entropy

Article

Geometric Learning of Canonical Parameterizations of 2*D*-Curves

Ioana Ciuclea [1,†], Giorgio Longari [2,3,†] and Alice Barbora Tumpach [2,3,4,*,†]

[1] Faculty of Physics and Mathematics, Department of Mathematics, West University of Timişoara, Vasile Pârvan 4, 300392 Timişoara, Romania; ioana.ciuclea@e-uvt.ro
[2] Computer Vision Lab, Technische Universität Wien, Karlsplatz 13, 1040 Vienna, Austria; giorgio.longari@unimib.it
[3] Wolfgang Pauli Institut, Oskar-Morgensternplatz 1, 1090 Vienna, Austria
[4] Laboratoire Painlevé, Lille University, 59650 Villeneuve d'Ascq, France
[*] Correspondence: alice-barbora.tumpach@univ-lille.fr
[†] The authors are listed in alphabetic order, see the Author contributions section for each contribution.

Abstract

Most datasets encountered in computer vision and medical applications present symmetries that should be taken into account in classification tasks. A typical example is the symmetry by rotation and/or scaling in object detection. A common way to build neural networks that learn the symmetries is to use data augmentation. In order to avoid data augmentation and build more sustainable algorithms, we present an alternative method to mod out symmetries based on the notion of section of a principal fiber bundle. This framework allows to use simple metrics on the space of objects in order to measure dissimilarities between orbits of objects under the symmetry group. Moreover, the section used can be optimized to maximize separation of classes. We illustrate this methodology on a dataset of contours of objects for the groups of translations, rotations, scalings and reparameterizations. In particular, we present a 2-parameter family of canonical parameterizations of curves, containing the constant-speed parameterization as a special case, which we believe is interesting in its own right. We hope that this simple application will serve to convey the geometric concepts underlying this method, which have a wide range of possible applications.

Keywords: principal fiber bundles; reparameterizations; group of diffeomorphisms; shape-preserving groups; plane curves; section of a fiber bundle; arc-length parameterization; curvature-weighted parameterization

Academic Editors: Fanzhang Li and Li Liu

Received: 20 October 2025
Revised: 7 December 2025
Accepted: 15 December 2025
Published: 30 December 2025

1. Extended Abstract

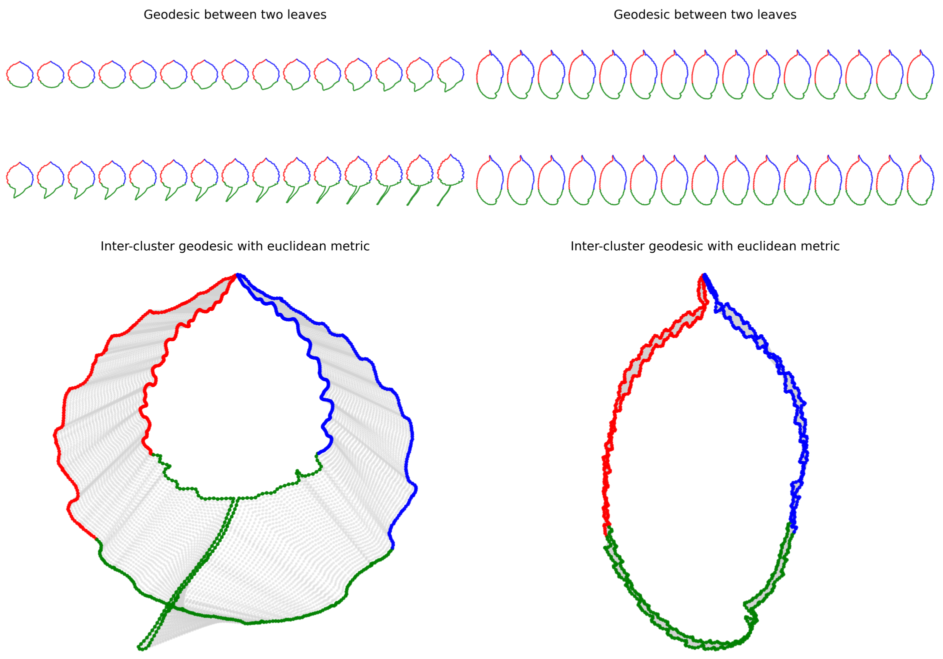
Our visual system is trained to identify objects that differ only by the action of a shape-preserving group, like the group of translations, rotations, and scalings. Consequently, these symmetries need to be taken into account in the design of algorithms for object detection and classification. A common way to build neural networks that learn the symmetries is to use data augmentation. This involves adding to the dataset new samples obtained by letting the symmetry group act on the original samples, for example, adding rotated images to the original images. In addition to the fact that data augmentation increases computational cost, it is also very memory-intensive. In this paper, we will consider, in particular, the symmetry group consisting of reparameterizations of contours in the plane, which is an infinite-dimensional Lie group.

In order to avoid data augmentation and build more sustainable algorithms, we present an alternative method to mod out symmetries based on the notion of section (also called cross-section) of a principal fiber bundle (see Sections 2.2 and 2.3). Within this framework, a distinguished object is selected in each orbit under the symmetry group. This amounts to normalization or standardization of samples with respect to the action of the groups of translations, rotations, scalings, and reparameterizations.

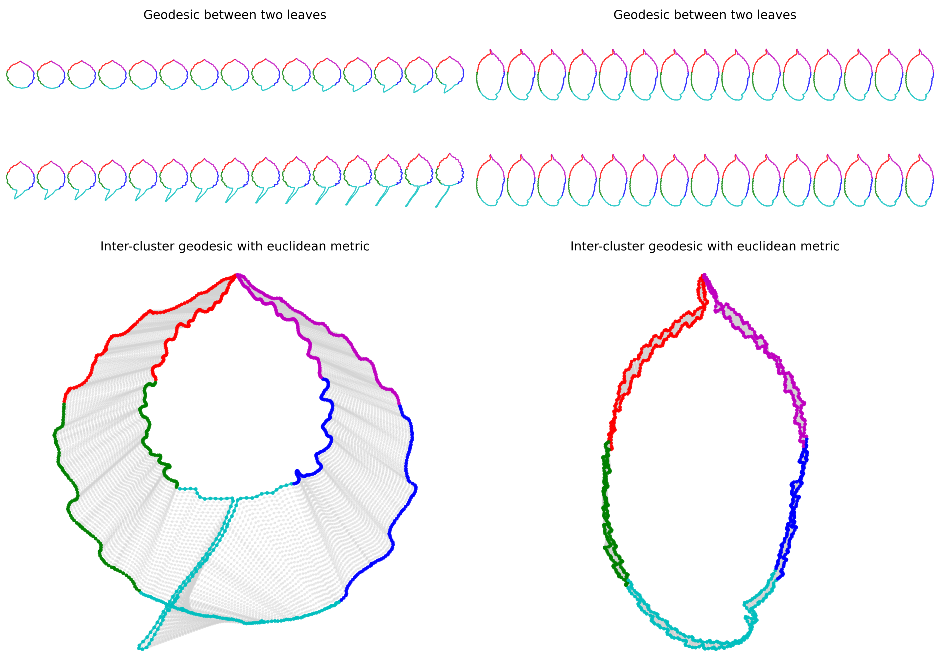
One aim of the present paper is to investigate canonical parameterizations of curves, which allow one to mod out the action of the infinite-dimensional group of diffeomorphisms acting on curves by reparameterizations. A canonical parameterization can be understood as an automatic way to re-sample a curve according to some of its geometric features. An example of a canonical parameterization is provided by the arc-length parameterization, which consists of a unit speed travel along the shape drawn by the curve. In Section 3.2, we present a new 2-parameter family of canonical curve parameterizations, called curvature-weighted clock parameterizations, inspired by the small hand trajectory on a traditional clock, which moves at a constant angle every hour. These canonical parameterizations are very natural and may be a good choice in many applications, particularly in the presence of noise.

When the quotient space by the group action is unique, sections, when they exist, are numerous. In fact, for trivial fiber bundles like the fiber bundle of parameterized curves studied in the present paper, the space of sections is infinite-dimensional. Therefore, the present approach allows for a lot of flexibility and can be customized for particular applications. It also allows us to use a simple distance function on the total space of the fiber bundle in order to measure dissimilarities between orbits of objects under the symmetry group. Indeed, restricting a simple distance function, such as the L^2 distance, to the range of a chosen section, we obtain a distance function on the quotient space, which is easy to compute. An example of this construction of distance functions between curves irrespective of their parameterization is given in Section 2.7. They are straightforward to compute, and do not rely on any energy minimization algorithm. During training for a classification task, the section used to design the distance function measuring the dissimilarities between orbits can be optimized to maximize the separation of classes, solving a metric learning problem (see Sections 2.8 and 3.3). Moreover, the optimal section gives rise to an optimal correspondence between points along any pair of contours in the dataset, solving a registration task. It therefore allows us to interpolate between contours, leading to optimal deformations between shapes (see Figure 1). Last but not least, our standardization procedure can be integrated into all classification algorithms for contours as a pre-processing step, allowing us to improve classification performance (see Section 4).

In Section 3, we illustrate this methodology with a dataset of leaves. More precisely, we optimize the Dunn index of clustering over a 2-parameter family of sections corresponding to the curvature-weighted clock parameterizations defined in Section 3.2. In Section 4.1, we show that this solution leads to good classification results for very low computational costs using classical machine learning algorithms. Indeed, with an optimization over only 2 parameters, our algorithm reaches 0.9602 accuracy (96.02% of correct classifications) with SVM for the dataset of Swedish leaves, whereas the state-of-the art model VGG-16 needs 138 million parameters to reach perfect accuracy (100% correct classifications) on the same dataset (see Section 4.1). We also show that taking into account all the shape-preserving groups boosts classification performance of all the classification algorithms that we considered, with even an increase of 25.71% of correct classifications for KNN on the Swedish leaf dataset (Section 4.1). Therefore, we argue that our method is a good pre-processing step that should be performed before any more complex feature extraction algorithm on contours.


Geodesic between two leaves
Geodesic between two leaves
Inter-cluster geodesic with euclidean metric
Inter-cluster geodesic with euclidean metric

(**a**)


Geodesic between two leaves
Geodesic between two leaves
Inter-cluster geodesic with euclidean metric
Inter-cluster geodesic with euclidean metric

(**b**)

Figure 1. (**a**) Left: The pair of leaves from the Swedish dataset that maximizes the intraclass distance is extracted from the training set, and the interpolation of their optimal parameterizations for the Dunn index is displayed for the parameters $(n = 3, \lambda = 2000)$. Right: The pair of leaves from the Swedish dataset that minimizes the interclass distance is extracted from the training set, and the interpolation of their optimal parameterizations for the Dunn index is displayed for $(n = 3, \lambda = 2000)$. (**b**) Left: The pair of leaves from the Swedish dataset that maximizes the intraclass distance is extracted from the training set, and the interpolation of their optimal parameterizations for the Davies Bouldin index is displayed for the parameters $(n = 5, \lambda = +\infty)$. Right: The pair of leaves from the Swedish dataset that minimizes the interclass distance is extracted from the training set, and the interpolation of their optimal parameterizations is displayed for $(n = 5, \lambda = +\infty)$. We can see that the same pair of leaves maximizes the intraclass distance both for the Dunn index and the Davies Bouldin index, and the same pair of leaves minimizes the interclass distance for both indices.

The main contributions of this paper are the following:

- The idea of using sections of principal fiber bundles in order to mod out symmetries is explained in a comprehensive manner and illustrated in the context of plane curves for classical shape-preserving groups (Section 2.2).
- A 2-parameter family of canonical contour parameterizations is introduced, called curvature-weighted clock parameterizations (Section 3.2).
- For a labeled dataset of contours, the separation of classes is optimized based on cluster validity indices such as the Dunn index (Section 3.3).
- We demonstrate and quantify how taking into account symmetries affects clustering and classification results (Section 4.1).
- The proposed method not only allows us to measure distances between shapes in a parameterization-invariant manner, but also provides a registration and optimal deformation between shapes at a very low computational cost.

The code is available at the following link: https://github.com/GiLonga/Geometric-Learning (accessed on 13 November 2025). A tutorial notebook showcasing an application of the code to a specific dataset is available at the following link: https://github.com/ioanaciuclea/geometric-learning-notebook (accessed on 13 November 2025).

2. Mathematical Background and Method

2.1. *Parameterized Versus Unparameterized 2D-Curves*

In this section, we recall the distinction between parameterized and unparameterized $2D$-curves [1,2]. We will be mainly interested in the contours of objects, like the contours of objects depicted in Figure 2, which mathematically correspond to Jordan curves in the plane. More precisely, we will consider the following space of smooth embedded closed curves in the plane:

$$\mathcal{P} = \{\gamma \in \mathcal{C}^\infty(\mathbb{S}^1, \mathbb{R}^2), \gamma \text{ injective}, \gamma'(s) \neq 0, \forall s \in \mathbb{S}^1\}. \tag{1}$$

In what follows, the unit circle $\mathbb{S}^1$ will be identified with $\mathbb{R}/\mathbb{Z} = \{t \in [0,1], 0 \sim 1\}$ via the map $\iota : \mathbb{R}/\mathbb{Z} \to \mathbb{C}$, $[t] \mapsto e^{2\pi i t}$. In particular, this identification distinguishes the point $\iota(0) = (1,0)$ in $\mathbb{S}^1 \subset \mathbb{C}$.

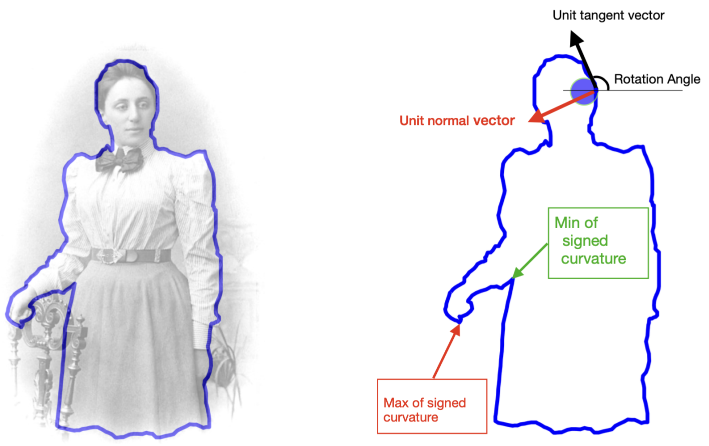


Figure 2. Emmy Noether and the moving frame associated with her profile. The signed curvature κ is defined as the rate of turning angle of the moving frame associated with a parameterized plane curve. The maximum and the minimum of the signed curvature correspond to two points where the curvature is extremal.

The space $\mathcal{P}$ has a natural structure of smooth Fréchet manifold [3]. Note that the parameterization of a contour with parameter space $\mathbb{S}^1$ is not unique. In fact, the group $\mathcal{G} = \text{Diff}^+(\mathbb{S}^1)$, consisting of orientation-preserving diffeomorphisms of $\mathbb{S}^1$, is a Fréchet Lie group acting smoothly on $\mathcal{P}$ by precomposition:

$$\begin{array}{ccc} \mathcal{G} \times \mathcal{P} & \to & \mathcal{P} \\ (\psi, \gamma) & \mapsto & \gamma \circ \psi^{-1} \end{array}$$

This action preserves the shapes of curves, and also the direction of travel along the curves. Moreover, two parameterized curves γ_1 and γ_2 in $\mathcal{P}$ corresponding to the same oriented contour in the plane are necessarily related by a diffeomorphism $\psi \in \mathcal{G}$: $\gamma_1 = \gamma_2 \circ \psi^{-1}$. Given a parameterized curve $\gamma \in \mathcal{P}$, one can consider its equivalence class $[\gamma]$ modulo the action of $\mathcal{G}$:

$$[\gamma] = \{\gamma \circ \psi^{-1}, \psi \in \mathcal{G}\}, \tag{2}$$

also called the orbit of γ for the $\mathcal{G}$-action. The equivalence class $[\gamma]$ is uniquely characterized by the range of $\gamma : \mathbb{S}^1 \to \mathbb{R}^2$, which is the shape drawn by γ in the plane, also called the unparameterized curve associated with γ, together with its orientation (the direction of travel). Consequently, the shape space of oriented contours in the plane is the quotient space $\mathcal{P}/\mathcal{G}$ of the manifold of smooth embeddings $\mathcal{P}$ modulo the action of the Fréchet Lie group $\mathcal{G}$. It was proven in [3] that this quotient space admits a natural structure of smooth manifold and that the canonical projection

$$\begin{array}{cccc} \pi : & \mathcal{P} & \longrightarrow & \mathcal{P}/\mathcal{G}, \\ & \gamma & \longmapsto & \pi(\gamma) = [\gamma], \end{array} \tag{3}$$

onto the quotient space defines a principal fiber bundle in the Fréchet category. This result was extended to freely immersed curves in [4], with some missing arguments in the proof, which were fully fixed in [5]. A visualization of a fiber bundle is given in Figure 3.

