Citation: Omid Aghajanzadeh, Miguel Aranda, Gonzalo López-Nicolás, Roland Lenain, and Youcef Mezouar. An offline geometric model for controlling the shape of elastic linear objects. In IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2022, Kyoto, Japan, October 23-27, 2022, pages 2175–2181. IEEE, 2022.

# An offline geometric model for controlling the shape of elastic linear objects

Omid Aghajanzadeh, Miguel Aranda, Gonzalo López-Nicolás, Roland Lenain and Youcef Mezouar

***Abstract*— We propose a new approach to control the shape of deformable objects with robots. Specifically, we consider a fixed-length elastic linear object lying on a 2D workspace. Our main idea is to encode the object's deformation behavior in an offline constant Jacobian matrix. To derive this Jacobian, we use geometric deformation modeling and combine recent work from the fields of deformable object control and multirobot systems. Based on this Jacobian, we then propose a robotic control law that is capable of driving a set of shape features on the object toward prescribed values. Our contribution relative to existing approaches is that at run-time we do not need to measure the full shape of the object or to estimate/simulate a deformation model. This simplification is achieved thanks to having abstracted the deformation behavior as an offline model. We illustrate the proposed approach in simulation and in experiments with real deformable linear objects.**

## I. Introduction

The manipulation of deformable –as opposed to rigid– objects with robots is considered instrumental for extending the capabilities of current systems. For this reason, this field has acquired great relevance recently [1]. A particular subdomain that has attracted considerable attention in the past few years is the manipulation of deformable linear objects (DLOs) such as cables, wires, flexible rods, plant stems, etc. [2]–[7]. This recent interest is due to the important associated applications in, e.g., industry or agriculture.

In the state of the art, successful methods have been proposed to manipulate DLOs based on a pre-existing deformation model [6]–[8] or on a model-free sensor-based adaptive scheme [2]–[5]. An alternative approach based on a diminishing rigidity assumption has also been proposed [9]. At run-time, the cited methods that perform closed-loop control require sensing the full shape of the object (in order to simulate the deformation model), or running adaptive schemes to estimate the deformation behavior. In this paper, we propose a simpler approach that does not have these requirements. The approach works by computing offline a model (a deformation Jacobian) based on geometric modeling and assuming an elastic behavior. At run-time, our method needs to sense the values of the servoed features only, and not the full shape. To develop our model, we build on recent techniques for deformable object manipulation [7] and multirobot systems control [10]. Our model is geometric and thus we also follow along the lines of recent work [11], [12] which proposed the robotic control of the shape of deformable objects using geometric deformation modeling techniques [13], [14]. Differently from our paper, these works [11], [12] required sensing the full shape of the object and simulating the deformation model at run-time.

Addressing DLO manipulation problems via the use of machine learning has gained popularity recently. A deep-neural-network-based dynamics model was proposed in [15] to predict the next shape of the object. In [16], a multi-layer neural network encoded the mapping between end-effector motions and object deformations. [17] proposed a scheme for DLO deformation control relying on both offline and online learning. The work [18] addressed DLO manipulation via the use of reinforcement learning and accommodated elastoplastic behaviors. Compared to these methods, our approach is simpler and does not need any training.

The contributions of this paper are:

- A novel method for 2D control of the shape of elastic linear objects based on an offline geometric model which, at run-time, does not require estimating/simulating the deformation behavior or sensing the full shape of the object.
- A novel link between prior work in two different domains: deformable object modeling and multirobot systems control. This link is a promising starting point for potential future developments in the same direction.
- Successful validation of the applicability of the method, in simulation and via robotic experiments with diverse objects.

The envisioned applications of the proposed approach are diverse, including for instance plant pruning tasks in agriculture, or assembly operations in industrial contexts.

## II. Problem formulation

The problem we address is the control of the shape of a linear elastic object by means of robotic manipulation. This problem has been given the name shape servoing in the literature [4], [12], [19]. Specifically, we consider the following conditions and assumptions:

This work was sponsored by a public grant overseen by the French National Research Agency as part of the "Investissements d'Avenir" through the IMobS3 Laboratory of Excellence (ANR-10-LABX-0016) and the IDEX-ISITE initiative CAP 20-25 (ANR-16-IDEX-0001). It is developed within the context of the project "SyncEA" ("Dynamic SYNChronization of mobile manipulators for new Environmental and Agricultural tools") of CAP 20-25 Challenge 2 (Domain: Agro Technologies).

This work was also supported via grants PGC2018-098719-B-I00 and PID2021-124137OB-I00 funded by MCIN/AEI/10.13039/501100011033 and by "ERDF A way of making Europe", and via a María Zambrano fellowship funded by the Spanish Ministry of Universities and the European Union-NextGenerationEU.

O. Aghajanzadeh and Y. Mezouar are with Université Clermont Auvergne, CNRS, Clermont Auvergne INP, Institut Pascal, F-63000 Clermont-Ferrand, France. E-mail: omid.aghajanzadeh@uca.fr, youcef.mezouar@sigma-clermont.fr. M. Aranda and G. López-Nicolás are with Instituto de Investigación en Ingeniería de Aragón, Universidad de Zaragoza, Spain. E-mail: {miguel.aranda, gonlopez}@unizar.es. R. Lenain is with Université Clermont Auvergne, INRAE, URTSCF, 9 av. Blaise Pascal CS 20085, F-63178 Aubière, France. E-mail: roland.lenain@inrae.fr.

• The object is a DLO and lies in a 2D workspace. It has fixed length and it deforms elastically (not plastically). The shape of the object at rest is known. The object's shape always remains stable (i.e., in quasi-static equilibrium) during its manipulation.
• One or multiple robots (i.e., grippers) grasp the object. The velocity of each gripper can be set by a robotic controller. Each gripper constrains a specific gripped region on the object. There may also be anchored regions on the object, i.e., whose position is constant due to external constraints (e.g., a grounded region). The gripped and anchored regions of the object are known before starting the task, and they remain fixed throughout the execution. The grasping configuration used is suitable for completing the task described next.
• The addressed task consists in controlling a set of features that encode information about the object's shape (e.g., the 2D positions of certain parts of the object). Each feature has a desired value, which is fixed and prescribed before starting the task. The current values of the features can be continually sensed at run-time (e.g., with vision). Our objective is to find a closed-loop control law for the velocity of every gripper so that every feature is driven to its desired value.

