Time Consistent Surface Mapping for Deformable Object Shape Control

Ignacio Cuiral-Zueco and Gonzalo López-Nicolás

***Abstract*—Shape control involves deforming objects to achieve a desired shape. One of the main challenges of this task is to define a suitable control reference, especially when 3D objects that lack distinctive visual texture and geometric features are involved. This paper addresses the problem of generating a suitable shape control reference using surface maps for 3D texture-less objects. The proposed surface mapping method is based on functional maps and ensures time consistency with robustness to non-isometries. Our time consistent method is validated within a shape control strategy, where local exponential stability analysis is provided. The effectiveness of the framework is illustrated through simulations and experiments.**

***Note to Practitioners*—We present a method for shape comparison to analyse the geometric similarities between the shape of an object and a desired target shape. The goal is to generate a point map between both surfaces so that, with the use of robots that grasp the object, a control strategy can deform the object towards the desired shape. The point map we compute, while adapting to the changing shape of the deforming object, remains stable enough to enable the shape control task. This can be of use for the automation of processes that involve shape control (e.g., object packaging, moulding, etc.). Our proposed shape control framework combines computer graphics, computer vision, and automation techniques to create a system that is well-suited for industrial setups equipped with commonly used range sensors like RGB-D cameras. Regarding the experiments presented in this paper, controlled lighting conditions are recommendable in order to perform the colour-based object segmentation (e.g. avoidance of abrupt light variations, uniform illumination). Automating the grasping process is not within the scope of this paper, therefore, in each experiment we manually defined the object grasping points according to the shape control task involved. The proposed framework has the potential to increase productivity, reduce costs and, improve safety in hazardous tasks by automating the manipulation of 3D objects that lack distinctive visual texture and geometric features.**

***Index Terms*—Shape control, Robot vision systems, Multiple manipulators, Deformable object manipulation.**

I. Introduction

Robot-based deformable object manipulation is present in a variety of tasks (e.g., deformable object transportation [1], [2], linear object manipulation [3], cable untangling [4], object cutting [5], etc.). These tasks may involve different types of deformable objects which, within the field of robotics, can be classified according to several criteria [6]–[8]. Our goal is to achieve 3D shape control by using robots to manipulate deformable objects into acquiring a desired target shape [9]. At a high level of abstraction, the notion of target shape refers to the shape we want the object to achieve. However, generating a mathematically grounded definition of shape error that covers a broad range of shape control cases is challenging, specially for 3D texture-less objects that present symmetries or lack geometric distinctive features. We propose a shape error formalization that adapts to deformations experienced by the object and remains temporally consistent, and thus constitutes a proper *shape control reference*.

Our approach, illustrated in Fig. 1, involves the use of a 3D sensor (e.g., an RGB-D camera) to perceive the object and generate a mesh of its visible surface. Our proposed time-consistent surface mapping allows for the comparison of the current shape with the reference (target) shape, and generates a surface map that enables us to generate an input reference for the shape control strategy. The shape control strategy generates 6 degrees of freedom (DoF) actions for each robot involved in the manipulation of the object, thus deforming it towards the desired shape.

A. Related work in shape control

Regarding the shape control literature [10], there is a variety of approaches and control reference definitions. Some methods define their control reference with the use of feature points on the object's surface (e.g. [11]–[15]). Method in [14] defines its error reference as the alignment of a set of (provided) points, whereas [13] defines the error reference using the nodes of a planar deformation mesh that is updated by means of visual features. The deformation planning approach in [15] uses visual markers (laser-reflective features) to determine the object's shape. Other methods focus on deformable object transport, this is the case of [1] which uses a rigid Procrustes optimization to define a homogeneous contour point matching between planar shapes. Some approaches like [12] focus on isometric deformations of planar objects with monocular perception by using few feature points and a Shape-from-Template analysis. Methods like [16] tackle both isometric and elastic deformations of planar objects by defining a contour mapping based on a multiscale Fast Marching Method. Strategies like [17]–[19] tackle the manipulation of deformable objects, defining the control strategy by means of (2D) contour moments. However, they are limited to the analysis of the 1D contour (embedded in 2D or 3D) of the object's visible silhouette. Similarly, deformable linear object (DLOs) manipulation methods (e.g., approach in [20]) typically define

I. Cuiral-Zueco and G. López-Nicolás are with Instituto de Investigación en Ingeniería de Aragón, Universidad de Zaragoza, Spain. ignaciocuiral@unizar.es, gonlopez@unizar.es.

This work was supported via project REMAIN S1/1.1/E0111 (Interreg Sudoe Programme, ERDF) and projects PID2021-124137OB-I00 and TED2021-130224B-I00 funded by MCIN/AEI/10.13039/501100011033, by ERDF A way of making Europe and by the European Union NextGenerationEU/PRTR.

This article has been accepted for publication in IEEE Transactions on Automation Science and Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASE.2025.3529180

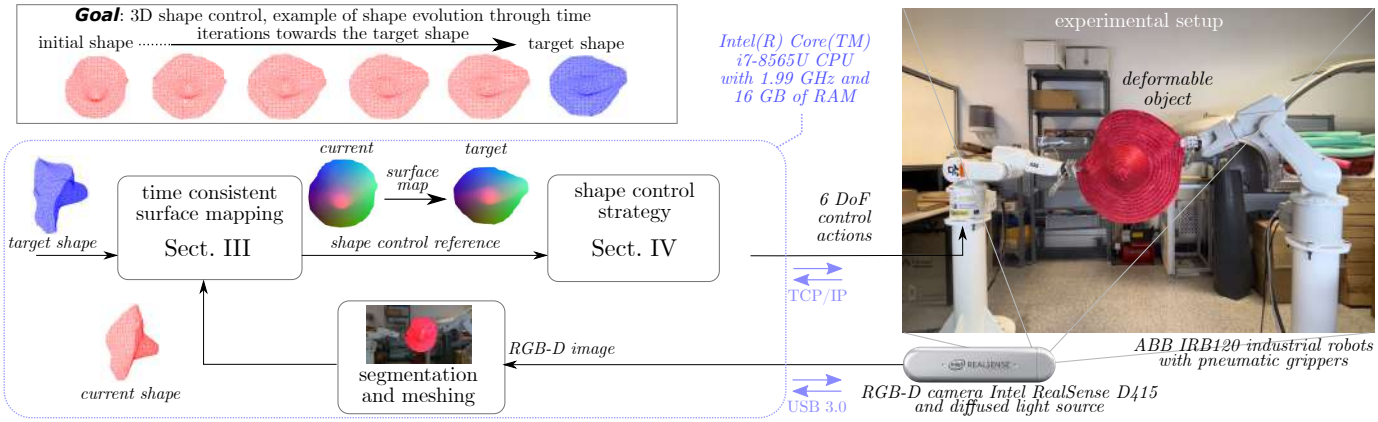


Fig. 1. Overview of the proposed approach. At the top, the main objective of deforming an object into a desired shape is illustrated with a sequence of deformation states (red mesh). A general outline of the proposed shape control framework is shown, and our experimental setup is illustrated on the right side of the figure. A 3D sensor (e.g., an RGB-D camera), provides the RGB-D images from which the object's mesh is retrieved at each iteration. The current object mesh is compared to the target mesh by means of our proposed time consistent surface mapping method. Our main contribution entails generating surface maps that allow defining the shape control reference, thus serving as input for the shape control strategy. A 6 DoF action is defined for each of the robots involved in the manipulation of the object.

shape references by mapping the curves (1D domains) that represent DLOs. For example, the method in [21] matches DLOs through B-spline representations. However, mapping 2D surfaces (embedded in 3D) for deformable object shape control remains a challenge.

B. Related work in surface correspondence

As this paper makes extensive use of functional maps, we provide a specific review of the relevant computer graphics literature. Functional maps, introduced in the shape correspondence context in [22], are a robust and efficient tool for isometric shape surface mapping. Since [22] was published, several proposals have modified and extended the use of functional maps for diverse and demanding challenges such as partial mapping of shapes [23], surface-orientation preserving correspondences [24] or vector-field transfer between surfaces with the use of complex functional maps [25]. Other functional map based methods focus on a coarse-to-fine shape analysis. This is the case of [26], in which smoothed versions of the shape, along with a Markov chain Monte Carlo initialization, allow refining the mapping process at different levels of detail. An interesting alternative is the ZoomOut method, proposed in [27], where a coarse-to-fine analysis is also performed.

C. Problem motivation

Surface mapping defines a continuous function that maps points from one surface to another, based on their geometry. This technique is relevant in computer graphics for producing realistic animations and effects, as it ensures that textures accurately adapt to the shape of an object, even when the object deforms. When applied in synthetic environments, such as simulations and animations, maintaining mapping consistency during surface deformations is straightforward, as the positions of all mesh nodes are always known. However,

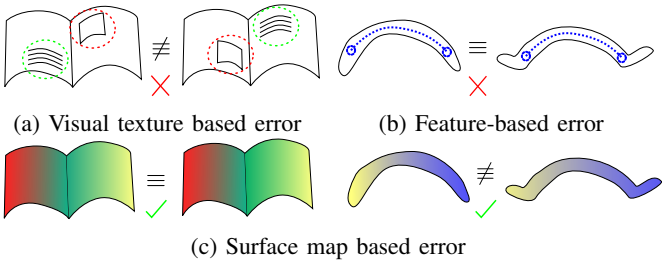


Fig. 2. Three different shape error criteria (a, b and c). (a) shows a visual texture-based criterion in which, although both shapes are the same, their visual descriptors present different positions and lead to non-zero error. (b) shows a feature based shape error definition that does not consider all the object's geometry. The descriptor (two discrete points and a segment's curvature) is identical in both cases and thus leads to zero shape error even-though the shapes are different. (c) involves a geometry based surface map that considers all the object's geometry and does not depend on visual texture, leading to proper shape analysis on both comparison cases.

applying surface mapping to texture-less real-world objects poses considerable challenges (e.g., ground-truth point positions are not known, sensor data may be noisy and incomplete, etc.). Therefore, significant efforts are needed to ensure mapping consistency during surface deformations in real scenarios. We define *time-consistent surface mapping* as the process of computing and updating surface maps so that they adapt to surface deformations, specifically considering surfaces acquired from sensor data. Thus, time consistency expands the applicability of surface mapping to real-world scenarios, closing the gap between synthetic models and real-world practical applications. To name some potential applications of time-consistent surface mapping: shape control of deformable objects in manufacturing processes (tackled in this

paper), quality control in the food industry, object texture-transfer in augmented reality, or surface mapping in real non-rigid environments (e.g., laparoscopies).

A challenging goal in the context of 3D shape control is to define a geometry based holistic shape error. We propose defining such shape error through 3D surface maps computed by means of functional maps (see a brief introduction to functional maps in section II-A). Functional maps can be of great interest in defining a reference for shape control because, in comparison to other shape control reference definitions from the literature, they:

- allow generating geometry based surface maps. Some shape control methods base their shape error on visual texture (e.g., [13]). However, even if an object presents rich visual texture, such texture is not necessarily representative of the object's shape (see Fig. 2.a). Furthermore, repetitive or symmetric visual patterns may lead to ambiguities, further limiting texture-based methods.
- a holistic shape analysis, in which all the object's available geometric information is considered. Existing shape control methods such as [11], [15], [28], [29] define their shape error through a reduced number of features that need to be properly defined beforehand (depending on the object's shape) and could lead to ambiguities (see Fig. 2.b). Thus, such approaches are limited to the amount and variety of available features that the object presents.
- favour isometries and seek maximising curvature resemblance. These two aspects lead to the minimisation of stretching/compression and bending deformation processes.