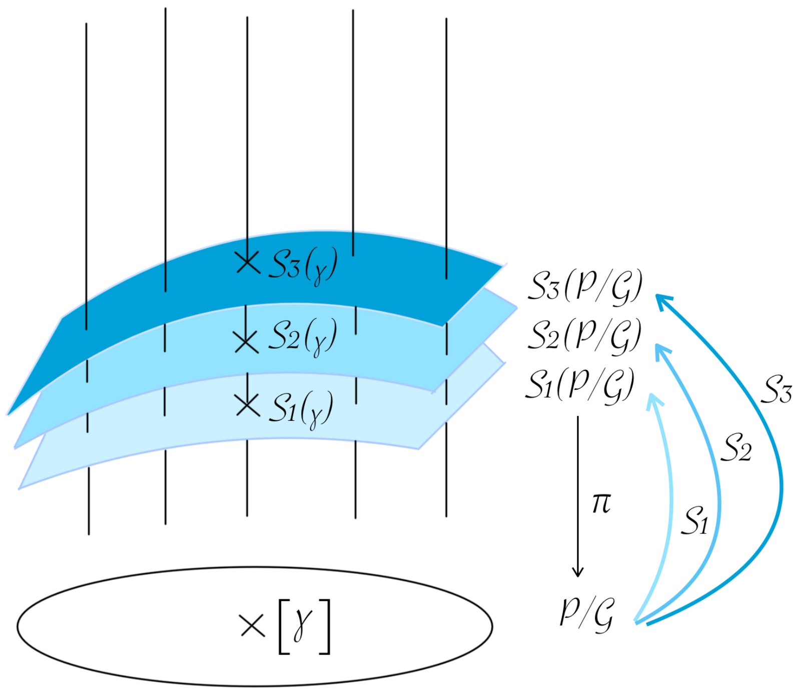

$S_3(\gamma)$
$S_2(\gamma)$
$S_1(\gamma)$
$S_3(\mathcal{P}/\mathcal{G})$
$S_2(\mathcal{P}/\mathcal{G})$
$S_1(\mathcal{P}/\mathcal{G})$
S_3
S_2
S_1
π
$\mathcal{P}/\mathcal{G}$
$\times[\gamma]$

Figure 3. Illustration of a fiber bundle $\pi : \mathcal{P} \to \mathcal{P}/\mathcal{G}$ with three different sections $S_i : \mathcal{P}/\mathcal{G} \to \mathcal{P}$.

2.2. Sections of Fiber Bundles

In the present paper, we will be interested in choosing smoothly a preferred parameterization in each equivalence class $[\gamma]$ defined by (2), where γ belongs to (some open subset of) the space of smooth embedded closed curves $\mathcal{P}$. This corresponds to the choice of a smooth section of the principal fiber bundle $\pi : \mathcal{P} \to \mathcal{P}/\mathcal{G}$ (see Figure 3). Let us recall the following definition.

Definition 1. *A* (global) smooth section *of a fiber bundle* $\pi : \mathcal{P} \to \mathcal{B}$ *is a smooth map* $s : \mathcal{B} \to \mathcal{P}$ *such that* $\pi \circ s = Id_{\mathcal{B}}$*.*

Remark 1. *It can be shown that the range of a smooth section* $s : \mathcal{B} \to \mathcal{P}$ *of a principal fiber bundle* $\pi : \mathcal{P} \to \mathcal{B}$ *is a smooth submanifold of* $\mathcal{P}$*. In particular, the manifold consisting of closed curves parameterized by arc length is a smooth manifold [6,7]. Using the parametrization with arc length of some particular curves, the authors of [8] were able to give the exact analytical solution of the linear static equation of curved Bernoulli–Euler beam.*

The notion of section can be applied to different quotient spaces, in particular to the quotient space of the space of embedded closed curves modulo shape-preserving groups. We will see in Sections 3 and 4.1 how the choice of a particular section can influence downstream analysis.

2.3. Canonical Parameterizations of 2*D-Curves as Smooth Sections*

An example of a smooth section for the fiber bundle $\pi : \mathcal{P} \to \mathcal{P}/\mathcal{G}$ is provided by the submanifold of curves parameterized proportional to arc-length. Let us recall how this particular parameterization is defined. Given a smooth parameterized curve in the plane $\gamma \in \mathcal{P}$, its length is defined as

$$\text{Length}(\gamma) = \int_0^1 \|\gamma'(t)\| dt, \tag{4}$$

where $\|\cdot\|$ denotes the Euclidean norm in $\mathbb{R}^2$. The length is a geometric invariant of the curve, i.e., it does not depend on the parameterization. Given a starting point, which in our case will be the image of $0 \in \mathbb{R}/\mathbb{Z}$, there is a canonical way to reparameterize a curve $\gamma \in \mathcal{P}$ by arc length, producing a unit speed curve. This procedure will change the parameter domain when the length of the curve is not equal to 1, and therefore may not belong to $\mathcal{P}$. However, there is a unique constantspeed reparameterization of $\gamma \in \mathcal{P}$ with parameter domain $\mathbb{R}/\mathbb{Z} = \{t \in [0,1], 0 \sim 1\}$, given as follows.

Proposition 1. *Given a curve* $\gamma \in \mathcal{P}$*, consider the map* ψ *defined as*

$$\psi(t) = \frac{1}{\text{Length}(\gamma)} \int_0^t \|\gamma'(s)\| ds, \tag{5}$$

where $t \in [0,1]$*. Then,* $\psi : \mathbb{R}/\mathbb{Z} \to \mathbb{R}/\mathbb{Z}$ *is an orientation-preserving diffeomorphism, fixing* $0 \in \mathbb{R}/\mathbb{Z}$*. Moreover, the parameterized curve* $p(\gamma) = \gamma \circ \psi^{-1} \in \mathcal{P}$ *is the unique constant-speed reparameterization of* γ *with parameter space* $\mathbb{R}/\mathbb{Z} = \{t \in [0,1], 0 \sim 1\}$*, which maps* $0 \in \mathbb{R}/\mathbb{Z}$ *to* $\gamma(0)$ *and has the same orientation as* γ*.*

Definition 2. *We will denote by* $\mathcal{A}$ *the subset of* $\mathcal{P}$ *consisting of constant speed curves with parameter space* $\mathbb{R}/\mathbb{Z} = \{t \in [0,1], 0 \sim 1\}$*. One has*

$$\mathcal{A} = \{\gamma \in \mathcal{P}, \|\gamma'(t)\| = \text{Length}(\gamma), \forall t \in \mathbb{R}/\mathbb{Z}\}. \tag{6}$$

The space $\mathcal{A}$ of constant-speed parameterized curves with parameter space $\mathbb{R}/\mathbb{Z} = \{t \in [0,1], 0 \sim 1\}$ is just one example of space of canonically parameterized curves. The possible choices are infinite. In the present paper, we will use the following terminology:

Definition 3. *Let $\mathcal{P}$ be the infinite-dimensional manifold of parameterized closed embedded curves in $\mathbb{R}^2$ defined in (1), and $\mathcal{G} = \mathrm{Diff}^+(\mathbb{S}^1)$ the Fréchet Lie group of orientation-preserving reparameterizations. A* canonical parameterization *will refer to the choice of a smooth section $s : \mathcal{P}/\mathcal{G} \to \mathcal{P}$ of the principal fiber bundle $\pi : \mathcal{P} \to \mathcal{P}/\mathcal{G}$, which depends only on the geometric features of oriented contours. It can be understood as an automatic procedure to parameterize curves. It allows us to single out a distinguished parameterization of an oriented contour $[\gamma] \in \mathcal{P}/\mathcal{G}$ by associating with $\pi(\gamma) = [\gamma]$ the parameterized curve $s([\gamma]) \in \mathcal{P}$. It also provides a (non-linear) projection $p : \mathcal{P} \to s(\mathcal{P}/\mathcal{G})$, i.e., satisfying $p^2 = p$, given by*

$$p(\gamma) = s([\gamma]). \tag{7}$$

2.4. Examples of Curvature-Weighted Canonical Parameterizations

In Definition 2, the parameterization proportional to arc-length with parameter space $\mathbb{R}/\mathbb{Z} = \{t \in [0,1], 0 \sim 1\}$ is defined, and the corresponding submanifold $\mathcal{A} \subset \mathcal{P}$ is given in (6). In [9], we have introduced the parameterization proportional to curvature-length, as well as a variant called the parameterization proportional to curvarc-length. In fact, these particular procedures to automatically parameterize curves belong to a one-parameter family of canonical parameterizations, and we recall their construction below (see Equation (8)). This family provides an interpolation between the parameterization proportional to curvature-length ($\lambda = 0$), the parameterization proportional to curvarc-length ($\lambda = 1$), and converges to the parameterization proportional to arc-length when $\lambda \to +\infty$ [9]. In order to have a picture in mind (see Figure 4) where the contour of Emmy Noether is sampled according to five different parameterizations from this family.

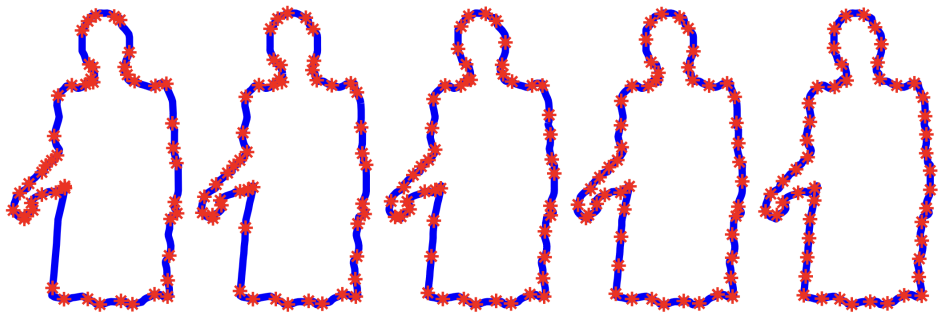


Figure 4. A one-parameter family of canonical parameterizations: Each contour of Emmy Noether is parameterized in a unique way using Equation (8) for a given parameter λ. The sample points are the images of a uniform sampling of the interval $[0;1]$. The leftmost contour is parameterized proportionally to the curvature-length with parameter space $\mathbb{R}/\mathbb{Z} = \{t \in [0,1], 0 \sim 1\}$ and corresponds to $\lambda = 0$. For this parameterization, sample points are concentrated on high-curvature portions of the curve, whereas flat pieces contain no sample points. The rightmost contour is parameterized proportionally to arc-length with parameter space $\mathbb{R}/\mathbb{Z} = \{t \in [0,1], 0 \sim 1\}$ and corresponds to $\lambda = +\infty$. In this case, sample points are uniformly distributed along the contour. In between, from left to right, the following parameters are used $\lambda = 0.3$, $\lambda = 1$, $\lambda = 2$ (see Equation (8)).

Equivalently, this one-parameter family of canonical parameterizations corresponds to a one-parameter family of sections $s_\lambda : \mathcal{P}/\mathcal{G} \to \mathcal{P}$, where $s_{+\infty}(\mathcal{P}/\mathcal{G}) = \mathcal{A}$ (see Figure 4). These parameterizations are defined using the local differential invariant of curves given by the signed curvature κ. The signed curvature κ is the rate of turning angle of the moving frame attached to a parameterized curve. A visualization of this moving frame is illustrated in Figure 2.

More precisely, we introduce a one-parameter family of canonical reparameterizations of curves $\gamma \in \mathcal{P}$ as follows. For a given $\lambda \in (0, +\infty)$, the corresponding reparameterization of a curve $\gamma \in \mathcal{P}$ is given by $p_\lambda(\gamma) = \gamma \circ \Phi_\lambda^{-1}$, where Φ_λ depends on γ through the following equation involving the signed curvature κ of γ:

$$\Phi_\lambda(s) = \frac{\int_0^s (\lambda \,\mathrm{Length}(\gamma) + |\kappa(\gamma(s))|)\|\gamma'(s)\|ds}{\int_0^1 (\lambda \,\mathrm{Length}(\gamma) + |\kappa(\gamma(s))|)\|\gamma'(s)\|ds}, \quad \lambda > 0. \tag{8}$$

Note that the function $s \mapsto \int_0^s (\lambda \,\mathrm{Length}(\gamma) + |\kappa(\gamma(s))|)\|\gamma'(s)\|ds$ is strictly increasing when $\lambda > 0$, or when $[\gamma]$ does not contain flat pieces. In these cases, Φ_λ is an orientation-preserving diffeomorphism of $\mathbb{R}/\mathbb{Z}$ fixing $0 \in \mathbb{R}/\mathbb{Z}$. In the case $\lambda = 0$ and $\kappa = 0$ on some non-empty interval, the map Φ_0 defined by

$$\Phi_0(s) = \frac{\int_0^s |\kappa(\gamma(s))|\|\gamma'(s)\|ds}{\int_0^1 |\kappa(\gamma(s))|\|\gamma'(s)\|ds}, \tag{9}$$

is not injective and its graph presents horizontal portions. Consequently, Φ_0 is not a diffeomorphism, but it belongs to the semi-group of generalized reparametrizations [10]. In other words, Φ_0 is the limit of the diffeomorphisms Φ_λ when $\lambda \to 0$, and $p_0(\gamma)$ can be defined as the limit of $p_\lambda(\gamma)$ in an appropriate topology.

Remark 2. *In Equation (6) [9], another family of curvature-weighted parameterizations was introduced to assign a prescribed anatomical location to sample points on bone contours extracted from X-ray scans. It was used to measure the evolution of Rheumatoid Arthritis in a consistent way.*

2.5. Different Ways to Define a Riemannian Metric on Unparameterized Curves

In ref. [7], the authors present three different methods for quantifying dissimilarities in quotient spaces based on Riemannian geometry. These methods consist of defining a Riemannian metric on the quotient space $\mathcal{P}/\mathcal{G}$, which allows us to compute the length of paths in $\mathcal{P}/\mathcal{G}$. The distance between two points $[\gamma_1]$ and $[\gamma_2]$ in $\mathcal{P}/\mathcal{G}$ (hence between two contours in the plane) is then defined as the infimum of the length of all paths connecting $[\gamma_1]$ to $[\gamma_2]$. We recall, briefly, these three points of view.

2.5.1. Quotient Metric

The first method consists of endowing the space $\mathcal{P}$ with a $\mathcal{G}$-invariant Riemannian metric. In this case, the Riemannian metric on $\mathcal{P}$ descends to a Riemannian metric on the quotient space, called the quotient metric. A large body of literature is devoted to this method (see [1,2,11] and the references therein). For this method,

(i) Computing the distance between two points $[\gamma_1]$ and $[\gamma_2]$ relies on two optimization steps: First, the computation of the minimal path between γ_1 and a element in the orbit of γ_2. Second, the optimization over the infinite-dimensional group of reparameterizations acting on γ_2.

