Representation of Surfaces with Normal Cycles. Application to Surface Registration

Pierre Roussillon, Joan Glaunès

Representation of Surfaces with Normal Cycles. Application to Surface Registration.

Pierre Roussillon* Joan Alexis Glaunès†

January 21, 2019

Abstract

In this paper, we present a framework for computing dissimilarities between surfaces which is based on the mathematical model of normal cycle from geometric measure theory. This model allows to take into account all the curvature information of the surface without explicitely computing it. By defining kernel metrics on normal cycles, we define explicit distances between surfaces that are sensitive to curvature. This mathematical framework also has the advantage of encompassing both continuous and discrete surfaces (triangulated surfaces). We then use this distance as a data attachment term for shape matching, using the Large Deformation Diffeomorphic Metric Mapping (LDDMM) for modeling deformations. We also present an efficient numerical implementation of this problem in PyTorch, using the KeOps library, which allows both the use of auto-differentiation tools and a parallelization of GPU calculations without memory overflow. We show that this method can be scalable on data up to a million points, and we present several examples on surfaces, comparing the results with those obtained with the similar varifold framework.

1 Introduction

This article takes place in the context of computational anatomy, an active research field which aims at providing a mathematical framework to study the variability of anatomical structures among a population of subjects. The term "computational anatomy" was first introduced by Grenander and Miller [28].

In the past decades, the development of acquisition techniques (among which Magnetic Resonance Imaging, coherence tomography, Diffusion-Tensor Imaging, Functional MRI) opens the way to an early diagnosis of diseases that cause or are caused by unexpected deformations. A qualitative approach is not anymore sufficient and one necessitates an automatized procedure for an analysis that fully exploits the size of the database. This automation implies a quantitative approach, and thus a mathematical modelling of shape variability and how to measure it.

This problematic is at the heart of computational anatomy. Formally, one estimates from a "healthy" dataset a statistical model of the shapes' variability. From there, it is possible to provide statistical tests to discriminate between a pathological and a normal shape variation. Of course, this is a very large and complex problem. From a mathematical point of view, the measure of shapes variability necessitates a framework that provides theoretical and numerical guarantees. From a medical point of view, the relevance and the interpretation of such applications need to be evaluated depending on the anatomical structure at stake. Yet, numerous studies has shown promising results. Let us quote

*LTCI, Télécom ParisTech (pierre.roussillon@telecom-paristech.fr).

†Université Paris Descartes, Sorbonne Paris Cité, MAP5 UMR 8145 (alexis.glaunes@mi.parisdescartes.fr)

for example works on Alzeihmer's disease [39, 47, 44, 18, 17], on DTI images [20, 38, 29], heart malformations [34], Down syndrom [21, 19] and retina layer for glaucoma diagnosis [7, 31, 32].

An elementary brick for carrying out such statistical analyses is the pairwise registration of geometric structures. This matching problem is often addressed as a variational problem, with a cost functional which is the sum of two terms: a data attachment term that measures how close the deformed shape is to the target shape, and a regularization term on the deformation that ensures that the problem is well posed. In this context, the construction of a data-attachment term to measure the residual distance between the deformed shape and the target is of importance. Indeed, depending on the features of the shapes it takes into account, this term will drive the registration procedure in a certain way. Ideally, we want a mathematical setting to represent the shapes (e.g. the surfaces) in a common framework for discrete and continuous shapes, and with theoretical guarantees of the kind of information that we are able to retrieve through this representation. Last but not least, this setting should provide an explicit metric between shapes, that is implementable numerically for discrete shapes.

In the past decade, several models for defining such data-attachment terms for curves, surfaces or points clouds have been proposed, based on mathematical tools coming from geometric measure theory. The first of these models consisted of defining kernel metrics on spaces of *currents* ([46], [27]) to provide an explicit distance between shapes, for both continuous and discrete cases.

This setting is now commonly used in computational anatomy; its advantages lie in its simple implementation, its parametrization-free representation and the fact that it provides a common framework for continuous and discrete shapes. However, currents are oriented objects and thus a consistent orientation of shapes is needed for a coherent matching. Moreover, due to this orientation property, artificial cancellation can occur with shapes with high local variations. To deal with this problem, a more advanced model based on *varifolds* has been introduced more recently [10, 30]. Such framework provides a non-oriented distance between shapes, that contains first order information. Note that optimal transport has also been used to construct a data attachment term for diffeomorphic registration in the same spirit [25].

In [42], we have introduced another shape representation in computational anatomy, using the mathematical model of *normal cycle*, and we have applied it to curves. The notion of normal cycle has been introduced by Zähle in [49], based on the work on curvature measures developed by Federer earlier [22]. The theory of normal cycles relies on the representation of shapes through their unit normal bundle: more precisely, the normal cycle of a shape is the current associated with its unit normal bundle. The benefit of such a model is that the normal cycle contains all the curvature information of the underlying shape, as it was shown in [49], [50]. The idea is that the current associated with a shape contains only first order information. Considering the current of the unit normal bundle, we get first order information of a first order model, and thus curvature information. One can find a more detailed explanation in [45, 42]. Defining kernel metrics on normal cycles, one gets an explicit data-attachment term between shapes which is sensitive to curvature and also to singularities of the shapes, such as branching points and boundaries. Taking into account the curvature during the registration process is an interesting property, since in most application regions with high curvature are features that we want to be matched together.

In this article, we propose to extend the construction of kernel metrics on normal cycles seen in [42] to surfaces of $\mathbb{R}^3$ (note that we presented preliminary work on surfaces in a short article [43]). The contributions of this paper are twofold: first, the extension of the previous work to surfaces requires a fine decomposition of normal bundle in the case of triangulated surfaces. The choice of the kernel defining the metric is also crucial, so that the distance can be calculated, and contains non-trivial curvature information. We also present an implementation of this algorithm, which makes extensive use of recent software developments in the neural networks community. More precisely, our implementation takes full advantage of the automatic differentiation module of the PyTorch library [37], together with a seamless use of parallelization of generic convolution operations on Graphics

Processing Units (GPU) provided by the recent KeOps module [8, 9], developed by Benjamin Charlier, Joan Glaunès and Jean Feydy. This allows to have a linear memory footprint memory on GPU for kernel convolutions, and thus to avoid memory overflows that happen because of the usual quadratic memory footprint. All this elements lead to a code for the matching problem that is both very efficient (because the parallelization of the calculations is optimized) and simple (because the calculation of the gradients is painless). In addition, the computation times presented are significantly improved because of the automatic and efficient calculation of the gradient. By doing so, we are able to define distances that are sensitive to curvature, with an efficient implementation and that significantly improve matching results.

The article is organized as follows: in Sect. 2, we present the bases of the representation of shapes with normal cycles, after which we describe the normal cycle in the case of triangulated surfaces. In sect. 3 and 4, we define the kernel metrics on normal cycles and express the associated distance in the case of discrete surfaces. We will see that we can interpret what type of curvature is encoded in these distances. In section 5, we detail the diffeomorphic deformation model that we will use for the inexact registration problem, namely the Large Deformation Diffeomorphic Mapping Metric (LDDMM, [48]) framework. Section 6 focuses on the numerical aspects of the registration problem. We detail the implementation in PyTorch, and provide benchmarks for computation times of the distance on normal cycles, as well as the gradient evaluation. We compare this computation times with the ones on varifolds, and we observe that they are of the same order of magnitude, despite the greater complexity of the normal cycle model. Finally, in Section 7, we present many examples of matching on surfaces using kernel metrics on normal cycles as a data attachment term. The various examples aim to show the advantages of introducing curvature into the data-attachment term.

2 Representation of shapes with normal cycles

The question of shapes representation is a central point in *geometric measure theory* whose reference book is from Federer [23]. This field was motivated in the second half of the 20th century by the calculus of variation, and more specifically by the *Plateau's problem* of finding a minimal area surface with constrained boundary: given a closed $(m-1)$-dimensional surface $\Gamma \subset \mathbb{R}^d$, find an m-dimensional surface S of least area such that $\partial S = \Gamma$. Solving this problem was a conceptual breakthrough. Indeed if the formulation is a classical minimization problem with constraint, the main difficulty was to provide a theoretical setting to embed surfaces in a topological space with nice properties. The specifications of this framework are the following: define a space where the surfaces can be represented, with a topology that allows some compactness properties for a minimizing sequence of the Plateau's Problem. The representation of oriented surfaces with *integral currents* ([24]) and later the representation of non-oriented surfaces with varifolds ([2], [1]) where introduced for this purpose.

As convenient as these two settings may be, in this article we investigate the finer shape representations of *normal cycles*. The theory of normal cycles originated from Federer's work on *curvature measures* [22] and was pushed forward by Zähle [49] who gave integral representation of these measures: this is the *normal cycle*. It provides another theoretical tool to represent surfaces and more generally shapes, encoding higher order information such as curvature.

The theory of normal cycles had already been used in the applied mathematics community, to perform curvature estimation from triangulations: in [15], the authors use the Lipschitz-Killing forms to generalize the mean and Gaussian curvatures in a mathematical setting that encompasses the polyhedral case as well as the continuous case. They also use vector valued differential forms to extend the notion of second fundamental form operator and get finer results: the principal curvatures as well as the principal directions may be estimated from this. Moreover, an upper bound of the error of the estimated curvature from a triangulated approximation of a smooth shape is proven. The estimation of the second fundamental form is refined in [16] in a Riemannian framework. [13] extends the stability

result of [15] that was valid only for approximation of smooth hypersurfaces. Introducing the μ-reach of a set, they provide curvature stability with respect to the Hausdorff distance of compact subset that has positive μ-reach, still using the theory of normal cycles. For example a finite set of points in $\mathbb{R}^3$ has a positive μ-reach and the authors derived an algorithm to explicit the curvature measure in this case from a description of its normal bundle. Some of the previous curvature estimation with normal cycles are summed up in [36].

In this section, we detail the representation of shapes with normal cycles. We start with a brief recall of currents, as it is underpinning for the theory of normal cycles. Then, we define the normal cycle of a shape with positive reach as the current associated with its unit normal bundle, and we explain how to extend this definition to sets that are union of sets with positive reach. These kinds of sets encompass the case of discrete shapes, such as triangulations. The next step is then to describe the normal cycles for elementary objects as a triangle and a union of triangle. The latter relies on a decomposition of the unit normal bundle into spherical, cylindrical and planar part, each part being associated with a precise kind of curvature.

2.1 Currents

We present the framework for m-dimensional currents in $\mathbb{R}^d$.

Let $0 \leq m \leq d$ be an integer. In the following, we consider $\Omega_0^m(\mathbb{R}^d) := \mathscr{C}_0\big(\mathbb{R}^d, (\Lambda^m\mathbb{R}^d)^*\big)$ the space of continuous differential forms on M, vanishing at infinity, endowed with the uniform norm. Here $\Lambda^m\mathbb{R}^d$ is the space of m-vector in $\mathbb{R}^d$ and $(\Lambda^m\mathbb{R}^d)^*$ is its algebraic dual.

In this article, a current will simply be an element of the topological dual of $\Omega_0(\mathbb{R}^d)$:

Definition 1 (Currents)**.** *The space of m-currents in $\mathbb{R}^d$ is defined as the topological dual of $\Omega_0^m(\mathbb{R}^d)$, denoted $\Omega_0^m(\mathbb{R}^d)'$. $T \in \Omega_0^m(\mathbb{R}^d)'$ maps every differential form ω to $T(\omega) \in \mathbb{R}$ and $T(\omega) \leq C_T \|\omega\|_\infty$.*

Note 1. *This definition differs from the original one of Federer that was introduced similarly to Schwartz' theory of distribution [24]. However, we do not need such refinements for our applications.*

A current can be seen as an object that integrates differential forms, and if we consider X an *oriented*, m-dimensional submanifold of $\mathbb{R}^d$, we can associate with X a current, denoted $[X]$, through the integration of differential forms over X: if $\omega \in \Omega_0^m(\mathbb{R}^d)$,

$$[X](\omega) := \int_X \langle \omega(x) | \tau_X(x) \rangle \, d\sigma_X(x)$$

where $\tau_X(x)$ is the m-vector associated with a positively oriented, orthonormal basis of T_xX (if $(e_1(x), \ldots, e_m(x))$ is an orthonormal frame of T_xX, then $\tau_X(x) := e_1(x) \wedge \cdots \wedge e_m(x)$), $\langle .|. \rangle$ stands for the dual pairing between $\omega(x) \in (\Lambda^m\mathbb{R}^d)^*$ and $\tau_X(x) \in \Lambda^m\mathbb{R}^d$, and σ_X is the volume form of X (that coincides with the m-dimensional Hausdorff measure of $\mathbb{R}^d$, $\mathscr{H}^m$). We have seen here that the space of m-dimensional currents contains, among others, all the m-dimensional submanifolds.

Another interesting object living in $\Omega_0^m(\mathbb{R}^d)'$ is the *Dirac current*. Considering $x \in \mathbb{R}^d$ and $\alpha \in \Lambda^m\mathbb{R}^d$, we define δ_x^α as:

$$\delta_x^\alpha(\omega) := \langle \omega(x) | \alpha \rangle .$$

These types of currents are useful to have compact approximation of currents, especially for discrete shapes.

2.2 Representation of surfaces with normal cycles

The convenient setting for introducing normal cycles is the one of sets with positive reach. In this section we give basic definitions and results of the theory, and refer to [22], [50] for proofs and more developments.

Definition 2 (reach of a set)**.** *The* reach *of a set $X \subset \mathbb{R}^3$ is the supremum of the $r > 0$ for which we can define a unique projection from ∂X_r on X, where $\partial X_r = \{x \in \mathbb{R}^3 | \mathrm{d}(x,X) = r\}$.*

One can find in figure 1 an illustration of a set X with zero-reach and a set X with positive reach.

$\mathrm{Reach}(X) = 0$ $\mathrm{Reach}(X) > 0$

Figure 1: On the left: a curve X with zero-reach. On the right: a curve X with positive reach. The red dotted line represents the medial-axis, i.e. the set of points which do not have a unique projection on X.

Example 1.

- *For a convex set X,* $\mathrm{Reach}(X) = +\infty$.
- *If X is a $\mathscr{C}^2$-submanifold, compact,* $\mathrm{Reach}(X) > 0$.

One can prove that for any $r < \mathrm{Reach}(X)$, ∂X_r is a closed surface of class $\mathscr{C}^1$ in $\mathbb{R}^3$.

Definition 3 (Unit normal bundle for sets with positive reach in $\mathbb{R}^3$)**.** *Let $X \subset \mathbb{R}^3$ be a set with positive reach, and $0 < r < \mathrm{Reach}(X)$, and let $x \in S$. The* unit normal bundle *of X, denoted $\mathscr{N}_X$ is defined as:*

$$\mathscr{N}_X := \left\{(x,n) \in X \times \mathbb{S}^2, x + rn \in \partial X_r\right\}.$$

It can be proven that this set $\mathscr{N}_X$ does not depend on r, and that $g_r : (x,n) \mapsto x + rn$ defines a bi-Lipschitz bijection between $\mathscr{N}_X$ and ∂X_r. Since ∂X_r is a $\mathscr{C}^1$ closed surface in $\mathbb{R}^3$, it is canonically oriented, and this orientation transfers via g_r to a canonical orientation of $\mathscr{N}_X$.

Definition 4 (Normal cycles for sets with positive reach in $\mathbb{R}^3$)**.** *Let $X \subset \mathbb{R}^3$ be a set with positive reach. The* normal cycle *of X, denoted $N(X)$, is the 2-current associated with the surface $\mathscr{N}_X \subset \mathbb{R}^3 \times \mathbb{S}^2$, endowed with the canonical orientation. If $\omega \in \Omega_0^2(\mathbb{R}^3 \times \mathbb{S}^2)$,*

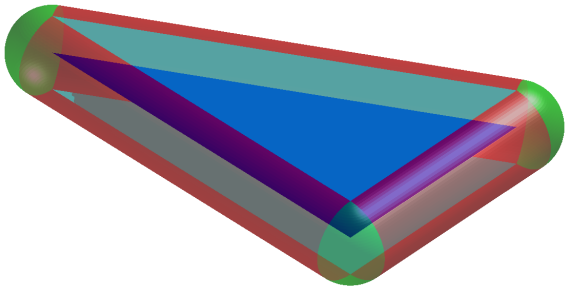
$$N(X)(\omega) := [\mathscr{N}_X](\omega) = \int_{\mathscr{N}_X} \left\langle \omega(x,n) \middle| \tau_{\mathscr{N}_X}(x,n) \right\rangle d\mathscr{H}^2(x,n).$$

The representation of a shape with its normal cycle is thus equivalent to consider the shape as an object which integrates differential forms over the unit normal bundle. As explained in [49], [45], and later in [42], there exists differential forms $\omega_0, \omega_1, \omega_2$ such that $N(X)(\omega_i)$ retrieves the integral of the i-th curvature of X when X is a smooth surface of $\mathbb{R}^3$. These differential forms are called the *Lipschitz-Killing* differential forms. This is why we can say that the representation of shapes with normal cycles encode all the curvature information.

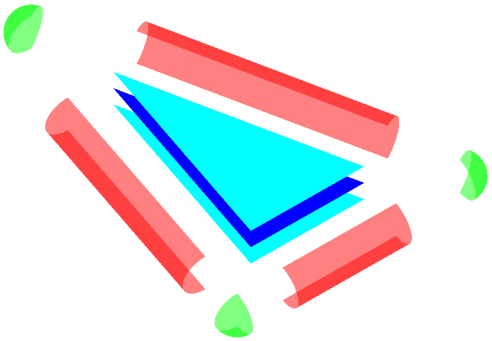
2.2.1 Description of the Unit Normal Bundle of a Triangle

Let T be a solid triangle of $\mathbb{R}^3$ with vertices x_1, x_2, x_3 and edges : $f_1 = x_2 - x_1$, $f_2 = x_3 - x_2$, $f_3 = x_1 - x_3$. The normal vectors of the face are : $n_T = \frac{f_1 \times f_2}{|f_1 \times f_2|}$ and $-n_T$.