Fig. 1 shows the block diagram of the method we propose to carry out this task. We describe the method next.

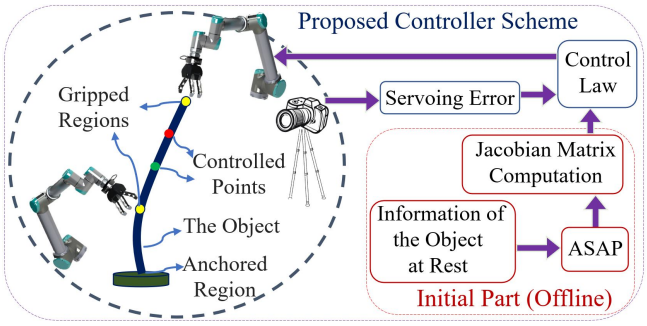


Fig. 1. Schematic diagram of the proposed method.

## III. Proposed DLO Modeling and Shape Control

We represent the DLO discretely with a set of $n$ nodes, indexed $1, ..., n$. The nodes are sampled uniformly and consecutively along the object's length. The object's rest shape, $\mathbf{c}$, is the stacking of the positions at rest of all nodes: $\mathbf{c} = [\mathbf{c}_1^\top,\ \mathbf{c}_2^\top,\ ...,\ \mathbf{c}_n^\top]^\top \in \mathbb{R}^{2n}$. We group the nodes in sets of three. We call each of these sets a *triad*. The full linear object has $m$ *chained* triads, $\mathcal{T}_k = \{k, k+1, k+2\}$ (where the indexes are taken $mod\ n$) for $k = 1, ..., m$. We consider that the object's rest shape is open; e.g., a rod with two ends. Therefore, $m = n - 2$. We denote the current shape of the object (i.e., the current node positions) as $\mathbf{q} = [\mathbf{q}_1^\top,\ \mathbf{q}_2^\top,\ ...,\ \mathbf{q}_n^\top]^\top \in \mathbb{R}^{2n}$.

Our approach is based on an As-Similar-As-Possible (ASAP) modeling. ASAP is a way to model deformations that has been used in the computer graphics domain for visualization applications [20]–[22]. It is closely related to, and sometimes used in conjunction with, the popular model ARAP (As-Rigid-As-Possible) [12], [13], [20], [23]. ASAP is based on the intuitive idea that every *local region* of the modeled object has a tendency to preserve its original shape up to translation, rotation and scaling: i.e., a similarity transformation of the original shape, hence the name of the method. In concrete terms, ASAP is formulated as a deformation energy, and the quasi-static configurations of the object's shape are local minima of that energy.

There are different formulations of ASAP/ARAP; all are based on the same underlying concepts, but they use, e.g., different definitions of the *local regions*. In particular, an ASAP energy in 2D was proposed in [10], in a context not related with deformable objects: specifically, the proposed energy was used as the cost function in a distributed controller of a multirobot formation. In [10], the elements forming the triads were robots, whereas they are the points on the object's body in our case. As we show next, the formulation of [10] is compact and adapts perfectly to our needs, which motivates our use of it. The energy proposed in [10] is a sum of energies over every triad, as follows:

$$E_a = \sum_{k=1}^{m} E_k,\ E_k = \frac{1}{2}\sum_{i\in\mathcal{T}_k} ||(\mathbf{q}_i - \mathbf{q}_{0_k}) - \mathbf{H}_k(\mathbf{c}_i - \mathbf{c}_{0_k})||^2. \tag{1}$$

$\mathbf{q}_{0_k}$ and $\mathbf{c}_{0_k}$ are the centroids of the current and rest positions of the three nodes in $\mathcal{T}_k$. $\mathbf{H}_k \in \mathbb{R}^{2\times 2}$ is the least-squares similarity (rotation and uniform scaling) transformation between these two sets of positions. $E_a$ can be expressed compactly:

$$E_a = -\frac{1}{2}\mathbf{q}^\top \mathbf{A}\mathbf{q}, \tag{2}$$

where $\mathbf{A}$ is a constant matrix that encapsulates the deformation (in the ASAP sense) of the object. $\mathbf{A}$ is a sparse symmetric matrix based on the triad structure and on the shape at rest, $\mathbf{c}$. Note that this compact 2D formulation cannot be directly extended to 3D, where non-linearities appear (see, e.g., [24]). An expression of $\mathbf{A}$ was given for general triad structures and general shapes at rest in [10]. Next, starting from that expression, we give the particular form of the matrix in the case of our DLO model.

We define $\mathbf{S} = [(0,1)^\top, (-1,0)^\top]$, i.e., a counterclockwise rotation of $\pi/2\,rad$, and $\mathbf{T} = \mathbf{I}_n \otimes \mathbf{S}$, where $\mathbf{I}_n$ denotes the $n\times n$ identity matrix and $\otimes$ the Kronecker product. Then, $\mathbf{A}, \mathbf{A}_k, \mathbf{L}_k \in \mathbb{R}^{2n\times 2n}$, $\mathbf{L}_{gk} \in \mathbb{R}^{n\times n}$ are as follows:

$$\mathbf{A} = \sum_{k=1}^{m} \mathbf{A}_k,\ \mathbf{A}_k = \frac{\mathbf{L}_k(\mathbf{c}\mathbf{c}^\top + \mathbf{T}\mathbf{c}\mathbf{c}^\top\mathbf{T}^\top)\mathbf{L}_k}{\mathbf{c}^\top\mathbf{L}_k\mathbf{c}} - \mathbf{L}_k, \tag{3}$$

$$\mathbf{L}_k = \mathbf{L}_{gk}\otimes\mathbf{I}_2,\ \mathbf{L}_{gk}[i,j] = \begin{cases} 2/3, & \text{if } i=j\ \&\ i\in\mathcal{T}_k \\ -1/3, & \text{if } i\neq j\ \&\ i,j\in\mathcal{T}_k \\ 0 & \text{otherwise.} \end{cases}$$

ASAP (and ARAP) are models that represent physical behavior at the geometric level, rather than at the mechanical one. Still, these models assume the object behavior is quasi-static, and they are based on formulating a deformation energy. In these aspects they are analogous to elastic FEM models [7], [25]. Therefore, it is possible to compute equivalent mechanical magnitudes (e.g., forces) in ASAP or ARAP.