(ii) The Riemannian metric on $\mathcal{P}$ is, in general, difficult to adjust to applications since the horizontal space may be difficult to compute.

(iii) The added dimensions (infinitely many) that are going from $\mathcal{P}/\mathcal{G}$ to $\mathcal{P}$ are dimensions that are irrelevant for the analysis of data living in the quotient space, but they need to be taken into account, particularly in the second optimization step.

A class of reparameterization-invariant Riemannian metrics on curves, called elastic metrics, was introduced in [12]. It corresponds to a 2-parameter family of Riemannian metrics $G^{a,b}$ penalizing bending as well as stretching. In [13], it was shown that, for a certain relation between the parameters, the resulting metric is flat on parameterized open curves. A similar method for simplifying the analysis of plane curves was introduced in [14]. These results have been generalized in [15], where the authors introduced another family of metrics, including the metrics from [12,14], which can be described using the restrictions of flat metrics to some cones. The flattening map has been significantly simplified in [16] and the previous cones interpreted as Regge cones. In [17], a precise algorithm for the matching problem of piecewise linear curves is implemented, giving a tool to compare contours in a meaningful way. For other parameter values, the $F^{a,b}$ transform introduced in [16] allows us to extend the precise algorithm of [17] to arbitrary parameter values (a, b). Approximations of these algorithms using neural networks were implemented in [18]. We believe that the results obtained do not justify the choice of these computationally intensive designs and are looking for more sustainable solutions.

2.5.2. Immersion Metric

The second method consists of identifying the quotient space with the range of a smooth section $s : \mathcal{P}/\mathcal{G} \to \mathcal{P}$ and endowing the submanifold $s(\mathcal{P}/\mathcal{G}) \subset \mathcal{P}$ with a Riemannian metric, such as those induced by a Riemannian metric on $\mathcal{P}$. In this case, the Riemannian metric on $\mathcal{P}$ does not need to be $\mathcal{G}$-invariant. For this method,

(i) Computing the distance between two points $[\gamma_1]$ and $[\gamma_2]$ relies on one optimization step with constraint: it consists of minimizing the length of paths constrained to remain in the submanifold $s(\mathcal{P}/\mathcal{G}) \subset \mathcal{P}$.
(ii) The dimension of the space is preserved, since the quotient space $\mathcal{P}/\mathcal{G}$ and the range of the section s are diffeomorphic.
(iii) The section s can be adapted to applications (we will see some optimization for sections s in the present paper).

Let us mention that, since the quotient space $\mathcal{P}/\mathcal{G}$ and the range of any section $s : \mathcal{P}/\mathcal{G} \to \mathcal{P}$ are diffeomorphic, any quotient metric on $\mathcal{P}/\mathcal{G}$ can be push-forward to the range $s(\mathcal{P}/\mathcal{G})$ of any section s. In [19], the authors have transported a particular family of quotient metrics, called elastic metrics, to the space of arc-length parameterized curves.

2.5.3. Gauge-Invariant Metric

The third method was introduced in [20] (see also [21]) and consists of defining a non-negative metric on $\mathcal{P}$ (i.e., a non-negative symmetric bilinear form on the tangent bundle $T\mathcal{P}$), called a gauge-invariant metric, whose kernel coincides exactly with the direction of the fibers of the canonical projection $\pi : \mathcal{P} \to \mathcal{P}/\mathcal{G}$, hence descending to a non-degenerate Riemannian metric on the quotient space. The idea behind this construction is that the vertical directions of the fiber bundle $\pi : \mathcal{P} \to \mathcal{P}/\mathcal{G}$ are irrelevant for the analysis of the data in the quotient space $\mathcal{P}/\mathcal{G}$; therefore, they should not interfere in the computation of distances in the quotient space. For this method,

(i) The dimensions irrelevant to the analysis of the quotient space do not play any role, since they do not contribute to the cost function.
(ii) A reparameterization of curves can be performed on the fly without affecting the minimization algorithms.
(iii) During a path-straightening algorithm for determining a geodesic in the quotient space, the paths can be lifted to $\mathcal{P}$ and reparameterized with time-dependent reparameterizations without affecting downstream analysis, allowing for more robust algorithms to be designed and improving their convergence.

An example of application of this method to curves for action recognition is given in [22].

2.6. The Geodesic Distance Function Associated with a Riemannian Metric

Recall that the geodesic distance between two points in a Riemannian manifold is defined as the infimum of the lengths of curves connecting these two points. For a finite-dimensional manifold, this distance is non-degenerate and allows one to separate points. In other words, the geodesic distance between two points in a finite-dimensional Riemannian manifold is zero if and only if these two points coincide.

In an infinite-dimensional setting, the geodesic distance function associated with a Riemannian metric can be degenerate. The first example of this infinite-dimensional phenomenon was explicitly given in [23]. In this paper, the authors considered the reparameterization-invariant L^2-Riemannian metric on the space of parameterized $2D$-curves, and the induced quotient metric on the space of unparameterized $2D$-curves. They proved that the quotient metric admits a vanishing geodesic distance function. In other words, the geodesic distance between any pair of curves is zero.

Clearly, when the distance function is degenerate, it cannot be used to measure the dissimilarities between pairs of points in the manifold. For this reason, as well as to avoid computationally costly optimization steps, we propose in this paper another strategy to measure the dissimilarity between contours in the plane.

2.7. Proposed Distance Between Oriented Contours

Recall that $\mathcal{P}$ defined in (1) is the space of embedded closed $2D$-curves. As a space of smooth functions on the compact manifold $\mathbb{S}^1$ with values in $\mathbb{R}^2$, it is contained in the Hilbert space of square-integrable functions on $\mathbb{S}^1$ with values in $\mathbb{R}^2$, denoted by $L^2(\mathbb{S}^1, \mathbb{R}^2)$. Recall that the scalar product in $L^2(\mathbb{S}^1, \mathbb{R}^2)$ is given by

$$\langle f, g \rangle_{L^2} = \int_{\mathbb{S}^1} f(t) \cdot g(t) dt, \tag{10}$$

where the dot denotes the scalar product on $\mathbb{R}^2$. The corresponding norm is given by

$$\|f\|_{L^2} = \left(\int_{\mathbb{S}^1} \|f(t)\|^2 dt \right)^{\frac{1}{2}}. \tag{11}$$

Since the scalar product (10) is not invariant by the group of reparameterizations $\mathcal{G}$, it cannot be used directly to measure the dissimilarity between oriented contours, since the result would depend on the way the contours are parameterized. However, if we fix the way contours are parameterized by choosing a canonical parameterization $s : \mathcal{P}/\mathcal{G} \to \mathcal{P}$, then any oriented contour $[\gamma]$ is associated with a unique function $s([\gamma])$ in $L^2(\mathbb{S}^1, \mathbb{R}^2)$, and we can measure the distance between $[\gamma_1]$ and $[\gamma_2]$ as

$$d_s([\gamma_1], [\gamma_2]) = \|s([\gamma_1]) - s([\gamma_2])\|_{L^2}. \tag{12}$$

In other words, the L^2-distance is restricted to the subset $s(\mathcal{P}/\mathcal{G})$, which is in one-to-one correspondence with the quotient space $\mathcal{P}/\mathcal{G}$ consisting of oriented contours. The distance on the space of contours $\mathcal{P}/\mathcal{G}$ given by (12) is non-degenerate:

Proposition 2. *For any section $s : \mathcal{P}/\mathcal{G} \to \mathcal{P}$, and any oriented contours $[\gamma_1]$ and $[\gamma_2]$ in $\mathcal{P}/\mathcal{G}$, one has*

$$d_s([\gamma_1], [\gamma_2]) = 0 \Leftrightarrow [\gamma_1] = [\gamma_2]. \tag{13}$$

Proof of Proposition 2. Suppose that $d_s([\gamma_1], [\gamma_2]) = 0$. By definition (12), $\|s([\gamma_1]) - s([\gamma_2])\|_{L^2} = 0$. Since $L^2(\mathbb{S}^1, \mathbb{R}^2)$ is a Hilbert space, this implies that $s([\gamma_1]) = s([\gamma_2])$ as elements in $L^2(\mathbb{S}^1, \mathbb{R}^2)$, and is thus almost everywhere. Since both $s([\gamma_1])$ and $s([\gamma_2])$

are smooth functions, one has $s([\gamma_1])(t) = s([\gamma_2])(t)$ for any $t \in \mathbb{R}/\mathbb{Z}$. Consequently, $\pi(s([\gamma_1])) = \pi(s([\gamma_2]))$. But by the Definition 1 of a section, $\pi \circ s = \mathrm{Id}_{\mathcal{P}/\mathcal{G}}$. Hence, $[\gamma_1] = [\gamma_2]$. The other implication is trivial. □

Proposition 3. *For any smooth section $s : \mathcal{P}/\mathcal{G} \to \mathcal{P}$, $d_s : \mathcal{P}/\mathcal{G} \times \mathcal{P}/\mathcal{G} \to [0, +\infty)$ defined by (12) satisfies the triangular inequality.*

Proof. Consider any smooth section $s : \mathcal{P}/\mathcal{G} \to \mathcal{P}$, as well as the contours $[\gamma_1], [\gamma_2]$ and $[\gamma_3]$ in $\mathcal{P}/\mathcal{G}$. One has

$$\begin{aligned} d_s([\gamma_1],[\gamma_3]) &= \|s([\gamma_1]) - s([\gamma_3])\|_{L^2} \leq \|s([\gamma_1]) - s([\gamma_2])\|_{L^2} + \|s([\gamma_2]) - s([\gamma_3])\|_{L^2} \\ &\leq d_s([\gamma_1],[\gamma_2]) + d_s([\gamma_2],[\gamma_3]), \end{aligned}$$

where we used the triangle inequality in $L^2(\mathbb{S}^1, \mathbb{R}^2)$. □

Remark 3. *It follows from Proposition 2 and 3 that d_s is indeed a distance function on the quotient space $\mathcal{P}/\mathcal{G}$, i.e., it is non-negative, symmetric, non-degenerate, and satisfies the triangle inequality.*

Remark 4. *Propositions 2 and 3 can be generalized to any norm on the space of functions from $\mathbb{S}^1$ to $\mathbb{R}^2$. The L^2-norm was chosen since it is well suited for the datasets we are considering in Sections 3 and 4.1. For instance, a leaf with peduncle and a leaf without peduncle belonging to the same class of leaves, see Section 3.1.4, are close in distance when induced by the L^2-norm but distant if we use the L_∞-norm instead.*

2.8. Metric Learning

Metric learning is a branch of Geometric Learning devoted to learning a distance function from a dataset. It emerged from the observation that the Euclidean distance of the ambient space in which the dataset is encoded may not be the best choice for measuring distances. Application-driven metric learning aims to design a distance function that measures similarities between sample points in a pertinent way for the application at hand.

In the present paper, we propose a metric learning algorithm based on an optimization over the section s. The distance defined in (12) clearly depends on the choice of section $s : \mathcal{P}/\mathcal{G} \to \mathcal{P}$. Given a contour classification task, we can optimize the section s to obtain the best separation between classes on the training set. The quality of a clustering in a metric space can be measured using different validation indices (see Section 2.9), such as the Dunn index (Equation (16)). In Section 3.2, we present a 2-parameter family of sections $s_{\lambda,n}$ that is used to define distance functions on contours using Equation (12). The optimization of the corresponding cluster validation indices is performed in Section 3.3 for the leaf dataset. The improvement of the classification performance is analyzed in Section 4.1.

In the finite-dimensional context, the field of supervised PAC (Probably Approximately Correct) learning provides theoretical guarantees that explain why and when supervised learning algorithms work. For PAC-Bayes guaranties to learning settings with non-compact, finite-dimensional symmetries, we refer the reader to the recent paper [24]. As far as we know, such a Bayesian approach has not been investigated for infinite-dimensional groups of symmetries.

2.9. Validation Indices of a Clustering

In order to quantify how the choice of different sections influences the distances between samples, we use two cluster validation indices, the Dunn index (Section 2.9.1) and the Davies Bouldin index (Section 2.9.2). A comparison of these two indices is made in

Section 4.1. For a discussion and comparison of more general cluster validation techniques, we refer the reader to [25,26].

2.9.1. Dunn Index of a Clustering

In order to measure clustering efficiency in the algorithms we describe below, we use the Dunn index [27], which measures the ratio between the minimal interclass distance to the maximal intraclass distance. A high Dunn index characterizes dense and well-separated clusters, with a small variance between members of a cluster and different clusters sufficiently far apart, as compared to the within-cluster variance.

The Dunn index is computed as follows. For each class C_k, $1 \leq k \leq K$ (K being the number of classes), we compute the centroid c_k of class C_k as the mean of this class. In practice, the average of the positions of the points along the contours gives the average shape. The distance between classes $D_{\text{inter}}(k_1, k_2)$ is calculated as the distance between the centroid c_{k_1} of class C_{k_1} and the centroid and c_{k_2} of class C_{k_2}

$$D_{\text{inter}}(k_1, k_2) = d_s(c_{k_1}, c_{k_2}), \tag{14}$$

where d_s is defined in Equation (12) for a given section $s : \mathcal{P}/\mathcal{G} \to \mathcal{P}$ of the fiber bundle of parameterized contours (we will start with the section of arc-length parameterized contours, and optimize over a two-parameter family of sections in Section 3.3). The distance between classes $D_{\text{intra}}(k)$ is measured as the maximum distance between any pair of elements in class C_k:

$$D_{\text{intra}}(k) = \max_{i,j \in C_k} d_s(i, j). \tag{15}$$

The Dunn index is defined as follows, with K being the number of classes:

$$\text{Dunn}_{\lambda,n} = \frac{\min_{1 \leq k_1 < k_2 \leq K} D_{\text{inter}}(k_1, k_2)}{\max_{1 \leq k \leq K} D_{\text{intra}}(k)}. \tag{16}$$

2.9.2. Davies Bouldin Index of a Clustering

An alternative measure of clustering efficiency is the *Davies Bouldin index* [28] that measures the maximal ratio between the spread of two classes and the distance between their centroids. The Davies Bouldin index varies between 0 and $+\infty$, where a low index corresponds to a better classification. As with the Dunn index, for each class C_k, $1 \leq k \leq K$ (K being the number of classes), the centroid c_k of class C_k is computed as the mean of this class. Then, the mean distance $\bar{\delta_k}$ of the elements of the class C_k to their centroid c_k is computed as

$$\bar{\delta_k} = \frac{1}{|C_k|} \sum_{i \in C_k} d_s(i, c_k). \tag{17}$$

Finally, the Davies Bouldin index $DB_{\lambda,n}$ is defined as follows, with K being the number of classes:

$$DB_{\lambda,n} = \frac{1}{K} \sum_{k=1}^{K} \max_{k' \neq k} \left(\frac{\bar{\delta}'_k + \bar{\delta}_k}{d_s(c'_k, c_k)} \right). \tag{18}$$

Due to the averaging of the distances to a centroid over all elements of a class, the Davies Bouldin index is more stable than the Dunn index in the presence of outliers (see Section 4.2). On the other hand, the Dunn index can help detect outliers. By extracting from the dataset the pairs of samples from the same class maximizing the intraclass distance and the pair of samples from different classes that minimizes the interclass distance (see Section 3.3), one can spot some inconsistencies.