Note that the description of the normal cycle of a 2-dimensional polyhedral has been studied in [15]. The description of the normal bundle of a triangle is quite straightforward. As illustrated in fig. 2, it can be decomposed into a planar part (associated with the unit normal vectors at the relative interior of T), composed of two triangles (in cyan), a cylindrical part (associated with the generalized unit normal vectors at the edges), composed of three "half" cylinders located at the edges (in red), and a spherical part (associated with the generalized unit normal vectors at the vertices), composed of three portions of sphere located at the vertices (in green).



(a) Representation of the normal bundle in $\mathbb{R}^3$.



(b) Separation of the normal bundle for better visualisation.

Figure 2: Illustration of the decomposition of the normal bundle of a dark blue triangle into a planar (in cyan), a cylindrical (in red) and a spherical (in green) parts. Note that the actual normal bundle lives in $\mathbb{R}^3 \times \mathbb{S}^2$ and not in $\mathbb{R}^3$.

This description of the unit normal bundle can be summed up as follows. Let us define

$$\begin{aligned} \mathcal{N}_T^{pln} &:= T \times \{-n_T, n_T\}, \\ \mathcal{N}_T^{cyl} &:= \cup_{i=1}^3 [x_i, x_{i+1}] \times S^{\perp +}_{f_i, f_i \times n_T}, \\ \mathcal{N}_T^{sph} &:= \cup_{i=1}^3 \{x_i\} \times S^{+}_{f_{i-1}, -f_{i+1}}. \end{aligned}$$

$\mathcal{N}_T^{pln}, \mathcal{N}_T^{cyl}, \mathcal{N}_T^{sph}$ stand for respectively for the planar, cylindrical and spherical part of the unit normal bundle. Here, for any non zero vectors $\alpha, \beta \in \mathbb{R}^3$, we denote the semicircle $S^{\perp +}_{\alpha,\beta} := \left(\mathbb{S}^2 \cap \alpha^{\perp}\right) \cap \{u | \langle u, \beta \rangle \geq 0\}$, and the portion of sphere $S^{+}_{\alpha,\beta} := \left\{u \in \mathbb{S}^2, \langle u, \alpha \rangle \geq 0, \langle u, \beta \rangle \geq 0\right\}$. Note that f_i and n_T depend on an orientation of T, but this is not the case for $f_i \times n_T$ which is always oriented outward from the triangle. We define the associated currents:

$$N(T)^{pln} = [\mathcal{N}_T^{pln}], N(T)^{cyl} = [\mathcal{N}_T^{cyl}], N(T)^{sph} = [\mathcal{N}_T^{sph}].$$

We have straightforwardly

$$N(T) = N(T)^{pln} + N(T)^{cyl} + N(T)^{sph}$$

because the sets $\mathscr{N}_T^{pln}$, $\mathscr{N}_T^{cyl}$ and $\mathscr{N}_T^{sph}$ intersect only along 1-dimensional subsets and their union equals $\mathscr{N}_T$.

Notice that as previously said, every part of the triangle appears in the unit normal bundle: for a triangle the edges are associated with half cylinders, and the vertices with portions of spheres. Starting from this description, we will see how to consider the normal cycle of a polyhedral mesh.

2.2.2 Decomposition of the normal cycle for triangulation meshes

The theory of normal cycles can be extended to a class of sets containing unions of sets with positive reach, as developed in [50] [40],[45]. We briefly introduce this extension here, referring to these works for all details. The $\mathscr{U}_{PR}$ class is defined as the class of sets X which can be written as a locally finite union of sets X_i, $i \in \mathbb{N}$, such that for any finite subset of indices $I \subset \mathbb{N}$, $\cap_{i\in I} X_i$ is of positive reach. In particular sets of positive reach belong of course to this class, and it contains also all finite unions of non-empty closed convex sets. The normal cycle $N(X)$ associated with a set $X \in \mathscr{U}_{PR}$ can be defined in a recursive way so that the following fundamental additive property is satisfied:

Definition 5 (Additive property)**.** *Assume that sets X, Y, $X \cap Y$ are with positive reach. Then we define*

$$N(X \cup Y) := N(X) + N(Y) - N(X \cap Y) \tag{1}$$

In the case where $X \cup Y$ is with positive reach, this definition is coherent : the left hand side and the right hand side in the definition are equal. In the case of a finite union of sets with positive reach: $X = \cup_{i=1}^n X_i$, belonging to $\mathscr{U}_{PR}$, it is easy to see that any combination of unions and intersections of the X_i also belongs to $\mathscr{U}_{PR}$. Hence the additive formula allows to write a recursive expression for the normal cycle of X, which can serve as a definition for normal cycle in this case: for $1 \leq k \leq n$, one has

$$\begin{aligned} N(X_1 \cup \cdots \cup X_k) = {} & N(X_1 \cup \cdots \cup X_{k-1}) + N(X_k) \\ & - N((X_1 \cup \cdots \cup X_{k-1}) \cap X_k) \end{aligned}$$

Since this formula is not ready-to-use, we will rather explicit a decomposition of the unit normal bundle such that the additive property is straightforward. This work has already been done for discrete curves in [42]. We recall it briefly here for convenience, starting with union of segments and considering union of triangles after.

If we start with a single segment, $C = [a,b]$, we define the normal cycle of its relative interior, the "open segment" $\tilde{C} := [a,b] \setminus \{a,b\}$ as

$$N(\tilde{C}) = N(C) - N(\partial C) = N(C) - N(\{a\}) - N(\{b\})$$

where $N(\{a\}) = \{a\} \times \mathbb{S}^2$, $N(\{b\}) = \{b\} \times \mathbb{S}^2$.

Now, let $C_1 \cup \cdots \cup C_n$ be a union of n segments in $\mathbb{R}^d$. We can consider without loss of generality that two segments C_i and C_j for $i \neq j$ either do not intersect or intersect at one of their end points. Using the additive property eq. (1) it can be easily seen that the normal cycle of a union of segments can be obtained by summing the normal cycles associated with open segments and vertices. More precisely, if we denote $\{v_1, \ldots, v_N\}$ the vertices of $\cup_{i=1}^n C_i$, our decomposition of the normal bundle satisfies:

$$N(C_1 \cup \cdots \cup C_n) = \sum_{i=1}^{n} N(\tilde{C}_i) + \sum_{j=1}^{N} N(\{v_i\}) \tag{2}$$

Even though the additive property is now straightforward, we will go a bit further in this decomposition, as it will prove to be more efficient with the kernel metric. We can decompose (2) into cylindrical and spherical parts as follows:

$$N(C_1 \cup \cdots \cup C_n) = \left(\sum_{i=1}^{n} N(C_i)^{cyl} \right) + \left(\sum_{i=1}^{n} N(\tilde{C}_i)^{sph} + \sum_{j=1}^{N} N(\{v_i\}) \right) \tag{3}$$

This decomposition is sketched in fig. 3.

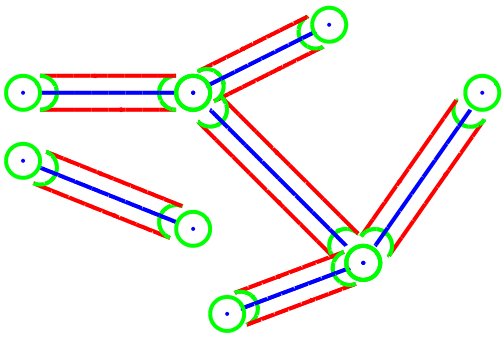


Figure 3: Decomposition of the normal bundle of a union of segments. In green, the spherical part (of a single point and of an extremity) and in red the cylindrical part. Note that this representation is only illustrative, as the true normal bundle belongs to the space $\mathbb{R}^2 \times \mathbb{S}^1$ in this case.

A slightly more complex decomposition is necessary for a union of triangles in $\mathbb{R}^3$. We apply the same process as for the union of segments. Let T be a triangle of $\mathbb{R}^3$ with vertices x_1, x_2, x_3 and edges : $f_1 = x_2 - x_1$, $f_2 = x_3 - x_2$, $f_3 = x_1 - x_3$. We denote by e_i the geometrical edges (i.e. the segments $[x_1, x_2]$, etc.) of the triangle. The normal vectors of the face are : $n_T = \frac{f_1 \times f_2}{|f_1 \times f_2|}$ and $-n_T$. First, we define the normal cycle of the relative interior of T, the "open triangle" $\tilde{T} := T \setminus \partial T$ where $\partial T := (e_1 \cup e_2 \cup e_3)$:

$$N(\tilde{T}) := N(T) - N(\partial T).$$

∂T is a union of the edges $(e_i)_{1 \leq i \leq 3}$, and the description of its normal bundle has been done right above:

$$N(\partial T) = \sum_{i=1}^{3} N(\tilde{e_i}) + \sum_{i=1}^{3} N(\{x_i\}).$$

Thus

$$N(\tilde{T}) = N(T) - \sum_{i=1}^{3} N(\tilde{e_i}) - \sum_{i=1}^{3} N(\{x_i\})$$

Since we know an explicit description of $N(T)$, $N(\tilde{e_i})$ and $N(\{x_i\})$, we can express $N(\tilde{T})$ as a sum of a spherical part, cylindrical part and planar part:

$$N(\tilde{T}) = N(\tilde{T})^{pln} + N(\tilde{T})^{cyl} + N(\tilde{T})^{sph},$$

with

$$\begin{aligned}\mathscr{N}^{pln}_{\tilde{T}} &:= \mathscr{N}^{pln}_{T} = T \times \{\pm n_T\},\\ \mathscr{N}^{cyl}_{\tilde{T}} &:= \cup_{i=1}^{3} e_i \times S^{\perp+}_{f_i,-f_i\times n_T},\\ \mathscr{N}^{sph}_{\tilde{T}} &:= \cup_{i=1}^{3} \{x_i\} \times S^{+}_{f_{i-1},-f_{i+1}},\end{aligned}$$

where $S^{+}_{\alpha,\beta} = \{u \in \mathbb{S}^2 \mid \langle u,\alpha\rangle \geq 0\,,\, \langle u,\beta\rangle \leq 0\}$, and

$$\begin{aligned}N(\tilde{T})^{pln} &:= [\mathscr{N}^{pl}_{\tilde{T}}] = \big[T \times \{\pm n_T\}\big],\\ N(\tilde{T})^{cyl} &:= -\sum_{i=1}^{3} \big[e_i \times S^{\perp+}_{f_i,-f_i\times n_T}\big],\\ N(\tilde{T})^{sph} &:= \sum_{i=1}^{3} \big[\{x_i\} \times S^{+}_{-f_{i+1},f_i}\big],\end{aligned}$$

In fig. 4 one can find an illustration of the normal bundle of an open triangle.

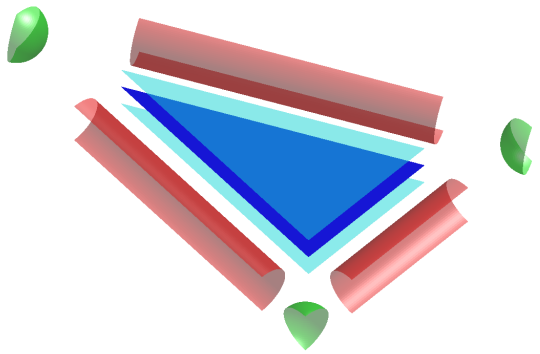


Figure 4: "Normal bundle" of an open triangle $\tilde{T}$ in dark blue. The normal bundle above the interior of the triangle, $\mathscr{N}^{pln}_{\tilde{T}}$, are two triangles in cyan. The normal bundle above the edges, $\mathscr{N}^{cyl}_{\tilde{T}}$ are three half cylinder, in red. The normal bundle over the vertices, $\mathscr{N}^{sph}_{\tilde{T}}$ are portions of sphere, in green.

After the introduction of $N(\tilde{T})$ we can proceed as for a union of segments. Suppose that $\mathscr{T} = \cup_{i=1}^{n_T} T_i$ is a triangulation where we require that the intersection of two triangles is either empty, or a single edge or a single vertex. We denote $(e_j)_{1\leq j\leq n_e}$ the edges and $(v_k)_{1\leq k\leq n_v}$ the vertices of the triangulation. Then one has:

$$N(\mathscr{T}) = \sum_{i=1}^{n_T} N(\tilde{T}_i) + \sum_{j=1}^{n_e} N(\tilde{e}_j) + \sum_{k=1}^{n_v} N(\{v_k\})$$

With this decomposition, the additive property is straightforward. Moreover, one can guess that the different parts of the unit normal bundle contain different kind of discrete curvature information: the area form for the planar part, the mean curvature for the cyclindrical part, and the Gaussian curvature for the spherical part. This will be clearer when introducing kernel metrics.

In this section we have developed the representation of shapes through normal cycles, in a common setting that encompasses both the smooth and the discrete case. We are left to design metric on such representations, so that we have a distance between shapes that is sensitive to curvature.

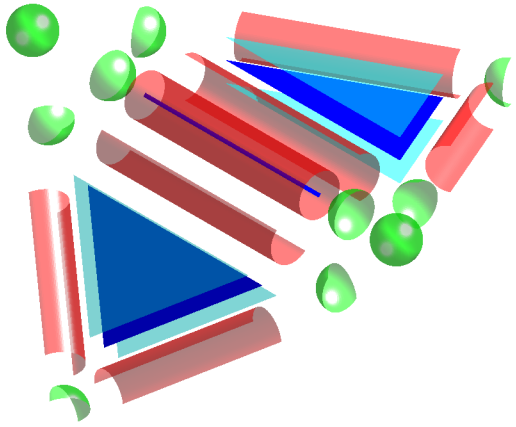


Figure 5: Decomposition of the normal bundle for two triangles with a common edge. In this figure, the two normal bundle of the open triangles appear. Then, we add (only once) the normal bundle of the open edge (the red cylinder and the two green half spheres). Then we add (only once) the normal bundle of the vertices of the edge (the two green spheres). Note that if the triangulation is reduced to this two triangles, we should add the normal bundle of the other edges of the triangles.

3 Kernel Metrics on Normal cycles

The idea of normal cycles (resp. current) is convenient because it embeds shapes in a vectorial space: the space of 2-currents in $\mathbb{R}^3 \times \mathbb{S}^2$ (resp. the space of 2-current in $\mathbb{R}^2$). These spaces, defined as dual to spaces of differential forms, come with a dual norm: if $T \in \Omega_0^m(\mathbb{R}^d)'$, we define $M(T) := \sup\left\{T(\omega), \omega \in \Omega_0^m(\mathbb{R}^d), \|\omega\|_\infty \leq 1\right\}$, called the mass norm in geometric measure theory. It would be tempting to use this norm as a distance between shapes. However it is not interesting for a matching purpose. Indeed, if C and S are two shapes, non intersecting, then one can show that $M([S]-[C]) = \mathscr{H}^m(C) + \mathscr{H}^m(S)$, and this independently of any closeness between the two sets. This happens because the set of test functions ω is too large, and thus discriminates completely the two shapes.

For our numerical purpose, we need a computable expression for the dissimilarity between shapes. In the very same spirit of [26] for currents and [11] for varifolds, we will use the theory of Reproducing Kernel Hilbert Space (RKHS, see [4] for the original article) to provide kernel metrics on normal cycles as dissimilarity measures. This work has already been presented in [42], and we present here the basis for self-completeness.

Example 2. *We illustrate this with the example of two curves C and S in $\mathbb{R}^2$. Representing these two curves as currents $[C]$ and $[S]$ (fig. 6), a kernel metric allows to consider a scalar product between those curves that takes explicit expression as integral over the curves:*

$$\langle [C], [S] \rangle_{W'} = \int_C \int_S k(x,y) \left\langle \tau_x, \tau_y \right\rangle d\mathscr{H}^1(x) d\mathscr{H}^1(y).$$

Now, if we represent the two curves as normal cycles (fig. 7), the kernel metric will consider integrals over the normal bundle rather than integrals over the curves themselves. Precisely, we will construct two scalar kernels k_p and k_n where k_p takes into account the relative spatial position of the curves and k_n the relative position of the normal vectors u and v at point $x \in C$ and $y \in S$.

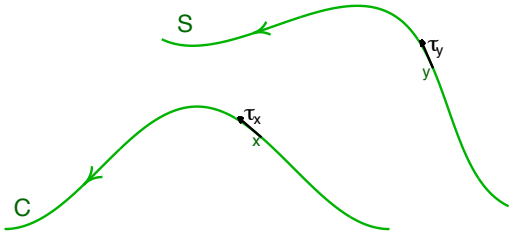


Figure 6: Representation of the curves C and S with currents

$$\langle N(C),N(S)\rangle_{W'} = \int_{\mathcal{N}_C}\int_{\mathcal{N}_S} k_p(x,y)k_n(u,v)\left\langle \tau_{(x,u)},\tau_{(y,v)}\right\rangle d\mathcal{H}^1(x,u)d\mathcal{H}^1(y,v).$$

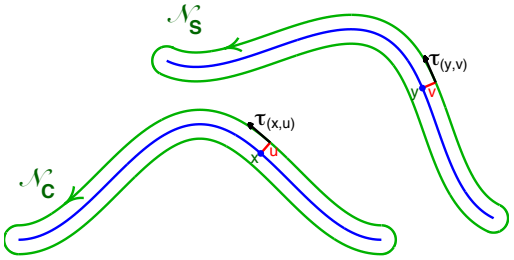


Figure 7: Representation of the curves C and S with normal cycle

It has been shown in [42] (in $\mathbb{R}^d$ and as a consequence in our framework of surface in $\mathbb{R}^3$) that by choosing scalar kernels k_n and k_p such that: $\forall x \in \mathbb{R}^3$, $k_p(x,.) \in \mathscr{C}_0(\mathbb{R}^d)$ and $\forall n \in \mathbb{S}^2$, $k_n \in \mathscr{C}(\mathbb{S}^2)$, then we generate a RKHS W of differential forms such that $W \hookrightarrow \Omega_0^2(\mathbb{R}^3 \times \mathbb{S}^2)$. This will always be the case in the following. The space W is generated through elementary differential forms :

$$k_p(x,.)k_n(n,.)\alpha,\ (x,n) \in \mathbb{R}^3 \times \mathbb{S}^2,\ \alpha \in \Lambda^2(\mathbb{R}^3 \times \mathbb{S}^2)^*$$

By considering the topological dual of these spaces, we obtain the inclusion: $\Omega_0^2(\mathbb{R}^3 \times \mathbb{S}^2)' \subset W'$. We recall that the normal cycle $N(S)$ of a surface $S \subset \mathbb{R}^3$ is an element of $\Omega_0^2(\mathbb{R}^3 \times \mathbb{S}^2)'$. The inclusion $\Omega_0^2(\mathbb{R}^3 \times \mathbb{S}^2)' \subset W'$ allows to have an explicit representation of $N(S)$ in W' through the kernels k_p and k_n.