We will therefore use an analysis similar to [7], [25] to derive a control law from ASAP.

The first step is to know the forces at the nodes due to the ASAP energy. By definition these forces are equal to the negated gradient of the energy. Therefore, noting that $\mathbf{A}$ is symmetric, the nodal forces $\mathbf{f}_a \in \mathbb{R}^{2n}$ are:

$$\mathbf{f}_a = -\frac{\partial E_a}{\partial \mathbf{q}} = \mathbf{A}\mathbf{q}. \tag{4}$$

It is interesting to note that this is a linear expression in $\mathbf{q}$. We compute its time derivative:

$$\mathbf{A}\dot{\mathbf{q}} = \dot{\mathbf{f}}_a. \tag{5}$$

From (5), we will next derive a control law following a similar strategy to the one used in [7] with a FEM model. We first define a partition of the nodes. Concretely, we divide the set of nodes into $n_g$ *gripped* nodes, $n_s$ *servoed* nodes, and $n_f$ *free* nodes, such that $n_g + n_s + n_f = n$. Note that the set of gripped nodes includes all regions of the object whose position is constrained externally: i.e., the grasped regions, and also the anchored regions. The positions of the nodes are denoted, respectively, by $\mathbf{q}_g \in \mathbb{R}^{2n_g}$, $\mathbf{q}_s \in \mathbb{R}^{2n_s}$, $\mathbf{q}_f \in \mathbb{R}^{2n_f}$. Matrix $\mathbf{A}$ and the vector of forces $\mathbf{f}_a$ are also partitioned accordingly, and hence (5) takes the form:

$$\begin{pmatrix} \mathbf{A}_{gg} & \mathbf{A}_{gs} & \mathbf{A}_{gf} \\ \mathbf{A}_{sg} & \mathbf{A}_{ss} & \mathbf{A}_{sf} \\ \mathbf{A}_{fg} & \mathbf{A}_{fs} & \mathbf{A}_{ff} \end{pmatrix} \begin{pmatrix} \dot{\mathbf{q}}_g \\ \dot{\mathbf{q}}_s \\ \dot{\mathbf{q}}_f \end{pmatrix} = \begin{pmatrix} \dot{\mathbf{f}}_{ag} \\ \dot{\mathbf{f}}_{as} \\ \dot{\mathbf{f}}_{af} \end{pmatrix}. \tag{6}$$

As the object is always in quasi-static equilibrium, all resultant forces (i.e., sum of ASAP force and external force) on all nodes are always zero. Hence, their time-derivative is zero too. Therefore, using subscript $ex$ to denote external forces, we have:

$$\dot{\mathbf{f}}_{ag}+\dot{\mathbf{f}}_{exg} = \mathbf{0}\,, \qquad \dot{\mathbf{f}}_{as}+\dot{\mathbf{f}}_{exs} = \mathbf{0}\,, \qquad \dot{\mathbf{f}}_{af}+\dot{\mathbf{f}}_{exf} = \mathbf{0}. \tag{7}$$

The gripped nodes are subjected to the external forces that cause the object to deform, which are in general time-varying forces. Therefore, $\dot{\mathbf{f}}_{exg}$ are not zero, and consequently $\dot{\mathbf{f}}_{ag}$ is not zero. On the other hand, the nodes that are not being gripped (i.e., the servoed nodes and free nodes) are subjected to constant external forces. Typically these constant external forces are zero, or equal to the gravity force. This means that $\dot{\mathbf{f}}_{exs} = \mathbf{0}$ and $\dot{\mathbf{f}}_{exf} = \mathbf{0}$. Hence, from (7), $\dot{\mathbf{f}}_{as} = \mathbf{0}$ and $\dot{\mathbf{f}}_{af} = \mathbf{0}$. Using these latter two conditions in (6), we get:

$$\begin{pmatrix} \mathbf{A}_{sg} & \mathbf{A}_{ss} & \mathbf{A}_{sf} \\ \mathbf{A}_{fg} & \mathbf{A}_{fs} & \mathbf{A}_{ff} \end{pmatrix} \begin{pmatrix} \dot{\mathbf{q}}_g \\ \dot{\mathbf{q}}_s \\ \dot{\mathbf{q}}_f \end{pmatrix} = \mathbf{0}. \tag{8}$$

From (8), we can obtain the key expression in our development: the relation between gripped and servoed node motions. This expression is as follows:

$$\dot{\mathbf{q}}_s = \mathbf{J}_{sg}\dot{\mathbf{q}}_g, \tag{9}$$

with $\mathbf{J}_{sg} = -(\mathbf{A}_{ss} - \mathbf{A}_{sf}\mathbf{A}_{ff}^{-1}\mathbf{A}_{fs})^{-1}(\mathbf{A}_{sg} - \mathbf{A}_{sf}\mathbf{A}_{ff}^{-1}\mathbf{A}_{fg})$. Similarly to [7], [17], [20], we assume the matrices that have to be inverted are full-rank. This comes from the fact that the shape is completely constrained by the gripped nodes.