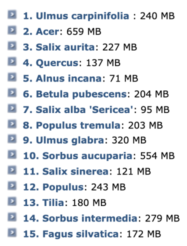
3. Illustration of the Methodology

3.1. Database and Pre-Processing Steps

3.1.1. Database

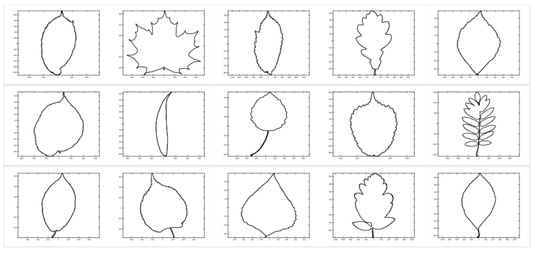
We used the Swedish leaves dataset from the Linköpling University, which can be freely downloaded from https://www.cvl.isy.liu.se/en/research/datasets/swedish-leaf/ (accessed on 8 September 2025).This dataset consists of pictures of leaves organized into 15 classes, with each class containing 75 leaves of the same variety. An element of each class is illustrated in Figure 5a, and the names of the corresponding varieties are listed in Figure 5b. In a preliminary step, we extract the contours of the leaves by transforming the pictures into black and white imprints, and then extract the boundaries of the resulting shapes with an appropriate algorithm (e.g., `bwboundaries` in Matlab). The resulting contours are illustrated in Figure 5c, and consist of an ordered set of points along the boundary of the leaves. This ordering gives us an initial parameterization γ of each contour.

(**a**)



1. **Ulmus carpinifolia** : 240 MB
2. **Acer**: 659 MB
3. **Salix aurita**: 227 MB
4. **Quercus**: 137 MB
5. **Alnus incana**: 71 MB
6. **Betula pubescens**: 204 MB
7. **Salix alba 'Sericea'**: 95 MB
8. **Populus tremula**: 203 MB
9. **Ulmus glabra**: 320 MB
10. **Sorbus aucuparia**: 554 MB
11. **Salix sinerea**: 121 MB
12. **Populus**: 243 MB
13. **Tilia**: 180 MB
14. **Sorbus intermedia**: 279 MB
15. **Fagus silvatica**: 172 MB

(**b**)



(**c**)

Figure 5. Dataset of Swedish leaves from the Linköpling University dataset https://www.cvl.isy.liu.se/en/research/datasets/swedish-leaf/ (accessed on 8 September 2025). (**a**) A sample image from each class of leaves is depicted (the classes are ordered from left to right and top to bottom) (**b**) corresponding classes (**c**) extracted contours using Matlab's function `bwboundaries` on binarized images.

We divide the resulting set of contours into a training set, containing 50 contours from each class, as well as a testing set containing the remaining contours. **In particular, the training set and the testing set are disjointed.**

3.1.2. Standardizing the Direction of Travel

The initial parameterizations of the contours obtained from the boundary extraction algorithm explained in Section 3.1.1 induce an orientation, leading to contours following clockwise or counterclockwise. As a first normalization step, we check if the contours are traveling counterclockwise, and flip the parameterization of those contours following clockwise. In order to automatically detect the orientation of a given contour, we compute the signed area enclosed by the contour. A positive signed area corresponds to a contour that traveled counterclockwise, and a negative area corresponds to a contour that traveled clockwise. The signed area can be computed using Stokes' Theorem by integration along the contour of a leaf:

$$\text{Area}(\gamma) = \int_{\gamma} x dy \tag{19}$$

In practice, for the dataset of Swedisch leaves, we did not encounter any contour following clockwise. The Dunn index defined by Equation (16), calculated on the training set containing 50 leaves of each of the $K = 15$ classes, is equal to 0.0286 when all contours have traveled counterclockwise. It decreases to 0.0127 when half of the contours chosen randomly have traveled clockwise, and the other half are traveled counterclockwise. To have a visual representation of the distance distribution of the leaves according to the distance function given by (12) with respect to the section s consisting of arc-length parameterized contours, we use the `tsne` algorithm. The resulting distribution of leaves in 2 dimensions with random direction of travel, as well as for contours that traveled counterclockwise, is given in Figure 6.

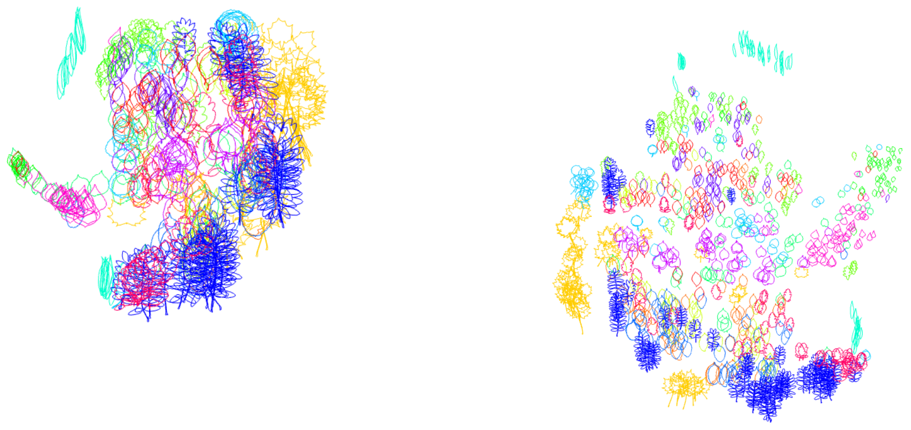


Figure 6. Normalization of the orientation variability. Two-dimensional representation of the distance distribution along the dataset using `tsne` algorithm (**a**). Before normalization of orientation (half of the contours are traveling clockwise, the other half counterclockwise), the Dunn index equals 0.0127. (**b**) After orientation normalization (all the contours are traveled counterclockwise), the Dunn index increases to 0.0286.

3.1.3. Standardizing the Starting Point of Parameterizations

Since the contour of a leaf is represented by an ordered set consisting of finitely many sample points along the contour, the starting point of this discretization induces variability that we need to take into account. In the continuous case, this amounts to standardizing the position of the starting point of the parameterization of contours. This corresponds to the normalization with respect to rotation in parameter space $\mathbb{S}^1$, i.e., with respect to the subgroup of rotations $\mathrm{Rot}(\mathbb{S}^1) \subset \mathrm{Diff}^+(\mathbb{S}^1)$, where $\mathrm{Diff}^+(\mathbb{S}^1)$ is the group of orientation-preserving reparameterizations.

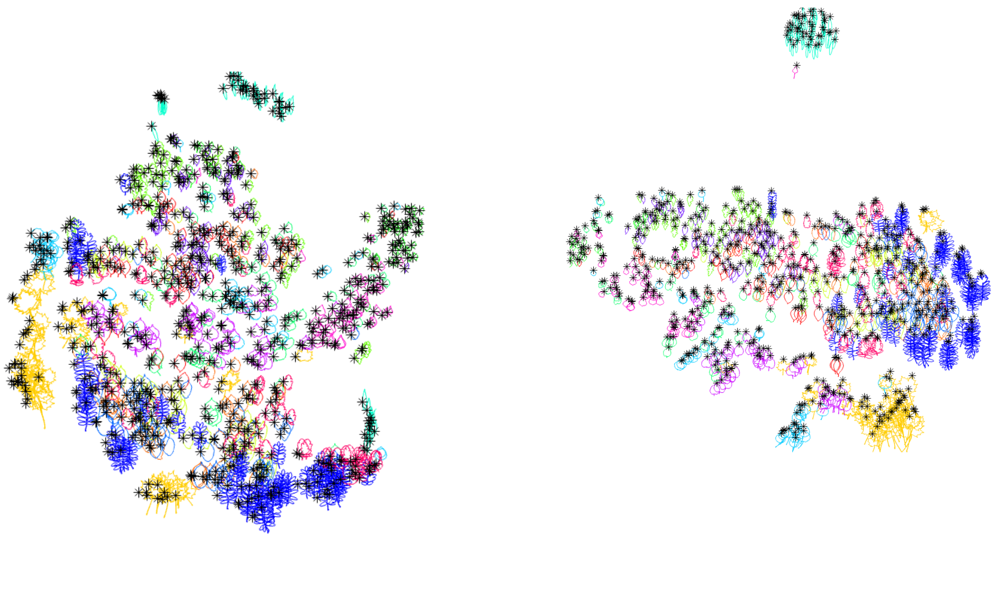
For the dataset of leaves at hand, we detect automatically the point of each contour with the largest vertical component (which was unique for all contours) and reorder the sample points in such a way that this particular point becomes the starting point. In Figure 7, we illustrate the distance distribution using the `tsne` algorithm before and after normalization of the starting points. The starting points are showcased as black dots along the contours. The Dunn index increases from 0.0286 to 0.0328 after this normalization step.

3.1.4. Standardizing the Scale Variability

The dataset contains leaves of different sizes, as can be seen in Figure 8a on 7 samples of Acer leaves. In order to recognize the class of a leaf irrespective of its size, we need to eliminate the variability of the scale. We tested two normalization procedures:

(a) **Normalization of the length of contours:** In this normalization method, we first compute the contour length of each leaf and then divide the initial parameterization by this length.

(b) **Normalization of the enclosed area:** Each contour is a Jordan curve in the plane and encloses a domain in the plane that corresponds to the surface of the corresponding leaf. In this normalization step, we compute the area of each leaf using Equation (19) and renormalize the initial parameterization to have a unit area by dividing the parameterization by the square-root of (the absolute value of) the area.



(**a**) (**b**)

Figure 7. Normalization of the starting point variability. Two-dimensional representation of the distance distribution along the dataset using `tsne` algorithm. (**a**) Before normalization of the starting points, the Dunn index equals 0.0286. (**b**) After starting point normalization, the Dunn index increases to 0.0328. The starting points are depicted as black dots.

As can be seen in Figure 8b,c, normalization to unit-length induces greater intraclass variability compared to normalization to unit-enclosed area. This is mainly due to the fact that, in the same class, leaves with peduncles as well as leaves without peduncles are present. Normalization by unit-length is heavily affected by the presence or absence of a peduncle. In contrast, the normalization to curves with unit-enclosed area is not affected by the presence or absence of peduncles, as peduncles barely contribute to the area.

Despite this fact, the Dunn index increases to 0.0587 after normalization by unit-length, and only to 0.0381 after normalization by unit area. This is due to the fact that the interclass distance increases more when normalization by the length is used, due to the characteristic boundary shape of different varieties of leaves. This can be seen in Figure 9. In the sequel, we therefore select the normalization by unit-length.

3.1.5. Standardizing the Position in Space

The shape of a leaf is invariant by translation in space. We have tested three normalization procedures that can be used to eliminate the variability in positions.

(a) **Starting point at the origin:** for this normalization method, we simply substract the coordinates of the first point visited by the initial parameterized contour, leading to a parameterized curve starting at $(0,0) \in \mathbb{R}^2$.

(b) **Center of mass of the contour at the origin:** in this normalization method, we compute the coordinates $(\bar{x},\bar{y})$ of the center of mass of the contour as the mean of the coordinated of points visited by the initial parameterization $\gamma(s) = (x(s), y(s))$:

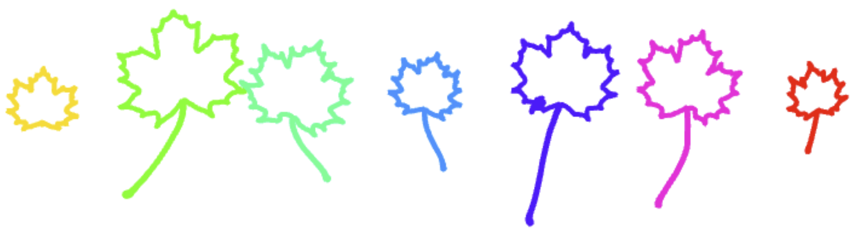
$$\begin{aligned} \bar{x} &= \tfrac{1}{\text{Length}(\gamma)} \int_0^1 x(s)\|\gamma'(s)\|ds \\ \bar{y} &= \tfrac{1}{\text{Length}(\gamma)} \int_0^1 y(s)\|\gamma'(s)\|ds \end{aligned} \tag{20}$$

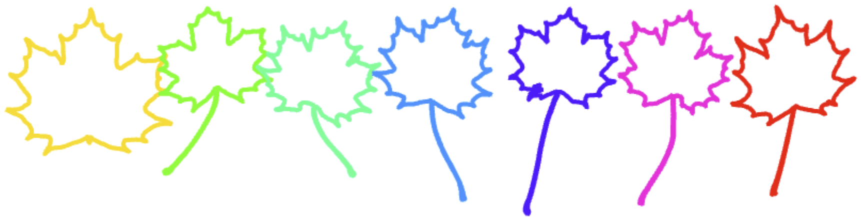
and then we substract the coordinates of this center of mass from the initial parameterization.

(c) **Center of gravity of the enclosed area at the origin:** in this normalization method, we compute the coordinated $(\hat{x}, \hat{y})$ of the center of gravity of the area enclosed by the contour (i.e., of the surface of the corresponding leaf) by using Stokes theorem:

$$\begin{aligned} \hat{x} &= \frac{1}{2\,\mathrm{area}(\gamma)} \int_\gamma x^2 dy \\ \hat{y} &= -\frac{1}{2\,\mathrm{area}(\gamma)} \int_\gamma y^2 dx \end{aligned} \tag{21}$$

and then we substract the coordinates of this center of gravity from the initial parameterization.


(**a**)


(**b**)

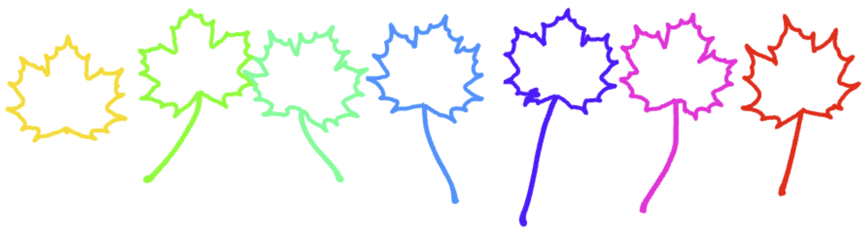

(**c**)

Figure 8. Normalization of the scale variability. Seven Acer leaves from the Swedish leaves dataset are used to illustrate two different normalizations of scaling. (**a**) Initial contours. (**b**) Each contour is rescaled in such a way that the length of the contour is equal to one. This scaling method has the effect of enlarging significantly the first leaf without peduncle. (**c**) Each contour is rescaled in such a way that the area enclosed by the contour is equal to one. For this scaling method, the leaves appear with the similar proportions.

As can be seen in Figure 10a, on seven Acer leaves from the Swedish dataset, the positions of the first points (in black), the centers of mass of the contours (in orange), and the centers of gravity of the enclosed areas (in purple) are different. In this experiment, the initial parameterization is counterclockwise, the starting point of each parameterized curve coincides with the point of the contour with the largest vertical coordinate (see Section 3.1.3) and the scaling is by unit-length.

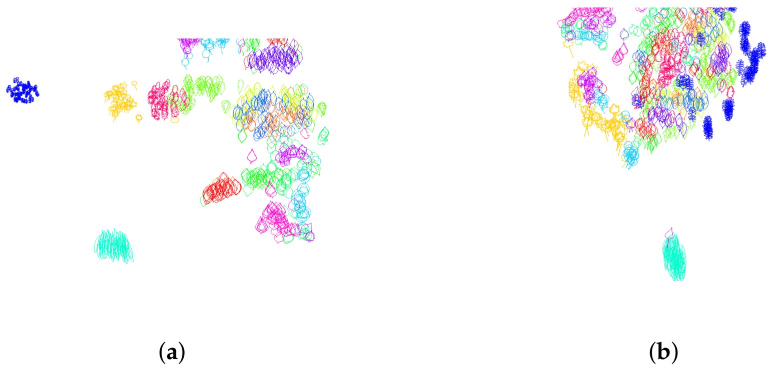


Figure 9. Normalization of the scale variability. Two-dimensional representation of the distance distribution along the dataset using `tsne` algorithm. (**a**) After normalization to unit-length curves, the Dunn increases to 0.0587. (**b**) After normalization to curves enclosing a unit area, the Dunn index increases to 0.0381.