Note that surface mapping methods from the functional maps literature are not directly applicable to shape control, as it constitutes a relatively different problem. We will now discuss the difficulties and challenges that need to be addressed.

D. Contributions

The following is a summary of the challenges we have addressed and the main contributions of the 3D shape control framework presented in this paper.

*1) We generate **time-consistent functional maps** along iterations:* The functional maps literature focuses on the computation of functional maps between two specific static shapes, whereas shape control requires the consistent computation along iterations of functional maps between an evolving (deforming) shape and a target shape. As the object deforms and acquires different shapes, new minima may appear in the functional map computation process, leading to solutions (functional maps) that may differ greatly from the initially computed functional map.

*2) We compute **consistent surface point tracking during deformation processes**:* Time consistent functional maps alone are not sufficient for generating a proper shape control error: a functional map based shape error would be defined in terms of the shape's Laplace-Beltrami basis and thus be invariant to isometries. That is, if two different shapes constitute an isometry, their bases are identical and thus would lead to zero error. Therefore, it is necessary to define a point-to-point based error, something that would be straightforward in virtual mesh deformations (simulations) as the shape's node ground-truth positions are known. However, in a real setup, a new set of the object's 3D points is acquired in each iteration (with varying number of points and arbitrary index/order). Our method allows to consistently track surface points during the deformation process and thus compute deformation Jacobians and time-consistent point-to-point error vectors that are suitable for shape control.

*3) Our method allows for **computation times that are suitable for industrial use**:* We managed to remain above 5 [Hz] in the computation of the time-consistent surface maps. The functional maps literature prioritises fineness over computational time; they tackle a different problem (mapping between two static virtual surfaces) and face challenging bench-marking in terms of accuracy. Generally, methods analyse highly detailed meshes ($\approx$ 10.000 mesh nodes), use sophisticated shape descriptors (e.g., wave-kernel signature [30]) and solution searching methods (e.g., MCMC initialization [26]) that typically require minutes of processing to obtain refined results. Our framework is based on one of the fastest methods in the computer graphics literature (ZoomOut method [27]) and, without the use of shape descriptors, we achieve a compromise between fineness (≈ 500 mesh nodes), computational time ($\leq 0.2\,[s]$), and robustness.

*4) Our proposed framework is **robust to non-isometries and problems arising from real data**:* Surface mapping methods in the functional maps literature typically analyse mesh datasets with fine and smooth meshes in which target shapes constitute isometries [31], [32]. Surface mapping of large non-isometries is still an ongoing challenge in the computer graphics literature and, in this paper, we do not solve it in a formal and general manner. However, we do achieve robustness of our system to non-isometric deformations. Regarding real data derived problems, a realistic industrial set-up will most likely involve the use of affordable 3D sensors, such as RGB-D cameras, that provide noisy and varying/incomplete data. Aspects like object occlusions (e.g., object self-occlusions), unsuitable sensor position or sensor's limitations (reflections, bad illumination, etc.) can cause sudden variations on the mesh. Our framework is robust to these problems, as it uses information from previous iterations in a coarse-to-fine manner.

II. Functional maps

A. Functional maps: background

Functional maps are widely used to compute point-to-point correspondences between pairs of 2D surfaces (embedded in 3D). In the computer graphics context, they were introduced in [22] for computing point-to-point correspondences between two surfaces (manifolds) $\mathcal{M}$ and $\mathcal{N}$. These surfaces are defined in the discrete setting as triangular meshes, being M the number of nodes in $\mathcal{M}$ and N the number of nodes in $\mathcal{N}$. Consider now the space of all real-valued functions on surfaces $\mathcal{M}$ and $\mathcal{N}$, i.e. $\mathcal{F}(\mathcal{M},\mathbb{R})$ and $\mathcal{F}(\mathcal{N},\mathbb{R})$ respectively, and suppose we can obtain a functional T_F that maps one space onto the other:

$$T_F : \mathcal{F}(\mathcal{M},\mathbb{R}) \rightarrow \mathcal{F}(\mathcal{N},\mathbb{R}). \tag{1}$$

This means that any real-valued function $f : \mathcal{M} \to \mathbb{R}$ can be mapped to an analogous function $g : \mathcal{N} \to \mathbb{R}$ by applying functional T_F as follows:

$$g = T_F(f). \tag{2}$$

Recall, the objective is to obtain a point-to-point matching between $\mathcal{M}$ and $\mathcal{N}$. Therefore, as input for (2), we are interested in a specific kind of function f that allows us to select specific points of surface $\mathcal{M}$ and find their corresponding image in $\mathcal{N}$. We denote a function that selects a point $x_s \in \mathcal{M}$ as: $f_s(x) = 1$ if $x = x_s$ and $f_s(x) = 0$ otherwise. After applying (2), we can obtain the corresponding selector function for $\mathcal{N}$, i.e. g_s, which provides us with the matched point $x_s \in \mathcal{N}$: one just needs to find $x \in \mathcal{N} \mid g_s(x) = 1$ and $g_s(x) = 0$ for the rest of points. The main problem now is how to obtain T_F in (1).

We can consider a function f as a linear combination of infinite basis functions $\{\phi_i^{\mathcal{M}}\}$, i.e. $f = \sum_i^{\infty} a_i \phi_i^{\mathcal{M}}$. Also consider $\{\phi_j^{\mathcal{N}}\}$ as the analogous set of infinite basis functions of g. This decomposition leads to the following definition of T_F (proposed in [22], remark 4.2):

$$T_F(f) = T_F(\sum_i^{\infty} a_i \phi_i^{\mathcal{M}}) = \sum_j^{\infty} \sum_i^{\infty} a_i c_{i,j} \phi_j^{\mathcal{N}}. \tag{3}$$

The functional representation T_F (or functional map, for now on) can be represented in matrix form as

$$T_F(\mathbf{A}) = \mathbf{C}_{\infty}\mathbf{A}, \tag{4}$$

being $\mathbf{C}_{\infty}$ (with elements $c_{i,j} \in \mathbb{R}$) an infinite matrix and $\mathbf{A} \in \mathbb{R}^{\infty}$ the infinite vector of coefficients $a_i \in \mathbb{R}$. The goal is to obtain a finite approximation of $\mathbf{C}_{\infty}$, i.e. $\mathbf{C}$, that allows us to compute a point-to-point map $T : \mathcal{M} \to \mathcal{N}$ between our two shapes' surfaces $\mathcal{M}$ and $\mathcal{N}$.

Before computing $\mathbf{C}$ and T we need to choose a basis $\{\phi_i^{\mathcal{M}}\}$ and $\{\phi_j^{\mathcal{N}}\}$. A specially convenient basis is provided by the Laplace-Beltrami eigenfunctions, as they present proper compactness (i.e. with a few components they provide a good approximation of functions) and stability under deformation processes (i.e. they are invariant to isometries).

Given the triangle mesh representation of surfaces $\mathcal{M}$ and $\mathcal{N}$, a common approach in the literature is to approximate the Laplace-Beltrami eigenfunctions with the eigenvectors of the meshes' discrete Laplacian matrices (obtained with cotangent weight criterion [33]). If we discard the zero-valued eigenvalue, the semi-definite Laplacian matrix $\mathbf{L}^{\mathcal{M}} \in \mathbb{R}^{M \times M}$ gives a finite set of non-zero eigenvectors $\Phi_I^{\mathcal{M}} = [\phi_1^{\mathcal{M}}, .., \phi_i^{\mathcal{M}}, ..., \phi_I^{\mathcal{M}}]$, where $I \in \mathbb{N}$, $I < M$ denotes the number of eigenvectors (omitting the null eigenvector) that, in column form, constitute matrix $\Phi_I^{\mathcal{M}} \in \mathbb{R}^{M \times I}$. Analogously, eigenvectors of $\mathbf{L}^{\mathcal{N}} \in \mathbb{R}^{N \times N}$ yields matrix $\Phi_J^{\mathcal{N}} \in \mathbb{R}^{N \times J}$, where $J \in \mathbb{N}, J < N$. Both bases, $\Phi_I^{\mathcal{M}}, \Phi_J^{\mathcal{N}}$, present columns (eigenvectors) ordered by increasing associated eigenvalues.

As described above, given a functional map $\mathbf{C}$ between two surfaces $\mathcal{M}$ and $\mathcal{N}$, a point-to-point map between the surfaces can be retrieved. Once bases $\Phi_I^{\mathcal{M}}$ and $\Phi_J^{\mathcal{N}}$ are obtained, the computation of $\mathbf{C}$ is tackled in different ways. Most approaches involve optimisation processes that make use of mesh feature descriptors such as SHOT [34] or WKS [30] (projected on the basis functions) combined with regularisation terms or stochastic analysis. Among all the current approaches we focus on [27] for two main reasons: it is significantly faster (in the order of tenths of a second instead of minutes on standard computers) and thus fits the closed-loop control paradigm; and it performs a coarse-to-fine optimisation process in the frequency domain that allows to prioritise coarse geometry of the shape over the fine details.

B. The ZoomOut method

Note that, as basis' eigenvalues increase, their associated eigenvectors represent a higher frequency component of the basis. Given low values of I or J, the lower frequency components of the basis are prioritised. This fact is well exploited in ZoomOut [27] where both bases ($\Phi_I^{\mathcal{M}}$ and $\Phi_J^{\mathcal{N}}$) are enlarged iteratively. This method iteratively computes the point-to-point map $T : \mathcal{M} \to \mathcal{N}$ in a logical $M \times N$ matrix form Π, being:

$$\Pi_{m,n} = \begin{cases} 1 & \text{if } T(x_m) = y_n, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

$x_m \in \mathcal{M}$ and $y_n \in \mathcal{N}$ in (5) refer to mesh points with indices $m, n \in \mathbb{N}$. We now introduce dependence of $\mathbf{C}$ on variable I to refer to the $I \times I$ finite approximation of $\mathbf{C}_{\infty}$ in (4), that is $\mathbf{C}(I) \in \mathbb{R}^{I \times I}$. The ZoomOut method refines functional maps $\mathbf{C}(I)$ through iterative increments of I, resulting in progressively larger and finer functional maps (see Fig. 3). The method seeks to force orthonormality of *sub-matrices* $\mathbf{C}(I)$ and thus ensuring locally area-preserving correspondences. It does so by minimising

$$\min_{\Pi} \sum_{I}^{I_{max}} \frac{1}{I} \left\| \mathbf{C}^{\intercal}(I)\mathbf{C}(I) - \mathbf{I}_{I \times I} \right\|_F^2, \tag{6}$$

$$\text{s.t. } \mathbf{C}(I) = \left(\Phi_I^{\mathcal{M}}\right)^{+} \Pi \Phi_I^{\mathcal{N}} \tag{7}$$

where F indicates the Frobenius norm, $\mathbf{I}_{I \times I}$ the identity matrix of size I and $()^+$ denotes pseudo-inverse (note that $M \gg I$). The constraint in (6) ensures sub-matrices of $\mathbf{C}(I_{max})$ (i.e. $\mathbf{C}(I)$) arise from a point-to-point map Π. ZoomOut [27] proposes splitting this optimisation problem into two sub-problems solved iteratively. That is, given I_{max} the maximum eigenvalue index we are considering, and the functional map initialisation $\mathbf{C}(I_{initial}) = \mathbf{I}_{I_{initial} \times I_{initial}}$ with $0 < I_{initial} < I_{max} \leq \min(M, N)$, compute iteratively:

$$\Pi = \arg\min_{\Pi} \left\| \Pi \Phi_I^{\mathcal{N}} \mathbf{C}^{\intercal}(I) - \Phi_I^{\mathcal{M}} \right\|_F^2 \tag{8}$$

$$\mathbf{C}(I+1) = (\Phi_{I+1}^{\mathcal{M}})^{\intercal} \mathbf{A}^{\mathcal{M}} \Pi \Phi_{I+1}^{\mathcal{N}} \tag{9}$$

for $I = I_{initial}, \ldots, I_{max} - 1$. Equation (8) seeks to minimise the cost presented in (6), and (9) presents the relaxed form of the constraint in (7) as it penalises the image of $\Pi \Phi_I^{\mathcal{N}} \mathbf{C}^{\intercal}(I)$ lying outside the span of $\Phi_I^{\mathcal{M}}$ (i.e. it seeks to ensure that map $\mathbf{C}(I)$, given a proper point-to-point map Π,