### A. Control law

Assuming the desired positions for the servoed nodes are $\mathbf{q}_{sd} \in \mathbb{R}^{2n_s}$, we define the servoing error as:

$$\mathbf{e}_s = \mathbf{q}_s - \mathbf{q}_{sd}. \tag{10}$$

Now, from (9) and (10), we can propose the following proportional control law:

$$\mathbf{v}_g = -k_g \mathbf{J}_{sg}^{+}\mathbf{e}_s, \tag{11}$$

where $k_g$ is a positive gain, $+$ denotes the pseudoinverse, and $\mathbf{v}_g$ is the velocity of the gripped nodes: $\mathbf{v}_g = \dot{\mathbf{q}}_g$.

Frequently, the gripper used in real applications can constrain not only the position of the gripped point of the linear object, but also the orientation (i.e., the line tangent to the object's curve) at that point. Interestingly, this can be introduced in a straightforward way in our model, as follows. For simplicity, assume a gripper $h$ grips two (adjacent) nodes. Calling the positions of these nodes $\mathbf{g}_1 = [g_{1x},\, g_{1y}]^\top$ and $\mathbf{g}_2 = [g_{2x},\, g_{2y}]^\top$, the 3-DOF configuration of the gripper in the global frame is $[x_g,\, y_g,\, \beta_g]^\top = [(g_{1x} + g_{2x})/2,\, (g_{1y} + g_{2y})/2,\, atan2(g_{2y} - g_{1y}, g_{2x} - g_{1x})]^\top$. Denoting the gripped length (i.e., length of the line segment between $\mathbf{g}_1$ and $\mathbf{g}_2$) by $l_g$, a Jacobian matrix block that maps gripper $h$'s velocities to the velocities of its gripped nodes is as follows:

$$\mathbf{J}_{gph} = \begin{pmatrix} 1 & 0 & (l_g/2)\sin\beta_g \\ 0 & 1 & -(l_g/2)\cos\beta_g \\ 1 & 0 & -(l_g/2)\sin\beta_g \\ 0 & 1 & (l_g/2)\cos\beta_g \end{pmatrix}. \tag{12}$$

Assuming there are $n_e$ such grippers, we assemble the Jacobian blocks for all of them in a full Jacobian matrix $\mathbf{J}_{gp}$ of size $4n_e \times 3n_e$. We also stack the gripper velocities in a vector $\mathbf{v}_p$ of length $3n_e$. Then we define the control law:

$$\mathbf{v}_p = -k_p \mathbf{J}_{sp}^{+}\mathbf{e}_s, \tag{13}$$

where $\mathbf{J}_{sp} = \mathbf{J}_{sg}\mathbf{J}_{gp}$ is the Jacobian relating servoed nodes to grippers. One can use a gain $k_p > 0$, or a gain matrix to weight differently translation and rotation velocities.

### B. Discussion

We make several remarks about the presented idea.

• For elastic linear objects, there are well-known instabilities at certain configurations where the shape changes non-smoothly [6]. Our formulation is only valid in the regions of the object's shape space where the changes are stable and smooth, i.e., where a relation such as (9) can be defined.

• In [7], a matrix $\mathbf{K}$ that plays an analogous role to our $\mathbf{A}$ is used. That $\mathbf{K}$ is the tangent stiffness matrix of the FEM structure that represents the object, and it is a function of $\mathbf{q}$. This means one needs to estimate $\mathbf{q}$ (i.e., the full shape of the object) at run-time to compute the control law, even if only a few of the nodes are servoed and most are not. We avoid this requirement since our matrix $\mathbf{A}$ is constant and can be computed offline. This is important because measuring the shape of deformable objects robustly is challenging.

• $\mathbf{A}$ and then $\mathbf{J}_{sg}$ can be computed from the rest shape of the object (which is direct to know for a straight elastic rod) and

the knowledge of what nodes are gripped and what nodes are to be servoed. At run-time, our control law (11) only needs to compute $\mathbf{e}_s$, which requires measuring the positions of the servoed nodes only (not the shape of the full object).

• Even if our Jacobian is computed using the rest shape, this does not mean that it is only valid at the rest shape. On the contrary, it is the valid and exact Jacobian of the object, under the ASAP modeling, at every shape that satisfies our prior assumptions (stable deformation).

• Recent works performed successful shape control with approximated deformation models (FEM with imprecise parameters [7], ARAP [12]). By using the measurement of the object's shape in a feedback loop, the control laws in these works are convergent even if the deformation model is only coarsely accurate. This justifies our use of ASAP.

• Our controller idea directly accommodates multiple robots (grippers) without any change in the methodology. Just by knowing which nodes are gripped by each gripper, one can derive directly the control laws for all grippers.

• However, in order to use the described idea to control the shape of real-world objects, there remains a major issue: in general, ASAP is not a good model of the deformation of real-world elastic objects. We solve this issue by identifying specific conditions under which it becomes a good model, as explained in the next section.

## IV. ASAP AS AN ARAP APPROXIMATION

ASAP is not a good representation of the behavior of elastic objects because it does not preserve their physical size. In general, an object simulated using ASAP can easily shrink or grow in size unrealistically [20], [21]. On the other hand, ARAP is a good model of deformation behaviors of elastic objects [13], [23]. ARAP tends to preserve rigidity locally and by doing so, it keeps the size of the object under control, unlike ASAP does. Indeed, ARAP has been used recently for controlling the shape of 3D deformable objects [12]. However, ARAP is a nonlinear model, and its Jacobian depends on the current shape $\mathbf{q}$. Avoiding this dependency is precisely why we want to use ASAP.

Our triad-based formulation can be used both for ASAP and ARAP. For both models, an essential element is the computation of an optimal transformation for each triad. In ASAP, this transformation is a similarity, as described above: $\mathbf{H}_k$, for triad $k$. In ARAP, the transformation is a pure rotation: $\mathbf{R}_k$, for triad $k$. These transformations satisfy:

$$\mathbf{H}_k = s_k \mathbf{R}_k \quad \forall k. \tag{14}$$

Therefore, we can make the observation that if the condition $s_k = 1\ \forall k$ holds, ASAP and ARAP are equivalent. Motivated by this observation, we next show that this condition approximately holds in the scenario we consider.