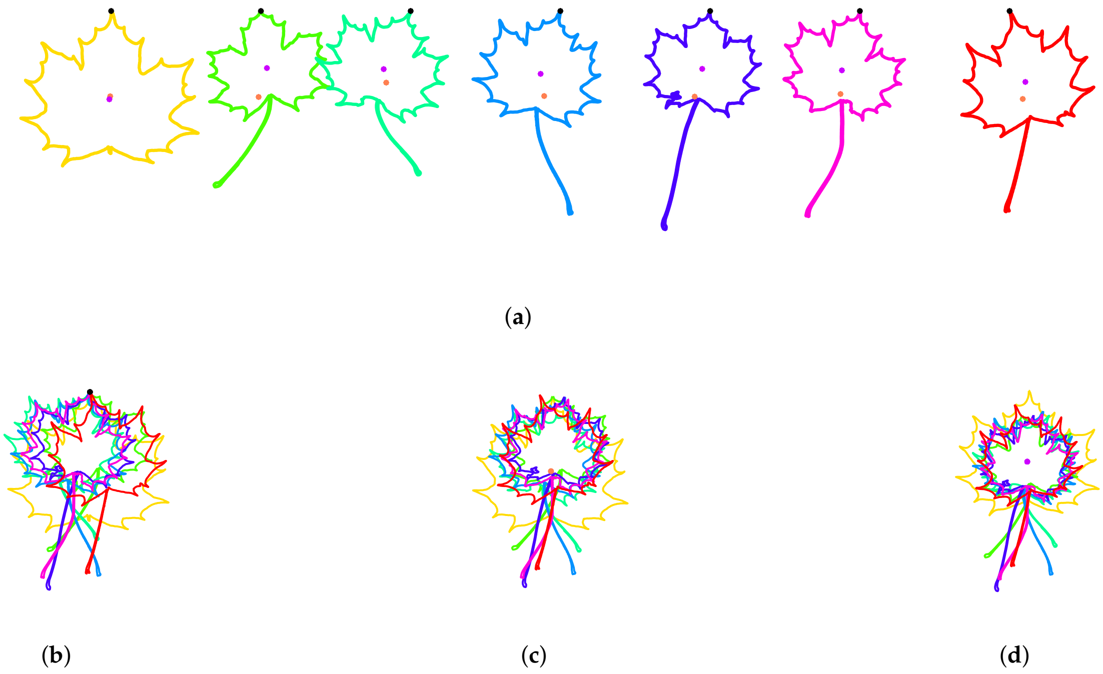


Figure 10. Normalization of the position in space. Seven Acer leaves from the Swedish leaves dataset are used to illustrate three different methods to normalize the position of contours in space. (**a**) Initial contours. Each black dot corresponds to the starting point of the parameterization and has been selected as the point of the contour with largest vertical coordinate. Each orange point corresponds to the center of mass of the contour. Each purple point corresponds to the center of gravity of the enclosed area. One can see that the length of the peduncle influences the position of the center of mass of the contour, but it has little effect on the position of the center of gravity of the enclosed area. (**b**) Each contour is translated in such a way that the starting point of the parameterization of the contour is at the origin. (**c**) Each contour is translated in such a way that the center of mass of the contour is at the origin. (**d**) Each contour is translated in such a way that the center of gravity of the enclosed area is at the origin.

In Figure 10b, the contours of the Acer leaves are centered so that the first point of their parameterization coincides with the origin. The corresponding clustering of the training set after this normalization can be visualized in Figure 11a. The Dunn index decreased by this normalization process from 0.0587 (see Section 3.1.5) to 0.0482.

In Figure 10c, the contours are centered so that the center of mass of the contours is at the origin. The corresponding clustering of the training set after this normalization can be visualized in Figure 11b. The Dunn index slightly decreases after this normalization process from 0.0587 to 0.0573.

In Figure 10d, the contours are centered so that the center of gravity of the enclosed area is at the origin. One can see that the length of the peduncle influences the position of the center of mass of the contour, but not the position of the center of gravity of the enclosed area, leading to a better alignment of the contours. The corresponding clustering of the training set after this normalization can be visualized in Figure 11c. The Dunn index increases after this normalization process from 0.0587 to 0.0702. Therefore, in what follows, the center of the enclosed area is used to center contours.

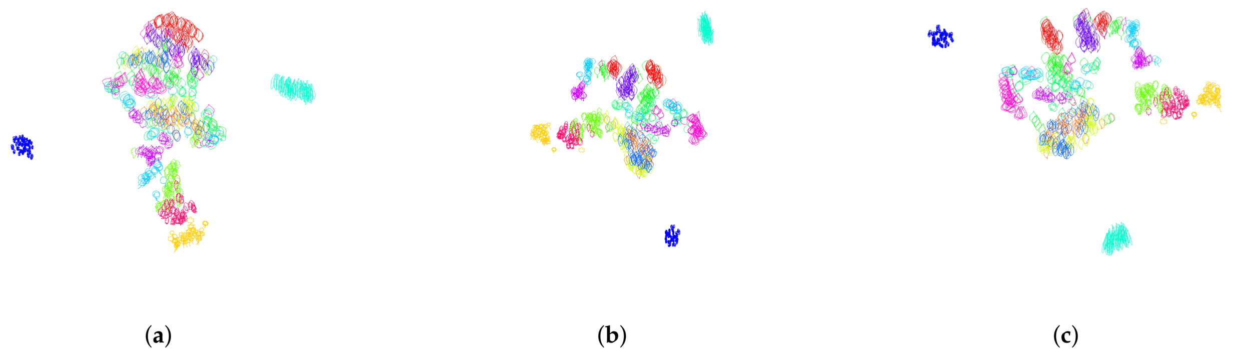

(**a**) (**b**) (**c**)

Figure 11. Normalization of the position in space. Two-dimensional representation of the distance distribution along the dataset using `tsne` algorithm. (**a**) After centering the curves to the same starting point, the Dunn index decreases from 0.0587 to 0.0482. (**b**) After centering the curves to have the center of mass at the origin, the Dunn index decreases from 0.0587 to 0.0573. (**c**) After centering the curves to have the center of gravity of enclosed area at the origin, the Dunn index increases from 0.0587 to 0.0702.

3.1.6. Standardizing the Orientation in Space

The leaves in the dataset we are considering have different orientations in space and need to be rotated in a consistent way to eliminate the orientation variability. We have tested two normalization procedures to align the orientations through the dataset.

(a) **Axes of the approximating ellipse aligned:** Each contour is rotated so that the ellipse that best approximates the contour has its minor axis along the horizontal axis, and its major axis vertically. We did not encounter contours with equal minor and major axes.
(b) **Segment that joins the tip of the leaf to the center of the enclosed area is placed vertically:** Each contour is rotated so as to position the center of the enclosed area vertically below the highest point of the contour.

The first normalization method does not lead to good results because of the presence of leaves with a peduncle and leaves without a peduncle in the same class. As can be seen in Figure 12b on the example of Acer leaves, the alignment of the major and minor axis of the approximating ellipse leads to inconsistent orientation of the leaf without peduncle with respect to the other leaves. After this normalization procedure, the Dunn index decreases from 0.0702 to 0.0268. The corresponding clustering can be visualized in Figure 13a.

The second normalization method gives better results (see Figure 12c), although the Dunn index decreases slightly from 0.0702 to 0.0636. We will choose this second normalization method, in order to normalize the orientation variability and obtain consistent classification results. The corresponding clustering can be visualized in Figure 13b.

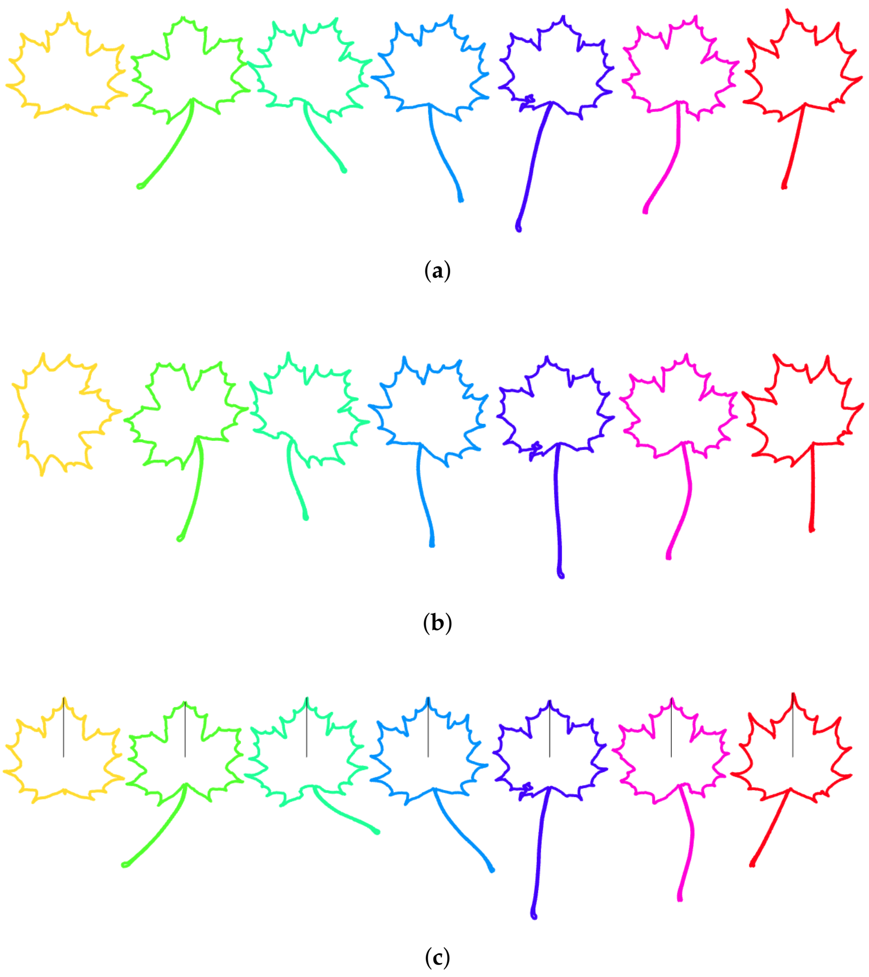


Figure 12. Normalization of the orientation variability. Seven Acer leaves are used to illustrate different methods to normalize the orientation in space in a consistent manner through the dataset. (**a**) Initial contours. (**b**) Each contour is rotated in such a way that the approximating ellipse has its minor axis along the horizontal axis, and its major axis vertically. Note that the first Acer leaf has an inconsistent orientation with respect to the other leaves with peduncles. (**c**) Each contour is rotated in such a way that the segment (in black) joining the center of gravity to the first point is vertical.

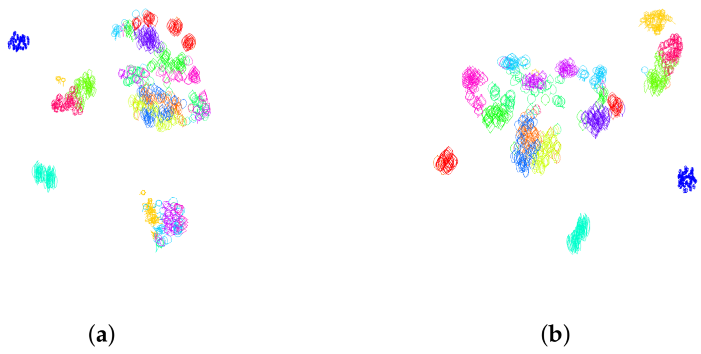


Figure 13. Normalization of the orientation in space. Two-dimensional representation of the distance distribution along the dataset using `tsne` algorithm. (**a**) After rotation of the curves to have their approximating ellipse aligned with the axis, the Dunn index decreases from 0.0702 to 0.0294. (**b**) After rotation of the curves so that the segment joining the center of gravity of the enclosed area and the tip of the leave is vertical, the Dunn index decreases slightly from 0.0702 to 0.0658.

3.1.7. Resulting Normalization over Finite-Dimensional Shape-Preserving Groups

The resulting normalization over the finite-dimensional shape-preserving group consisting of scalings, translations, rotations in space and rotations in parameter space is illustrated for different classes of leaves in Figure 14. Let us summarize here the normalization steps that were selected:

- Counterclockwise travel along the curves (Section 3.1.2).
- Starting point at the tip of the leaves (Section 3.1.3).
- Unit-length curves (Section 3.1.4).
- Center of gravity of the enclosed area at the origin (Section 3.1.5).
- Segment joining the tip of the leaf to the center of gravity vertical (Section 3.1.6)

The remaining shape-preserving group is infinite-dimensional and consists of orientation-preserving reparameterizations fixing the starting (and ending) point. Mathematically, this group corresponds to the following subgroup of $\mathrm{Diff}^+(\mathbb{S}^1)$:

$$\mathrm{Diff}_0^+(\mathbb{S}^1) = \{\Phi \in \mathrm{Diff}^+(\mathbb{S}^1), \Phi(0) = 0\}.$$

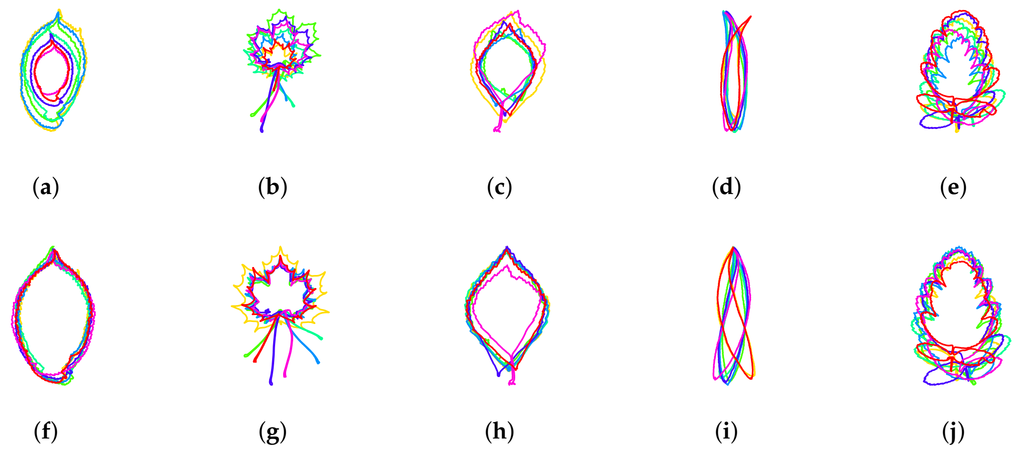


Figure 14. Resulting normalization over the group of scalings, translations, rotations in space, and rotation in parameter space. Several leaves of the same class are depicted before normalization (upper row (**a**–**e**)) and after normalization (lower row (**f**–**j**)).

3.2. A New 2-Parameter Family of Canonical Parameterizations

3.2.1. Clock Parameterization of Jordan Curves

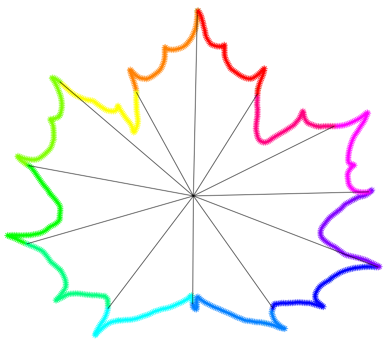
In this section, we introduce a new canonical parameterization of simple plane curves, called the clock parameterization. We will make use of the analogy with a traditional clock to explain how this parameterization is constructed. Suppose that we have $720 = 12 \times 60$ points to place along the contour of the Acer leaf depicted in Figure 15a. If we place 720 points uniformly along the contour and cut the enclosed area as a pizza from its center of gravity to the points corresponding to a multiple of 60, then we obtain 12 pieces of different angles. This is illustrated in Figure 15a by a color change with every 60 points. In contrast, the clock parameterization automatically places each point numbered by a multiple of 60 in such a way that the corresponding angle is precisely 360/12 degrees, hence at the positions of the hours on a traditional clock (see Figure 15b). To place these 12 keypoints at the hours positions, we compute the angle between the vertical line and the segment connecting the center of gravity to a point traveling along the contour at constant speed. The graph of the angle function for the Acer leaf is illustrated in Figure 15c. It allows us to detect the constant speed parameter of the first point reaching an angle multiple of 360/12 degrees. In Figure 15c, the horizontal lines are spaced every 360/12 degrees and hit the angle function graph precisely at these constant speed parameters. Between two consecutive points that have these particular constant speed parameters, we distribute exactly 60 points uniformly along the portion of the curve between them. The resulting reparameterization of the Acer leaf is such that each colored portion of the curve describes the same angle with respect to the center of gravity and contains exactly the same number of points. In Figure 15d–f, the same procedure is applied to the more challenging shape of Sorbus leaf. Note the difference in the density of points on the light blue portion and on the dark blue portion of the curve. This is due to the structure of compound of Sorbus leaves, which are made up of multiple leaflets arranged along a central stalk.