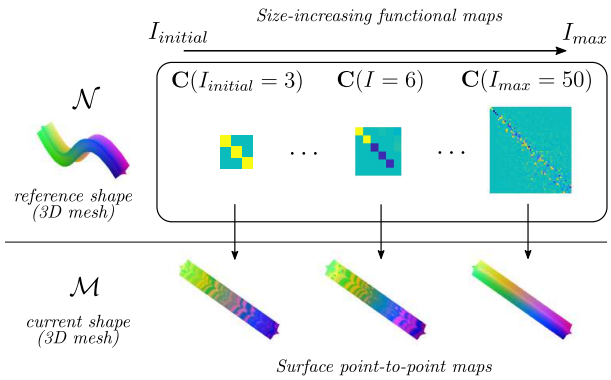


Fig. 3. Illustration of the ZoomOut method where size-increasing functional maps $\mathbf{C}(I)$ are computed for a reference shape $\mathcal{N}$ and a target shape $\mathcal{M}$. The iteration process begins with a map $\mathbf{C}(I_{initial})$ of size $I_{initial} \times I_{initial}$ which then evolves (is refined) and grows in size until becoming $\mathbf{C}(I_{max})$ of size $I_{max} \times I_{max}$. The surface map is represented by the colour map on the object's surface.

maps one eigenbasis to the other as $\Phi_I^{\mathcal{M}}\mathbf{C}(I) = \Pi\Phi_I^{\mathcal{N}}$). The orthonormality of $\Phi_I^{\mathcal{M}}$ with respect to its inner product with the mesh's lumped area matrix $\mathbf{A}^{\mathcal{M}} \in \mathbb{R}^{M\times M}$ ($\mathbf{A}^{\mathcal{M}}$ is a diagonal matrix containing the sum of the areas associated to node m and its neighbours) allows substituting $(\Phi_I^{\mathcal{M}})^+$ for $(\Phi_I^{\mathcal{M}})^\intercal\mathbf{A}^{\mathcal{M}}$ in (9). Note that as values of I increase in (9), more columns are added to $\Phi_I^{\mathcal{M}}$ and $\Phi_I^{\mathcal{N}}$ and larger (and thus finer) functional maps $\mathbf{C}(I)$ are obtained. Once the iterative process finishes, (8) provides a point-to-point map Π between both surfaces. See [27] for more detailed explanations of the method.

III. Time Consistent and Non-Isometry Robust Surface Mapping

So far, we discussed surface maps Π between two static shapes $\mathcal{M}$ and $\mathcal{N}$. Our goal now is to provide maps through time iterations $k \in \mathbb{N}$ between an evolving current shape $\mathcal{M}(k)$ and a target shape $\mathcal{N}$. Such maps can then be used to deform $\mathcal{M}(k)$ towards $\mathcal{N}$ in a shape control strategy like the one introduced in sect. IV. For effective shape control, as the object deforms, we want functional maps $\mathbf{C}(I_{max}, k)$ and point-to-point maps $\Pi(k)$ to evolve smoothly and thus be time consistent (note dependence on k has been introduced). We identified three conditions to achieve consistency and smooth map evolution in real-world applications with deformable objects: (i) **Time consistency of basis** $\Phi_{I_{max}}^{\mathcal{M}}(k)$ (Section III-B), (ii) **Time consistency of functional maps** $\mathbf{C}(I_{max}, k)$ (Section III-C), and (iii) **Robustness to non-isometries** (Section III-D). To address these challenges, we propose a solution developed in the following sections and embodied in a compact and practical form in Algorithm 1. Our proposed solution yields a time-consistent surface map $\Pi(k)$ that, when applied to a set of points $\mathbf{Y} \in \mathbb{R}^{N\times 3}$ from the target shape $\mathcal{N}$, generates the set of points $\Pi(k)\mathbf{Y} \in \mathbb{R}^{M\times 3}$ that are mapped to points $\mathbf{X} \in \mathbb{R}^{M\times 3}$ from the current shape $\mathcal{M}$. These point maps can be incorporated into a shape control scheme, as explained in Section IV.

Algorithm 1 Time consistent and non-isometry robust method.

Require: $k > 0$ ▷ Current time iteration.
Require: $0 < I_{initial} \leq I_{update} < I_{max} \leq \min(M, N)$.
if $k = 1$ **then**
  $I = I_{initial}$
  Initialise $\mathbf{C}(I, k=1)$ ▷ III-A
else
  Time consistent update of $\Phi_{I_{max}}^{\mathcal{M}}(k)$ ▷ III-B
  $I = I_{update}$
  $\mathbf{C}(I, k) = \mathbf{C}(I_{update}, k-1)$ ▷ III-C
  $J = I_{update}$
end if
while $I \leq I_{max} - 1$ **do**
  $\Pi(k) = \underset{\Pi(k)}{\arg\min} \left\|\Pi(k)\Phi_J^{\mathcal{N}}\mathbf{C}^\intercal(I,k) - \Phi_I^{\mathcal{M}}(k)\right\|_F^2$
  $I = I + 1$
  $J = \max_r\{r \mid \lambda_r^{\mathcal{N}} \leq \lambda_I^{\mathcal{M}}(k)\rho(k)\}$ ▷ III-D
  $\mathbf{C}(I, k) = (\Phi_I^{\mathcal{M}}(k))^\intercal\mathbf{A}^{\mathcal{M}}(k)\Pi(k)\Phi_J^{\mathcal{N}}(k)$
end while
return $\Pi(k), \mathbf{C}(I_{max}, k)$

A. Initialization of $\mathbf{C}(I, k=1)$

As previously stated, certain shape characteristics, such as symmetries, may result in numerous solutions for functional maps $\mathbf{C}(I, k)$ that present similar minima. We now tackle the problem of defining an initial solution that is appropriate for a shape control application. One may arbitrarily define an initial solution $\mathbf{C}(I, k=1) = \mathbf{I}_{I\times I}$, like ZoomOut does. However, this initialization is not appropriate for shape control given two main reasons:

1) Functional maps do not straightforwardly differentiate between surface orientations and inversions. An initialization $\mathbf{C}(I, k=1) = \mathbf{I}_{I\times I}$ may result in a solution that requires *inverting* object surface, thus leading to the infeasibility of the control task and to object collapse. Methods in the functional map literature such as [24] can deal with this limitation as they also consider the surface orientation through its normals.

2) Even when considering surface orientation, two mapping solutions that share similar minima might lead to very different control action requirements (e.g., unnecessarily rotating a symmetric object 180º around its symmetry axis). It is desirable to obtain the mapping solution that leads to the least Euclidean distance error. Methods such as [26] consider extrinsic error in their initialization, thus achieving more desirable solutions for our particular application of shape control.

Initialization is not required to be as fast as in-loop processes, and a more time-costly method such as [24] or [26] can be applied to achieve an initial solution that can be later updated using our proposed method. However, seeking a more straightforward approach, in this paper we opted for an ICP (Iterative Closest Point) and closest neighbour based initialization of $\mathbf{C}(I_{initial}, k=1)$. We conduct an ICP optimisation between the two sets of point coordinates $\mathbf{Y} \in \mathbb{R}^{N\times 3}$ (which stacks the node positions of the target mesh) and $\mathbf{X}$, thus obtaining a rigidly aligned set of point coordinates $\mathbf{X}_{\text{ICP}} \in \mathbb{R}^{M\times 3}$. A closest neighbour search between $\mathbf{X}_{\text{ICP}}$ and $\mathbf{Y}$ provides us with a point-to-point map $\Pi(k)$ that allows

to compute our initial functional map solution:

$$\mathbf{C}(I_{initial}, k=1) = (\Phi^{\mathcal{M}}_{I_{initial}}(k))^{\intercal}\mathbf{A}^{\mathcal{M}}(k)\Pi(k)\Phi^{\mathcal{N}}_{J}, \quad (10)$$

where the considered amount of eigenvectors J in $\Phi^{\mathcal{N}}_{J}(k)$ is responsible for the robustness to non-isometries. The computation of J will be further explained in Section III-D.

B. Time consistency of basis $\Phi^{\mathcal{M}}_{I_{max}}$ and object point tracking

The time consistency of the basis $\Phi^{\mathcal{M}}_{I_{max}}(k)$ is a necessary (but not sufficient) condition for the consistency of functional maps $\mathbf{C}(I_{max}, k)$ and point-to-point maps $\Pi(k)$ across iterations. Consider $\Phi^{\mathcal{M}}_{I_{max}}(k=1)$ our reference basis. Computing new bases across iterations $k>1$ using newly retrieved object points $\mathbf{X}_{\text{ret}}(k) \in \mathbb{R}^{M_{\text{ret}}(k)\times 3}$ ($M_{ret}(k)$ varies with time and is not necessarily equal to M) results in two main issues:

- Each row in basis $\Phi^{\mathcal{M}}_{I_{max}}(k)$ corresponds directly to a specific point on the object's surface. However, variations occur between iterations due to factors like object movement, deformation, camera noise, and irregular sampling patterns. These variations affect the number of retrieved points ($M_{\text{ret}}(k) \neq M$), their 3D positions, and their associated row indexes, causing an inconsistency in the row-to-point correspondence of $\Phi^{\mathcal{M}}_{I_{max}}(k)$ across iterations. Maintaining this consistency is essential, whether for tracking object points (e.g., for computing a deformation Jacobian) or for properly updating functional maps $\mathbf{C}(I_{max}, k)$ across iterations.
- Furthermore, properly updating $\Phi^{\mathcal{M}}_{I_{max}}(k)$ would require analysing the sign of the eigenvectors that comprise the basis, as the signs of the eigenvectors are arbitrarily defined and can change among time iterations k.

Consequently, we need to update basis $\Phi^{\mathcal{M}}_{I_{max}}(k)$ in a manner that preserves the row-to-point correspondence and the signs of eigenvectors across iterations.

Our approach serves a dual purpose. First, it solves the time inconsistency of the basis $\Phi^{\mathcal{M}}_{I_{max}}(k)$ by ensuring a consistent correspondence between rows of $\Phi^{\mathcal{M}}_{I_{max}}(k)$ and points on the object's surface. Second, it provides a time consistent vector $\mathbf{X}(k) \in \mathbb{R}^{M\times 3}$, which tracks the movement of the same set of object points (from the initial instant) over time. Through the use of $\mathbf{X}(k)$ it becomes possible to compute the deformation Jacobian of the object and to define an error vector that remains time-consistent for shape control purposes (the control system will be explained in more detail in Section IV).