Note 2. *The question whether the dual inclusion $\Omega_0^2(\mathbb{R}^3 \times \mathbb{S}^2)' \subset W'$ is an injection is called the* universality *property. It can be shown that it is equivalent to W being dense in $\Omega_0^2(\mathbb{R}^3 \times \mathbb{S}^2)$ [6]. We have already shown that if k_n is a Sobolev kernel, then we have indeed $\Omega_0^2(\mathbb{R}^3 \times \mathbb{S}^2)' \hookrightarrow W'$ ([42]). However, this will not be the case with the normal kernels k_n that we will chose in this article.*

In W', the scalar product between two normal cycles $N(S)$ and $N(C)$ associated with two surfaces S and C is explicit:

Proposition 1. ***[42]*** *With such spatial scalar kernel k_p and normal scalar kernel k_n, we generate a RKHS of differential forms $W \hookrightarrow \Omega_0^2(\mathbb{R}^3 \times \mathbb{S}^2)$. In W', the scalar product between the normal cycles*

$N(C)$ and $N(S)$ associated with two surfaces (discrete or smooth) C and S is:

$$\langle N(C), N(S)\rangle_{W'} = \int_{\mathscr{N}_C}\int_{\mathscr{N}_S} k_p(x,y)k_n(u,v)\left\langle \tau_{(x,u)}, \tau_{(y,v)}\right\rangle d\mathscr{H}^2(x,u)d\mathscr{H}^2(y,v). \tag{4}$$

The unit normal bundles $\mathscr{N}_C$ and $\mathscr{N}_S$ have been described for smooth surfaces and triangulations in the previous section. Our aim in the following is to specify some kernels k_n and k_p that allow for a computable scalar product between two triangulations. The main limitation for the choice of the kernels will be our ability to compute explicitly the scalar product, and to do so, we will chose two simple normal kernels : $k_n(u,v) = 1$ and $k_n(u,v) = \langle u,v\rangle$. We will see that even though these kernels may seem coarse at first, we are able to retrieve curvature-related information. For the spatial kernel k_p, we will use a Gaussian kernel $k_p(x,y) = \exp\left(\frac{-|x-y|^2}{\sigma_W^2}\right)$.

4 Expression of the Kernel Metric for Discrete Surfaces with Constant and Linear Normal Kernel

Note 3. *We derive expression of the kernel metrics on normal cycles with the linear normal kernel and the constant normal kernel k_n. However, for the moment and for the linear normal kernel we are limited to the implementation phase which is more delicate than for the constant kernel. This is why we only present experiments with the constant normal kernel in this article. It is however interesting to see that the normal linear kernel gives Gaussian curvature information as it will be seen in Prop. 4*

In this section, we derive the expression of the kernel metric for discrete surfaces in $\mathbb{R}^3$, with constant and linear normal kernels. For this, we will use the decomposition of the unit normal bundle that has been presented in sect. 2.2. The computation of the scalar product between two triangulations (eq. (4)) involves integration over the planar part (in cyan in fig. 5), the cylindrical part (in red) and the spherical part (in green). After introducing the notations in 4.1, we will detail the computation and the approximation of eq. (4) for each part of the normal bundle. One can find the scalar product with the constant and linear normal kernel k_n in 4.2 and 4.3. In sect. 6.2, we present an implementation of such metrics in PyTorch, using automatic differentiation libraries in order to compute the gradient of the metric without implementing it.

4.1 Notations for triangulated surfaces and general remarks

Let us first introduce the notations that we will use in the following.

Let $\mathscr{T} = \cup_{i=1}^N T_i$ and $\mathscr{T}' = \cup_{i=1}^M T_i'$ be two triangulated meshes. We denote $x_1,\dots,x_{n_v}$ (resp. $y_1,\dots,y_{m_v}$) the vertices of $\mathscr{T}$ (resp. of $\mathscr{T}'$). Given a triangle T_i (resp. T_j'), v_i^1, v_i^2, v_i^3 are its three vertices and b_i its barycentre: $b_i = \frac{1}{3}(v_i^1 + v_i^2 + v_i^3)$ (resp. b_j'). $(f_l)_{1\le l\le n_e}$ (resp $(g_l)_{1\le l\le m_e}$) are the edges of $\mathscr{T}$ (resp. $\mathscr{T}'$). $\pm n_{T_i}$ are the normal vectors of the triangle T_i. Moreover:

- $x_{f_i^1}$ and $x_{f_i^2}$ are the two vertices of f_i: $f_i = x_{f_i^2} - x_{f_i^1}$.
- c_i (resp. d_j) is the middle of the edge f_i (resp. g_j).
- n_{T,f_i} is the normal vector of the triangle T such that $n_{T,f_i}\times f_i$ is oriented inward for the triangle T.

We recall that with the kernel metric, the planar, cylindrical and spherical parts are orthogonal one with another ([42], prop. 36).

Proposition 2. ***[42], prop. 36.*** *For any two triangulations $\mathscr{T}$ and $\mathscr{T}'$, the planar part $N(\mathscr{T})^{pln}$, the cylindrical part $N(\mathscr{T})^{cyl}$ and the spherical part $N(\mathscr{T})^{sph}$ are orthogonal with respect to the kernel metric:*

$$\begin{aligned}\left\langle N(\mathscr{T})^{pln}, N(\mathscr{T}')^{cyl} \right\rangle_{W'} &= \left\langle N(\mathscr{T})^{cyl}, N(\mathscr{T}')^{sph} \right\rangle_{W'} \\ &= \left\langle N(\mathscr{T})^{sph}, N(\mathscr{T}')^{pln} \right\rangle_{W'} = 0\end{aligned}$$

The calculation of the expression of (4) in this case is simplified and we see here how convenient the decomposition introduced in sect. 2.2 is: we only need to compute scalar products between spherical parts, cylindrical parts and planar parts. This will be done below.

We start with $\left\langle N(\mathscr{T})^{pln}, N(\mathscr{T}')^{pln} \right\rangle_{W'}$. We would like to compute

$$\begin{aligned}\left\langle N(\mathscr{T})^{pln}, N(\mathscr{T}')^{pln} \right\rangle_{W'} = &\int_{\mathscr{N}_{\mathscr{T}}^{pln}} \int_{\mathscr{N}_{\mathscr{T}'}^{pln}} k_p(x,y) k_n(u,v) \\ &\times \left\langle \tau_{(x,u)}, \tau_{(y,v)} \right\rangle d\mathscr{H}^2(x,u) d\mathscr{H}^2(y,v)\end{aligned}$$

We have seen in sect. 2.2, and illustrated in fig. 5 that

$$N(\mathscr{T})^{pln} = \left[\mathscr{N}_{\mathscr{T}}^{pln} \right],$$

with

$$\mathscr{N}_{\mathscr{T}}^{pln} = \bigsqcup_{i=1}^{N} T_i \times \{\pm n_{T_i}\}.$$

$\mathscr{N}_{\mathscr{T}}^{pln}$ corresponds to the cyan triangles in fig. 5. Since the planar parts are disjoint, we can restrict ourselves to the computation of the scalar product for the planar part associated with a single triangle $T \in \mathscr{T}$ and $T' \in \mathscr{T}'$.

For a point $(x,u) \in \mathscr{N}_T^{pln}$, we denote $\tau_{(x,u)}$ the 2-tangent vector of $\mathscr{N}_T^{pln}$. We recall that this is a 2-vector associated with a positively oriented, orthonormal basis of $T_{(x,u)}\mathscr{N}_T^{pln}$. One can show that

$$\tau_{(x,u)} = \begin{pmatrix} e_1(x,u) \\ 0 \end{pmatrix} \wedge \begin{pmatrix} e_2(x,u) \\ 0 \end{pmatrix},$$

where $(e_1(x,u), e_2(x,u), u)$ is a positively oriented orthonormal basis of $\mathbb{R}^3$. Moreover, if $\tau_{(y,v)}$ is a 2-tangent vector of $\mathscr{N}_T^{pln}$ at point (y,v), then we have $\left\langle \tau_{(x,u)}, \tau_{(y,v)} \right\rangle = \langle u, v \rangle$. Combining all these quantities, we have:

$$\begin{aligned}\left\langle N(T)^{pln}, N(T')^{pln} \right\rangle_{W'} = &\int_T \int_{T'} k_p(x,y) k_n(\pm n_T, \pm n_{T'}) \\ &\times \langle \pm n_T, \pm n_{T'} \rangle d\mathscr{H}^2(x) d\mathscr{H}^2(y).\end{aligned}$$

In order to have a fast implementation, we approximate the integrals over T and T' with a single evaluation of the spatial kernel at the barycenters of the triangles c_T and $c'_{T'}$:

$$\begin{aligned}\left\langle N(T)^{pln}, N(T')^{pln} \right\rangle_{W'} \simeq &|T||T'| k_p(c_T, c'_{T'}) \\ &k_n(\pm n_T, \pm n_{T'}) \langle \pm n_T, \pm n_{T'} \rangle.\end{aligned} \tag{5}$$

where $|T| = \text{Area(T)}$. For the whole triangulation:

$$\left\langle N(\mathscr{T})^{pln}, N(\mathscr{T}')^{pln} \right\rangle_{W'} \simeq \sum_{i=1}^{N}\sum_{j=1}^{M} |T_i||T'_j| k_p(c_i, c'_j) \\ k_n(\pm n_{T_i}, n_{T'_j}) \left\langle \pm n_{T_i}, \pm n_{T'_j} \right\rangle.$$

Note 4. *If we choose $k_n(u,v) = \langle u, v\rangle$, we obtain:*

$$\left\langle N(T)^{pln}, N(T')^{pln} \right\rangle_{W'} \simeq 4\sum_{i=1}^{N}\sum_{j=1}^{M} |T_i||T'_j| k_p(c_i, c'_j) \left\langle n_{T_i}, n_{T'_j} \right\rangle^2.$$

which is exactly the kernel metric on varifolds for the linear kernel on the Grassmanian [11]. We see here that with the planar part, we retrieve the metric on varifolds. This shows that the metric on normal cycles contains more information about the shape that the one on varifolds.

The same work can be done for the cylindrical and the spherical part.

We consider two cylinders (or half cylinders) $C_i = f_i \times S_i$ and $C'_j = g_j \times S'_j$, where S_i (resp. S'_j) is a circle or a half circle, in a plane orthogonal to f_i (resp. g_j). C_i represents a generic elementary of the cylindrical part of the unit normal bundle for a triangulation, as it is illustrated in fig. 5. Thus, $[C_i]$, the current associated with C_i represents a typical contribution of the cylindrical part in the normal cycle.

For a point $(x,u) \in \mathscr{N}_T^{cyl}$, one can show ([16, 42]) that:

$$\tau_{(x,u)} = \begin{pmatrix} e_1(x,u) \\ 0 \end{pmatrix} \wedge \begin{pmatrix} 0 \\ e_2(x,u) \end{pmatrix},$$

where $(e_1(x,u), e_2(x,u), u)$ is a positively oriented orthonormal basis of $\mathbb{R}^3$. We then obtain:

$$\left\langle [C_i], [C'_j] \right\rangle_{W'} = \int_{f_i}\int_{g_j} k_p(x,y) \left\langle f_i/|f_i|, g_j/|g_j| \right\rangle \int_{S_i}\int_{S'_j} k_n(u,v) \langle u, v\rangle \\ \times d\mathscr{H}^2(x,u) d\mathscr{H}^2(y,v)$$

and with a similar approximation as for the planar part, we end up with:

$$\left\langle [C_i], [C'_j] \right\rangle_{W'} \simeq k_p(c_i, d_j) \left\langle f_i, g_j \right\rangle \\ \int_{S_i}\int_{S'_j} k_n(u,v) \langle u, v\rangle \, d\mathscr{H}^1(u) d\mathscr{H}^1(v) \tag{6}$$

One should notice that we do not approximate the integration over the normal part (i.e. integrations over S_i and S'_j).

For the spherical part, for a point $(x,u) \in \mathscr{N}_T^{sph}$, one can show ([16, 42]) that:

$$\tau_{(x,u)} = \begin{pmatrix} 0 \\ e_1(x,u) \end{pmatrix} \wedge \begin{pmatrix} 0 \\ e_2(x,u) \end{pmatrix},$$

where $(e_1(x,u), e_2(x,u), u)$ is a positively oriented orthonormal basis of $\mathbb{R}^3$

We compute explicitely the spherical scalar product: we end up with

$$\langle [\{x\} \times S_1], [\{y\} \times S_2] \rangle = k_p(x,y) \\ \times \int_{S_1}\int_{S_2} k_n(u,v) \langle u, v\rangle \, d\mathscr{H}^2(u) d\mathscr{H}^2(v) \tag{7}$$

where x and y are the vertices at stake, and S_1 and S_2 are portions of sphere (see fig. 5, in green).

In the next sections, we will specify the different scalar product with the constant ($k_n(u,v) = 1$) or the linear normal kernel ($k_n(u,v) = \langle u, v\rangle$).

4.2 Discrete scalar product with constant normal kernel

In this subsection, we express the discrete version of the scalar product eq. (4), with the constant normal kernel,

$$k_n(u,v) = 1.$$

With such kernel, it is easy to show that the planar scalar product vanishes, as well as the spherical scalar product. Only the cylindrical part, and the boundary term remain:

Proposition 3. *Let $\mathscr{T}$ and $\mathscr{T}'$ be two triangulated meshes. The approximated scalar product between the associated normal cycles with spatial kernel k_p and constant normal kernel $k_n(u,v) = 1$ is*

$$\begin{aligned}\left\langle N(\mathscr{T}), N(\mathscr{T}')\right\rangle_{W'} = \frac{\pi^2}{4}\sum_{i=1}^{n_e}\sum_{j=1}^{m_e} k_p(c_i,d_j)\left\langle f_i, g_j\right\rangle \\ \times\left\langle \sum_{\{T|f_i \text{ edge of } T\}} n_{T,f_i}, \sum_{\{T'|g_j \text{ edge of } T'\}} n_{T',g_j}\right\rangle \\ + \frac{\pi^2}{4}\sum_{\substack{x_i \text{ vertex}\\ \text{of } \partial\mathscr{T}}}\sum_{\substack{y_j \text{ vertex}\\ \text{of } \partial\mathscr{T}'}} k_p(x_i,y_j)\left\langle A_i, B_j\right\rangle\end{aligned} \tag{8}$$

where $A_i = \sum_k f_k^i/|f_k^i|$ is the sum of the normalized edges of the border, with x_i as vertex, and oriented outward from x_i, and n_{T_i,f_i} is the normal vector of the triangle T_i such that $n_{T_i,f_i} \times f_i$ is oriented inward for the triangle T.

Proof. See appendix. □

This can be re-written:

$$\begin{aligned}\left\langle N(\mathscr{T}), N(\mathscr{T}')\right\rangle_{W'} = \frac{\pi^2}{4}\sum_{i=1}^{n_e}\sum_{j=1}^{m_e} k_p(c_i,d_j)\left\langle f_i, g_j\right\rangle \\ \times\left\langle \sum_{\{T|f_i \text{ edge of } T\}} n_{T,f_i}, \sum_{\{T'|g_j \text{ edge of } T'\}} n_{T',g_j}\right\rangle \\ + \left\langle N(\partial\mathscr{T}), N(\partial\mathscr{T}')\right\rangle_{W'}\end{aligned} \tag{9}$$

The expression $\langle N(\partial\mathscr{T}), N(\partial\mathscr{T}')\rangle_{W'}$ is exactly the scalar product of the curves $\partial\mathscr{T}$ and $\partial\mathscr{T}'$ that has been computed in [42]. Notice that the planar part and the spherical part are not involved in this scalar product (except for the spherical part of the border).

Some remarks: first, we recall that the previous expression does not necessitate a coherent orientation for the mesh. Secondly, even with a constant kernel k_n for the normal part, the metric is sensitive to curvature. Indeed, for an edge f, the cylindrical part of the scalar product involves scalar products between normal vectors of the adjacent triangles which are required quantities to compute the discrete mean curvature. Another interesting feature to notice is that the scalar product involves a specific term for the boundary which will enforce the matching of the boundaries of the shapes. The fact that the boundary has a special behaviour for the normal cycle metric is not surprising. Indeed a normal cycle encodes generalized curvature information of the shape. Hence, the boundary corresponds to a singularity of the curvature and has a specific behaviour in the kernel metric. We will see that this feature is of interest for a matching purpose.

In term of computational complexity, we see in (8) that the model of normal cycles on surfaces is more sophisticated than varifolds (see sect. 4.4 for a recall on varifolds), even with a constant normal kernel. The scalar product involves a double loop on the edges of the triangulations, as well as for

each edge, the computation of the sum of the normal vector of the adjacent triangles. However, it is the same order of complexity as varifolds for the computation of the scalar product, i.e. $O(n_e^2)$ where n_e is the number of edges which is often the same order as the number of triangles.