In the previous procedure, the number of subdivisions of 360 was set to 12. For each choice of the number n of subdivisions, we obtain a different reparameterization procedure for Jordan curves. In Figure 16, we illustrate how the resulting parameterization of a curve depends on the number of subdivisions n. In this case, we distribute 1000 points along the contour of an Acer leaf, this time with a peduncle. The first row in Figure 16 corresponds to a parameterization with constant speed. From left to right, we use 20, 50, and 100 subdivisions to color the curve. The corresponding clock parameterizations are depicted in Figure 16d–f, respectively. The graph of the angle function with equally spaced horizontal lines is depicted in Figure 16g–i for 20, 50, and 100 subdivisions of 360 degrees. In contrast to the constant speed parameterization, for the clock parameterization, the density of points along the peduncle decreases with the number of subdivisions. Indeed, while the number of subdivisions increases, the angle formed by each colored piece of curve decreases. Since, on each colored piece of curve, we distribute the same number of points, the density of points on the piece containing the long peduncle decreases drastically.

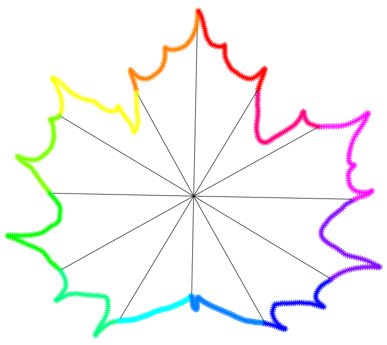
Remark 5. *The clock parameterization is well defined as long as the center of gravity is within the interior of the contour. In practice, this was generally the case, but we encountered some leaves with a center of gravity outside the interior. In these cases, the center of gravity was replaced by a reference point nearby but located inside the leaf. There are many possible automatic procedures for doing so:*

- *After computing the closest point of the contour to the center of gravity, the reference point is initialized at the center of gravity and moved in the direction of this closest point until its index with respect to the contour increases from* 0 *to* 1.
- *The reference point is initialized at the center of gravity and moved in the direction of the tip of the leaf until its index with respect to the contour increases from* 0 *to* 1.
- *After computing the Delaunay triangulation for the contour and subsequently creating the Voronoi diagram, the leaf is translated so that the closest Voronoi vertex is at the origin.*
- *After computing the closest point of the contour to the center of gravity, we consider triangles with one vertex at the closest point and two other vertices on the contour, and compute their centroids. We choose as a reference point a centroid near the center of gravity that has the property to be inside the shape, and we perform a translation so that this reference point is at the origin.*

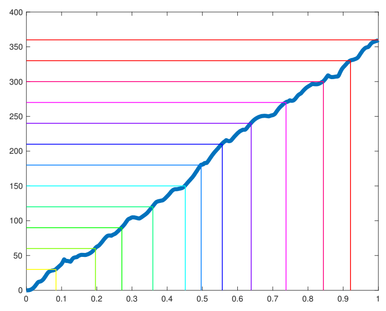
The first solution has the advantage of generalizing to datasets without distinguished point along the contour (which could take the role of the tip of the leaf), the second solution has the advantage of being compatible with the rotation alignment performed in Section 3.1.6. However, these two solutions are dependent on the step size of the displacement, which is an extra data-dependent parameter. In contrast, the last two translation procedures do not require learning an extra hyperparameter and are therefore preferred.



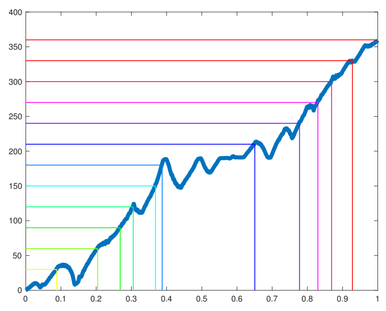
(**a**)



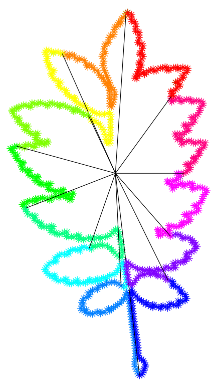
(**b**)



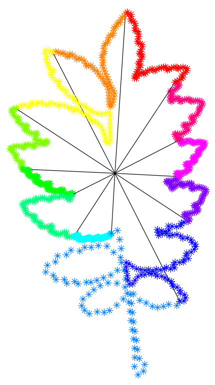
(**c**)



(**d**)

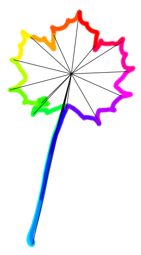


(**e**)

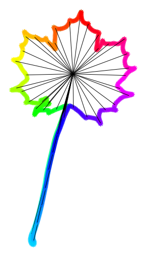


(**f**)

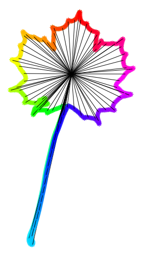
Figure 15. Clock parameterization of Jordan curves. (**a**) A leaf of Acer is sampled uniformly with 720 points. Every 60 points the color is changed. The angle between the first point of a colored portion, the center of gravity, and the last point of the same colored portion is illustrated. These angles are not equal. (**b**) 12 points are placed successively along the contour to form an angle of 360/12 with the center of gravity and the previous such point. Now the angles formed by each colored portion of the curve are the same. On each colored portion, 60 points are distributed uniformly. (**c**) The graph of the angle function of the Acer leaf is represented in the constant speed parameterization. (**d**) The graph of the angle function of the Sorbus leaf is represented in the constant speed parameterization. (**e**) A leaf of Sorbus is parameterized with constant speed and sampled with 720 points. (**f**) 12 keypoints are detected to form equal angles to the center of gravity and the portions of curve between them are resampled with 60 points.



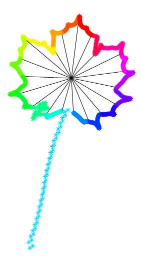
(**a**)



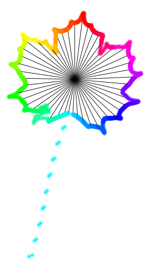
(**b**)



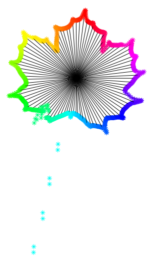
(**c**)



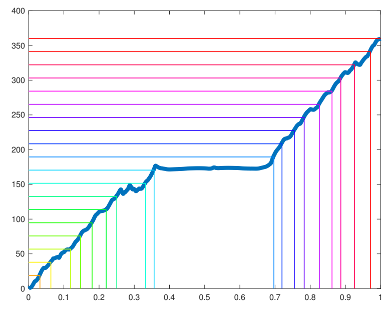
(**d**)



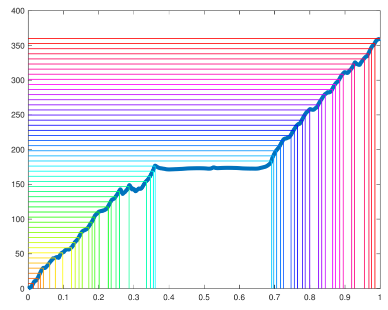
(**e**)



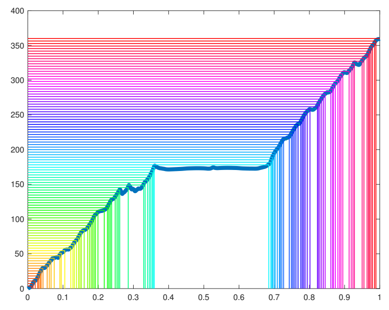
(**f**)



(**g**)



(**h**)



(**i**)

Figure 16. Dependence of the clock parameterization with respect to the number of subdivisions. First row: An Acer leaf with peduncle is parameterized at constant speed and sampled with 1000 points. The color is changed every (**a**) 20 points, (**b**) 50 points, (**c**) 100 points. Second row: An Acer leaf with peduncle is parameterized with clock parameterization according to (**d**) 20 subdivisions, (**e**) 50 subdivisions, (**f**) 100 subdivisions. The density of points along the peduncle decreases drastically with the number of subdivisions. Last row: The graph of the corresponding angle function is illustrated with (**g**) 20 equally spaces horizontal lines, (**h**) 50 equally spaces horizontal lines, (**i**) 100 equally spaces horizontal lines.

3.2.2. Curvature-Weighted Clock Parameterizations of Jordan Curves

In this section, we introduce a 2-parameter family of canonical parameterizations obtained by combining curvature-weighted parameterizations with parameter λ (see Section 2.4) and clock parameterizations with n subdivisions (Section 3.2.1). More precisely, each contour is first decomposed into n subdivisions forming n equal angles to the center of gravity. Secondly, each portion of the curve is reparameterized according to a curvature-weighted parameterization with parameter λ as in Section 2.4, Equation (8).

A sampling of a curve with N points according to the curvature-weighted clock parameterization with parameters (λ, n) goes as follows: the curve is subdivided into n portions forming equal angles at the center of gravity; N/n points are distributed on each portion according to the curvature-weighted parameterization with parameter $\lambda > 0$;

see Section 2.4 (for low parameter λ, the density of points decreases on flat parts of the contour and increases on curved parts, while for large parameter λ, the curvature-weighted parameterization tends to the constant speed parameterization).

In Figure 17, an Acer leaf (with peduncle) is resampled with 1000 points according to curvature-weighted clock parameterizations with different parameters (λ, n). The first column corresponds to $\lambda = 0.3$, the second to $\lambda = 1$, and the third column to $\lambda = 2$. At the same time, the first row corresponds to $n = 12$, the second row to $n = 24$, and the last row to $n = 36$. One can observe that the density of points along the peduncle decreases when λ decreases and/or the number of subdivisions increases.

3.3. Geometric Learning of Canonical Parameterizations

In this section, we consider the 2-parameter family of curvature-weighted clock parameterizations with parameters (λ, n) introduced in Section 3.2.2, as well as the corresponding sections $s_{\lambda,n} : \mathcal{P}/\mathcal{G} \to \mathcal{P}$ of the fiber bundle consisting of embedded closed curves. The aim is to optimize clustering based on the distance between shapes defined in (12), which depends on the section $s_{\lambda,n}$ chosen.

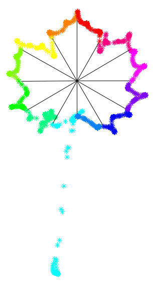
A table containing the Dunn index for various values of parameters λ (weighting the parameterization by curvature) and n (number of segments in the clock parameterization) is given in Table 1. The Dunn index values were averaged over a 30-fold cross-validation. For this experiment, we used the training set consisting of 15 classes of leaves with 50 leaves each. We can see in Table 1 that the largest Dunn index (corresponding to the best clustering for this metric) is obtained for $n = 3$ subsections along the contours of the leaves, and $\lambda = 2000$, which corresponds to a curvature-weighted parameterization on each of the three portions of the curve. In Figure 1a, we visualize the pair of curves that maximizes the intraclass distance, as well as the pair of curves that minimizes the interclass distance. These two pairs of curves are responsible for the value of the Dunn index. We illustrate the segment in $L^2(\mathbb{S}^1, \mathbb{R}^2)$,which connects the leaves parameterized by the optimal parameterization ($n = 3$, $\lambda = 2000$). In the left picture of Figure 1a, we can see that the south portion of the contour of the leaf without peduncle deforms to create a peduncle. In comparison in the right picture of Figure 1a, two leaves from two different classes seem perfectly aligned. This illustrates the challenges of clustering or classifying this dataset of leaves, where very different shapes belong to the same class and similar shapes belong to different classes. Figure 18 illustrates a 2-dimensional representation of the distance distribution along the dataset using the tsne algorithm before any normalization and after normalization using the optimal parameterization for the Dunn index. One can see that the classes are significantly better clustered after normalization.

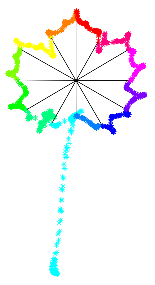
Table 1. Dunn index for various clustering of the Swedish dataset based on clock parameterization with parameters λ (weighting the parameterization by curvature) and n (number of segments in the clock parameterization), cross-validated over 30 partitions of the dataset into training set and testing set. Each column corresponds to the parameter λ given at the top of the column; each row corresponds to the values n given on the left. A larger Dunn index corresponds to a better clustering. The frame highlights the highest Dunn index.

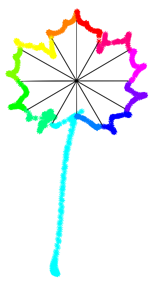
$\lambda =$	**0.5**	**1**	**2**	**5**	**10**	**100**	**1000**	**2000**	$+\infty$
n = 0	0.0535	0.0535	0.0536	0.0539	0.0543	0.0584	0.0670	0.0706	0.0774
n = 2	0.0382	0.0382	0.0383	0.0384	0.0387	0.0432	0.0522	0.0560	0.0639
n= 3	0.0768	0.0768	0.0768	0.0769	0.0770	0.0785	0.0821	$\boxed{0.0827}$	0.0826
n = 4	0.0656	0.0657	0.0657	0.0658	0.0659	0.0677	0.0723	0.0735	0.0766
n = 5	0.0711	0.0711	0.0711	0.0711	0.0712	0.0720	0.0748	0.0759	0.0780

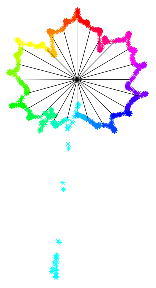
Table 1. *Cont.*

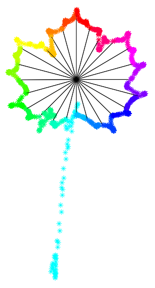
$\lambda =$	**0.5**	**1**	**2**	**5**	**10**	**100**	**1000**	**2000**	$+\infty$
n = 7	0.0703	0.0703	0.0703	0.0703	0.0704	0.0714	0.0755	0.0772	0.0811
n = 9	0.0698	0.0698	0.0698	0.0699	0.0699	0.0707	0.0739	0.0755	0.0798
n = 10	0.0672	0.0672	0.0672	0.0672	0.0673	0.0680	0.0712	0.0727	0.0771
n = 20	0.0642	0.0642	0.0642	0.0642	0.0643	0.0647	0.0667	0.0677	0.0710

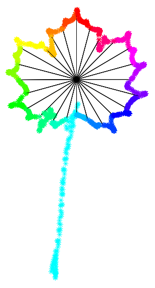

(**a**)

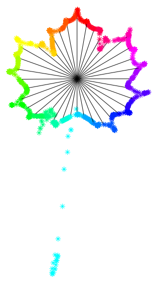

(**b**)

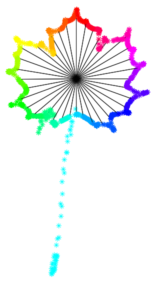

(**c**)


(**d**)


(**e**)


(**f**)


(**g**)


(**h**)

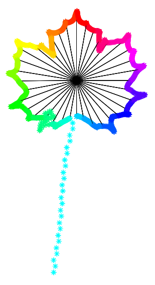
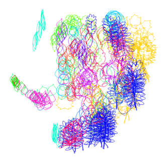
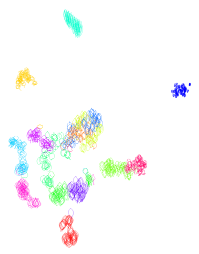

(**i**)

Figure 17. Curvature-weighted clock parameterizations with different parameters. An Acer leaf (with peduncle) is resampled with 1000 points according to curvature-weighted clock parameterizations with different parameters (λ, n). The first column (a,d,g) corresponds to $\lambda = 0.3$, the second (b,e,h) to $\lambda = 1$, and the third column (c,f,i) to $\lambda = 2$. At the same time, the first row (a,b,c) corresponds to $n = 12$, the second row (d,e,f) to $n = 24$, and the last row (g,h,i) to $n = 36$. One can observe that the density of points along the peduncle decreases when λ decrease and/or the number of subdivisions increases.