### A. Geometric analysis for a triad

Our geometric modeling of a generic triad, shown in Fig. 2, is motivated next. This modeling assumes the rest shape of every triad is a straight line. Note that our modeling of the object is discrete. Hence, the shape of the object between

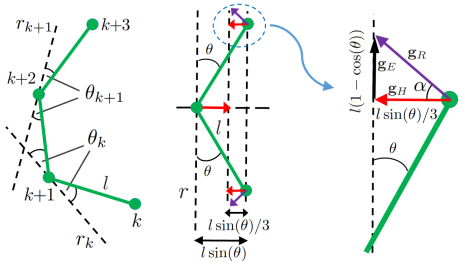


Fig. 2. Geometric modeling. Left: four nodes in two triads $\mathcal{T}_k$, $\mathcal{T}_{k+1}$, with their angles $\theta_k$, $\theta_{k+1}$. Center: a generic triad. Right: close-up of a region of the center plot. Nodes are shown as circles, gradient vectors as arrows.

two adjacent nodes is represented as a straight line segment. Recall that the object's length (equal to the sum of the segments' lengths in our modeling) is fixed. Therefore all segments always have equal and fixed lengths ($l$). The change of curvature along the object is represented via the angle $\theta$ relative to $r$, which is the line tangent to the object's curve according to our discrete modeling. Due to the symmetry of the layout, the variables of the ASAP/ARAP models can be obtained via direct trigonometric relations. We can obtain:

$$s = \cos(\theta). \tag{15}$$

Our goal was to have values of $s$ close to 1 (14): it is clear from (15) that this will happen when $\theta$ is small. Therefore, we want $\theta$ to be always small for all triads.

Notice that $\theta$ is small if the local curvature at every point is small: this will typically be true if $n$ is high enough to sample the object densely, and there are no extreme local deformations (i.e., very high bending at some point along the rod). Observe that the global deformation of the object can still be large even if the local curvature is small at every one of its points, as shown in our experiments (Sect. V). We used the gradients (which express the forces) of the ASAP energy to derive our controller. To support the use of this controller, next we study how well the ASAP gradients can approximate the ARAP ones. In Fig. 2, the ASAP gradient for the shown triad is denoted by $\mathbf{g}_H$, the ARAP gradient as $\mathbf{g}_R$, and the approximation error as $\mathbf{g}_E = \mathbf{g}_R - \mathbf{g}_H$. Due to the properties of least-squares alignment, the three endpoints of the gradient vectors form the rest (straight) shape and their centroid is the same as the nodes' centroid. Moreover, the vectors have certain symmetries due to the symmetry of the layout (the nodes form an isosceles triangle). We can obtain:

$$\alpha = \arctan \frac{||\mathbf{g}_E||}{||\mathbf{g}_H||} = \arctan \frac{3(1-\cos(\theta))}{\sin(\theta)}. \tag{16}$$

As $|\theta| \leq \pi/2\ rad$ by definition, (16) implies that $|\alpha| < \pi/2\ rad$. This means that the inner product of the true gradient vector ($\mathbf{g}_R$) and the one we use as its approximation ($\mathbf{g}_H$) is always non-negative. In gradient-based algorithms, this is usually a sufficient condition for obtaining the desired performance (e.g., in gradient descent). Note that we do not need to study the center node of the triad, because for that node $\mathbf{g}_H = \mathbf{g}_R$, i.e., the approximation is exact.

Overall, this geometric analysis supports the appropriateness of approximating elastic behavior with an ASAP model

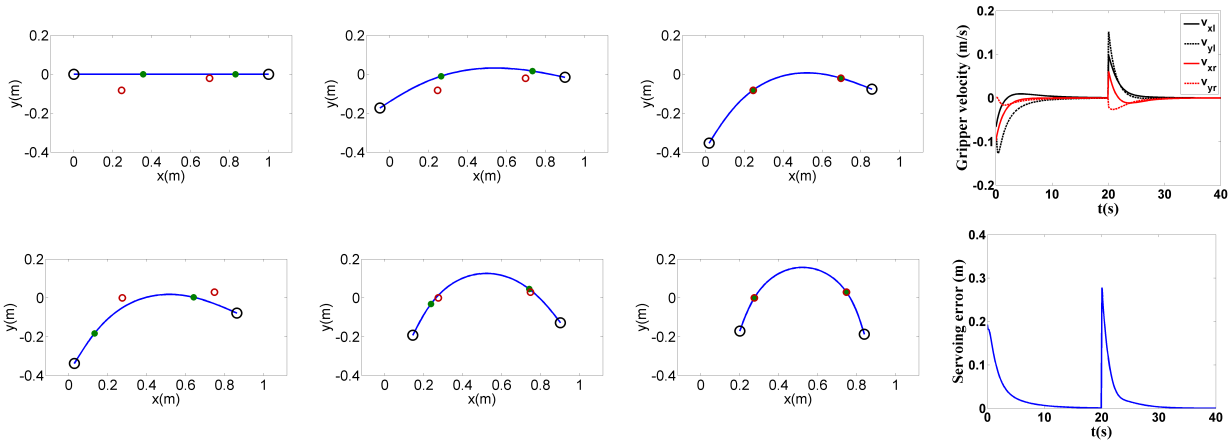


Fig. 3. Simulation results. Six sequential $xy$ plots are displayed, showing convergence to the two targets (third plots from the left on both rows). The targets are represented by small red hollow circles, the controlled points (servoed nodes) by small solid green circles, and the robotic grippers by big black hollow circles. Time plots show servoing error ($||\mathbf{e}_s||$) and $x, y$ gripper velocities (subscript $l$: left gripper, $r$: right gripper).

in our scenario. Note that this is only a partial analysis restricted to one triad; for each point, its total gradient is the sum of gradients over all the triads the point belongs to.