4.3 Discrete scalar product with linear normal kernel

Now, we focus on the linear normal kernel, $k_n(u,v) = \langle u, v \rangle$.

Proposition 4. *Suppose that $\mathscr{T}$ and $\mathscr{T}'$ are two triangulated meshes. The scalar product between the associated normal cycles with spatial kernel k_p and linear normal kernel $k_n(u,v) = \langle u, v \rangle$ is*

$$\begin{aligned}\left\langle N(\mathscr{T}), N(\mathscr{T}') \right\rangle_{W'} &= 4 \sum_{i=1}^{N} \sum_{j=1}^{M} k_p(b_i, b'_j) |T_i| |T'_j| \left\langle n_{T_i}, n_{T'_j} \right\rangle^2 \\ &+ \frac{4}{3} \sum_{k=1}^{N_v} \sum_{l=1}^{M_v} k_p(x_k, y_l) G_{\mathscr{T}}(x_k) G_{\mathscr{T}'}(y_l) \\ &+ \textit{second order spherical terms}\end{aligned}$$

where

$$\begin{cases} G_{\mathscr{T}}(x_k) = \left[\pi(2 - n_{x_k} + N_{x_k}) - \sum_{i=1}^{N_{x_k}} \varphi_{i,x_k} \right] \\ G_{\mathscr{T}'}(y_l) = \left[\pi(2 - m_{y_l} + N_{y_l}) - \sum_{j=1}^{M_{y_k}} \varphi_{j,y_l} \right] \end{cases}$$

with N_{x_k} the number of triangles with vertex x_k and m_{x_k} the number of edges with vertex x_k, and φ_{i,x_k} is the angle at vertex x_k of the triangle T_i.

Proof. See appendix, the second order spherical terms are also explained. □

Now, suppose for sake of simplicity that the two triangulated meshes have no border and no branching edge or vertex. Then, it is easy to see that for every vertex x of the triangulations $N_x = m_x$ and then, $G_{\mathscr{T}}(x)$ is the discrete Gaussian curvature of the triangulation $\mathscr{T}$ at vertex x. The scalar product is then a classical varifold scalar product, with an additional measure term, located at the vertices, and with intensity equal to the discrete Gaussian curvature.

We remind that we put this linear term to be complete, but that for the moment we do not present any results with such a metric. There is indeed not yet a satisfactory implementation.

4.4 Comparison with kernel metrics on varifolds

This subsection is for comparison purpose. We briefly express the discrete metric on varifolds. For more details, one can see [30].

Rigorously speaking, a 2-varifold in $\mathbb{R}^3$ is a Borel finite measure on $\mathbb{R}^3 \times G_2(\mathbb{R}^3)$ where $G_2(\mathbb{R}^3)$ is the Grassmanian: the space of all unoriented planes of $\mathbb{R}^3$. $G_2(\mathbb{R}^3)$ can be made in correspondence with $\mathbb{R}^3$ by associating to a plane one of its normal vectors.

Now, considering $(x,T) \in \mathbb{R}^3 \times G_2(\mathbb{R}^3)$, we define the *Dirac varifold* $\delta_{(x,T)}$ which is simply the dirac measure located at (x,T).

We can define a scalar product between two such measures using two scalars kernel k_p for the spatial part ($\mathbb{R}^3$), and k_t for the Grassmanian part ($G_2(\mathbb{R}^3)$):

$$\left\langle \delta_{(x,T)}, \delta_{(y,P)} \right\rangle = k_p(x,y) k_t(T,P).$$

Associating T and P with a unit normal vector u and v, and considering $k_t(T,P) = \langle u, v\rangle^2$ which defines a proper reproducing kernel on $G_2(\mathbb{R}^3)$, then we have:

$$\left\langle \delta_{(x,T)}, \delta_{(y,P)} \right\rangle = k_p(x,y) \langle u, v\rangle^2 .$$

Now if we consider a triangulations $\mathscr{T}$, we can approximate $\mathscr{T}$ in the space of varifolds with dirac varifolds located at the barycenter of the triangles, and with unit normal vectors as a unit normal vector of the triangle, and with an area information of the triangles. We denote $\mu_{\mathscr{T}}$ this approximation in the space of varifolds. $\mu_{\mathscr{T}} = \sum_{i=1}^{N} \text{Area}(T_i)\delta_{(b_i, n_{T_i})}$. Then the scalar product between two triangulations writes immediately:

$$\langle \mu_{\mathscr{T}}, \mu_{\mathscr{T}'} \rangle = \sum_{i=1}^{N}\sum_{j=1}^{M} k_p(b_i, b'_j)|T_i||T'_j| \left\langle n_{T_i}, n_{T'_j} \right\rangle^2 \tag{10}$$

Comparing this expression with the one obtained with the linear normal kernel on normal cycles in prop. 4, one can see that the metric on varifolds contains only planar information and thus is not sensitive to curvature. Moreover, it formalizes the fact the model of normal cycles is more complex, and even that the latter encompass the former.

This metric is the one with which we will compare the results in the experiment.

5 Deformation model : LDDMM

Previous sections provide a theoretical framework to compute distance between shapes with kernel metrics on normal cycles. This distance can be used for various applications. In this article, we propose an application for a registration purpose: we use this distance as a residual distance between a deformed shape and a target shape, or in other words, we use this distance as a data-attachment term for an inexact matching problem.

This data attachment term can be fitted in any kind of registration framework. In the following, we will consider diffeomorphic deformations that are generated through flows of regular time varying vector fields. This is the Large Deformation Diffeomorphic Metric Mapping (LDDMM) framework, that we will briefly recall in the following.

After stating an existence result for this problem, we detail in 6.1 a practical algorithm to minimize the functional associated with the inexact matching problem (see eq. (11)) with discrete shapes. The concrete numerical implementation is detailed in the next section. Several examples of registration with normal cycles are proposed on synthetic and real data in the following sections.

As explained in [48], in the LDDMM framework, the study of shape variability is carried by the study of geometrical transformations from one shape to another. The group of deformations at stake, G_V, is generated through integration of time-varying vector fields living in a a Hilbert space V, with $V \hookrightarrow \mathscr{C}_0^1(\mathbb{R}^d)$. With this hypothesis, V is a Reproducing Kernel Hilbert Space with kernel K_V and G_V is endowed with a nice Riemannian structure. For example, the Riemannian distance between the identity and a deformation $\varphi \in G_V$ writes:

$$\begin{cases} d_{G_V}(\text{Id}, \varphi)^2 = E(\varphi) \\ \qquad\qquad := \inf\left\{ \displaystyle\int_0^1 \|v_t\|_V^2 \, dt \,\middle|\, (v_t)_{0\le t\le 1} \in L^2([0,1], V) \right\} \\ \dfrac{\partial \varphi_t}{\partial t} = v_t \circ \varphi_t,\ \varphi_0 = \text{Id},\ \text{and}\ \varphi_1 = \varphi. \end{cases}$$

This distance between the identity and φ can be interpreted as the energy of the deformation φ. Thus, the optimal deformation between two shapes C and S will be the deformation φ with least energy and such that $\varphi(C) = S$. For practical purpose, we can not assume that any two shapes can be

registered with a deformation $\varphi \in G_V$. That is why we relax this hypothesis, and say that the optimal deformation is the one that minimize the sum of the energy and a discrepancy measure between the deformed shape and the target, $A(\varphi(C), S)$. This new registration problem, called inexact matching problem, is a trade-off between the regularity of the deformation, quantified by the energy $E(\varphi)$ and the closeness of the registration, quantified by a term $A(\varphi(C), S)$. The aim of this section is to use kernel metrics on normal cycles for the dissimilarity measure A. Given two shapes C and S

$$A(C,S) := \|N(C) - N(S)\|_{W'}^2 .$$

where W is a RKHS such that $W \hookrightarrow \Omega_0^{d-1}(\mathbb{R}^d \times \mathbb{S}^{d-1})$. The minimization problem with dual Hilbert norm on normal cycles as data attachment term is then:

$$\min_{v \in L_V^2} \gamma \int_0^1 \|v_t\|_V^2 \, dt + \|\varphi^v . N(C) - N(S)\|_{W'}^2 \tag{11}$$

where γ is a trade-off parameter and φ^v is the deformation obtained at time 1 through the flow of $(v_t)_{0 \leq t \leq 1}$.

One should notice that we have defined the action $\varphi.N(C)$ of diffeomorphism on normal cycles for sets $C \in \mathscr{U}_{PR}$ in [42]. This includes smooth submanifolds of $\mathbb{R}^d$, but also polyhedral meshes. This has been studied with more details in [41]. This general framework will be the one we work with in the following.

5.1 Existence of a minimizer

We remind that for smooth submanifold C, we have $\varphi.N(C) = N(\varphi(C))$. For discrete shapes, we refer to [42] for a rigorous definition of such transport.

We now state the theorem of existence of a minimizer for (11) that encompasses both the case of smooth shapes and the one of polyhedral shapes:

Theorem 1 (Existence of a minimizer for (11))**.** *Suppose that C, S are either smooth or discrete shapes and assume that one has the embeddings $V \hookrightarrow \mathscr{C}_0^3(\mathbb{R}^d, \mathbb{R}^d)$, and $W \hookrightarrow \Omega_{1,0}^{d-1}(\mathbb{R}^d \times S^{d-1})$. Then there exists a minimizer for the problem* (11)*.*

The proof of this theorem relies on the weak continuity of $v \in L_V^2 \mapsto \|\varphi^v.N(C) - N(S)\|_{W'}^2$ and is fully proved in [42, 41].

In the following, K_V will be, depending on the application, a Cauchy kernel with width σ_V:

$$K_V(x,y) = \frac{1}{1 + \frac{|x-y|^2}{\sigma_V^2}},$$

a Gaussian kernel with width σ_V:

$$K_V(x,y) = \exp(-\|x-y\|^2 / \sigma_V^2),$$

or a sum of Gaussian kernel with decreasing width. W is generated through the kernels k_p and k_n as in Sect. 4. So that we have existence of a minimizer for (11).

5.2 Discrete framework

Knowing that a minimizer exists is a first step, and we will focus now on the problem of finding such a minimizer.

In the following, we focus on the discrete problem: we consider discrete shapes C_d and S_d. The geodesic equation followed by φ_t^v are simpler and we will explicit the approximations made for the data attachment term in order to have a tractable algorithm for the minimization procedure.

A discrete shape C_d is defined by a set of N points $(x_i)_{1\leq i\leq N}$ in $\mathbb{R}^d$ (the vertices), with a connectivity matrix describing the connexion between the vertices. This applies for curves in $\mathbb{R}^3$ but also for any polyhedral shape in $\mathbb{R}^d$. However, we will restrain our problem to curves and surfaces in $\mathbb{R}^d$, and we will use the metric on surfaces seen in sect. 4. The functional to minimize is then:

$$J_1(v) = \gamma \int_0^1 \|v_t\|_V^2 \, dt + \|\varphi_1^v.N(C_d) - N(S_d)\|_{W'}^2 \tag{12}$$

However, $\varphi_1^v.N(C_d)$ is too complex to be implemented numerically. To overcome this difficulty, we approximate the action of φ^v on C_d. For this purpose, we define C_{d,φ^v} as the discrete curve or surface with vertices $(\varphi_1^v(x_i))_{1\leq i\leq N}$ with the same connectivity matrix as C_d. This means that we consider that φ^v induces a displacement of the vertices only, and the displaced vertices are linked with straight lines. From this, we introduce the approximate matching problem, with the functional $\tilde{J}$:

$$\tilde{J}(v) = \gamma \int_0^1 \|v_t\|_V^2 \, dt + \left\|N(C_{d,\varphi^v}) - N(S_d)\right\|_{W'}^2 \tag{13}$$

As shown in [26], if we denote by $q_i(t) = \varphi_t^v(x_i)$ the points trajectories, the energy term in (13) enforces the optimal vector field to be a geodesic path and to write

$$v_t = \sum_{i=1}^{N} K_V(\cdot, q_i(t)) p_i(t) \tag{14}$$

where the $p_i(t) \in \mathbb{R}^d$ are auxiliary variables and are called momentum vectors. Further, it was shown in [35] (and detailed in an optimal control point of view in [3]) that the problem can be written in Hamiltonian form: if we denote H_r the reduced Hamiltonian:

$$\begin{aligned} H_r(p(t), q(t)) &= \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} p_j(t)^T K_V(q_i(t), q_j(t)) p_i(t) \\ &= \frac{1}{2} \|v_t\|_V^2 , \end{aligned}$$

q_i and p_i must satisfy coupled geodesic equations which write

$$\begin{cases} \dot{q}_{i,t} = \dfrac{\partial H_r}{\partial p_i} = \displaystyle\sum_{j=1}^{n} K_V(q_{i,t}, q_{j,t}) p_{j,t} \\ \dot{p}_{i,t} = -\dfrac{\partial H_r}{\partial q_i} = -\left(\displaystyle\sum_{j=1}^{n} d_1(K_V(q_{i,t}, q_{j,t}) p_{j,t}) \right)^T p_{i,t}. \end{cases} \tag{15}$$

This Hamiltonian is constant along geodesic path and thus is a function of the initial momenta p_0 and the initial positions q_0. As could be expected, this implies that the optimal velocity vector field v_t in (14) is of constant norm: $\|v_t\|_V^2 = cste = H_r(q_0, p_0)$. Initial positions being fixed, we can consider H_r and further φ^v as function of the p_0 only, and denote it φ^{p_0}. The Hamiltonian formalism reduces the initial problem of minimization on an infinite dimensional Hilbert space V (13) to a minimization on $(\mathbb{R}^d)^N$:

$$\min_{p_0 \in (\mathbb{R}^d)^N} 2\gamma H_r(p_0, q_0) + \left\|N(C_{d,\varphi^{p_0}}) - N(S_d)\right\|_{W'}^2 \tag{16}$$

and where q and p follow the coupled geodesic equations (15). The second term depends only on the position of the final vertices: $(q_i(1))_{1\leq i\leq N} = \left(\varphi_1^{p_0}(q_i(0))\right)_{1\leq i\leq N}$ that we will denote $q(1)$. The data attachment term is then a function of $q(1)$: $g(q(1))$.

$$\min_{p_0 \in (\mathbb{R}^d)^N} J(p_0) := 2\gamma H_r(p_0, q_0) + g(q(1)) \tag{17}$$

with q and p following (15). As said before, g is a measure of the residual dissimilarity between the deformed shape at time 1 with vertices $q(1)$ and the target shape S_d.

6 Numerical implementation

6.1 Registration Algorithm

The functional (17) is explicit using the expressions for the scalar products of normal cycles appearing in $g(q(1))$ and that have been computed in Sect. 4. In order to minimize it depending on the initial momenta, one classically uses a geodesic shooting algorithm [35, 3]. We explain here briefly the heuristic of this aglorithm. In order to compute $\nabla_{p_0} J(p_0)$, we need to compute $\nabla_{p_0} g(q(1))$. However, $g(q(1))$ depends on p_0 through the integration of the geodesic equations (eq. (15)). Hopefully, we have explicitly access to $\nabla_{q(1)} g(q(1))$, and starting with this gradient, we obtain $\nabla_{p_0} g(q(1))$ through *backward integration* of the linearized geodesic equations which are recalled in [3], and which involve the Hessian of the Hamiltonian. Integrating these equations from time 1 to time 0, we end up with $\nabla_{p_0} g(q(1))$.

Algorithm 1 Geodesic shooting with fixed-step gradient descent.

Input: q_0 (initial source shape), δ (step size)
Output: $\arg\min_{p_0 \in (\mathbb{R}^d)^n} J(p_0)$
initialization: $p_0 = 0$
while Convergence **do**
  Compute $(q(1), p(1))$ through *forward integration*
  Compute $\nabla_{q(1)} g(q(1))$
  Compute $\nabla_{p_0} g(q(1))$ through *backward integration*
  Compute $\nabla_{p_0} J(p_0) = K_V(q_0, q_0) p_0 + \nabla_{p_0} g(q(1))$
  $p_0 \leftarrow p_0 - \delta \nabla_{p_0} J(p_0)$.
end while

In fact, once we can compute $\nabla_{p_0} J(p_0)$, one can plug any optimization procedure in order to minimize the functional with respect to p_0. We use a quasi Newton Broyden Fletcher Goldfarb Shanno algorithm with limited memory (L-BFGS) [33] rather than the gradient-descent with fixed step presented in Alg. 1. The BFGS agorithm is a quasi-Newton method that computes an approximation of the Hessian, which is updated and improved at each step. With the limited memory implementation of BFGS, there is no storage of a $N \times N$ (where N is the number of variables) matrix and the memory storage is linear with respect to N. See [33] for more details. This method provides a direction of descent and the step in this direction is fixed by a Wolfe line search.

For our numerical implementation, the forward integration scheme is done with a Ralston numerical scheme. This is a higher order discrete ode solver than the classical Euler scheme.

The challenging part of the implementation is the calculation of the gradient, which requires the integration of linearized backward differential equations involving the second order derivatives of the deformation kernel (as briefly explained above and extensively studied in [3]). In the next section we will see how to take advantage of PyTorch's auto-differentiation libraries to automatically evaluate the gradient $\nabla_{p_0} J(p_0)$, without implementing it, and therefore without implementing the backward step in Alg. 1.

6.2 PyTorch and KeOps

Our code for surface registration with normal cycles as data-attachment term is available at `https://proussillon.gitlab.io/en/code/surface-registration-lddmm/projects/`.

We provide a Python implementation using the PyTorch library, together with the KeOps library developed by Benjamin Charlier, Jean Feydy, and Joan Glaunès and freely available at `https://plmlab.math.cnrs.fr/benjamin.charlier/libkeops`.

This allows for a user friendly implementation which performs automatic differentiation (to compute the gradient of the metrics). Moreover, the PyTorch + KeOps library automatically transfers the calculus on GPU (Graphics Processing Unit), with a smart paralellization. All this is briefly detailed in the following. See [9, 8] for more details.