(**a**)


(**b**)

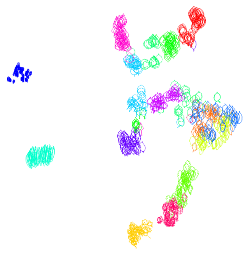

(**c**)

Figure 18. Two-dimensional representation of the distance distribution along the dataset using `tsne` algorithm (**a**) before any normalization; (**b**) after normalization over finite-dimensional shape-preserving groups, as explained in Section 3.1, and normalization over the infinite-dimensional group of orientation-preserving reparameterization by using the optimal parameterization for the Dunn index obtained as clock parameterization (see Section 3.2.1) with $n = 3$ subdivisions and curvature-weighted parameterization ($\lambda = 2000$) on all portions of the subdivisions; (**c**) after normalization over finite-dimensional shape-preserving groups, as explained in Section 3.1, and normalization over the infinite-dimensional group of orientation-preserving reparameterization by using the optimal parameterization for the Davies Bouldin index obtained as clock parameterization (see Section 3.2.1) with $n = 5$ subdivisions and arc-length parameterization ($\lambda = +\infty$) on all portions of the subdivisions.

4. Classification Results

4.1. Testing on the Dataset of the Swedish Leaves

4.1.1. Clustering Evaluation Using Another Cluster Validation Index

Recall that the Swedish leaves dataset was divided into a training set, containing 50 contours from each class, and a testing set containing the remaining 25 contours per class. The standardization steps performed in Sections 3.1 and 3.3 were necessary to define a distance between contours in the plane that is independent of their position and orientation in space, their scaling, and their parameterization (starting point, traveling direction, and velocity). The best standardization procedures were selected to optimize the clustering of the classes of the labeled training set. The quality of the clustering obtained can be measured by computing a cluster validity index, like the Dunn index or the Davies Bouldin index, as explained in Section 2.9, for the distance defined in (12). Tables 1 and 2 contain, respectively, the values of the Dunn index and Davies Bouldin index, cross-validated over 30 partitions of the dataset into training set and testing set. As mentioned in Section 2.9.2, the averaging of the distances to a centroid over all elements of a class allows the Davies Bouldin index to be more stable than the Dunn index in the presence of outliers. We therefore select the Davies Bouldin index for classification task.

Table 2. Davies Bouldin index for various clustering of the Swedish dataset based on clock parameterization with parameters λ (weighting the parameterization by curvature) and n (number of segments in the clock parameterization), cross-validated over 30 partitions of the dataset into training set and testing set. Each column corresponds to the parameter λ given at the top of the column, each row corresponds to the values n given on the left. A lower Davies Bouldin index corresponds to a better clustering. The lowest Davies-Bouldin index is highlighted with a frame.

$\lambda =$	**0.5**	**1**	**2**	**5**	**10**	**100**	**1000**	**2000**	$+\infty$
n = 0	2.7540	2.7524	2.7491	2.7394	2.7238	2.5351	2.2157	2.1575	2.0848
n = 2	2.8729	2.8725	2.8717	2.8694	2.8657	2.8121	2.6816	2.6337	2.5677
n= 3	2.1682	2.1678	2.1671	2.1649	2.1614	2.1129	1.9709	1.9301	1.8636
n = 4	2.4506	2.4505	2.4502	2.4495	2.4483	2.4317	2.3666	2.3343	2.2473
n = 5	2.0851	2.0848	2.0843	2.0827	2.0801	2.0446	1.9508	1.9201	$\boxed{1.8574}$
n = 7	2.0892	2.0890	2.0885	2.0870	2.0847	2.0539	1.9715	1.9414	1.8707
n = 9	2.1272	2.1270	2.1265	2.1251	2.1229	2.0959	2.0197	1.9886	1.9101
n = 10	2.2697	2.2695	2.2690	2.2677	2.2656	2.2397	2.1573	2.1207	2.0176
n = 20	2.2241	2.2239	2.2236	2.2225	2.2207	2.2015	2.1493	2.1258	2.0457

4.1.2. Improvement of the Classification Results After Normalization

In the present section, we illustrate how standardization procedures affect classification performance of samples from the testing set. We have used the following:

- Logistic Regression with L^2-norm regularization;
- Random Forest with 400 trees;
- Support Vector Machine (SVM) with a non-linear Radial Basis Function (RBF) kernel;
- k-Nearest Neighbors (KNN) with $k = 5$ nearest neighbors.

Complete parameter specifications are available in the code. The Support Vector Machine with an RBF kernel achieved the highest performance, with $C = 25$ (the regularization parameter controlling the trade-off between margin size and classification error) and $\gamma = 1.5$ (the kernel coefficient that determines the influence radius of individual training samples). To assess classification performance, we used accuracy as an evaluation metric, defined as follows:

$$\text{Accuracy} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}\{\, Y_i = \hat{Y}_i \,\}, \tag{22}$$

where Y_i is the true label of the i-th element in the testing set, and $\hat{Y}_i$ is the corresponding predicted label.

The results are displayed in Table 3. We see that all four classification algorithms perform significantly better after normalization, i.e., when a representative is chosen in each orbit of the shape-preserving groups in a consistent way. In particular, we observe an increase of 25,85% of correct classifications for the KNN algorithm between the first line of Table 3 (no normalization performed) and the last line (all finite and infinite-dimensional shape-preserving groups taken into account using optimized sections). This illustrates that including standardization of the representative of each orbits under shape-preserving groups in the pre-processing step improves classification results irrespective of the classification algorithms.

Table 3. Classification results on the dataset of Swedish leaves in terms of average accuracy across pre-processing stages, with different classifiers and over a 30-fold cross-validation. The Dunn index and the Davies Bouldin indices are also reported, cross-validated over 30 partitions of the dataset into the training set, as well as testing set and reported in the first column. We reparametrized the curve with 1000 points and, for the clock parametrization, we set the number of subsections to $n = 5$ and $\lambda = \infty$, which corresponds to arc-length parameterization on each portion of the curves and minimizes the Davies Bouldin index (Table 2). For comparison, the last line corresponds to the curvature-weighted clock parameterization with a number of subsections equal to $n = 5$ and the weight of the curvature equal to $\lambda = 2000$. The best result in term of accuracy is highlighted with a frame.

Pre-Processing Steps	Dunn	DB	Logistic	RF	SVM	KNN
No normalization Section 3.1.1	0.0325	3.7981	0.7273	0.7489	0.8310	0.6713
Std the travel direction Section 3.1.2	0.0314	3.8766	0.7743	0.7483	0.8411	0.6800
Std the starting point Section 3.1.3	0.0367	3.2387	0.8604	0.7708	0.8829	0.6826
Std the scale variability Section 3.1.4	0.0566	2.4960	0.9233	0.8754	0.9449	0.8913
Std the position Section 3.1.5	0.0667	2.4239	0.9132	0.8902	0.9364	0.8892
Std the orientation Section 3.1.6	0.0774	2.0847	0.9192	0.8990	0.9496	0.9228
Clock parametrization Section 3.2	0.0759	1.9201	0.9395	0.9063	$\boxed{0.9602}$	0.9332
Curvature-weighted Section 3.2.2	0.0780	1.8574	0.9357	0.8992	0.9562	0.9284

4.1.3. Comparison with State-of-the-Art Classification Results

Compared to the state-of-the art classification results displayed in Table 4 for classical machine learning algorithms (without Neural Networks) and in Table 5 for Neural Network-based algorithms, we observe that, with an optimization over only 2 parameters, our algorithm reaches 0.9602 accuracy (96.02% of correct classifications) with SVM on the dataset of Swedish leaves, whereas the state-of-the art model VGG-16 needs 138 million parameters to reach perfect accuracy (100% correct classifications) on the same dataset. This illustrates that algorithms using fewer but well-chosen parameters can compete with brute force algorithms using millions of parameters. We hope that this can motivate the investigation of more sustainable solutions for classifications tasks, as well as meaningful parameter optimization. Moreover, as shown in Section 4.1.2, our proposed method could be a beneficial pre-precessing step before applying fine-tuned algorithms since it leads to an optimal point-to-point correspondence across the dataset. Contrary to the other classification methods present in Tables 4 and 5, the standardization procedure that we propose allows us to interpolate between elements in the dataset (as in Figure 1). It could be interesting to test whether the methods of [29,30] improve if we apply our standardization method as a pre-precessing step.

Table 4. Comparison of classification results on the Swedish leaves dataset using different classical machine learning methods (no Neural Networks) taken from [30,31]. We see that with an optimization over only two parameters, our method is comparable to the state-of-the art classical machine learning algorithms. Moreover, it could serve as a pre-processing step for more complex algorithms.

Methods	Classification Rate (%)
Multi-features fusion [32]	77.24
MSRA	91.87
MARCH	93.20
MDM [33]	93.60
IDSC [34]	94.13
MCC [35]	94.75
SPTC [34]	95.33
TAR [29]	95.97

Table 4. *Cont.*

Methods	Classification Rate (%)
OURS	96.02
MSSD [30]	96.85

Table 5. Comparison of classification results on the Swedish leaves dataset using Neural Networks methods; table taken from [36]. Note that the state-of-the art model VGG-16 needs 138 million parameters to reach perfect accuracy, whereas our method achieves similar accuracy with an optimization over only two geometrically explainable parameters.

Methods	Classification Rate (%)
AlexNet	99.70
GoogleLeNet	99.39
VGG16	100.00
ResNet18	99.39
ResNet50	99.39
ResNet101	99.70

4.2. Testing on Flavia Dataset

To further assess the effectiveness of the proposed pipeline, we evaluated it on a second dataset. We use the Flavia dataset, which contains 1,907 leaf images belonging to 32 classes and is available at https://www.kaggle.com/datasets/gauravneupane/flavia-dataset (accessed on 13 November 2025). Figure 19 illustrates the different types of leaves present in this dataset. Achieving high classification accuracy on this dataset is more challenging due to the larger number of classes and the extremely similar shapes among many of them.

While applying our pipeline on the Flavia dataset, we were surprised to see that normalization of the orientation in space deteriorated the clustering drastically. After taking a closer look at the dataset, we discovered that the original Flavia dataset contains an alignment bias. Indeed, for some classes, all the leaves are oriented in a class-dependent direction in space. In Figure 20, the angle distribution of the leaves in each class is depicted. As we can see, for instance on classes 15, 19, and 32, the distribution is very concentrated around a mean orientation, and this mean orientation differs from class to class. This is probably due to the way the dataset was collected. Consequently, the orientation in space can be used to determine the belonging of a sample to a given class, which is unfortunate. In order to test our algorithm on an unbiased dataset, we applied random rotations to the samples of the dataset. The unbiased dataset is available at the following links: https://github.com/GiLonga/Geometric-Learning (accessed on 19 October 2025) and https://github.com/ioanaciuclea/geometric-learning-notebook (accessed on 19 October 2025).

Starting from the unbiased Flavia dataset, we can see in Table 6 that our normalization procedure improves the classification performance of all the algorithms tested. Since this dataset contains very similar shapes but with different scales, scale normalization was not performed because the scale contains valuable information in order to distinguish between classes. In order to optimize over the parameterization, we have used the Davies Bouldin index, which is more stable than the Dunn index in the presence of outliers. Table 7 contains the Davies Bouldin index for different values of the parameters, 30-fold cross-validated.

Figure 19. Sample of the Flavia leaf dataset. Picture taken from [31].

As a concluding remark, let us note that the optimal normalization procedure and the optimal parameters (n, λ) depend on the dataset and the selected cluster validity index. However, for a given dataset, the optimal parameterizations $s_{n,\lambda}$ for various cluster validity indices seem to be close in the space of sections over the sample points. Indeed, as can be seen in Figure 1 for the Swedish leaves dataset, the optimal section $s_{n,\lambda}$ for the Dunn index is different from the optimal section for the Davies Bouldin index (the former is associated with $(n = 3, \lambda = 2000)$ whereas the latter with $(n = 5, \lambda = +\infty)$), but the corresponding contours parameterizations look very similar. This can be explained by the fact that the various cluster validity indices are linked to each other [25] and are continuous functions of the distances between samples, while these distances depend continuously on the section $s_{n,\lambda}$. From this perspective, it becomes clear that the standardization procedure improves classification performance, as the optimal distance function better reflects the intrinsic geometry of the dataset.

Table 6. Classification results in terms of average accuracy across pre-processing stages, with different classifiers and over a 30-fold cross-validation. The Dunn and Davies Bouldin indices at each step are also reported. We reparametrized the curve with 1000 points and, for the clock parametrization, we set the number of subsections to $n = 3$. For the curvature-weighted clock parameterization (last line) we set $n = 3$ and $\lambda = 1000$ which corresponds to the curvature-weighted clock parameterization that minimizes the Davies Bouldin index (Table 7). The best result in term of accuracy is highlighted with a frame.