## V. Experimental Validation

We test the approach in simulation and in hardware experiments with real objects, in different scenarios[1].

### A. Simulation

We use Matlab and model with ARAP an elastic rod of length $1\,m$ and $n = 60$. We use two grippers, one at each end-node. We do not consider any anchored regions. We implement (11), i.e., we do not consider gripper rotation. Two control stages are executed, each with a different set of two servoed nodes and different desired values. As seen in Fig. 3, the two grippers move the object to make the servoed nodes reach their desired positions. The value of $s$ for all triads always stayed between $0.995$ and $1$, which is consistent with our analysis in Sect. IV-A.

### B. Robotic experiments with real objects

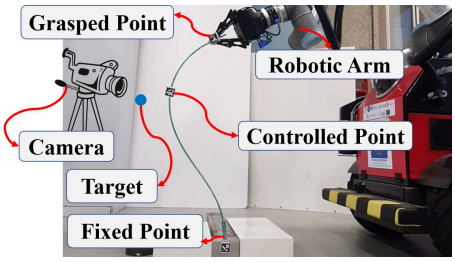


Fig. 4. The experimental setup used in this work.

The proposed method is implemented in a robotic manipulation system to study its performance, as shown in Fig. 4. The experiments are performed with a UR10 arm. Fifty points are used to discretize the object (i.e., $n = 50$). Other values of $n$ in a similar range (several tens) produce similar results. The controlled points (servoed nodes) and the gripped points are tracked with a fixed Logitech C270 camera, and ArUco markers have been used to obtain the pose of these points using the C++ OpenCV library. To validate the performance of the proposed method, various experiments are performed to control one, two, and three different points along the length of the objects. We use three different deformable linear objects with various characteristics to conduct the tests: a thin flexible rod made of plastic with the length of $0.87\ m$, a deformable foam rod with the length of $0.87\ m$, and a small deformable foam rod with the length of $0.26\ m$. Note that by selecting an appropriate $n$, the object's length does not affect the controller's performance. The Jacobian matrix is obtained using the proposed method offline, and then, the states of the gripped points are updated using control law (13) at each instant. In the plots of this section, the error of each controlled point is defined from (10) as follows ($\iota$ indicates the controlled point number):

$$error_\iota = \sqrt{(x_{s-\iota} - x_{sd-\iota})^2 + (y_{s-\iota} - y_{sd-\iota})^2}, \qquad (17)$$

where $[x_{s-\iota},\ y_{s-\iota}]^\top$ is the current position of the controlled point and $[x_{sd-\iota},\ y_{sd-\iota}]^\top$ is its desired position.

To investigate the controller's performance, in the first step, we try to control pose (position, and angle of the line tangent to the object's curve) of a single point along the object's length. To do that, the pose of the point is obtained using the position of its two closest nodes. Several experiments have been performed to validate the accuracy of the proposed algorithm, and the results are shown in Fig. 5 to Fig. 7. As one can see, by applying the controller during the deformation process, the errors of the controlled points converge to zero. In these experiments, we define $error_\beta = \beta_{s-1} - \beta_{sd-1}$ where $\beta_{s-1}$ and $\beta_{sd-1}$ are the current and target value of the angle of the point, respectively.

In the next step, we try to control position of two and three non-adjacent points using the proposed method. It is evident

[1]A video of our simulation and experimental results is attached and can be found at: https://www.youtube.com/watch?v=LI8JdTFPlR8

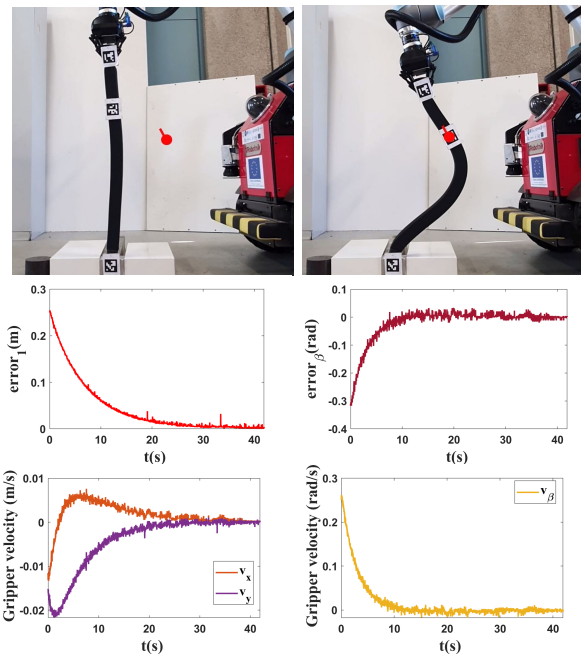


Fig. 5. Experiment 1. Pose control of one point. First row: initial and final shapes. Target point shown as a red circle, target angle as a line segment. Second row: errors plots. Third row: gripper velocities on different axes.

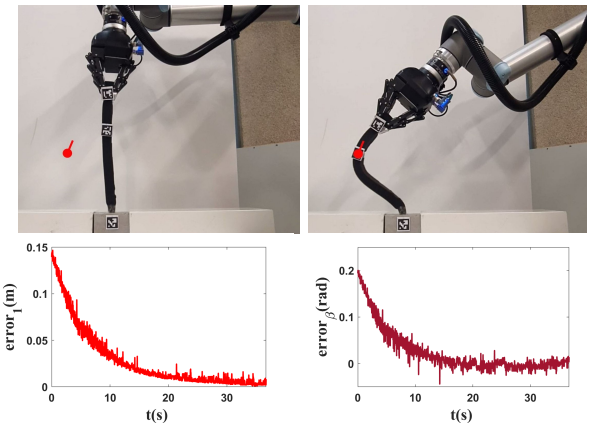


Fig. 6. Experiment 2. Pose control of one point. The initial and final shapes can be found in the first row. The errors are plotted in the second row.

that the system becomes underactuated since the number of actuated degrees of freedom is lower than the number of degrees of freedom of the controlled points. Hence, the system can only be stable in a local sense. The final shape of the object and the errors of the controlled points concerning their targets are presented in Fig. 8 to Fig. 10. As one can see, the errors converge to zero, and the presented method achieves its objective. The initial and target states of the controlled points in all experiments are presented in Table I.