PyTorch is a Python native language, developed by Facebook for neural networks applications. It handles the computation on GPU, allowing for a pain free parallelization. On top of that, PyTorch performs automatic differentiation. Indeed, with the recent development of neural network, there has been an increasing necessity to compute the gradient of loss functions, obtained through elementary operations, linear or non-linear, across different layers. Even though each operation is simple, with an explicit differential, it may be hard to compute the whole gradient (i.e. across all the layers). With PyTorch, it is possible to keep track of the sequence of operations, and to automatically differentiate it through *backward propagation*.

Now, if we look closely at our scalar products of the kernel metrics (e.g. eq. (8)), the typical expressions that we want to evaluate and differentiate are of the form $\gamma_i = \sum_j c_{ij}$ with $c_{ij} = k_p(x_i, y_j)\langle b_i, c_j\rangle$ or $c_{ij} = k_p(x_i, y_j)\langle b_i, c_j\rangle^2$.

The main limitation inherent with PyTorch is that we need to store the matrix $(c_{ij})_{i,j}$, transfer this matrix to the GPU and then perform the computations. However, this matrix is of size N^2 where N is the number of vertices which may be huge. For surfaces with 100 000 vertices, this may exceed the memory capacity of the GPU.

In order to solve this problem, Benjamin Charlier, Jean Feydy, and Joan Glaunès developed the KeOps library which is an interface for a CUDA implementation that computes and automatically differentiate such expressions on the fly, without storing the whole matrix on GPU.

The code provided in this article uses with benefits all the functionality allowed by the coupling PyTorch + CUDA in terms of automatic differentiation and GPU implementation thanks to the KeOps Library.

6.3 Computation time for the data-attachment term and its gradient

We present here indications on the calculation time of the kernel distance with normal cycles, and compare them with those on varifolds. For this purpose, we provide benchmarks of calculation time, for mesh size ranging from 10^1 to 10^6 points. In figure 8 , we present the calculation times for a single evaluation of the distance and its gradient (with respect to the vertices) on the normal cycles and on the varifolds according to the number of points.

In fig. 9, we present the computational time with respect to the number of vertices for the evaluation of the total loss in the LDDMM framework, i.e. eq. (17), with varifolds or normal cycles as data-attachment term. The time needed here is higher because on top of the evaluation of the previous distance, we also need to integrate the geodesic equation and to do backpropagation to compute the gradient.

Let us now make some comments on these results. First of all, it is important to note that the implementation coupling auto-differentiation on PyTorch and parallelization on GPU *via* KeOps provides a method that is fast and scalable to surfaces up to a million points (for the distance evaluation, as well as for the gradient evaluation). Second, even if the normal cycle model is more complex, the computation times are comparable with those of varifolds (from figures 8, it can be said that there is a factor 3 between the two methods). And the difference in computation time is reduced even further

when this distance is integrated into the LDDMM machinery. Indeed, we observe in figure 9 that the calculation times are almost identical with varifolds or normal cycles. This means that in this context, the time-consuming part of the calculation comes more from the forward and backward integration of geodesic equations than from the evaluation of the data-attachment term and its gradient. This is even if the normal cycle model is more complex.

Note 5. *The integration of the geodesic equations are made using a Ralston's method. It is a Runge-Kutta method of order 2 (an Euler scheme method).*

Note 6. *These results are obtained with a NVIDIA GeForce GTX 1080. With a standard graphic card on a laptop (e.g. a Quadro M1200), the computation time is approximately 8 times higher.*

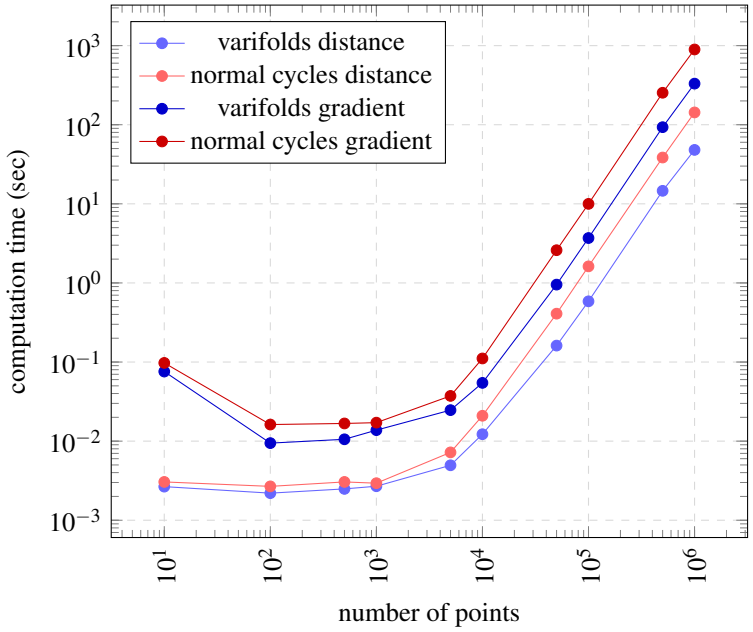


Figure 8: Time needed to compute the distance between two triangulated meshes, as well as its gradient (with respect to the vertices) for varifolds and normal cycles. On the y-axis: time of computation in seconds, on the x-axis : number of vertices of the meshes. The computation is scalable to surfaces up to millions of points. Note that the calculation time is of the same order of magnitude for normal cycles and varifolds. This benchmark is made with a NVIDIA GeForce GTX 1080.

7 Experiments

Let us now move to the experiments of surface matching using LDDMM and kernel metrics on normal cycles. The aim of this section is to illustrate the properties of a matching with normal cycles, as well as some limitations.

For each type of data, the different synthetic examples aim to illustrate the curvatures properties and a comparison with varifolds ([11],[30]) are shown when it is relevant. The examples on real data are a first step to show the advantage of this new dissimilarity metric for applicative purpose.

The algorithm is run until convergence with a stopping criterion on the norm of the successive iterations, with a tolerance of 10^{-6}. Our implementation with the use of GPU allows to perform

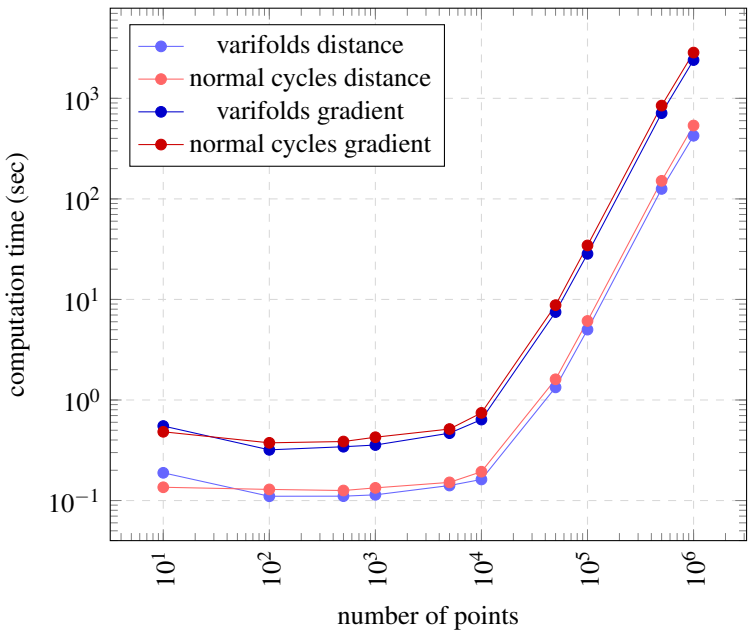


Figure 9: Time needed to compute the total loss of the LDDMM framework (eq. (17)) as well as its gradient (with respect to the initial momenta) for varifolds and normal cycles as data-attachment term. On the y-axis: time of computation in seconds, on the x-axis : number of vertices of the meshes. The computation is scalable to surfaces up to millions of points. Note that the calculation time is of the same order of magnitude for normal cycles and varifolds. This benchmark is made with a NVIDIA GeForce GTX 1080.

matching of surfaces with 10 000 points in a reasonable time, which will be specified for each experimentation.

For all the following matchings, the geometric kernel k_p is a Gaussian kernel of width σ_W, k_n is constant kernel as in sect. 4.2. The kernel K_V is a sum of 4 Gaussian kernels of decreasing sizes, in order to capture different features of the deformation. The trade-off parameter γ is fixed at 0.01 for all the experiments.

7.1 Synthetic data: illustration of the curvature properties

7.1.1 Registration of an ellipsoid to a duck.

Let us start with the simple, yet interesting example of fig. 10. We want to perform a matching between an ellipsoid (the source shape, in blue) and a duck shape (the target, in orange). The duck shape contains 2000 points and the ellipsoid 10 000.

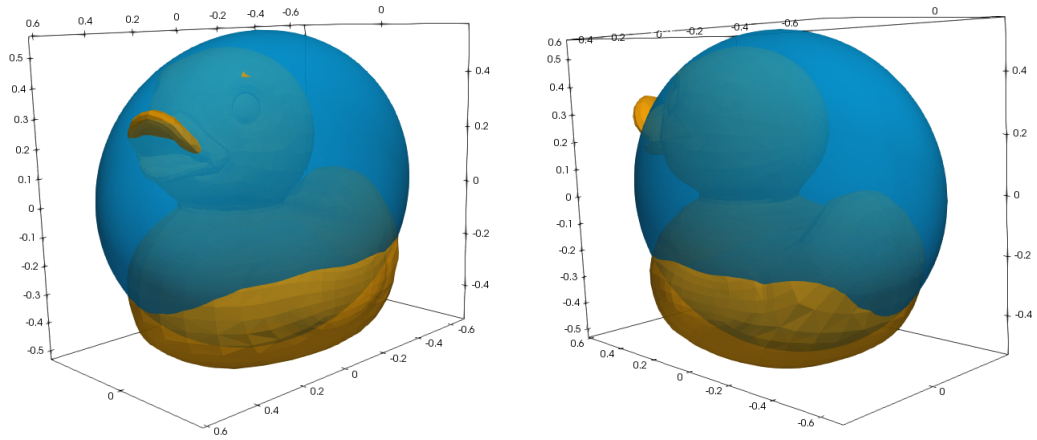


Figure 10: Two views of the matching problem of an ellipsoid to an duck.

The registration is performed with normal cycles and varifolds. We chose a Gaussian kernel for the spatial kernel and a sum of 4 Gaussian kernels of decreasing size ($\sigma_V = 0.2, 0.1, 0.05, 0.025$) for the deformation kernel k_V. We recall that for normal cycles, we chose a constant normal kernel and for varifolds, a linear kernel on the Grassmanian. One run is performed at size $\sigma_W = 0.075$ for spatial kernel. For normal cycles the 1000 iterations were made in 1018 seconds (1s/it). For varifolds, the run ended in 890 seconds (0.9s/it). These computation times were obtained with a Nvidia GeForce GTX 1080. The registrations can be found in fig. 11. As expected, the matching with normal cycles is more accurate that the one with varifolds. This appears clearly in the neighbourhood of regions with high curvature as the beak or the eyes. It is interesting to notice that even the coarse mesh of the duck appears in the deformed ellipsoid for normal cycles.

For validation purposes, in order to have a measurement of the closeness between the deformed shape and the target, and compare the different registrations, we computed the Hausdorff distance and a Root Mean Square (RMS) Hausdorff distance between two surfaces S and S', defined as:

$$
\begin{aligned}
d(S,S') &= \max\left(\sup_{x\in S} d(x,S'), \sup_{y\in S'} d(y,S)\right), \\
d_{RMS}(S,S') &= \sqrt{\frac{1}{\mathscr{H}^2(S)}\int_S d(x,S')^2 d\mathscr{H}^2(x)} \\
&\quad + \sqrt{\frac{1}{\mathscr{H}^2(S')}\int_{S'} d(y,S)^2 d\mathscr{H}^2(y)}.
\end{aligned} \tag{18}
$$

To compute these distances, we used the MeshLab software ([14]). In practice these quantities were approximated by subsampling meshes and evaluating all pairwise distances between vertices. Note that for an interpretation purpose, we also renormalized these distances with the typical size of the data (i.e. the diagonal of the box containing the data). The reader can find the distances on the figures for each experiment.

Figure 11: Registration of a blue ellipsoid to an orange duck. The left column represents the matching with normal cycles and the right column the one with varifolds. The registration with normal cycles is more accurate as it can be seen with the beak or the eyes. One can even notice that the coarse mesh of the duck appears in the deformed ellispoid.
Hausdorff distance with normal cycles: $d = 0.015, d_{RMS} = 0.004$. Hausdorff distance with varifolds: $d = 0.021, d_{RMS} = 0.005$. See eq. (18) for the definition of d and d_{RMS}.

Beyond the precision of the matching, we want to insist on the stability of the results in relation to the parameters. Unlike varifolds, where precise parameter adjustment is sometimes required, the result of matching for normal cycles is less crucially dependent on the parameters of the spatial kernel σ_W, as well as the trade-off parameter γ. As an illustration, we can look at Fig. 12,13 which shows the result of the final matching on the duck as a function of the size of σ_W. It is noteworthy that even at very small scale ($\sigma_W = 0.0375$), the metric on normal cycles still contains global information. Indeed, if we compare the results for $\sigma_W = 0.0375$ for normal cycles and varifolds (bottom right in fig. 12, 13), we see that the matching with normal cycles is very accurate, whereas with varifolds, we end up in a poor local minima. On fig. 14,15, we present the same experiment with $\gamma = 0.1$. That is, in the trade-off between regularization and proximity of the matching, the importance of regularization is increased. We can see that this strongly deteriorates the results for varifolds, whereas it remains fully satisfactory for normal cycles.

A way to avoid such behaviour for varifolds is to use a coarse to precise scale strategy, by decreasing progressively σ_W. The lower sensitivity of the metric on normal cycles with respect to the

parameters allows us to avoid this costly step in time and adjustment.

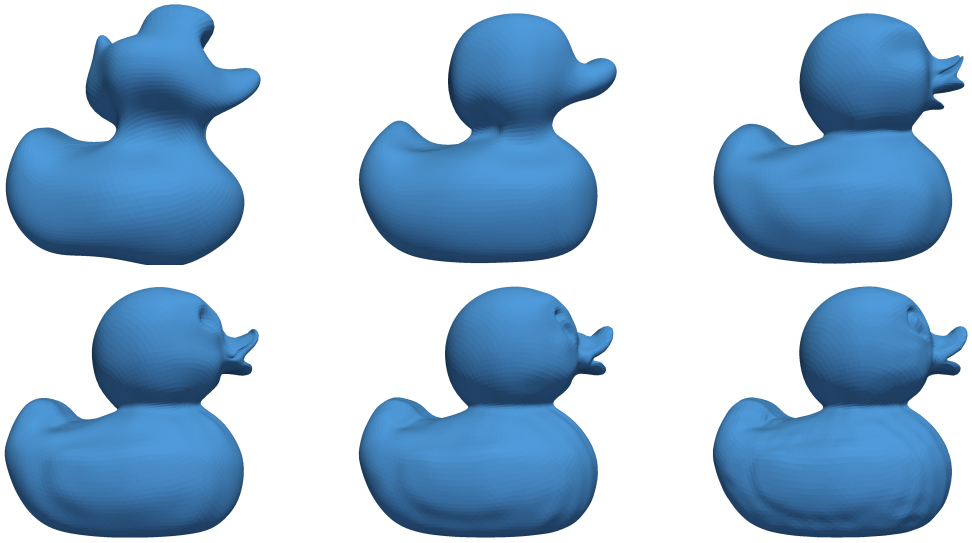


Figure 12: Sensitivity to the size of the kernel σ_W, normal cycles, $\gamma = 0.01$. We show here the final matching of the blue ellipsoid to the orange duck with normal cycles, depending on σ_W. σ_W decreases from top left to right bottom, respectively $\sigma_W = 1.2, 0.6, 0.3, 0.15, 0.075, 0.0375$.

7.1.2 Registration of an ellipsoid to a hippopotamus.

Now let us move to an other example (fig. 16). We want to perform a matching between an ellipsoid (source shape, in blue) and a hippopotamus shape (target, in orange). The hippopotamus contains 20000 points and the ellipsoid 10 000. Notice that the size of the mesh of the hippopotamus is curvature-dependant (coarse near flat regions, precise around region with high curvature). This feature will be of importance to compare the matching results.

The hippopotamus fits in a box of size $52 \times 26 \times 16$. The registration was performed with normal cycles and varifolds. We chose a Gaussian kernel for the spatial kernel and a sum of 4 Gaussian kernels of decreasing size ($\sigma_V = 10, 5, 2.5, 1.25$) for the deformation kernel k_V. We recall that for normal cycles, we chose a constant normal kernel and for varifolds, a linear kernel on the Grassmanian. Three runs were performed, one at size $\sigma_W = 10$, one at size $\sigma_W = 5$ and the last one at size $\sigma_W = 2$. The first runs can be seen as initialization procedures in order to avoid local minima. In this example these runs are compulsory since the target has fine details (located in the head), but also coarse features, such as the legs.

For normal cycles, each run was stopped after 200 iterations, for a total time of 660 seconds (1,1 s/it). The same number of iterations was used for varifolds, for a total time of 600 seconds (1 s/it). The registrations are depicted in fig. 17. The matching with normal cycles is satisfactory considering the difficulty of the registration. Not all details are retrieved in the head, but this could be achieved with another run at smaller scale.