We first introduce matrix $\Pi^{\mathcal{M}}(k) \in \mathbb{R}^{M\times M_{\text{ret}}(k)}$ which, in the fashion of $\Pi(k)$, constitutes the point-map between newly retrieved points $\mathbf{X}_{\text{ret}} \in \mathbb{R}^{M_{\text{ret}}(k)\times 3}$ and the time-consistent tracked points from the previous iteration $\mathbf{X}(k-1)$. In each iteration $k>1$, $\Pi^{\mathcal{M}}(k)$ is initialised through a closest neighbour search between point positions stacked in $\mathbf{X}_{\text{ret}}(k)$ and $\mathbf{X}(k-1) \in \mathbb{R}^{M\times 3}$. This initialization facilitates the convergence of the next iterative process towards an embedding-coherent mapping solution in which the object moves and deforms continuously in space. The already initialised map $\Pi^{\mathcal{M}}(k)$ is updated and refined in a coarse-to-fine manner for increasing values of I, that is for $I = I_{update}, \ldots, I_{max}-1$ we iteratively compute:

$$\begin{aligned}\Pi^{\mathcal{M}}(k) &= \\ &= \underset{\Pi^{\mathcal{M}}(k)}{\arg\min} \left\| \Pi^{\mathcal{M}}(k)\Phi^{\mathcal{M}}_{\text{ret},I}(k)(\mathbf{C}^{\mathcal{M}}(I,k))^{\intercal} - \Phi^{\mathcal{M}}_{I}(k-1) \right\|^2_F.\end{aligned} \quad (11)$$

$$\mathbf{C}^{\mathcal{M}}(I+1,k) = (\Phi^{\mathcal{M}}_{I+1}(k-1))^{\intercal}\mathbf{A}^{\mathcal{M}}(k-1)\Pi^{\mathcal{M}}(k)\Phi^{\mathcal{M}}_{\text{ret},I+1}(k), \quad (12)$$

being $\Phi^{\mathcal{M}}_{\text{ret},I}(k) \in \mathbb{R}^{M_{\text{ret}}\times I}$, $\Phi^{\mathcal{M}}_{\text{ret},I+1}(k) \in \mathbb{R}^{M_{\text{ret}}\times I+1}$ in (11) and (12) the basis obtained from the newly retrieved set of object points truncated to the I-th and $(I+1)$-th eigenvector respectively.

Functional map $\mathbf{C}^{\mathcal{M}}$ in (12) (with elements $c^{\mathcal{M}}_{i,j}(k)$) maps the previous state (time consistent) basis $\Phi^{\mathcal{M}}_{I_{max}}(k-1)$ and the obtained basis from the newly retrieved set of points $\Phi^{\mathcal{M}}_{\text{ret},I_{max}}(k)$. Using $\mathbf{C}^{\mathcal{M}}$, the diagonal matrix $\mathbf{C}^{\mathcal{M}}_{\text{sgn}}(k) \in \mathbb{R}^{I_{max}\times I_{max}}$ is computed:

$$\mathbf{C}^{\mathcal{M}}_{\text{sgn}}(k) = \text{blkdiag}(\{\text{sgn}(c^{\mathcal{M}}_{i,i}(k))\}^{I_{max}}_{i=1}), \quad (13)$$

being sgn() the sign operator and blkdiag() the block diagonal matrix building operator. Matrix $\mathbf{C}^{\mathcal{M}}_{\text{sgn}}(k)$ contains the sign correction that should be applied to eigenvectors in $\Phi^{\mathcal{M}}_{\text{ret},I_{max}}(k)$ in order to be sign-consistent with the previous state basis. Note that map $\Pi^{\mathcal{M}}(k)$ contains the point mapping that allows to pick and re-order the rows of $\Phi^{\mathcal{M}}_{\text{ret},I_{max}}(k)$ so that they are consistent with the rows of $\Phi^{\mathcal{M}}_{I_{max}}(k-1)$ in the sense of both representing the same set of tracked object points. Combining the row re-ordering provided by map $\Pi^{\mathcal{M}}(k)$ from (11) and the sign correction provided by $\mathbf{C}^{\mathcal{M}}_{\text{sgn}}(k)$ from (13) we compute a time consistent basis:

$$\Phi^{\mathcal{M}}_{I_{max}}(k) = \Pi^{\mathcal{M}}(k)\Phi^{\mathcal{M}}_{\text{ret},I_{max}}(k)\mathbf{C}^{\mathcal{M}}_{\text{sgn}}(k). \quad (14)$$

Then, also using $\Pi^{\mathcal{M}}(k)$ from (11), the positions of the object tracked points $\mathbf{X}(k)$ can be updated as

$$\mathbf{X}(k) = \Pi^{\mathcal{M}}(k)\mathbf{X}_{\text{ret}}(k). \quad (15)$$

C. Time consistency of functional maps $\mathbf{C}(I_{max}, k)$

Even if time consistency of the basis $\Phi^{\mathcal{M}}_{I_{max}}(k)$ is achieved, when shapes undergo deformations or present symmetries, new or multiple similar minima might arise in the computation of $\mathbf{C}(I_{max}, k)$. This can lead to significantly different functional map solutions $\mathbf{C}(I_{max}, k)$ across iterations that, in turn, lead to time inconsistent point-to-point maps $\Pi(k)$. For this reason, our method needs to ensure time consistency of functional maps $\mathbf{C}(I_{max}, k)$.

To address this problem, our method exploits the fact that slow/smooth deformations do not largely affect geometry features presented at low frequencies. We propose taking advantage of sub-matrices from previously computed functional maps $\mathbf{C}(I_{max}, k-1)$ under the assumption that $\Phi^{\mathcal{M}}_{I}(k) \approx \Phi^{\mathcal{M}}_{I}(k-1)$ for low $I = I_{update}$. That is, We initialise the functional map $\mathbf{C}(I_{update}, k)$ of iteration $k>1$ with a

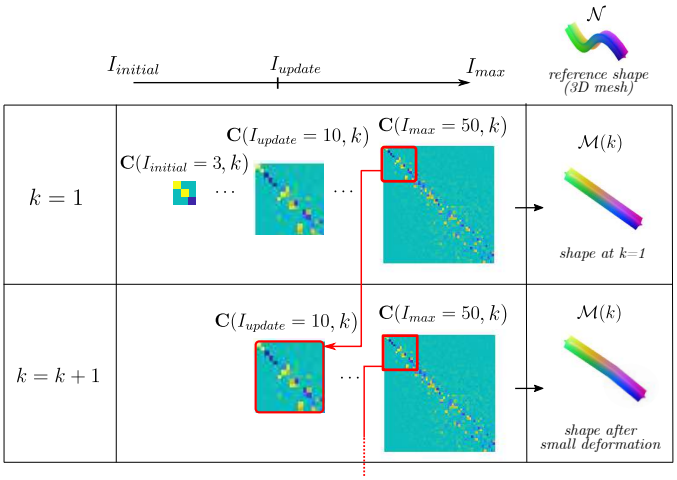


Fig. 4. Illustration of our time consistent surface mapping method. The first time iteration $k=1$ requires to iterate from $\mathbf{C}(I_{initial},k=1)$ to $\mathbf{C}(I_{max},k=1)$. For $k>1$ the process begins with $\mathbf{C}(I_{update},k)$, which is obtained as a sub-matrix of $\mathbf{C}(I_{max},k-1)$.

matrix equal to the top-left $I_{update}\times I_{update}$ sub-matrix of $\mathbf{C}(I_{max},k-1)$ (see Fig. 4). Thus, elements of $\mathbf{C}(I_{update},k)$ are:

$$c_{i,j}(I_{update},k)=c_{i,j}(I_{max},k-1),\text{ for } i,j=1,\dots,I_{update}. \tag{16}$$

This process not only allows updating the matching in a time-consistent manner (i.e. with time continuity), but also reduces the computational cost in iterations $k>1$ as the number of optimisations is reduced by $I_{update}-I_{initial}$. Using our time-consistent basis $\Phi^{\mathcal{M}}_{I_{max}}(k)$ from (14) (recall $\Phi^{\mathcal{M}}_{I}(k)$ is $\Phi^{\mathcal{M}}_{I_{max}}(k)$ truncated to the I-th column), we can iterate through increasing values of $I=I_{update},...,I_{max}-1$ to refine point-to-point maps $\Pi(k)$ and functional maps $\mathbf{C}(I,k)$:

$$\Pi(k)=\underset{\Pi(k)}{\arg\min}\left\|\Pi(k)\Phi^{\mathcal{N}}_{J}\mathbf{C}^{\intercal}(I,k)-\Phi^{\mathcal{M}}_{I}(k)\right\|^2_F, \tag{17}$$

$$\mathbf{C}(I+1,k)=(\Phi^{\mathcal{M}}_{I+1}(k))^{\intercal}\mathbf{A}^{\mathcal{M}}(k)\Pi(k)\Phi^{\mathcal{N}}_{J}. \tag{18}$$

Note that, even though we re-use data from the previous $\mathbf{C}(I_{max},k-1)$ to initialise our functional map estimation in iteration k, in (18) we allow those elements of $\mathbf{C}(I_{update},k)$ that map low frequencies (i.e. $c_{i,j}(I,k)$ with $i,j\leq I_{update}$) to be updated as well. Regarding the value of J in (17) and (18), the classic refining method in [27] would be equivalent to setting $J=I$. That is, truncating the current and target shape basis to the same number of columns (see how (8)-(9) do not require defining J). However, in the following section, we propose defining J to increase the robustness of the method to non-isometries.

D. Robustness to non-isometries

A challenge of real setups is that there is no guarantee that the desired target shape constitutes an isometry of the current shape. Our method improves robustness against non-isometries, thus achieving adequate performance in real applications.

We will now develop on the computation of J in (17) and (18) to increase robustness to non-isometric deformations. In [23], a method for matching an incomplete surface (i.e. a shape with missing parts) to its full version (i.e. the complete shape) was presented. This problem is referred to as partial functional correspondence. It is solved by taking advantage of the fact that the basis of the incomplete shape generates a subset of the basis of the complete shape. This leads to a non-square functional map $\mathbf{C}$ that presents a dominant slanted diagonal. This diagonal's slope is proportional to the ratio of the shapes' areas $A(\mathcal{N})/A(\mathcal{M})$, being $\mathcal{N}$ the complete shape and $\mathcal{M}$ the incomplete shape.

In [27], in order to solve the partial correspondence problem, they propose an update rule of the basis' size (i.e. an update rule for I and J) that weakly enforces $\mathbf{C}\in\mathbb{R}^{I\times J}$ to be non-square and to present a slanted diagonal with a slope proportional to $A(\mathcal{N})/A(\mathcal{M})$. One intuition behind the slanted diagonal slope is that the lower frequencies of the incomplete shape begin to appear *later* (in higher harmonic indexes) of the complete shape's spectrum, and thus the slanted diagonal compensates for that.