## VI. Conclusion and Future Work

In this paper, we have investigated the problem of controlling deformable linear objects. We have presented an approach to control one or several arbitrary points along the length of an object. The proposed approach allows deforming elastic linear objects in a controlled way and needs only

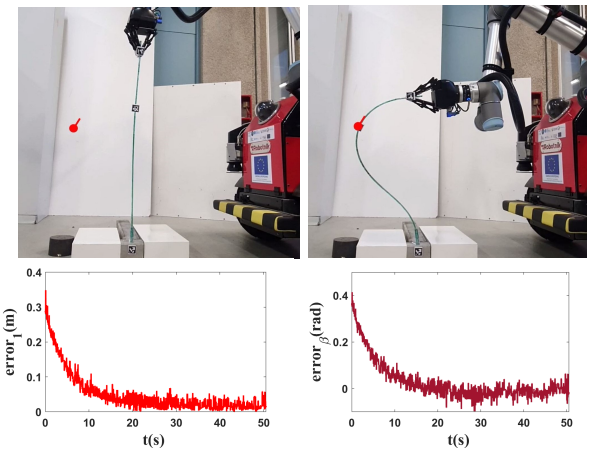


Fig. 7. Experiment 3. Pose control of one point. The initial and final shapes can be found in the first row. The errors are plotted in the second row.

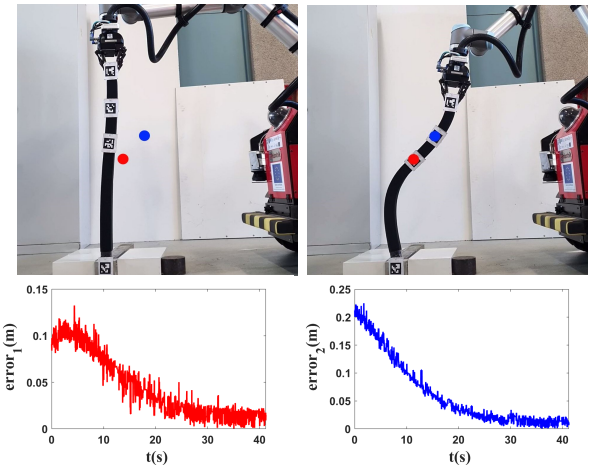


Fig. 8. Experiment 4. Position control of two points. The initial and final shapes of the object are indicated in the first row. The errors of the two controlled points are plotted in the second row.

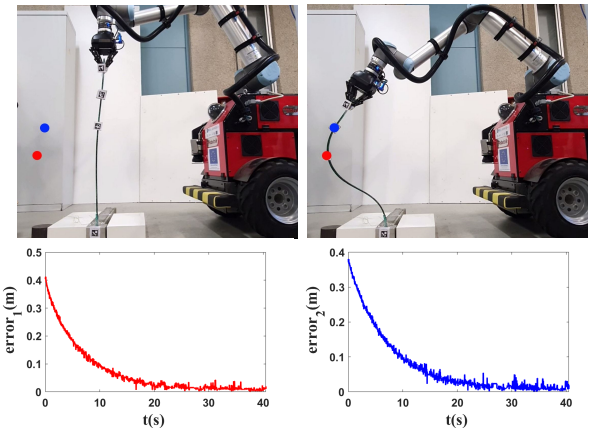


Fig. 9. Experiment 5. Position control of two points. The initial and final shapes of the object are indicated in the first row. The errors of the two controlled points are plotted in the second row.

a simple offline geometric model, which is an important practical advantage. The limitations of the approach include: it is restricted to 2D workspaces, it cannot handle non-smooth unstable deformation behaviors, the type of objects

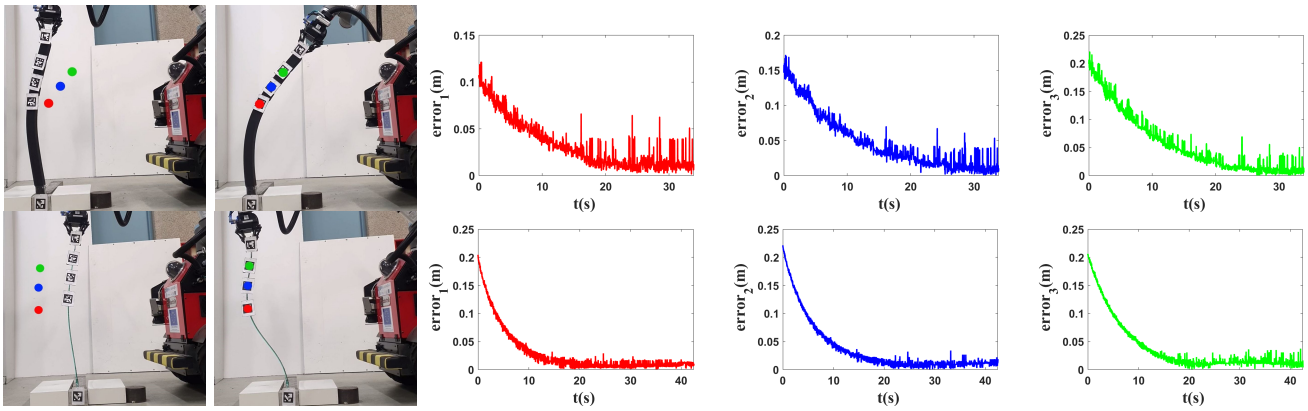


Fig. 10. Experiments 6 (top row) and 7 (bottom). Position control of three points. From the left in each row: initial shape, final shape, error of each point.

it can be applied on (fixed-length elastic DLOs) is limited, and it assumes the grasping remains fixed all throughout the control task. Overcoming some of these limitations is a clear avenue for future work. Another relevant problem is how to optimize the object's deformation trajectory. We are also interested in exploring the use of our formulation to find suitable grasping points for a given shape control task.