The result obtained with varifolds may seem surprising at first. If the registration is good near the head of the hippopotamus, it is much worse on the body. On this region, one can observe pinches of the deformed shape, whereas the target is flat. In order to understand this behaviour, the figure 18 is illuminating. On this figure, one can observe the superimposition of the target mesh and the final matching mesh. The first thing to notice is that for the varifolds (right column), the deformation tends to concentrate triangles at different locations (the pinches). And on the last row, one can observe that

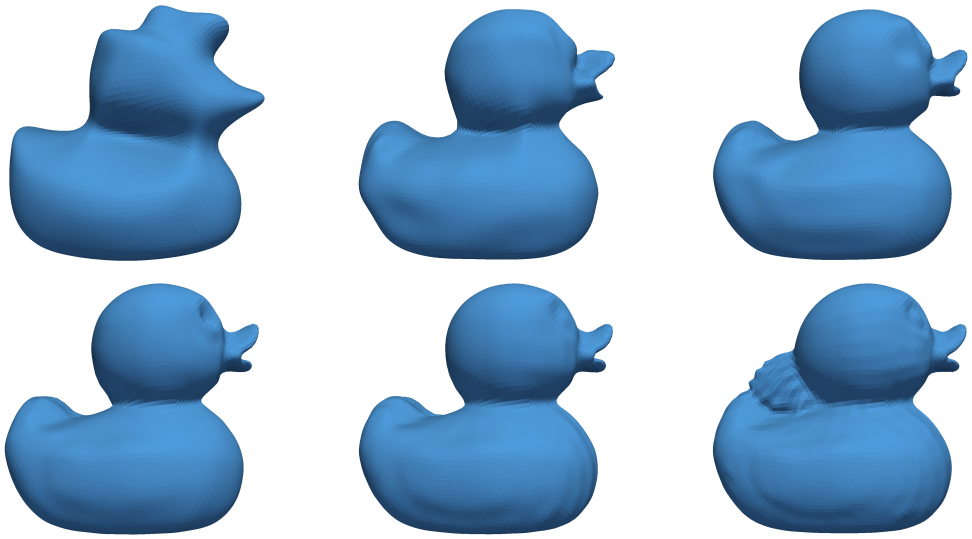


Figure 13: Sensitivity to the size of the kernel σ_W, varifolds, $\gamma = 0.01$. We show here the final matching of the blue ellipsoid to the orange duck with varifolds, depending on σ_W. σ_W decreases from top left to right bottom, respectively $\sigma_W = 1.2, 0.6, 0.3, 0.15, 0.075, 0.0375$.

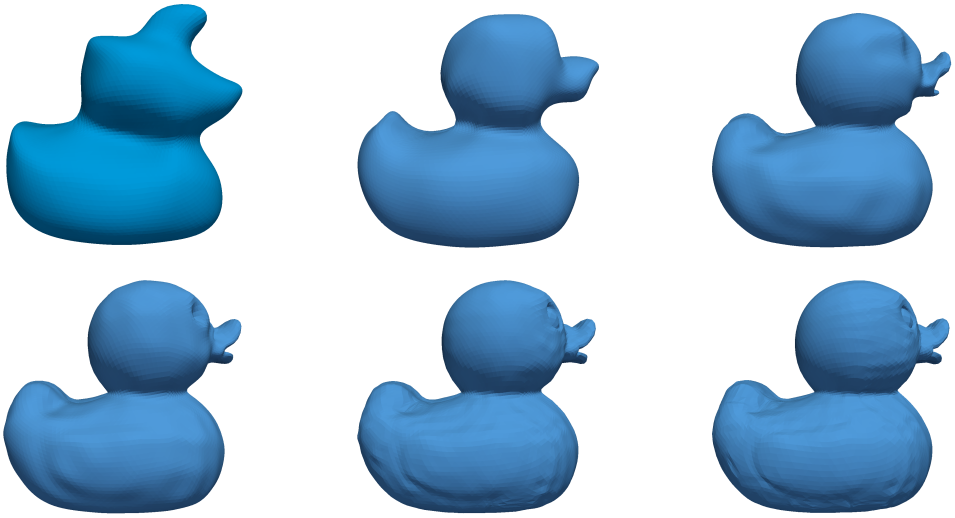


Figure 14: Sensitivity to the size of the kernel σ_W, normal cycles, $\gamma = 0.1$. We show here the final matching of the blue ellipsoid to the orange duck with normal cycles, depending on σ_W. σ_W decreases from top left to right bottom, respectively $\sigma_W = 1.2, 0.6, 0.3, 0.15, 0.075, 0.0375$.

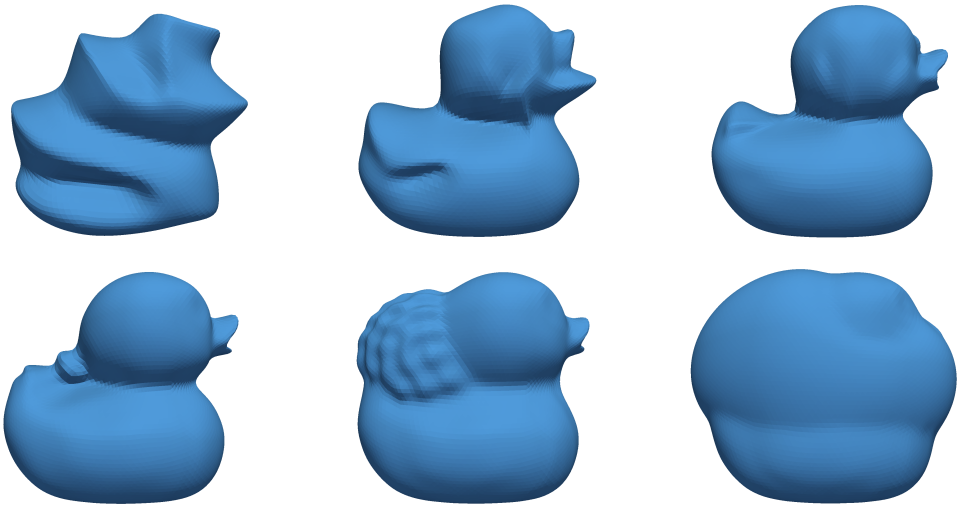


Figure 15: Sensitivity to the size of the kernel σ_W, varifolds, $\gamma = 0.1$. We show here the final matching of the blue ellipsoid to the orange duck with varifolds, depending on σ_W. σ_W decreases from top left to right bottom, respectively $\sigma_W = 1.2, 0.6, 0.3, 0.15, 0.075, 0.0375$.

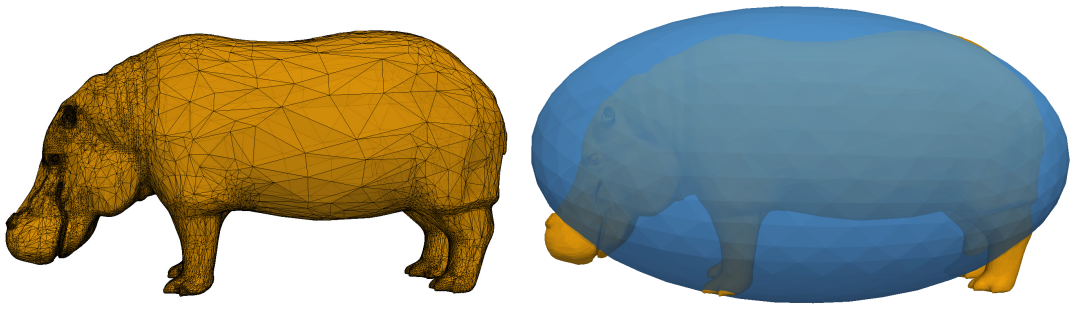


Figure 16: Matching problem between an ellipsoid and a hippopotamus shape. Note that the mesh of the target is coarse at flat regions, and becomes more precise around regions with high curvature.

these locations can be made in correspondence with vertices of the target that are in the middle of big triangles. To explain this phenomena, one needs to remember that in the discrete case, the varifold is approximated as Diracs located in the couple (barycenter of the triangles $\times$ tangent plane of the triangles), with mass equals to the area of the triangle. Thus, one way to reduce the varifold norm is to concentrate the mass of the deformed shape at points far from all the barycenters of the surrounding triangles, i.e. the vertices. This is particularly true at small scales σ_W, and this is why we can observe this behaviour for the varifolds norm. However, the runs at small scales are neccessary in order to fit the details of the head.

For normal cycles, the metric penalizes the difference of curvatures between the source and the target, and one can observe a nice fitting of the flat region with a coarse mesh (fig. 18, left column).

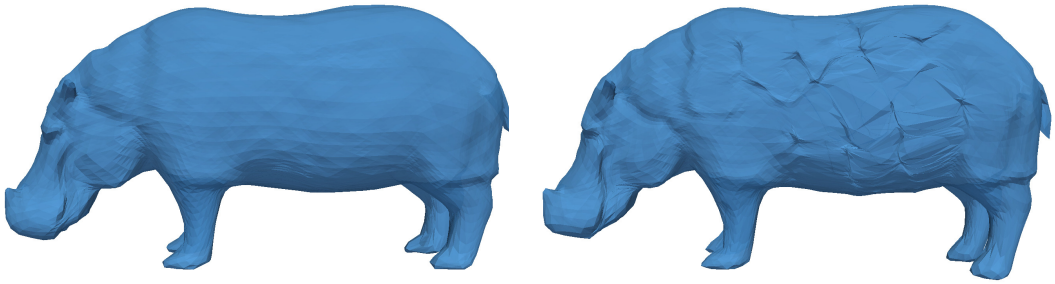


Figure 17: Registration of an ellipsoid (in blue) to a hippopotamus shape (in orange). The left column represents the matching with normal cycles and the right column the one with varifolds. The registration with normal cycles is satisfactory. The one with varifolds is bad near the flat regions of the target.
Hausdorff distance with normal cycles: $d = 0.014, d_{RMS} = 0.0037$. Hausdorff distance with varifolds: $d = 0.024, d_{RMS} = 0.0055$. See eq. (18) for the definition of d and d_{RMS}.

7.1.3 Registration of hippocampi.

The second example is a matching of two human hippocampi, of typical size $10 \times 20 \times 40$. Each shape is around 7000 points. Three runs at different geometric kernel sizes are performed (see Fig. 19). We can see the the final deformation matches well the two hippocampus, even the high curved regions of the shape.

7.1.4 Real data: retinas

This data set was provided by B. Charlier, N. Charon and M.F. Beg is a set of retina layers from different subjects. Originally, each surface comes with a signal that represents the thickness of the retina layers at each vertex. In [31], a statistical analysis of these functional shapes is made using atlas estimation in the framework of LDDMM and with a varifolds kernel metric. We refer to this article for the procedure of generation of this data set. In the following, we only use the geometrical information of the shapes to illustrate the properties of a matching with normal cycles. The difficulty of this example is to perform a matching that is convincing for the interior of the retina, as well as for the boundary. One should notice that the border has no real physical meaning but is the result of the data acquisition. The hole in the center of each retina corresponds to optical nerve. Even though these boundaries are not the interesting part for a medical application, they make the registration harder. We will see that the matching with normal cycles will incorporate the boundaries during the registration, resulting to a much smoother deformation.

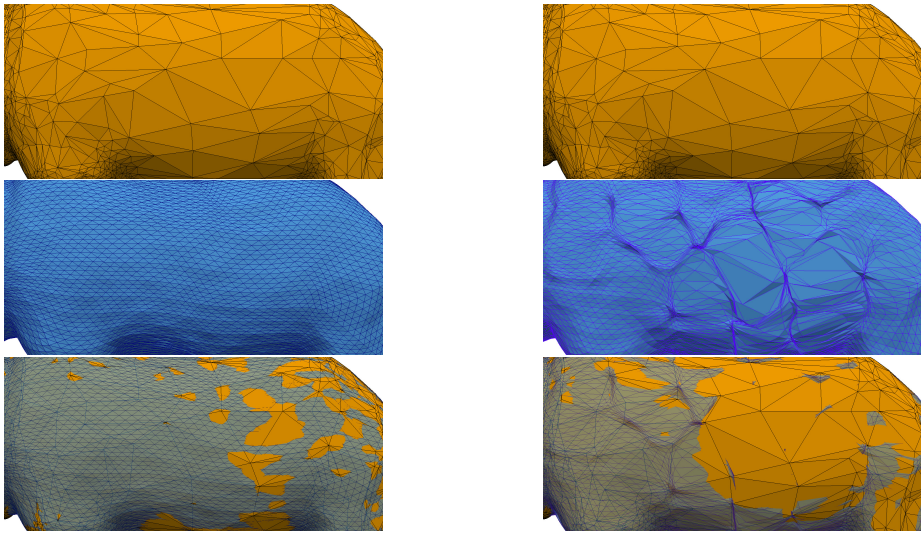


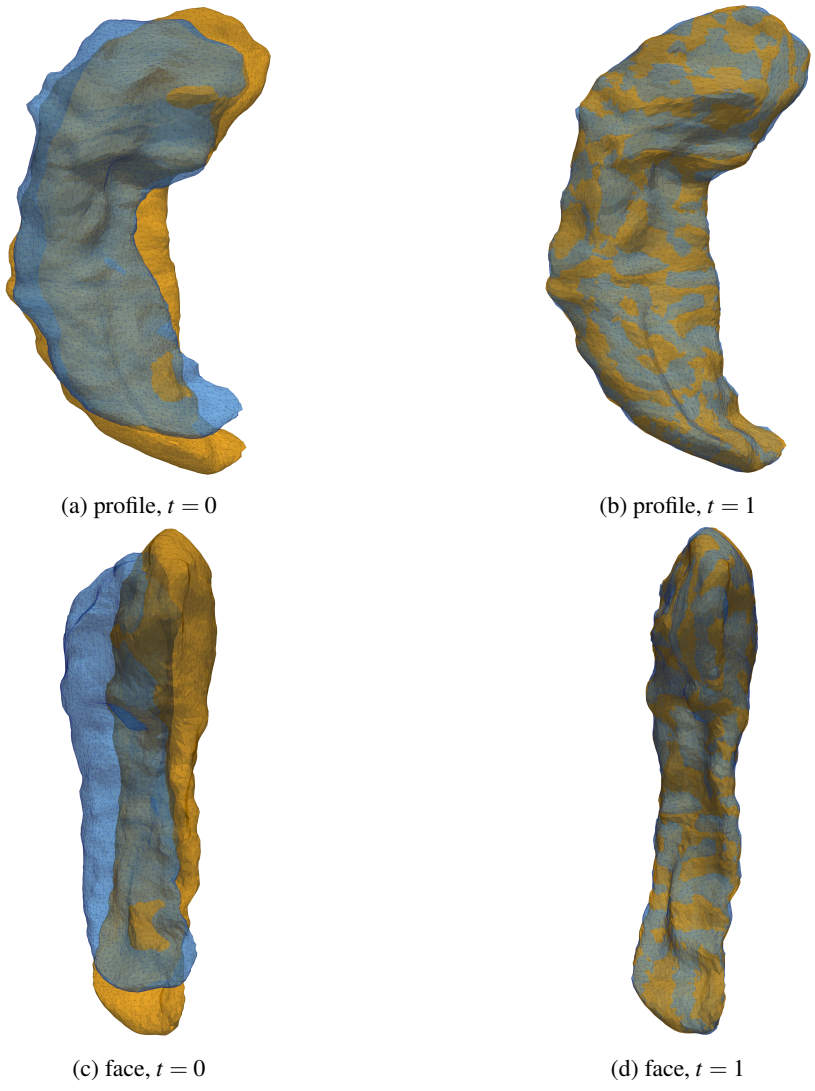
Figure 18: View 1. Registration of an ellipsoid to a hippopotamus (zoom, with the mesh). The left column represents the matching with normal cycles and the right column the one with varifolds. The last row is the overlay of the target and the final resgistration. In order to reduce the varifolds norm between the source and the target at small scales, the deformation concentrates small triangles of the source near vertices of big triangles for the target. This is beacause, with the discrete varifolds setting, the mass is concentrated at the barycenter of the triangles.

The retina are surfaces of typical size 8×8mm. Each retina is sampled with approximately 5000 points. As for hippocampi, three runs were performed, with $\sigma_W = 0.8, 0.4, 0.2$ and the deformation kernel K_V used was a sum of 4 Gaussian kernels, $\sigma_V = 2.4, 1.2, 0.6, 0.3$. All the details of the matching are in Fig. 20. The retinas have a boundary which will be seen as a region with singularities for the kernel metric on normal cycles. This is not the case for the varifolds metric which makes the matching of the corresponding corners harder. The matching of the boundaries is better with normal cycles, and provides a much more regular deformation (see Fig. 20).

In the last example (Fig. 21), the two retinas are the result of an unsatisfactory segmentation . This leads to artifacts in each retina : two triangles for the source retina (in blue, Fig. 21) and only one for the target, in orange. We would like that during the matching, these artificial features are not taken into account. However, these are regions of high curvature and as we could expect, the kernel metric on normal cycles will make a correspondence between those points. As we can see in the second row of Fig. 21, the two triangles are crushed together, into one triangle, even though the cost of the resulting deformation is high. This example shows how sensitive to noise or artifacts normal cycles are. The data must be smooth and well segmented so that the matching works well.

8 Conclusion and perspectives

In this paper, we have used the theory of the shapes representation with normal cycles to define a distance between surfaces. Using reproducing kernels, we are able to construct a distance that becomes completely explicit in the case of triangulated surfaces. In addition, this distance still contains curvature information that is inherent in the normal cycle model, even if the selected kernels are simple. We also proposed an implementation in PyTorch, using both auto-differentiation libraries, and optimized GPU calculations, and with a linear memory footprint (with the KeOps library). Gradient calculations are greatly simplified, despite the complexity of the model (both for normal cycles and



(a) profile, $t = 0$

(b) profile, $t = 1$

(c) face, $t = 0$

(d) face, $t = 1$

Figure 19: Two views (profile and face) at times $t = 0$ and $t = 1$ of the matching of two hippocampi with normal cycles. The target shape is in orange and the source in blue. Each shape has 6600 points. Three runs at different geometric kernel sizes are performed ($\sigma_W = 25, 10, 5$) and the kernel of deformation is a sum of Gaussian kernels with $\sigma_V = 10, 5, 2.5, 1.25$. Each run ended respectively at 72, 100 and 100 iterations for a total time of 565 seconds (2seconds per iteration).