Our method exploits the slanted-diagonal concept from a different perspective. The fact that the objects we want to control deform isotropically implies that non-isometric deformations mainly affect the object's geometry in a narrow bandwidth of low frequency features. That is, if the object stretches, it will do it uniformly and surface details will be *dragged* along in a quasi-isometric manner. For this reason, we compensate for the frequency miss-match by enforcing a slanted diagonal with a slope

$$\rho(k)=\sqrt{A(\mathcal{M}(k))/A(\mathcal{N})}. \tag{19}$$

Note ρ in (19) is inversely proportional to the square root of the slope of the partial correspondence problem in [23] and [27]). We enforce the slope ρ in our functional map solution by setting J of $\lambda^{\mathcal{N}}_{J}$ to:

$$J=\max_{r}\left\{r\mid\lambda^{\mathcal{N}}_{r}\leq\lambda^{\mathcal{M}}_{I}(k)\rho(k)\right\}, \tag{20}$$

where $\lambda^{\mathcal{N}}_{J}$, and respectively $\lambda^{\mathcal{M}}_{I}$, are the eigenvalues associated to eigenvectors in columns I and J of the bases. Update of J (20) takes place between (17) and (18) and for the first computation of (17) is initialized as $J=I_{update}$. The intuition behind our update rule is that lower frequency components (i.e. components associated to low eigenvalues $\lambda^{\mathcal{M}}_{i}$) are inversely proportional to their periods $t^{\mathcal{M}}_{i}$ and therefore $\lambda^{\mathcal{N}}_{j}/\lambda^{\mathcal{M}}_{i}=t^{\mathcal{M}}_{i}/t^{\mathcal{N}}_{j}$. We approximate the objects' period ratio $t^{\mathcal{M}}_{i}/t^{\mathcal{N}}_{j}$ with the square root of the objects' area ratio ρ (19) (Fig. 5). Introducing (20) in Algorithm 1 we enhance the retrieval of point-to-point matches in non-isometric deformation scenarios and, in some cases, (20) can become a critical element in the success of the method (see Fig. 6). Note that (20) is not applied in the iterative refinement process (11)-(12). This is because

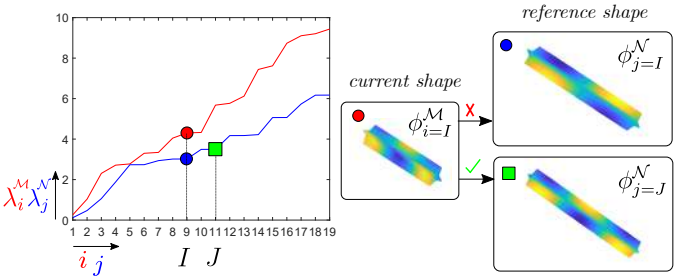
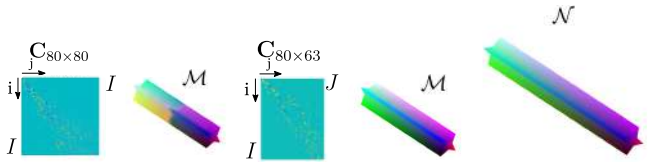


Fig. 5. Process for choosing index J in non-isometric deformations (short beam to long beam). An initial trivial guess would be $j = I$ for matching harmonic $i = I$ to harmonic $j = I$. We propose using (20) so $i = I, j = J$ with J such that $\lambda^{\mathcal{N}}_J$ is closer to $\lambda^{\mathcal{M}}_I \rho$ (in this particular example $\rho \approx 0.82$). See how our guess (i.e. $j = J$) finds a better eigenvector correspondence (eigenvectors are depicted on the shapes with colour maps).



(a) Zoomout [27] (b) Ours (Sect.III-D) (c) Target shape

Fig. 6. Example of a current shape $\mathcal{M}$ (a, b) being matched to a target shape $\mathcal{N}$ (c) by using the regular ZoomOut method (a) and our non-isometry robust method (b). Since the target shape has a significantly larger area than the current shape, ZoomOut's area-preserving optimization leads to inconsistency, evident by the colour map's discontinuity in the centre of the beam (a). Our method, using a slanted $\mathbf{C}$ matrix, resolves the non-isometry between shapes.

our method assumes a smooth deformation process, allowing the shapes of consecutive object states to be safely considered as isometries.

IV. Shape Control Application

In this section, we validate our proposed method for time-consistent surface mapping (presented in section III) within a shape control framework. Laplace-based eigenbases are useful for solving surface mapping, as they are invariant to isometries. However, this same characteristic makes them inappropriate for defining an error metric within the shape control context: different configurations of the 3D embedding can share the same intrinsic features (e.g. aspects such as concavity or convexity may not be distinguishable). We therefore propose defining a shape error based on extrinsic tracked point positions $\mathbf{X}(k)$ from (15), the target shape nodes $\mathbf{Y}$, and our computed maps $\Pi(k)$. We define a shape error in matrix form as

$$\mathbf{E}(k) = \mathbf{X}(k) - \Pi(k)\mathbf{Y}, \tag{21}$$

with $\mathbf{E}(k) \in \mathbb{R}^{M\times3}$. Rearranging $\mathbf{E}(k)$ in column form, we define the error vector $\mathbf{e}(k) \in \mathbb{R}^{3M}$:

$$\mathbf{e}(k) = [\mathbf{e}_1^\intercal(k), \ldots, \mathbf{e}_m^\intercal(k), \ldots, \mathbf{e}_M^\intercal(k)]^\intercal, \tag{22}$$

being $\mathbf{e}_m(k) \in \mathbb{R}^3$ the error vector of the m-th surface point.

A. Control law

We now present the robot-based shape control law for reducing $\|\mathbf{e}\| = \sqrt{\mathbf{e}^\intercal\mathbf{e}}$ and thus bringing a deformable object shape closer to the desired target shape (Fig. 7). The robot setup involves grippers $g = 1, ..., G$ that can perform 6 degrees of freedom actions $\mathbf{u}_g = (\Delta\mathbf{t}_g^\intercal, \Delta\mathbf{r}_g^\intercal)^\intercal$, where $\Delta\mathbf{t}_g \in \mathbb{R}^3$ is the translation increment and $\Delta\mathbf{r}_g \in \mathbb{R}^3$ is the change in orientation represented with the Rodrigues' rotation vector. We rearrange $\mathbf{X}(k)$ to define the column vector $\mathbf{x}(k) \in \mathbb{R}^{3M}$ and its variation along iterations $\Delta\mathbf{x}(k) = \mathbf{x}(k) - \mathbf{x}(k-1)$, $\Delta\mathbf{x}(k) \in \mathbb{R}^{3M}$. If we actuate the grippers simultaneously, we cannot estimate the contribution to $\Delta\mathbf{x}(k)$ that each gripper generates. For this reason, we propose operating them sequentially.

In order to design our control law, we define two data buffers (or historic information) for each gripper. One contains previous gripper g actions as $\mathbf{U}_g(k) = (\mathbf{u}_g^1, ..., \mathbf{u}_g^b, ..., \mathbf{u}_g^B), \mathbf{U}_g(k) \in \mathbb{R}^{6\times B}$, being $b = 1, ..., B$ the buffer index and $B \in \mathbb{N}, B \geq 6$ the buffer size. The other data buffer contains the change in the state (i.e. $\Delta\mathbf{x}$) that each action $\mathbf{u}_g^b$ has generated, i.e. $\Delta\mathbf{X}_g(k) = (\Delta\mathbf{x}_g^1, ..., \Delta\mathbf{x}_g^b, ..., \Delta\mathbf{x}_g^B), \Delta\mathbf{X}_g(k) \in \mathbb{R}^{3M\times B}$. Grippers actuate sequentially and thus buffers (note that they depend on k) are updated cyclically, i.e. each iteration k a different gripper actuates the system and thus its data buffer is updated. Buffers are updated by removing the oldest measurement B, shifting all the remaining measurements one position and adding the new measurements as $\mathbf{u}_g^1 = \mathbf{u}_g(k)$ and $\Delta\mathbf{x}_g^1 = \Delta\mathbf{x}(k)$. The initialization of the buffers is performed by actuating $B \geq 6$ times each gripper individually with small actions that ensure small deformations and full rank of $\mathbf{U}_g(k = 1)$.

We use the data buffers to estimate an interaction matrix $\mathbf{L}_g \in \mathbb{R}^{3M\times6}$ that defines the dynamics of the system as

$$\Delta\mathbf{x} = \mathbf{L}_g\mathbf{u}_g. \tag{23}$$

The estimation is:

$$\hat{\mathbf{L}}_g = \Delta\mathbf{X}_g\mathbf{U}_g^\intercal\left(\mathbf{U}_g\mathbf{U}_g^\intercal - \gamma\mathbf{I}_6\right)^{-1}, \tag{24}$$

where, for clarity, dependence on k has been omitted. Parameter $\gamma > 0$ (γ very small) is the Tikhonov's regularisation parameter and enhances the system's stability in those cases in which $\mathbf{U}_g\mathbf{U}_g^\intercal$ happens to be near-singular. Matrix $\hat{\mathbf{L}}_g(k) \in \mathbb{R}^{3M\times6}$ allows us to define the control law by using its left pseudo-inverse $\hat{\mathbf{L}}_g^+ \in \mathbb{R}^{6\times3M}$:

$$\mathbf{u}_g = -\alpha\hat{\mathbf{L}}_g^+\mathbf{e}, \tag{25}$$

where $\alpha > 0$ is the control gain. For good-enough estimations of $\mathbf{L}$ and considering $\Pi(k)$ in (21) is a time-consistent selector matrix of static reference points $\mathbf{Y}$, the error dynamics are obtained by substituting (25) in (23):

$$\Delta\mathbf{e} = \Delta\mathbf{x} = -\alpha\mathbf{L}_g\hat{\mathbf{L}}_g^+\mathbf{e} = -\alpha\mathbf{e}, \tag{26}$$

and thus we conclude local exponential stability.

This article has been accepted for publication in IEEE Transactions on Automation Science and Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASE.2025.3529180

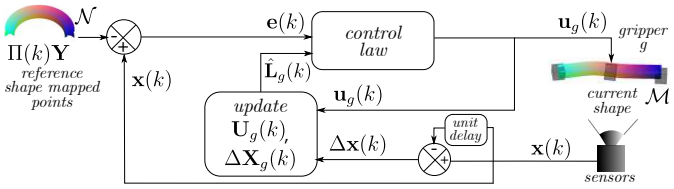


Fig. 7. Control scheme that shows how data buffers $\mathbf{U}_g(k)$ and $\Delta\mathbf{X}_g(k)$ are updated when gripper g is actuated. Our proposed time consistent tracked points $\mathbf{X}(k)$ and surface maps $\Pi(k)$ are key in the definition of our control law and shape error $\mathbf{e}(k)$.

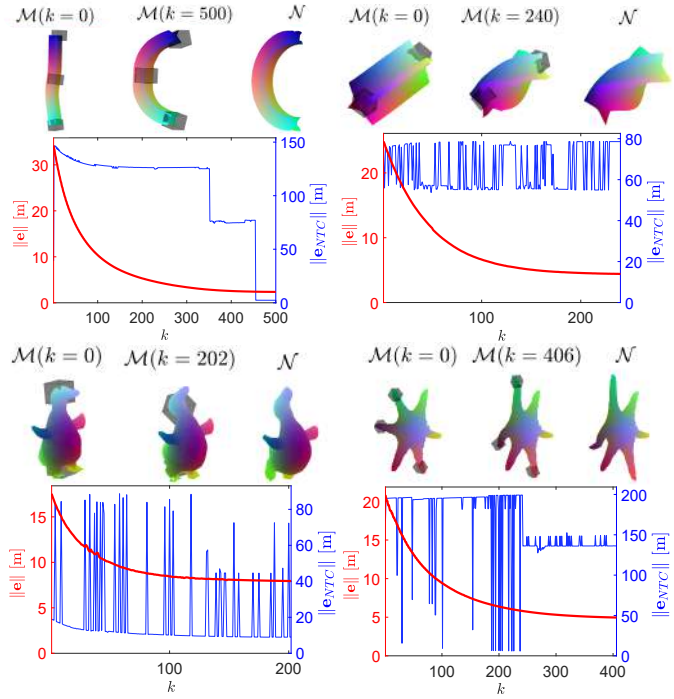


Fig. 8. Results of 4 different shape deformation simulations. For each simulation, the initial, final, and target shapes are depicted (left to right). Underneath them, a plot shows the error $\|\mathbf{e}\|$ (left axis, in red) and the evolution of the error that would result from applying ZoomOut directly (without our time consistent approach), i.e. $\|\mathbf{e}_{NTC}\|$ (right axis, in blue).