Pre-processing Steps	**Dunn**	**DB**	**Logistic**	**RF**	**SVM**	**KNN**
No normalization Section 3.1.1	0.0137	37.8766	0.0317	0.0392	0.0319	0.0331
Std the travel direction Section 3.1.2	0.0096	44.8592	0.1333	0.3188	0.6392	0.2192
Std the position Section 3.1.5	0.0103	38.8019	0.0905	0.5311	0.6916	0.3586
Std the orientation Section 3.1.6	0.0070	48.3259	0.1075	0.7002	0.7100	0.5649
Std the starting point Section 3.1.3	0.0132	20.0387	0.4514	0.6464	0.6782	0.5798
Clock parametrization Section 3.2	0.0188	4.7575	0.6959	0.7361	0.7565	0.6756
Curvature-weighted Section 3.2.2	0.0203	4.5763	0.6531	0.7384	$\boxed{0.7679}$	0.6759

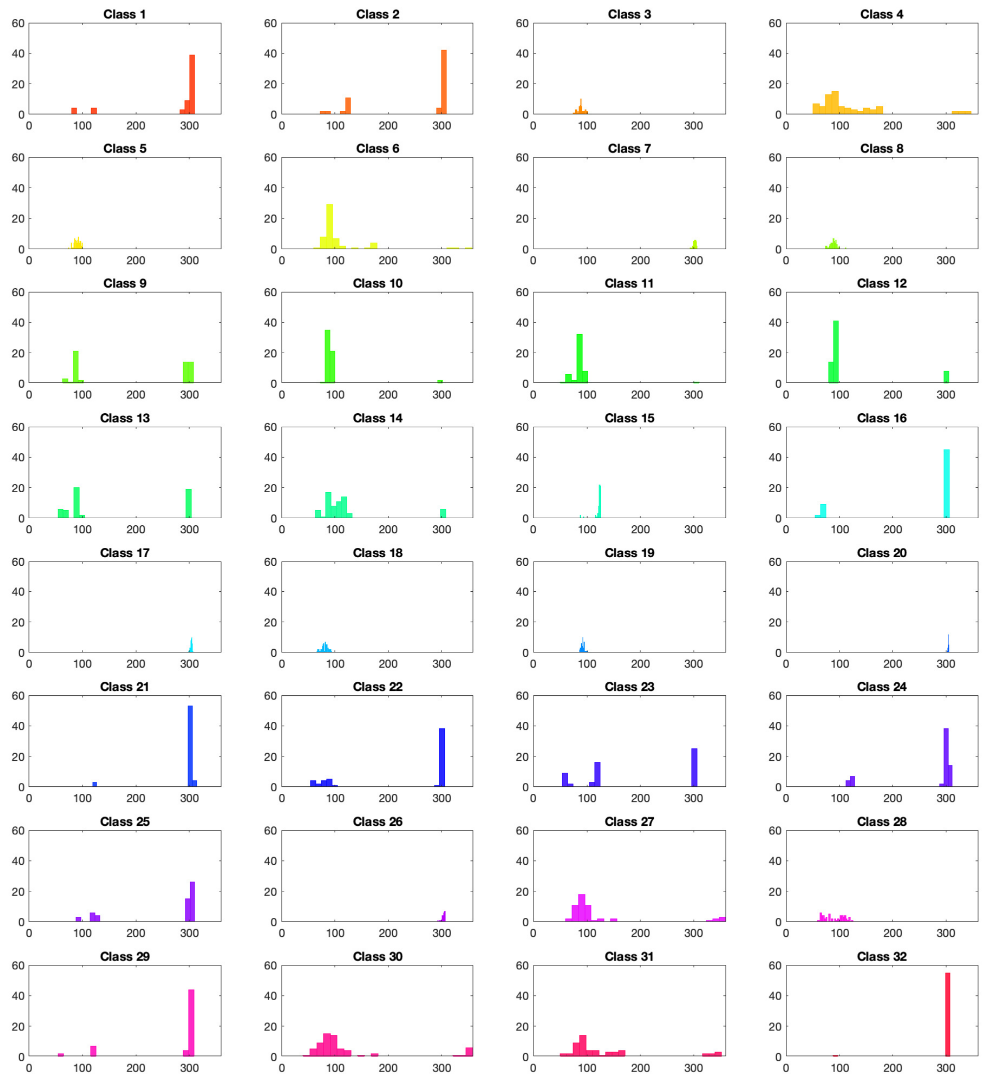


Figure 20. Distribution of orientation angles in each class of the Flavia dataset of leaves. We see that for some classes, all the leaves are oriented in the same direction in space, with a mean orientation differing from class to class, like for the classes 15, 19, and 32, for example. This implies that the dataset is biased with respect to orientation.

Table 7. Davies Bouldin index for various clustering of the Flavia dataset based on clock parameterization with parameters λ (weighting the parameterization by curvature) and n (number of segments in the clock parameterization), over a 30-fold cross-validation. Each column corresponds to the parameter λ given at the top of the column, each row corresponds to the values n given on the left. A lower Davies Bouldin index corresponds to a better clustering. The lowest Davies-Bouldin index is highlighted with a frame.

$\lambda =$	**0.5**	**1**	**2**	**5**	**10**	**100**	**1000**	**2000**	$+\infty$
n = 0	6.5694	6.5654	6.5576	6.5347	6.4983	6.0650	5.2734	5.1864	5.1206
n = 2	6.0932	6.0911	6.0869	6.0748	6.0559	5.8459	5.5693	5.5721	5.6204
n= 3	5.6204	4.8815	4.8790	4.8741	4.8662	4.7501	$\boxed{4.5763}$	4.6257	4.7575
n = 4	4.8376	4.8375	4.8373	4.8367	4.8360	4.8297	4.8644	4.9224	5.0768
n = 5	4.7432	4.7425	4.7410	4.7371	4.7308	4.6594	4.6479	4.7174	4.8260

Table 7. *Cont.*

$\lambda =$	0.5	1	2	5	10	100	1000	2000	$+\infty$
n = 7	4.7803	4.7801	4.7797	4.7788	4.7773	4.7739	4.9330	5.0198	5.0973
n = 9	4.7388	4.7388	4.7388	4.7388	4.7389	4.7672	4.9784	5.0584	5.1095
n = 10	4.8379	4.8377	4.8372	4.8359	4.8343	4.8250	4.9542	5.0271	5.0772
n = 20	5.2147	5.2145	5.2141	5.2131	5.2117	5.2127	5.2593	5.2638	5.2056

5. Discussion

In this paper, a supervised classification task is considered on contours in the plane. We have shown that classification performance is significantly improved when shape-preserving groups are taken into account and the dataset is appropriately normalized. In order to design classification algorithms that are independent of the action of shape-preserving groups and hence make sense on the quotient space, we propose to use customized sections of the corresponding fiber bundle for standardization or normalization along the dataset. This amounts to choosing a representant in each orbit of the shape-preserving group in a standardized way. We have introduced a distance on the manifold of contours in the plane based on a simple L^2 distance function and the choice of a section. We have presented multiple normalization procedures for the finite-dimensional groups of translations, rotations, and scalings, as well as for the infinite-dimensional group of reparameterizations (which act on the starting point and the velocity along the contours). In particular, for the latter group, we have introduced a new two-parameter family of canonical parameterizations of curves, called curvature-weighted clock parameterizations, that may be of interest for other applications. By optimizing a cluster validation index, like the Dunn or Davies Bouldin indices, of the resulting clustering in the training set, we are able to achieve high classification performance on the testing set, without the use of any neural network and by optimizing over only two parameters. This method can serve as a beneficial pre-processing step for more complex algorithms since it gives optimal point-to-point correspondances, solving a registration task. It can be easily generalized to curves in a Euclidean space of any dimension, and we will explore this in a future work. We hope that this work can serve as a guide for the design of more sustainable AI algorithms on manifolds of curves.

Author Contributions: Conceptualization, A.B.T.; methodology, A.B.T.; software, G.L.; validation, G.L. and I.C.; formal analysis, G.L., I.C. and A.B.T.; investigation, G.L.; data curation, G.L. and I.C.; writing—original draft preparation, A.B.T.; writing—review and editing, G.L., I.C. and A.B.T.; visualization, I.C.; supervision, A.B.T.; project administration, A.B.T.; funding acquisition, A.B.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Austrian Science Fund (FWF), AI Austria call, grant number PAT1179524. I.C. was financially supported by the START Grant of the West University of Timişoara during the writing of this paper.

Data Availability Statement: In this paper, we analyzed the dataset of Swedish leaves from the Linköping University, which is publicly available and can be freely downloaded from the following website: https://www.cvl.isy.liu.se/en/research/datasets/swedish-leaf/ (accessed on 13 December 2025), and Flavia leaves dataset is available at https://www.kaggle.com/datasets/gauravneupane/flavia-dataset (accessed on 13 December 2025). The code used is available at the following link: https://github.com/GiLonga/Geometric-Learning (accessed on 13 December 2025). A Tutorial notebook is available at the following link: https://github.com/ioanaciuclea/geometric-learning-notebook (accessed on 13 December 2025).

Acknowledgments: A.B.T and I.C. would like to acknowledge the excellent working conditions and interactions at Erwin Schrödinger International Institute for Mathematics and Physics, Vienna, during the thematic programme "Infinite-dimensional Geometry: Theory and Applications" where part of this work was completed. I.C was supported by the START Grant from the West University of Timişoara during the writing of this paper. This research was funded in whole or in part by the Austrian Science Fund (FWF) Grant "Geometric Green Learning on Groups and Quotient Spaces" [https://www.fwf.ac.at/en/research-radar/10.55776/PAT1179524] (accessed on 1 February 2025). For open access purposes, the author has applied a CC BY public copyright license to any author accepted manuscript version arising from this submission. The authors acknowledge TU Wien Bibliothek for financial support through its Open Access Funding Program.

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

References

1. Bauer, M.; Bruveris, M.; Michor, P.W. Overview of the geometries of shape spaces and diffeomorphism groups. *J. Math. Imaging Vis.* **2014**, *50*, 60–97. [CrossRef]
2. Mennucci, A.C.G. Metrics of Curves in Shape Optimization and Analysis. In *Level Set and PDE Based Reconstr*; Springer Lecture Notes in Mathematics; Springer: Berlin/Heidelberg, Germany, 2013; pp. 205–319. [CrossRef]
3. Binz, E.; Fischer, H.R. The manifold of embeddings of a closed manifold. In *Differential Geometric Methods in Theoretical Physics, Proceedings of the International Conference Held at the Technical University of Clausthal, Clausthal-Zellerfeld, Germany, July 1978*; Lecture Notes in Physics 139; Springer: Berlin/Heidelberg, Germany, 1981.
4. Cervera, V.; Mascaró, F.; Michor, P.W. The action of the diffeomorphism group on the space of immersions. *Diff. Geom. Appl.* **1991**, *1*, 391–401. [CrossRef]
5. Mennucci, A.C.G. Neighborhoods and Manifolds of Immersed Curves. *Int. J. Math. Math. Sci.* **2021**, *2021*. [CrossRef]
6. Preston, S.C. The geometry of whips. *Ann. Glob. Anal. Geom.* **2012**, *41*, 281–305. [CrossRef]
7. Tumpach, A.B.; Preston, S.C. Three methods to put a Riemannian metric on Shape Space. In *Geometric Science of Information, Proceedings of the 6th International Conference, GSI 2023, St. Malo, France, 30 August–1 September 2023*; Proceedings, Part I, Springer: Berlin/Heidelberg, Germany, 2023; pp. 3–11. [CrossRef]
8. Maksimović, S.; Borković, A. A New Class of Plane Curves with Arc Length Parametrization and Its Application to Linear Analysis of Curved Beams. *Mathematics* **2021**, *9*, 1778. [CrossRef]
9. Tumpach, A.B. On canonical parameterizations of 2*D*-curves. In *Geometric Science of Information, Proceedings of the 6th International Conference, GSI 2023, St. Malo, France, 30 August–1 September 2023*; Proceedings, Part I; Springer: Berlin/Heideberg, Germany, 2023; pp. 31–40. [CrossRef]
10. Bruveris, M. Optimal Reparametrizations in the Square Root Velocity Framework. *SIAM J. Math. Anal.* **2015**, *48*, 4335–4354. [CrossRef]
11. Sundaramoorthi, G.; Mennucci, A.; Soatto, S.; Yezzi, A. A new geometric metric in the space of curves, and applications to tracking deforming objects by prediction and filtering. *SIAM J. Imaging Sci.* **2011**, *4*, 109–145. [CrossRef]
12. Mio, W.; Srivastava, A.; Joshi, S.H. On shape of plane elastic curves. *Int. J. Comput. Vis.* **2007**, *73*, 307–324. [CrossRef]
13. Srivastava, A.; Klassen, E.; Joshi, ; S.H.; Jermyn, I.H. Shape analysis of elastic curves in Euclidean spaces. *IEEE Trans. Pattern Anal. Mach. Intell.* **2011**, *33*, 1415–1428. [CrossRef]
14. Younes, L.; Michor, P.W.; Shah, J.; Mumford, D. A metric on shape space with explicit geodesics. *Mat. Appli.* **2008**, *19*, 25–57. [CrossRef]
15. Bauer, M.; Bruveris, M.; Marsl, S.; Michor, P.W. Constructing reparametrization invariant metrics on spaces of plane curves. *Diff. Geom. Its Appl.* **2014**, *34*, 139–165. [CrossRef]
16. Needham, T.; Kurtek. S. Simplifying transforms for general elastic metrics on the space of plane curves. *SIAM J. Imaging Sci.* **2020**, *13*, 445–473. [CrossRef]
17. Lahiri, S.; Robinson, D.; Klassen, E. Precise matching of PL curves in $\mathbb{R}^n$ in the square root velocity framework. *Geom. Imaging Comput.* **2015**, *2*, 133–186. [CrossRef]
18. Hartman, E.; Sukurdeep, Y.; Charon, N.; Klassen, E.; Bauer, M. Supervised deep learning of elastic SRV distances on the shape space of curves. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Virtual, 19–25 June 2021; pp. 4425–4433. [CrossRef]
19. Tumpach, A.B.; Preston, S.C. Quotient Elastic Metrics on the manifold of arc-length parameterized plane curves. *J. Geom. Mech.* **2017**, *9*, 227–256. [CrossRef]
20. Tumpach, A.B.; Drira, H.; Daoudi, M.; Srivastava, A. Gauge Invariant Framework for Shape Analysis of Surfaces. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *38*, 1. [CrossRef]

21. Tumpach, A.B. Gauge Invariance of degenerate Riemannian metrics. *Not. Am. Math. Soc.* **2016**, *63*, 342–350. [CrossRef]
22. Drira, H.; Tumpach, A.B.; Daoudi, M. Gauge Invariant Framework for Trajectories Analysis. In Proceedings of the 1st International Workshop on DIFFerential Geometry in Computer Vision for Analysis of Shapes, Images and Trajectories (DIFF-CV), Swansea, UK, 7–10 September 2015. [CrossRef]
23. Michor, P.W.; Mumford, D.B. Riemannian geometries on spaces of plane curves. *J. Eur. Math. Soc.* **2006**, *8*, 1–48. [CrossRef]
24. Beck, A.; Ochs, P. Symmetries in PAC-Bayesian Learning. *arXiv* **2025**. [CrossRef]
25. Bezdek, J.C.; Pal, N.R. Some new indexes of cluster validity. *IEEE Trans. Syst. Man Cybern. B Cybern.* **1998**, *28*, 301–315. [CrossRef]
26. Bolshakova, N.; Azuaje, F. Cluster validation techniques for genome expression data. *Signal Process.* **2003**, *83*, 825–833. [CrossRef]
27. Dunn, J.C. A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. *J. Cybern.* **1973**, *3*, 32–57. [CrossRef]
28. Davies, D.L.; Bouldin, ; D.W. A Cluster Separation Measure. *IEEE Trans. Pattern Anal. Mach. Intell.* **1979**, *PAMI-1*, 224–227. [CrossRef]
29. Alajlan, N.; El Rube, I.; Kamel, M.S.; Freeman, G. Shape retrieval using triangle-area representation anddynamic space warping. *Pattern Recognit.* **2007**, *40*, 1911–1920. [CrossRef]
30. Xu, G.; Li, C. Plant leaf classification and retrieval using multi-scale shape descriptor. *J. Eng.* **2021**, *8*, 467–475. [CrossRef]
31. Almodfer, R.; Mudhsh, M.; Zhao, J. Pyramided and optimized blurred shape model for plant leaf classification. *IET Image Process.* **2023**, *17*, 2838–2854. [CrossRef]
32. Rojas-Hernández, R.; López-Chau, A.; Trujillo-Mora, V.; Rojas-Hernández, C.A. Plant identification using new geometricfeatures with standard data mining methods, In Proceedings of the 2016 IEEE 13th International Conference on Networking, Sensing, and Control (ICNSC), Mexico City, Mexico, 28–30 April 2016; pp. 1–4.
33. Hu, R.; Jia, W.; Ling, H.; Huang, D. Multiscale distance matrix for fast plant leaf recognition. *IEEE Trans. Image Process.* **2012**, *21*, 4667–4672. [CrossRef]
34. Ling, H.; Jacobs, D.W. Shape classification using the inner-distance. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 286–299. [CrossRef]
35. Adamek, T.; Connor, N.E.O. A multiscale representation method for non-rigid shapes with a single closed contour. *IEEE Trans. Circuits Syst. Video Technol.* **2004**, *14*, 742–753. [CrossRef]
36. Li, G.; Zhang, R.; Qi, D.; Ni, H. Plant-Leaf Recognition Based on Sample Standardization and Transfer Learning. *Appl. Sci.* **2024**, *14*, 8122. [CrossRef]