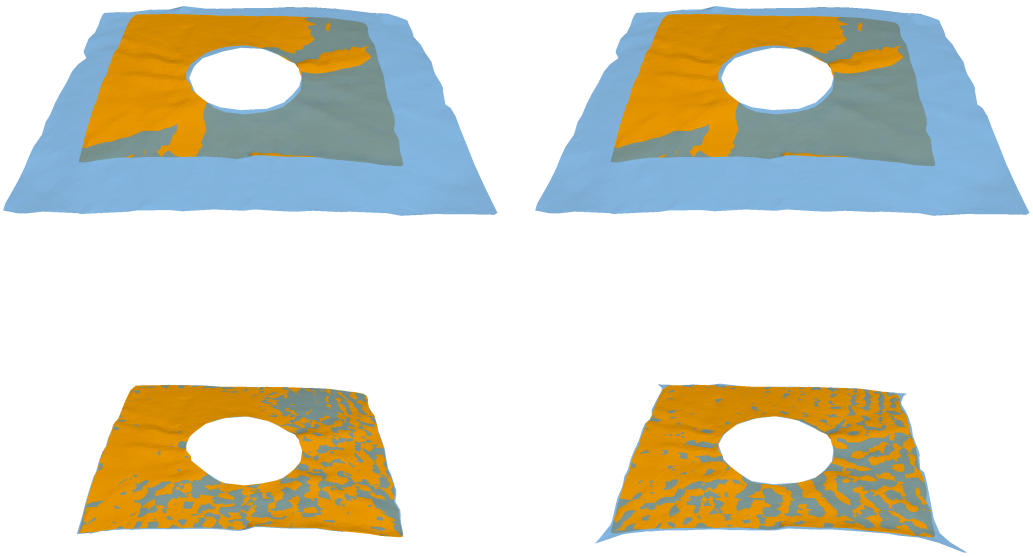
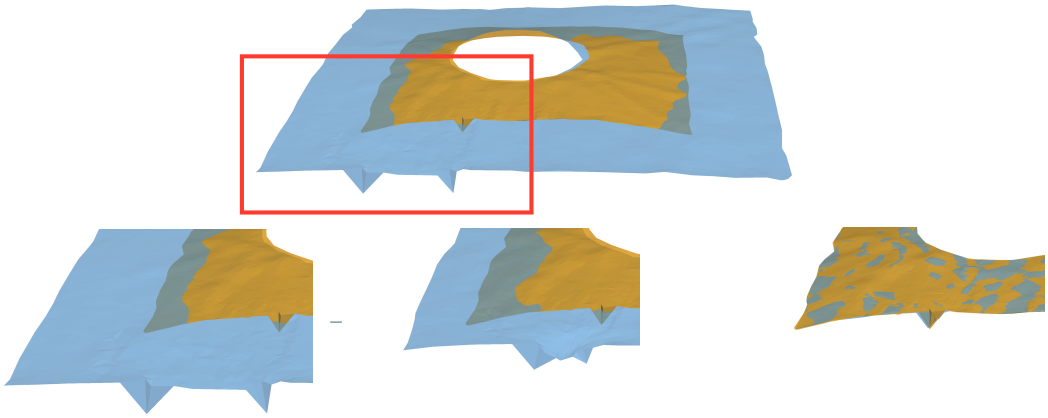


Figure 20: Each column represents the matching of two retinas with kernel metric on normal cycles (left) and varifolds (right). The target shape is in orange and the source shape is in blue. Each shape has 5000 points. For the varifolds metric, the geometric kernel is Gaussian. The kernel on the Grassmanian is chosen linear so that no additional parameter is involved. The same parameters are used for each data attachment term. Three runs at different geometric kernel sizes are performed ($\sigma_W = 0.8, 0.4, 0.2$). K_V is a sum of Gaussian kernels with $\sigma_V = 2.4, 1.2, 0.6, 0.3$. For normal cycles, the registration algorithm took 1000 seconds (0.5 seconds per iteration).
Hausdorff distance with normal cycles: $d = 0.1997, d_{RMS} = 0.0017$. Hausdorff distance with varifolds: $d = 0.2086, d_{RMS} = 0.0036$. See eq. (18) for the definition of d and d_{RMS}.



(a) $t = 0$ (b) $t = 0.5$ (c) $t = 1$

Figure 21: Matching of two retinas with normal cycles : the target is in orange and the source in blue. Three runs at different geometric kernel sizes are performed ($\sigma_W = 0.8, 0.4, 0.2$). K_V is a sum of Gaussian kernels with $\sigma_V = 2.4, 1.2, 0.6, 0.3$. The first row shows the initial configuration. The second row shows the matching in the specific zone delimited by the red rectangle. The metric on normal cycles enforces the matching of corresponding high curvature points, which leads to the alignment of the two triangles into the single one of the target.

for the deformation model).

The examples presented show that the use of normal cycles improves matching results (even when the size of the spatial kernel is large compared to the features to be matched), at a cost that is similar compared to varifolds. We can also highlight the fact that normal cycles take naturally into account the boundary of the shapes. This implies a good matching of the boundary for surfaces. This implies also that normal cycles are sensitive to topological changes, as opposed to currents or varifolds. This may be a drawback if we have uncertainty on the data (for example a poor segmentation that creates artificial holes). However it also allows the use of coarse meshes on regions of low curvature without affecting the registration (see example of fig. 18).

An obvious future work is to develop the computation of the metric in the case of discrete surfaces for non constant k_n, at least for the linear normal kernel. This is of interest since we have already seen that the kernel metric with linear kernel encodes Gaussian curvature information of the surfaces. A normal kernel $k_n(u,v) = 1 + \langle u, v \rangle$ would thus contains all the curvatures informations. However, the calculus remains intricate and this may lead to another approach: to find interesting compact approximation of the spherical part of a normal cycle. Indeed in our discretization strategy, replacing the spherical part of the normal cycle by Dirac masses as we do for the spatial part would not be directly possible, as it would not guarantee convergence of the discrete model to a continuous version when the size of the mesh goes to zero. In place of that we decided not to approximate this part, which leads to heavy computations. Finding efficient approximations of integrations over the spherical part of the normal bundle would be of great interest.

We believe also that kernel metrics on normal cycles can prove useful outside the LDDMM framework, in the spirit of [15, 16, 12, 13], where the authors study the curvature information of a smooth surface that one could retrieve from a surface approximation using normal cycles. The estimation of the mean curvature of a surface from a points cloud approximation has also been done using the first variation of varifolds in [5].The advantage of our setting is that it provides a Hilbert space W' where all the representation of shapes lives, and it might be possible to obtain convergence rate of

the approximation on W' and retrieve information on the curvatures convergence. Of course these are only guess for now, and we need to work further on this direction.

The theoretical study of the link between the kernels defined on the normal cycles and the information of curvatures that we retrieve is also an aspect that we would like to study. Going further in this direction, it seems that we can formalize a precise link between normal cycles and varifolds. It is obvious from the expression of prop. 4 that with the projection on the planar space of the discrete scalar product we retrieve the scalar product on varifolds. We would like to investigate this projection independently of the metric.

A Discrete scalar product with the constant kernel

A.1 Discrete surfaces

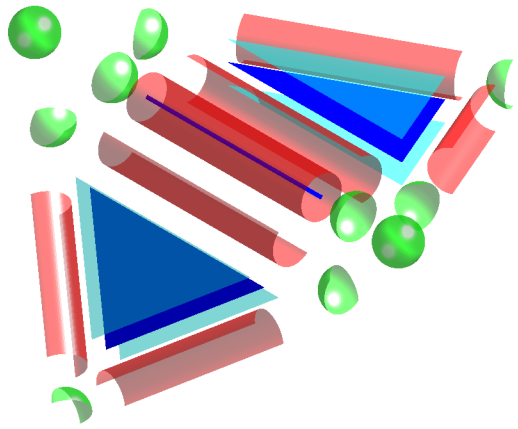


Figure 22: Decomposition of the normal bundle for two triangles with a common edge. In this figure, the two normal bundle of the open triangles appear. Then, we add (only once) the normal bundle of the open edge (the red cylinder and the two green half spheres). Then we add (only once) the normal bundle of the vertices of the edge (the two green spheres).

For discrete surfaces, and with the constant normal kernel, it can be easily seen that the planar part is not involved. The scalar product of normal cycles above vertices (i.e. the spherical scalar product) involves terms as:

$$\begin{aligned}
&k_p(x,y)\int_{S_1}\int_{S_2} k_n(u,v)\left\langle \tau_{\mathcal{N}_C}(x,u),\tau_{\mathcal{N}_C}(y,v)\right\rangle d\mathcal{H}^2(u)d\mathcal{H}^2(v)\\
&= k_p(x,y)\left\langle \int_{S_1} \underbrace{\tau_{\mathcal{N}_C}(x,u)}_{=u \text{ for the spherical part}} d\mathcal{H}^2(u), \int_{S_2}\tau_{\mathcal{N}_C}(y,v)d\mathcal{H}^2(v)\right\rangle\\
&= k_p(x,y)\left\langle \int_{S_1} u d\mathcal{H}^2(u), \int_{S_2} v d\mathcal{H}^2(v)\right\rangle
\end{aligned}$$

If we focus on portion of sphere, one can show that if

$$S_1 = \left\{ \begin{pmatrix} s\theta_u c\varphi_u \\ s\theta_u s\varphi_u \\ c\theta_u \end{pmatrix} \middle| \theta_u \in [0,\pi],\ \varphi_u \in [0,\varphi_0] \right\}$$

is a portion of sphere, then $\int_{S_1} u d\mathcal{H}^2(u) = \pi \sin\left(\varphi_0/2\right) \begin{pmatrix} \cos(\varphi_0/2) \\ \sin(\varphi_0/2) \\ 0 \end{pmatrix}$. Note that we retrieve the half sphere with $\varphi_0 = \pi$ and the total sphere with $\varphi_0 = 2\pi$. Now, taking into account the orientation, we have to compute for the spherical part:

$$\left\langle x \times \left([s.] - \sum[h.s] + \sum[p.s]\right), y \times \left([s.] - \sum[h.s] + \sum[p.s]\right)\right\rangle_{W'}$$

where s stands for sphere, $h.s$ for half sphere and $p.s$ for portion of spheres. In fact, one can show that for a given vertex, summing the contributions of the sphere, the half spheres (associated with the edges), and the portion of spheres (associated with the triangles), then the spherical part vanishes for a vertex that is not in the border. Moreover, we have the following equality:

$$\left\langle N(C)^{sph}, N(S)^{sph} \right\rangle_{W'} = \langle N(\partial C), N(\partial S) \rangle_{W'}$$

thus, the spherical part is exactly the scalar product of the curves associated with the border, scalar product that have been computed right above.

Now let us focus on the cylindrical part. Using expression (6), with $k_n = 1$, one can see that the scalar product involving a full cylinder is null, and thus, only the half cylinders remains. Consider thus the scalar product between two half cylinders. If we denote $HCyl_1 = [a,b] \times S^{\perp}_{b-a}$, $HCyl_2 = [c,d] \times S^{\perp}_{d-c}$ two half cylinders (where $S^{\perp+}_{b-a,\alpha} = \left\{ u \in \mathbb{S}^2 | \, \langle u, b-a \rangle = 0, \langle u, \alpha \rangle \geq 0 \right\}$ is a half circle), we compute the scalar product in W' between these two half cylinders. With the approximations of (6):

$$\begin{aligned}
&\langle HCyl_1, HCyl_2 \rangle_{W'} \simeq k_p\Big(\frac{a+b}{2}, \frac{c+d}{2}\Big) \langle b-a, d-c \rangle \\
&\times \int_{S^{\perp,+}_{b-a,\alpha}} \int_{S^{\perp}_{d-c,\beta},+} \left\langle \frac{b-a}{|b-a|} \times u, \frac{d-c}{|d-c|} \times v \right\rangle d\mathscr{H}^1(u) d\mathscr{H}^1(v) \\
&\simeq k_p\Big(\frac{a+b}{2}, \frac{c+d}{2}\Big) \langle b-a, d-c \rangle \\
&\times \left\langle \frac{b-a}{|b-a|} \times \int_{S^{\perp,+}_{b-a,\alpha}} u d\mathscr{H}^1(u), \frac{d-c}{|d-c|} \times \int_{S^{\perp}_{d-c,\beta},+} v d\mathscr{H}^1(v) \right\rangle \\
&\simeq \frac{\pi^2}{4} k_p\Big(\frac{a+b}{2}, \frac{c+d}{2}\Big) \langle b-a, d-c \rangle \left\langle \frac{b-a}{|b-a|} \times \alpha, \frac{d-c}{|d-c|} \times \beta \right\rangle
\end{aligned}$$

In a triangle T, $[a,b]$ corresponds to an edge and α corresponds to a unitary vector orthogonal to $[a,b]$, in the plane defined by the triangle and oriented in the interior of the triangle. Finally, if we consider two triangulations $\mathscr{T}$ and $\mathscr{T}'$, we have:

$$\begin{aligned}
&\left\langle N(\mathscr{T}), N(\mathscr{T}') \right\rangle_{W'} = \left\langle N(\mathscr{T})^{cyl}, N(\mathscr{T}')^{cyl} \right\rangle_{W'} + \left\langle N(\partial\mathscr{T}), N(\partial\mathscr{T}') \right\rangle_{W'} \\
&= \frac{\pi^2}{4} \sum_{i=1}^{n_e} \sum_{j=1}^{m_e} k_p(c_i, d_j) \langle f_i, g_j \rangle \left\langle \sum_{\substack{T_i \text{ triangles} \\ \text{with edge } f_i}} n_{T_i,f_i}, \sum_{\substack{T'_j \text{ triangles} \\ \text{with edge } g_j}} n_{T'_j,g_j} \right\rangle \qquad (19) \\
&+ \left\langle N(\partial\mathscr{T}), N(\partial\mathscr{T}') \right\rangle_{W'}
\end{aligned}$$

where n_{T_i,f_i} is the normal vector of the triangle T_i such that $n_{T_i,f_i} \times f_i$ is oriented inward for the triangle T.

For the boundary, $\langle N(\partial\mathscr{T}), N(\partial\mathscr{T}') \rangle_{W'}$, we can show similarly that:

$$\left\langle N(\partial\mathscr{T}), N(\partial\mathscr{T}') \right\rangle_{W'} = \frac{\pi^2}{4} \sum_{x_k \in \partial\mathscr{T}} \sum_{y_l \in \partial\mathscr{T}'} k_p(x_k, y_l) \langle A_k, B_l \rangle \qquad (20)$$

where $A_k = \sum_i f_i^k / |f_i^k|$ is the sum of the normalized edges of the boundary with x_k as vertex, and oriented outward from x_k.

B Discrete scalar product with linear normal kernel

We can show that only the planar and the spherical parts are involved in this scalar product. For the planar part, we have already seen that

$$\left\langle N(T)^{pln}, N(T')^{pln} \right\rangle_{W'} \simeq 4 \sum_{i=1}^{N} \sum_{j=1}^{M} |T_i||T'_j| k_p(c_i, c'_j) \left\langle n_{T_i}, n_{T'_j} \right\rangle^2.$$

In this appendix, we want to compute explicitely the spherical scalar product for normal cycles, with the linear normal kernel. The generic expression involved is the following integral:

$$\int_{S_1} \int_{S_2} \langle u, v \rangle^2 \, du dv$$

where S_1 and S_2 are two portions of spheres (that may be the whole sphere, half sphere) with no assumption on the relative position of one sphere compared to the other.