B. Simulation results

We performed several tests using the ARAP [35] deformation model to compute the object evolution. In Fig. 8, we present two simulations with synthetic shapes generated using Blender (both the initial and the target shapes) and two simulations that involve shapes obtained from real data: the duck is obtained from RGB-D images whereas the sea star is obtained from 3D scanner data. All used meshes contain around 100 to 500 nodes. In each simulation result we begin by showing the initial, final and target shape configurations (first three elements).

The plots in Fig. 8 present the error $\|\mathbf{e}\|$ evolution (left axis) along with another error $\|\mathbf{e}_{NTC}\|$ (right axis, note the different scale). Error $\|\mathbf{e}_{NTC}\|$ (non-time-consistent error) illustrates the evolution of the error according to the ZoomOut mapping method [27]. That is, the ZoomOut surface maps (not time-consistent) have been computed on surface data recorded from the simulations and experiments. Such simulations and experiments have been performed by feeding control law (25) with the error $\mathbf{e}$ in (22), generated through our time-consistent mapping method. No shape control has been performed with error $\mathbf{e}_{NTC}$ generated from ZoomOut maps, as its discontinuities and time inconsistencies do not allow the computation of deformation Jacobians nor provide a continuous and consistent error signal. Note that our mapping comparisons with ZoomOut are not a standard bench-marking evaluation, as ZoomOut was not created to achieve time consistency nor to deal with non-synthetic data. Instead, our goal is to underscore the significance of time consistency and robustness of surface maps for shape control and to demonstrate that our method meets these requirements as it allows for the computation of (24) and provides a proper error signal for (25). The simulations along with their associated action plots are shown in the **accompanying video**.

In the simulations, note how the proposed control reference $\|\mathbf{e}\|$ is smooth and continuous (deformable objects constitute highly under-actuated systems and thus $\|\mathbf{e}\|$ does not usually reach zero). On the other hand, error $\|\mathbf{e}_{NTC}\|$ presents discontinuities and noisy behaviour, leading to the conclusion that the control reference it represents would not be suitable for a shape control law. It is interesting to see how in the first experiment (top left) in Fig. 8, $\|\mathbf{e}_{NTC}\|$ still manages to stabilise at certain times, however, it remains around larger error values ($\sim$ 100 [m]) until the shape has almost converged to the solution. The second simulation (top right) presents more complex symmetries (axial-wise) and thus $\|\mathbf{e}_{NTC}\|$ oscillates around a relatively small range of values. The third simulation in Fig. 8 (bottom left) presents certain stability of $\|\mathbf{e}_{NTC}\|$ at the beginning as, at a certain range, $\|\mathbf{e}_{NTC}\|$ is coincident with $\|\mathbf{e}\|$ (axes have different scales). However, large discontinuous oscillations appear during the rest of the deformation process. The fourth simulation (bottom right) presented in Fig. 8 begins with a highly symmetric shape (radial symmetry), however, as the deforming shape gets closer to the target shape (which presents fewer symmetries), $\|\mathbf{e}_{NTC}\|$ presents discontinuities in a narrower range of values. Note that the axis of $\|\mathbf{e}_{NTC}\|$ contains large values compared to that of $\|\mathbf{e}\|$. This is due to the fact that, without the time-consistent approach, functional maps might be inverting, flipping or rotating the shape mapping completely between iterations and thus generating large variations in distances between matched points in the 3D embedding.

C. Experiments

This section describes five experiments involving the manipulation of objects with different shapes and materials. The setup (outlined in Fig. 1) includes two ABB IRB120 industrial robots with pneumatic grippers, a diffused light source, and an Intel Realsense D415 RGB-D camera. Objects are retrieved using colour-based segmentation in the CieLAB colour space; the first homogeneously sampled object points constitute the points to be tracked in subsequent iterations

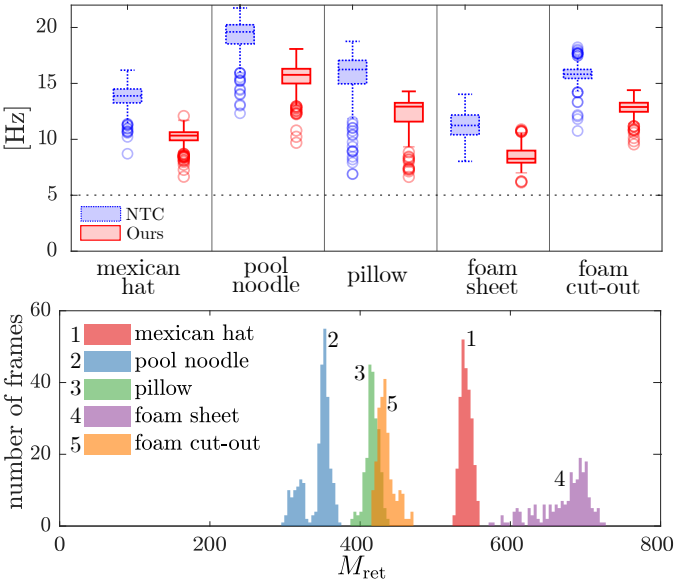


Fig. 9. Processing frequency (top box plots) and number of nodes processed M_{ret} (bottom histogram) for each experiment. The top plot compares the processing frequency distribution for the non-time-consistent method (blue and dashed contour boxes) with our method's (red).

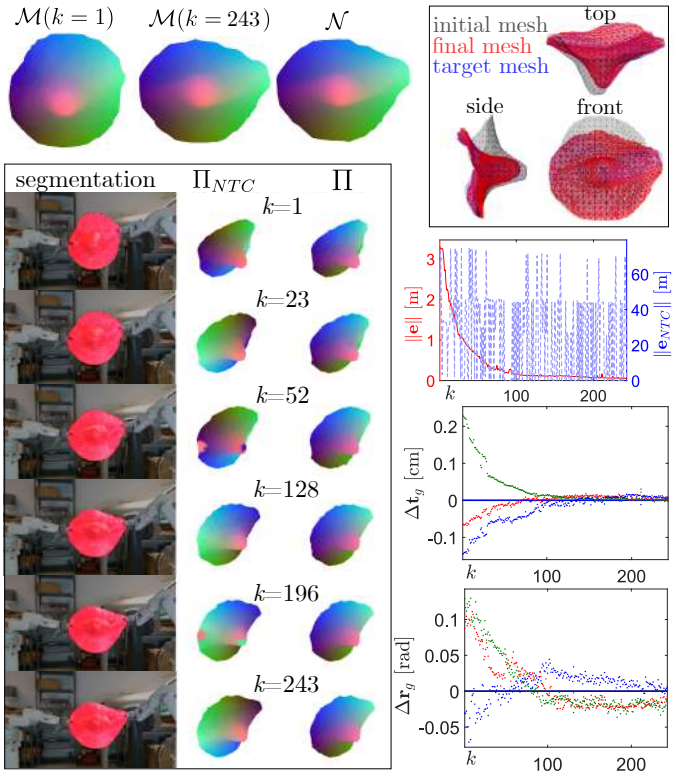


Fig. 10. Experiment: Mexican hat (see Section IV-C).

(i.e., $\mathbf{X}(k=1) = \mathbf{X}_{ret}(k=1)$). To ensure that the target shapes were feasible given the robot's workspace limitations, they were pre-acquired by teleoperating the robots. The code was implemented using Matlab R2022b, and the computer-to-robot communication was established via TCP/IP protocol. The experiments were conducted on an Intel(R) Core(TM) i7-8565U CPU with 1.99 GHz and 16 GB of RAM.

The experiments are illustrated in Figs. 9 to 15, where Figs. 9 and 15 show comparative analysis and Figs. 10 to 14 presents the experiments showing the same element distribution. On the top left, three meshes depict the initial, final, and target shape, along with the surface maps generated by our proposed method (colour map on the mesh). The same three meshes are shown in different colours (gray, red, and blue) and from three different viewpoints (front, side, and top) in the top right corner. On the bottom left, an image sequence displays the segmentation of the object on the acquired RGB images. Nearby the RGB images, the surface maps obtained using the ZoomOut method (Π_{NTC}) and our proposed method (Π) are depicted on the different state meshes as colour maps. On the right, three plots illustrate the evolution of the control process. Similar to the error plots in the simulations, the first plot on the right shows the evolution of the error associated with our mapping method, i.e., $\|\mathbf{e}\|$ (left axis, red), and the error associated with the non-time-consistent maps Π_{NTC}, i.e., $\|\mathbf{e}_{NTC}\|$ (right axis, dashed blue line, note the different scale). Below the error plot, two action plots, one for the translation component and the other for the rotation component of the action, display the evolution of actions $\mathbf{u}_g = (\Delta\mathbf{t}_g^\intercal, \Delta\mathbf{r}_g^\intercal)^\intercal$ for both robots (x, y and z components of translation and rotations depicted in red, green, and blue respectively). For a better understanding of the experiments, these results are presented in the **attached video**.

Figure 9 provides information on the processing frequency and the number of nodes processed in each experiment. Our method is more complex, which leads to a lower processing frequency as compared to the non-time-consistent method. However, it is worth noting that, in our method, all the processed frames stay over the minimum frequency of 5 [Hz] (value highlighted with a dotted line in the figure). In those sequences of $M_{\text{ret}} \approx 400$ (e.g., the pool noodle, the pillow, and the foam cut-out), sometimes the processing frequency rises up to 15 [Hz].

1) Mexican hat (Fig. 10): The flat object used in this experiment serves as an excellent example of a texture-less object that requires a 3D analysis method like the one proposed in this paper. This object lacks visual texture and has limited geometry defining features due to its radial symmetry, which makes it unsuitable for methods that rely on extracting 3D features. In this particular experiment, only the right gripper is active (the left gripper is fixed, as it is near a singular robot configuration). Despite these challenges, our proposed method successfully manipulates the hat towards the desired target shape. Note how the time inconsistent maps (obtained from directly applying ZoomOut at each iteration) not only lead to large variations derived from the symmetries of the shape but also find difficulties in converging to a map with proper continuity (see iterations $k = 52$ and $k = 196$).

This article has been accepted for publication in IEEE Transactions on Automation Science and Engineering. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASE.2025.3529180

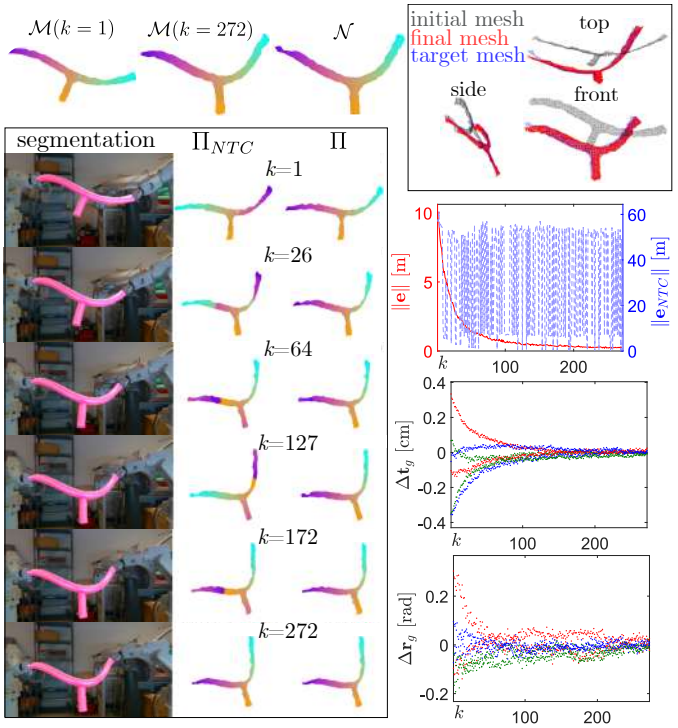


Fig. 11. Experiment: pool noodle (see Section IV-C).

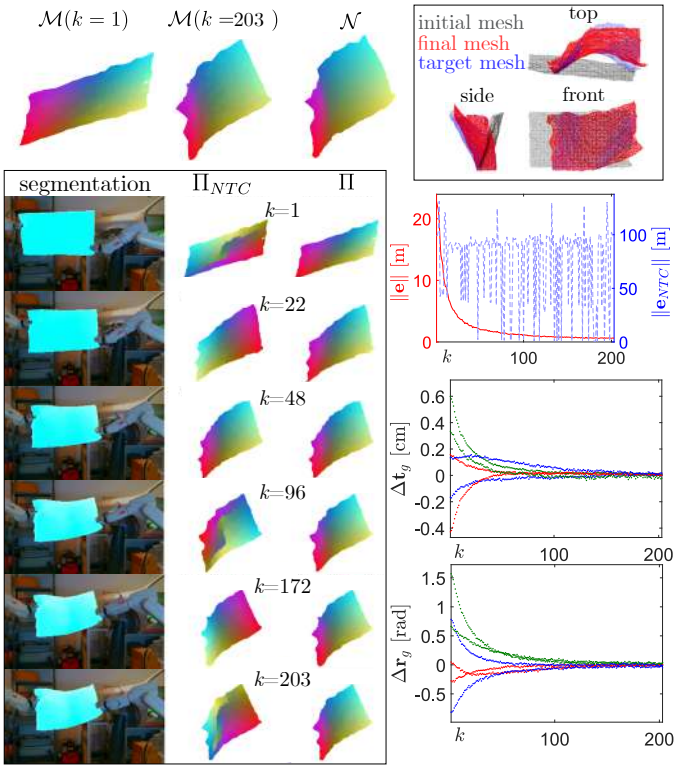


Fig. 13. Experiment: foam rectangular sheet (see Section IV-C).

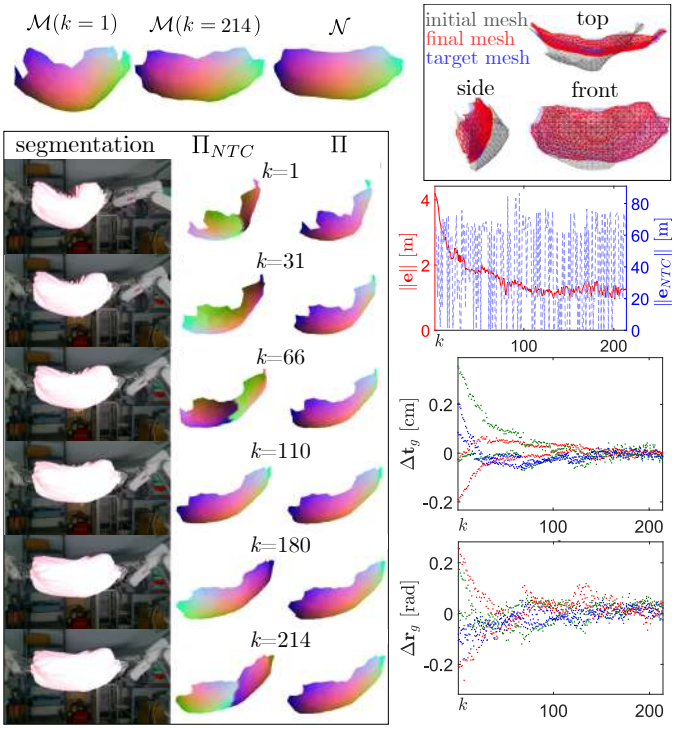


Fig. 12. Experiment: pillow (see Section IV-C).

2) Pool noodle (Fig. 11): This experiment presents a challenging sensing scenario due to the limitations of the RGB-D camera. Object points around the visual contour of the noodle, given its rounded profile, constitute slanted surfaces that greatly hamper the extraction of 3D information and lead to noisy information. Furthermore, the thin (elongated) shape of the noodle results in a high proportion of the object's retrieved information being noisy (it has a high contour-to-area ratio). This is reflected in the histogram in Fig. 9, as it reveals a non-uniform distribution of the number of retrieved points M_{ret} characterised by two clusters. Nevertheless, our time-consistent mapping method performs satisfactorily, allowing the shape control law to significantly reduce the error. In contrast, the time-inconsistent mapping method, which deals with the object's vertical symmetry and large amounts of noise, leads to map discontinuities and a highly varying error that ranges from 10 to almost 60 [m] in some cases.

3) Pillow (Fig. 12): The initial shape of the pillow, as it presents a certain inclination with respect to the camera, leads to temporary gaps on both sides of the pillow's associated mesh. This poses a challenge for the generation of surface maps, as there can be large variations of area or mesh shapes between consecutive iterations. Our time consistent mapping method handles this difficulty properly. The error $\|\mathbf{e}\|$ appears noisier in this experiment due to the smaller range of values covered by the error plot and the similarity in scale between the camera noise, mesh variations, and error values.

4) Rectangular foam sheet (Fig. 13): This shape's symmetries (with four 90º corners and straight sides) pose a challenge

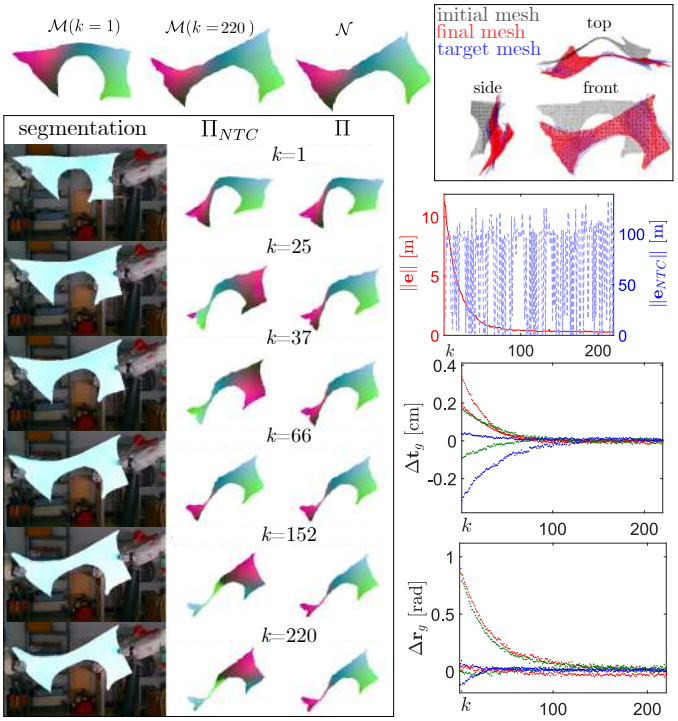


Fig. 14. Experiment: foam cut-out (see Section IV-C).

for surface mapping. The time inconsistent method struggles to converge to a continuous map, with discontinuities at certain frames (e.g., $k = 96$ or $k = 203$ in the sequence). Even when producing continuous maps, it oscillates between symmetric solutions (e.g., $k = 22$ versus $k = 48$). In this experiment, although our mapping method works properly, the proposed control law does not converge as effectively as in other experiments. See on the top right meshes that a significant portion of the final mesh does not overlap with the target mesh. This is due to the local nature of the Jacobian estimation that, although being updated, remains a rough estimation of the object's complex non-linear deformable behaviour.

5) Foam cut-out (Fig. 14): In this experiment, the challenge comes from the deformation process rather than the object's shape. The shape control problem combines bending, stretching, and twisting in 3D space. However, both the mapping method and shape control strategy perform well, resulting in a shape that closely matches the target shape. This is demonstrated by the high level of mesh overlap visible in the three viewpoints on the top right. Using this experiment as a case study, Fig. 15 illustrates the distribution of the point-to-point error $\|\mathbf{e}_m\| = \sqrt{\mathbf{e}_m^\mathsf{T}\mathbf{e}_m}$, $(m = 1, \ldots, M)$ for each time instant k, being $\mathbf{e}_m \in \mathbb{R}^3$ the error vector of the m-th object point. Below, the error distribution generated by the ZoomOut mapping method (i.e., $\|\mathbf{e}_{NTC,m}\|$) exhibits discontinuities and a lack of discernible trends. These characteristics render it impractical for calculating deformation Jacobians or for implementing shape control laws. On the bottom plot, we compare the mean values $(\bar{\mathbf{e}}, \bar{\mathbf{e}}_{NTC})$ and standard deviations $(\sigma_{\mathbf{e}}, \sigma_{\mathbf{e}NTC})$ of each method. Although the ZoomOut method sometimes converges to the desired solution (the one systematically provided by our method), these cases are sporadic and isolated.

V. Conclusions

We presented a method for obtaining time-consistent surface maps between deforming shapes, such maps consider all the object's geometry and allow defining shape control strategies in real scenarios. We applied the surface mapping method with our proposed control strategy to several deformation problems involving different objects, materials and 3D shapes. The method performed properly both in simulation and in real experiments. It is worth noting that even when used in simulations, with ground-truth tracked object points directly obtained from the simulation mesh, our proposed method still is relevant for maintaining time-consistent surface maps. Our time-consistent surface mapping method provides a versatile solution for the automation of deformable object manipulation, as it is capable of addressing a vast variety of shape mapping scenarios. It efficiently processes large numbers of point correspondences at industry-relevant frequencies. However, limitations such as unfavourable gripper positioning [36], low robustness against poor lighting conditions and large occlusions, or requirements such as faster computation times still leave room for improvement.

Regarding future work, our proposed time-consistent mapping method has the potential to be applied to other existing control strategies (e.g., [37]). Additionally, the method could be used to complement and monitor other deformable object tasks, such as cloth folding (e.g., [38]). Potential future research includes addressing the reliance on connected meshes: if occlusions *separate* the object's visible region into several regions, the resulting disconnected meshes can lead to incorrect basis computations. Other research lines could focus on making the gripper actuation concurrent rather than sequential, or using the time-consistent functional maps for analysing the feasibility of shape control tasks. Another avenue of research is the exploration of datasets based on synthetic surfaces (e.g., videos of simulated deformations). These datasets could serve as benchmark for other quantitative comparisons of emerging time-consistent surface mapping methods, not just in terms of computation cost as discussed in this paper, but also in accuracy, precision, and other quantitative measures.

References

[1] G. López-Nicolás, R. Herguedas, M. Aranda, and Y. Mezouar. Simultaneous shape control and transport with multiple robots. In *IEEE International Conference on Robotic Computing*, pages 218–225, 2020.

[2] D. Sirintuna, A. Giammarino, and A. Ajoudani. An object deformation-agnostic framework for human–robot collaborative transportation. *IEEE Transactions on Automation Science and Engineering*, 21(2):1986–1999, 2024.

[3] N. Lv, J. Liu, and Y. Jia. Dynamic modeling and control of deformable linear objects for single-arm and dual-arm robot manipulations. *IEEE Transactions on Robotics*, 38(4):2341–2353, 2022.