To compute this expression, without loss of generality, we can suppose that S_1 is parametrized as follow:

$$S_1 = \left\{ \begin{pmatrix} \sin\theta_u \cos\varphi_u \\ \sin\theta_u \sin\varphi_u \\ \cos\theta_u \end{pmatrix} \;\middle|\; \theta_u \in [0, \pi], \varphi_u \in [0, \varphi_1] \right\}$$

where φ_1 is the apperture angle of the portion of sphere ($\varphi_1 = 2\pi$ for a whole sphere, and π for a half sphere). Suppose that $v = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix}$, then:

$$\int_{S_1} \langle u, v \rangle^2 \, du = \int_0^{\pi} \int_0^{\varphi_1} \left\langle \begin{pmatrix} \sin\theta_u \cos\varphi_u \\ \sin\theta_u \sin\varphi_u \\ \cos\theta_u \end{pmatrix}, \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} \right\rangle^2 \sin\theta_u d\theta_u d\varphi_u$$

which can be made explicit using the fact that

$$\int_0^{\pi} \sin^3\theta_u d\theta_u = \frac{4}{3}, \int_0^{\pi} \sin^2\theta_u \cos\theta_u d\theta_u = 0, \int_0^{\pi} \sin\theta_u \cos^2\theta_u d\theta_u = \frac{2}{3}.$$

Integrating first with respect to θ_u and then to φ_u, we end up with:

$$\int_{S_1} \langle u, v \rangle^2 \, du = \frac{2}{3}\varphi_1 + \frac{1}{3}(v_1^2 - v_2^2)\sin(2\varphi_1) + \frac{4}{3} v_1 v_2 \sin^2(\varphi_1).$$

The quantity of interest is now:

$$\int_{S_1} \int_{S_2} \langle u, v \rangle^2 \, du dv = \int_{S_2} \Big[\frac{2}{3}\varphi_1 + \frac{1}{3}(v_1^2 - v_2^2)\sin(2\varphi_1) + \frac{4}{3} v_1 v_2 \sin^2(\varphi_1) \Big] dv$$

The main limitation is that we do not have an obvious parametrization of S_2 since there is no assumption on the relative disposition of S_1 and S_2. Suppose now that R is the rotation which brings S_2 to S'_2, where

$$S'_2 = \left\{ \begin{pmatrix} \sin\theta_v \cos\varphi_v \\ \sin\theta_v \sin\varphi_v \\ \cos\theta_v \end{pmatrix} \;\middle|\; \theta_v \in [0, \pi], \varphi_v \in [0, \varphi_2] \right\}$$

We have:

$$\int_{S_2}(v_1^2-v_2^2)dv = \int_{S_2}\left(\langle v,e_1\rangle^2-\langle v,e_2\rangle^2\right)dv$$
$$= \int_{RS_2}\left(\left\langle R^{-1}v,e_1\right\rangle^2-\left\langle R^{-1}v,e_2\right\rangle^2\right)dv$$
$$= \int_0^{\pi}\int_0^{\varphi_2}\left(\left\langle\begin{pmatrix}\sin\theta_v\cos\varphi_v\\ \sin\theta_v\sin\varphi_v\\ \cos\theta_v\end{pmatrix},Re_1\right\rangle^2-\left\langle\begin{pmatrix}\sin\theta_v\cos\varphi_v\\ \sin\theta_v\sin\varphi_v\\ \cos\theta_v\end{pmatrix},Re_2\right\rangle^2\right)$$
$$\times\sin\theta_v d\theta_v d\varphi_v$$

This computation is similar to the one of $\int_{S_1}\langle u,v\rangle^2 du$ and we obtain:

$$\int_{S_2}(v_1^2-v_2^2)dv = \frac{1}{3}\left[r_{11}^2-r_{12}^2+r_{22}^2-r_{21}^2\right]\sin(2\varphi_2)+\frac{4}{3}\left[r_{11}r_{21}-r_{12}r_{22}\right]\sin^2(\varphi_2)$$

where $R=(r_{ij})_{1\le i,j\le 3}$. The same reasoning for the term v_1v_2 leads to

$$\int_{S_2}v_1v_2dv = \int_0^{\pi}\int_0^{\varphi_2}\langle v,Re_1\rangle\langle v,Re_2\rangle dv.$$
$$= \frac{1}{3}\left[r_{11}r_{12}-r_{21}r_{22}\right]\sin(2\varphi_2)+\frac{2}{3}\left[r_{11}r_{22}+r_{12}r_{21}\right]\sin^2(\varphi_2)$$

Finally, if we combine all the terms, we obtain:

$$\begin{aligned}\int_{S_1}\int_{S_2}\langle u,v\rangle^2 dudv = &\frac{4}{3}\varphi_1\varphi_2+\frac{1}{9}\left[r_{11}^2-r_{12}^2+r_{22}^2-r_{21}^2\right]\sin(2\varphi_1)\sin(2\varphi_2)\\ &+\frac{4}{9}\left[r_{11}r_{21}+r_{12}r_{22}\right]\sin(2\varphi_1)\sin^2(\varphi_2)\\ &+\frac{4}{9}\left[r_{11}r_{12}-r_{21}r_{22}\right]\sin^2(\varphi_1)\sin(2\varphi_2)\\ &+\frac{8}{9}\left[r_{11}r_{22}+r_{12}r_{21}\right]\sin^2(\varphi_1)\sin^2(\varphi_2)\end{aligned} \tag{21}$$

with

$$r_{11} = \langle e_1,f_1\rangle$$
$$r_{21} = \langle e_1,f_2\rangle = \left\langle e_1,\frac{1}{\sin\varphi_2}(f_{\varphi_2}-\cos\varphi_2 f_1)\right\rangle$$
$$r_{12} = \langle e_2,f_1\rangle = \left\langle \frac{1}{\sin\varphi_1}(e_{\varphi_1}-\cos\varphi_1 e_1),f_1\right\rangle$$
$$r_{22} = \langle e_2,f_2\rangle = \left\langle \frac{1}{\sin\varphi_1}(e_{\varphi_1}-\cos\varphi_1 e_1),\frac{1}{\sin\varphi_2}(f_{\varphi_2}-\cos\varphi_2 f_1)\right\rangle$$

Note 7. *Expression* (21) *simplifies greatly when one of the involved sphere is a half sphere or a sphere. Indeed, all the terms with a sinus vanish, and it remains:* $\int_{S_1}\int_{S_2}\langle u,v\rangle^2 dudv = \frac{4}{3}\varphi_1\varphi_2$.

Note 8. *Note that this computation is useful to compute the scalar product between portions of sphere in the triangulation. For an implementation, remember that the portion of sphere is not defined by the edges of the triangles, say* e_1,e_2, *but by the orthogonals :* $-e_2^{\perp},e_1^{\perp}$.

Now, if we are given two triangulated meshes, we will express one part of the spherical scalar product: at two vertex x and y, as we have done for the constant normal kernel, we need to compute

$$\left\langle x\times\left([s.]-\sum[h.s]+\sum[p.s]\right),y\times\left([s.]-\sum[h.s]+\sum[p.s]\right)\right\rangle_{W'}$$

If we use the previous expressions of $\int_{S_1}\int_{S_2}\langle u,v\rangle^2\,dudv$, ad gathering all the terms $4/3\varphi_1\varphi_2$, we end up with:

$$\left\langle N(\mathscr{T})^{sph},N(\mathscr{T}')^{sph}\right\rangle_{W'}=\frac{4}{3}\sum_{k=1}^{N_v}\sum_{l=1}^{M_v}k_p(x_k,y_l)G_{\mathscr{T}}(x_k)G_{\mathscr{T}'}(y_l)$$
$$+\text{second order spherical terms}$$

where

$$\begin{cases}G_{\mathscr{T}}(x_k)=\left[\pi(2-n_{x_k}+N_{x_k})-\sum_{i=1}^{N_{x_k}}\varphi_{i,x_k}\right]\\ G_{\mathscr{T}'}(y_l)=\left[\pi(2-m_{y_l}+N_{y_l})-\sum_{j=1}^{M_{y_k}}\varphi_{j,y_l}\right]\end{cases}$$

and where the second order spherical terms appear when considering the crossed terms

$$\begin{aligned}&\frac{1}{9}\left[r_{11}^2-r_{12}^2+r_{22}^2-r_{21}^2\right]\sin(2\varphi_1)\sin(2\varphi_2)\\&+\frac{4}{9}\left[r_{11}r_{21}+r_{12}r_{22}\right]\sin(2\varphi_1)\sin^2(\varphi_2)\\&+\frac{4}{9}\left[r_{11}r_{12}-r_{21}r_{22}\right]\sin^2(\varphi_1)\sin(2\varphi_2)\\&+\frac{8}{9}\left[r_{11}r_{22}+r_{12}r_{21}\right]\sin^2(\varphi_1)\sin^2(\varphi_2)\end{aligned}$$

we do not explicit these second order terms since we still not have a satisfying implementation of this metric.

References

[1] Allard, W.: On the first variation of a varifold. Annals of Mathematics **95(3)** (1972)

[2] Almgren, F.: Plateau's Problem: An Invitation to Varifold Geometry. Mathematical Library (1966)

[3] Arguillère, S., Trélat, E., Trouvé, A., Younès, L.: Shape deformation analysis from the optimal control viewpoint. Journal de Mathématiques Pures et Appliquées **104**(1), 139 – 178 (2015)

[4] Aronszajn, N.: Theory of reproducing kernels. Transactions of the American Mathematical Society **68**, 337–404 (1950)

[5] Buet, B., Leonardi, G.P., Masnou, S.: A varifold approach to surface approximation. Archive for Rational Mechanics and Analysis **226**(2), 639–694 (2017)

[6] Carmeli, C., De Vito, E., Toigo, A., Umanit, V.: Vector valued reproducing kernel Hilbert spaces and universality. arXiv:0807.1659 [math] (2008)

[7] Charlier, B., Charon, N., Trouvé, A.: The fshape framework for the variability analysis of functional shapes. Foundations of Computational Mathematics pp. 1–71 (2015). DOI 10.1007/s10208-015-9288-2. URL `http://dx.doi.org/10.1007/s10208-015-9288-2`

[8] Charlier, B., Feydy, J., Glaunès, J.: Kernel operations on the gpu, with autodiff, without memory overflows. URL http://www.kernel-operations.io/. Accessed: 2018-12-21

[9] Charlier, B., Feydy, J., Glaunès, J.: KeOps: Calcul rapide sur GPU dans les espaces à noyaux. In: Proceedings of Journées de Statistique de la SFdS. Paris, France (2018). URL https://toltex.u-ga.fr/users/RCqls/Workshop/jds2018/resumesLongs/subm309.pdf

[10] Charon, N.: Analysis of geometric and functionnal shapes with extension of currents. Application to registration and atlas estimation. Ph.D. thesis, École Normale Supérieure de Cachan (2013)

[11] Charon, N., Trouvé, A.: The varifold representation of nonoriented shapes for diffeomorphic registration. SIAM J. Imaging Sciences **6**(4), 2547–2580 (2013)

[12] Chazal, F., Cohen-Steiner, D., Lieutier, A., Thibert, B.: Stability of curvature measures. CoRR **abs/0812.1390** (2008). URL http://arxiv.org/abs/0812.1390

[13] Chazal, F., Cohen-Steiner, D., Lieutier, A., Thibert, B.: Stability of Curvature Measures. Computer Graphics Forum (2009). DOI 10.1111/j.1467-8659.2009.01525.x

[14] Cignoni, P., Callieri, M., Corsini, M., Dellepiane, M., Ganovelli, F., Ranzuglia, G.: MeshLab: an Open-Source Mesh Processing Tool. In: V. Scarano, R.D. Chiara, U. Erra (eds.) Eurographics Italian Chapter Conference. The Eurographics Association (2008). DOI 10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136

[15] Cohen-Steiner, D., Morvan, J.M.: Restricted Delaunay Triangulations And Normal Cycle. SoCG'03 (2003)

[16] Cohen-Steiner, D., Morvan, J.M.: Second fundamental measure of geometric sets and local approximation of curvatures. J. Differential Geom. **74**(3), 363–394 (2006). DOI 10.4310/jdg/1175266231. URL http://dx.doi.org/10.4310/jdg/1175266231

[17] Csernansky, J., Wang, L., Swank, J., Miller, J., Gado, M., McKeel, D., Miller, M., Morris, J.: Preclinical detection of alzheimer's disease: hippocampal shape and volume predict dementia onset in the elderly. NeuroImage **25**(3), 783 – 792 (2005). DOI https://doi.org/10.1016/j.neuroimage.2004.12.036. URL http://www.sciencedirect.com/science/article/pii/S1053811904007852

[18] Csernansky, J.G., Wang, L., Joshi, S.C., Ratnanather, J.T., Miller, M.I.: Computational anatomy and neuropsychiatric disease: probabilistic assessment of variation and statistical inference of group difference, hemispheric asymmetry, and time-dependent change. NeuroImage **23**(Supplement 1), S56 – S68 (2004). DOI https://doi.org/10.1016/j.neuroimage.2004.07.025. URL http://www.sciencedirect.com/science/article/pii/S1053811904003970. Mathematics in Brain Imaging

[19] Durrleman, S., Allassonnière, S., Joshi, S.: Sparse adaptive parameterization of variability in image ensembles. International Journal of Computer Vision **101**(1), 161–183 (2013)

[20] Durrleman, S., Fillard, P., Pennec, X., Trouv, A., Ayache, N.: Registration, atlas estimation and variability analysis of white matter fiber bundles modeled as currents. NeuroImage **55**(3), 1073 – 1090 (2011). DOI https://doi.org/10.1016/j.neuroimage.2010.11.056. URL http://www.sciencedirect.com/science/article/pii/S105381191001534X

[21] Durrleman, S., Prastawa, M., Charon, N., Korenberg, J.R., Joshi, S., Gerig, G., Trouvé, A.: Morphometry of anatomical shape complexes with dense deformations and sparse parameters. NeuroImage **101**, 35–49 (2014)

[22] Federer, H.: Curvature measures. Trans. Amer. Maths. Soc. **93** (1959)

[23] Federer, H.: Geometric Measure Theory. Springer (1969)

[24] Federer, H., Fleming, W.: Normal and integral currents. Annals of Mathematics **72**, 458–520 (1960)

[25] Feydy, J., CHARLIER, B., Vialard, F.X., Peyré, G.: Optimal Transport for Diffeomorphic Registration. In: MICCAI 2017, Proc. MICCAI 2017. Quebec, Canada (2017). URL `https://hal.archives-ouvertes.fr/hal-01540455`

[26] Glaunès, J.: Transport par difféomorphismes de points, de mesures et de courants pour la comparaison de formes et l'anatomie numérique. Ph.D. thesis, Université Paris 13 (2005)

[27] Glaunès, J., Qiu, A., Miller, M., Younes, L.: Large deformation diffeomorphic metric curve mapping. International Journal of Computer Vision **80**(3), 317–336 (2008). DOI 10.1007/s11263-008-0141-9

[28] Grenander, U., Miller, M.I.: Computational anatomy: An emerging discipline. Q. Appl. Math. **LVI**(4), 617–694 (1998). URL `http://dl.acm.org/citation.cfm?id=309082.309089`

[29] Helm, P.A., Younes, L., Beg, M.F., Ennis, D.B., Leclercq, C., Faris, O.P., McVeigh, E., Kass, D., Miller, M.I., Winslow, R.L.: Evidence of structural remodeling in the dyssynchronous failing heart. Circulation Research **98**(1), 125–132 (2006). DOI 10.1161/01.RES.0000199396.30688.eb. URL `http://circres.ahajournals.org/content/98/1/125`

[30] Kaltenmark, I., Charlier, B., Charon, N.: A general framework for curve and surface comparison and registration with oriented varifolds. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)

[31] Lee, S., Charon, N., Charlier, B., Popuri, K., Lebed, E., Sarunic, M.V., Trouvé, A., Beg, M.F.: Atlas-based shape analysis and classification of retinal optical coherence tomography images using the functional shape (fshape) framework. Medical Image Analysis **35**, 570–581 (2017)

[32] Lee, S., Heisler, M.L., Popuri, K., Charon, N., Charlier, B., Trouvé, A., Mackenzie, P.J., Sarunic, M.V., Beg, M.F.: Age and glaucoma-related characteristics in retinal nerve fiber layer and choroid: Localized morphometrics and visualization using functional shapes registration. Frontiers in Neuroscience **11** (2017)

[33] Liu, D.C., Nocedal, J.: On the limited memory BFGS method for large scale optimization. Mathematical Programming **45**(1-3), 503–528 (1989). DOI 10.1007/BF01589116

[34] Mansi, T., Voigt, I., Leonardi, B., Pennec, X., Durrleman, S., Sermesant, M., Delingette, H., Taylor, A.M., Boudjemline, Y., Pongiglione, G., Ayache, N.: A statistical model for quantification and prediction of cardiac remodelling: Application to tetralogy of fallot. IEEE Transactions on Medical Imaging **30**(9), 1605–1616 (2011). DOI 10.1109/TMI.2011.2135375

[35] Miller, M.I., Trouvé, A., Younes, L.: Geodesic Shooting for Computational Anatomy. Journal of Mathematical Imaging and Vision **24**(2), 209–228 (2006)

[36] Morvan, J.M.: generalized curvatures. Springer (2008)

[37] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch. In: NIPS-W (2017)

[38] Pennec, X.: Intrinsic statistics on riemannian manifolds: Basic tools for geometric measurements. Journal of Mathematical Imaging and Vision **25**(1), 127–154 (2006)

[39] Qiu, A., Younes, L., Miller, M.I., Csernansky, J.G.: Parallel transport in diffeomorphisms distinguishes the time-dependent pattern of hippocampal surface deformation due to healthy aging and the dementia of the alzheimer's type. NeuroImage **40**(1), 68 – 76 (2008). DOI https://doi.org/10.1016/j.neuroimage.2007.11.041. URL http://www.sciencedirect.com/science/article/pii/S1053811907010932

[40] Rataj, J., Zähle, M.: Curvatures and currents for unions of sets with positive reach, ii. Annals of Global Analysis and Geometry **20**(1), 1–21 (2001). URL http://dx.doi.org/10.1023/A:1010624214933

[41] Roussillon, P.: Modèle de cycles normaux pour l'analyse des dformations. Ph.D. thesis, Université Paris Descartes (2017)

[42] Roussillon, P., Glaunès, J.: Kernel metrics on normal cycles and application to curve matching. SIAM J. Imaging Sciences **9**, 1991–2038 (2016)

[43] Roussillon, P., Glaunès, J.: Surface matching using normal cycles (2017). GSI'17: Geometric Science Information, 2017, Paris

[44] Tang, X., Holland, D., Dale, A.M., Younes, L., Miller, M.I., for the Alzheimer's Disease Neuroimaging Initiative: Shape abnormalities of subcortical and ventricular structures in mild cognitive impairment and alzheimer's disease: Detecting, quantifying, and predicting. Human Brain Mapping **35**(8), 3701–3725 (2014). DOI 10.1002/hbm.22431. URL http://dx.doi.org/10.1002/hbm.22431

[45] Thäle, C.: 50 years sets with positive reach, a survey. Surveys in Mathematics and its Applications **3** (2008)

[46] Vaillant, M., Glaunès, J.: Surface Matching via Currents. In: G.E. Christensen, M. Sonka (eds.) Information Processing in Medical Imaging, no. 3565 in Lecture Notes in Computer Science, pp. 381–392. Springer Berlin Heidelberg (2005)

[47] Wang, L., Beg, F., Ratnanather, T., Ceritoglu, C., Younes, L., Morris, J.C., Csernansky, J.G., Miller, M.I.: Large deformation diffeomorphism and momentum based hippocampal shape discrimination in dementia of the alzheimer type. IEEE Transactions on Medical Imaging **26**(4), 462–470 (2007). DOI 10.1109/TMI.2006.887380

[48] Younès, L.: Shapes and Diffeomorphisms. Springer (2010)

[49] Zähle, M.: Integral and current representation of Federer's curvature measure. Arch. Maths. **23**, 557–567 (1986)

[50] Zähle, M.: Curvatures and currents for unions of set with positive reach. Geometriae Dedicata **23**, 155–171 (1987)