[4] X. Huang, D. Chen, Y. Guo, X. Jiang, and Y. Liu. Untangling multiple deformable linear objects in unknown quantities with complex backgrounds. *IEEE Transactions on Automation Science and Engineering*, 21(1):671–683, 2024.

[5] L. Han, H. Wang, Z. Liu, W. Chen, and X. Zhang. Vision-based cutting control of deformable objects with surface tracking. *IEEE/ASME Transactions on Mechatronics*, 26(4):2016–2026, 2020.

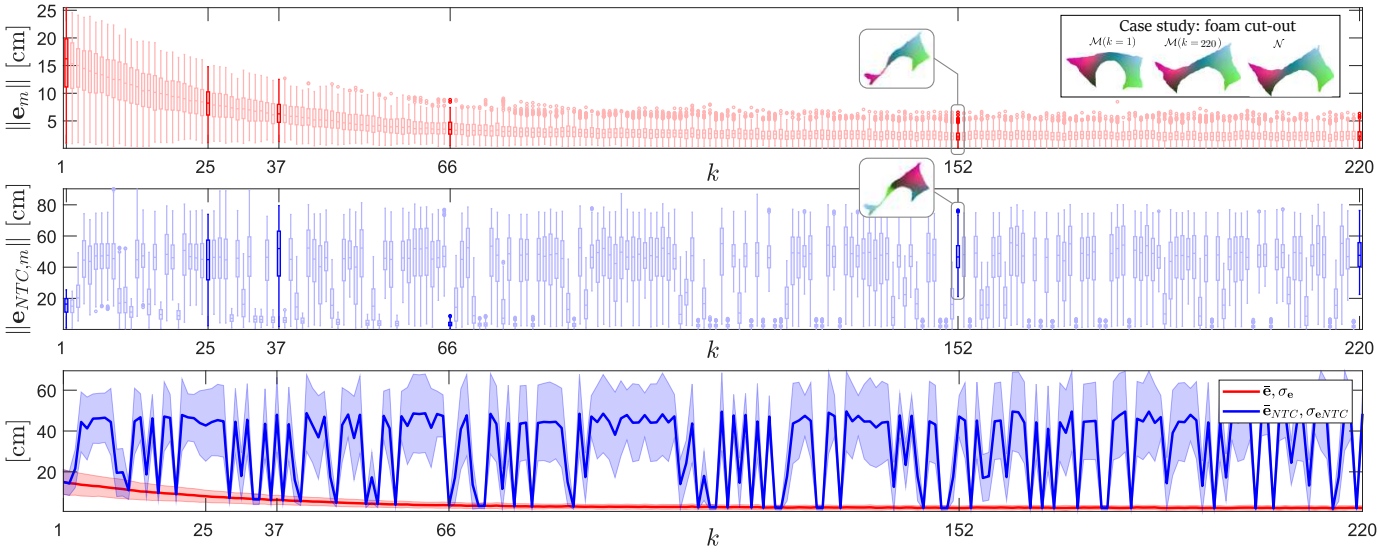


Fig. 15. Case study of the foam cut-out experiment (see Section IV-C5). Our mapping method generates smooth-varying surface maps: see top plot, with smooth variations on the point-to-point error distribution $\|\mathbf{e}_m\|$. On the other hand, the ZoomOut mapping method leads to discontinuities in the distribution of the point-to-point error $\|\mathbf{e}_{NTC,m}\|$. For reference, the numbered time instants on the abscissa axis correspond to those of the functional maps illustrated in Fig. 14. The bottom plot compares the means $(\bar{\mathbf{e}}, \bar{\mathbf{e}}_{NTC})$ and standard deviations $(\sigma_{\mathbf{e}}, \sigma_{\mathbf{e}NTC})$ of both distributions.

[6] R. Herguedas, G. López-Nicolás, R. Aragüés, and C. Sagüés. Survey on multi-robot manipulation of deformable objects. In *24th IEEE International Conference on Emerging Technologies and Factory Automation*, pages 977–984, 2019.

[7] J. Sanchez, J.A. Corrales, B.C. Bouzgarrou, and Y. Mezouar. Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey. *The International Journal of Robotics Research*, 37(7):688–716, 2018.

[8] H. Yin, A. Varava, and D. Kragic. Modeling, learning, perception, and control methods for deformable object manipulation. *Science Robotics*, 6(54):eabd8803, 2021.

[9] I. Cuiral-Zueco and G. López-Nicolás. Taxonomy of deformable object shape control. *IEEE Robotics and Automation Letters*, 9(10):9015–9022, 2024.

[10] J. Zhu, C. Dune, M. Aranda, Y Mezouar, J.M. Corrales, P. Gil, and G. López-Nicolás. Editorial: Robotic handling of deformable objects. *IEEE Robotics and Automation Letters*, 7(3):8257–8259, 2022.

[11] D. Navarro-Alarcon, H.M. Yip, Z. Wang, Y.H. Liu, F. Zhong, T. Zhang, and P. Li. Automatic 3-D manipulation of soft objects by robotic arms with an adaptive deformation model. *IEEE Transactions on Robotics*, 32(2):429–441, 2016.

[12] M. Aranda, J.A. Corrales Ramon, Y. Mezouar, A. Bartoli, and E. Özgür. Monocular visual shape tracking and servoing for isometrically deforming objects. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 7542–7549, 2020.

[13] M. Shetab-Bushehri, M. Aranda, Y. Mezouar, and E. Ozgur. As-rigid-as-possible shape servoing. *IEEE Robotics and Automation Letters*, 7(2):3898–3905, 2022.

[14] D. Berenson. Manipulation of deformable objects without modeling and simulating deformation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4525–4532, 2013.

[15] H. Deng, F. Ahmad, J. Xiong, and Z. Xia. A robot-object unified modeling method for deformable object manipulation in constrained environments. *IEEE/ASME Transactions on Mechatronics*, 29(6):4262–4273, 2024.

[16] I. Cuiral-Zueco and G. López-Nicolás. Multi-scale Laplacian-based FMM for shape control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3792–3797, 2021.

[17] D. Navarro-Alarcon and Y.H. Liu. Fourier-based shape servoing: a new feedback method to actively deform soft objects into desired 2-D image contours. *IEEE Transactions on Robotics*, 34(1):272–279, 2017.

[18] J. Zhu, D. Navarro-Alarcon, R. Passama, and A. Cherubini. Vision-based manipulation of deformable and rigid objects using subspace projections of 2-D contours. *Robotics and Autonomous Systems*, 142:103798, 2021.

[19] J. Qi, G. Ma, J. Zhu, P. Zhou, Y. Lyu, H. Zhang, and D. Navarro-Alarcon. Contour moments based manipulation of composite rigid-deformable objects with finite time model estimation and shape/position control. *IEEE/ASME Transactions on Mechatronics*, 27(5):2985–2996, 2022.

[20] A. Caporali, P. Kicki, K. Galassi, R. Zanella, K. Walas, and G. Palli. Deformable linear objects manipulation with online model parameters estimation. *IEEE Robotics and Automation Letters*, 9(3):2598–2605, 2024.

[21] A. Caporali, K. Galassi, and G. Palli. Deformable linear objects 3D shape estimation and tracking from multiple 2D views. *IEEE Robotics and Automation Letters*, 8(6):3852–3859, 2023.

[22] M. Ovsjanikov, M. Ben-Chen, J. Solomon, A. Butscher, and L. Guibas. Functional maps: a flexible representation of maps between shapes. *ACM Transactions on Graphics*, 31(4):1–11, 2012.

[23] E. Rodolà, L. Cosmo, M. M. Bronstein, A. Torsello, and D. Cremers. Partial functional correspondence. In *Computer Graphics Forum*, volume 36, pages 222–236. Wiley Online Library, 2017.

[24] J. Ren, A. Poulenard, P. Wonka, and M. Ovsjanikov. Continuous and orientation-preserving correspondences via functional maps. *ACM Transactions on Graphics*, 37(6):1–16, 2018.

[25] N. Donati, E. Corman, S. Melzi, and M. Ovsjanikov. Complex functional maps: A conformal link between tangent bundles. In *Computer Graphics Forum*, volume 41, pages 317–334. Wiley Online Library, 2022.

[26] M. Eisenberger, Z. Lahner, and D. Cremers. Smooth shells: Multi-scale shape registration with functional maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12265–12274, 2020.

[27] S Melzi, J. Ren, E. Rodolà, A. Sharma, P. Wonka, M. Ovsjanikov, et al. ZoomOut: spectral upsampling for efficient shape correspondence. *ACM Transactions on Graphics*, 38(6):1–14, 2019.

[28] Z. Hu, P. Sun, and J. Pan. Three-dimensional deformable object manipulation using fast online Gaussian process regression. *IEEE Robotics and Automation Letters*, 3(2):979–986, 2018.

[29] H. Mo, B. Ouyang, L. Xing, D. Dong, Y. Liu, and D. Sun. Automated 3-D deformation of a soft object using a continuum robot. *IEEE Transactions on Automation Science and Engineering*, 18(4):2076–2086, 2020.

[30] M. Aubry, U. Schlickewei, and D. Cremers. The wave kernel signature: A quantum mechanical approach to shape analysis. In *International conference on computer vision workshops (ICCV workshops)*, pages 1626–1633, 2011.

[31] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis. Scape: shape completion and animation of people. In *ACM SIGGRAPH 2005 Papers*, pages 408–416. 2005.

[32] L. Cosmo, E. Rodola, M. M. Bronstein, A. Torsello, D. Cremers, and Y. Sahillioglu. SHREC'16: Partial matching of deformable shapes. *Eurographics Workshop on 3D Object Retrieval*, 2(9):12, 2016.

[33] U. Pinkall and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental mathematics*, 2(1):15–36, 1993.

[34] F. Tombari, S. Salti, and L. Di Stefano. Unique signatures of histograms for local surface description. In *European conference on computer vision*, pages 356–369. Springer, 2010.

[35] O. Sorkine and M. Alexa. As-rigid-as-possible surface modeling. In *Symposium on Geometry processing*, volume 4, pages 109–116, 2007.

[36] I. Cuiral-Zueco, G. López-Nicolás, and H. Araujo. Gripper positioning for object deformation tasks. In *IEEE International Conference on Robotics and Automation*, pages 963–969. IEEE, 2022.

[37] D. McConachie and D. Berenson. Estimating model utility for deformable object manipulation using multiarmed bandit methods. *IEEE Transactions on Automation Science and Engineering*, 15(3):967–979, 2018.

[38] Y. Li, Y. Wang, Y. Yue, D. Xu, M. Case, S.F. Chang, E. Grinspun, and P.K. Allen. Model-driven feedforward prediction for manipulation of deformable objects. *IEEE Transactions on Automation Science and Engineering*, 15(4):1621–1638, 2018.

Ignacio Cuiral-Zueco received the Ph.D. degree in Systems Engineering and Computer Science from Universidad de Zaragoza, Spain, in 2024. He is a member of the Robotics, Perception, and Real-Time research group. His current research interests include computer vision, control engineering, and robotics.

Gonzalo López-Nicolás (Senior Member, IEEE) received the Ph.D. degree in Systems Engineering and Computer Science from Universidad de Zaragoza, Spain, in 2008. He is currently a full professor at the Department of Systems Engineering and Automation, Universidad de Zaragoza. He is a member of the Instituto de Investigación en Ingeniería de Aragón (I3A). His current research interests are focused on shape control, visual control, multi-robot systems, and application of computer vision to robotics.