TABLE I

STATE OF THE CONTROLLED POINTS IN THE PERFORMED EXPERIMENTS.

| Experiment number | Initial pose $(m, m, rad)$ | Target pose $(m, m, rad)$ |
|---|---|---|
| **1** | $[0.086, 0.650, 1.609]$ | $[0.296, 0.502, 1.915]$ |
| **2** | $[-0.027, 0.266, 1.467]$ | $[-0.146, 0.198, 1.258]$ |
| **3** | $[0.019, 0.678, 1.571]$ | $[-0.283, 0.551, 1.144]$ |
| | Initial position $(m, m)$ | Target position $(m, m)$ |
| **4** | $[0.047, 0.601]$<br>$[0.072, 0.772]$ | $[0.096, 0.505]$<br>$[0.226, 0.619]$ |
| **5** | $[0.037, 0.564]$<br>$[0.072, 0.735]$ | $[-0.345, 0.425]$<br>$[-0.273, 0.584]$ |
| **6** | $[-0.065, 0.529]$<br>$[-0.044, 0.640]$<br>$[-0.012, 0.753]$ | $[0.040, 0.519]$<br>$[0.101, 0.614]$<br>$[0.177, 0.715]$ |
| **7** | $[-0.044, 0.529]$<br>$[-0.024, 0.659]$<br>$[-0.006, 0.787]$ | $[-0.232, 0.470]$<br>$[-0.234, 0.601]$<br>$[-0.204, 0.719]$ |

## REFERENCES


[1] H. Yin, A. Varava, and D. Kragic, "Modeling, learning, perception, and control methods for deformable object manipulation," *Science Robotics*, vol. 6, no. 54, p. eabd8803, 2021.

[2] J. Zhu, B. Navarro, P. Fraisse, A. Crosnier, and A. Cherubini, "Dual-arm robotic manipulation of flexible cables," in *2018 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2018, pp. 479–484.

[3] R. Lagneau, A. Krupa, and M. Marchal, "Automatic shape control of deformable wires based on model-free visual servoing," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5252–5259, 2020.

[4] 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 Trans. Mech.*, pp. 1–12, 2021.

[5] O. Aghajanzadeh, M. Aranda, J. A. Corrales Ramon, C. Cariou, R. Lenain, and Y. Mezouar, "Adaptive deformation control for elastic linear objects," *Frontiers in Robotics and AI*, vol. 9, p. 868459, 2022.

[6] T. Bretl and Z. McCarthy, "Quasi-static manipulation of a Kirchhoff elastic rod based on a geometric analysis of equilibrium configurations," *Int. Journ. of Rob. Research*, vol. 33, no. 1, pp. 48–68, 2014.

[7] A. Koessler, N. Roca Filella, B. C. Bouzgarrou, L. Lequièvre, and J.-A. Corrales Ramon, "An efficient approach to closed-loop shape control of deformable objects using finite element models," in *2021 IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2021, pp. 1637–1643.

[8] A. Sintov, S. Macenski, A. Borum, and T. Bretl, "Motion planning for dual-arm manipulation of elastic rods," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6065–6072, 2020.

[9] D. Berenson, "Manipulation of deformable objects without modeling and simulating deformation," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013, pp. 4525–4532.

[10] M. Aranda, G. López-Nicolás, and Y. Mezouar, "Distributed linear control of multirobot formations organized in triads," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 8498–8505, 2021.

[11] M. Aranda, J. A. Corrales Ramon, Y. Mezouar, A. Bartoli, and E. Özgür, "Monocular visual shape tracking and servoing for isometrically deforming objects," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 7542–7549.

[12] M. Shetab-Bushehri, M. Aranda, Y. Mezouar, and E. Özgür, "As-rigid-as-possible shape servoing," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3898–3905, 2022.

[13] O. Sorkine and M. Alexa, "As-rigid-as-possible surface modeling," in *Eurographics Symp. on Geometry Processing*, 2007, pp. 109–116.

[14] M. Müller, B. Heidelberger, M. Hennix, and J. Ratcliff, "Position based dynamics," *Journal of Visual Communication and Image Representation*, vol. 18, no. 2, pp. 109–118, 2007.

[15] M. Yan, Y. Zhu, N. Jin, and J. Bohg, "Self-supervised learning of state estimation for manipulating deformable linear objects," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2372–2379, 2020.

[16] Z. Hu, T. Han, P. Sun, J. Pan, and D. Manocha, "3-D deformable object manipulation using deep neural networks," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4255–4261, 2019.

[17] M. Yu, H. Zhong, and X. Li, "Shape control of deformable linear objects with offline and online learning of local linear deformation models," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 1337–1343.

[18] R. Laezza and Y. Karayiannidis, "Learning shape control of elastoplastic deformable linear objects," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 4438–4444.

[19] 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 Trans. Rob.*, vol. 34, no. 1, pp. 272–279, 2018.

[20] T. Igarashi, T. Moscovich, and J. F. Hughes, "As-rigid-as-possible shape manipulation," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 1134–1141, 2005.

[21] L. Liu, L. Zhang, Y. Xu, C. Gotsman, and S. J. Gortler, "A local/global approach to mesh parameterization," in *Proceedings of the Symposium on Geometry Processing*, 2008, pp. 1495–1504.

[22] R. Chen and C. Gotsman, "Generalized as-similar-as-possible warping with applications in digital photography," *Computer Graphics Forum*, vol. 35, no. 2, pp. 81–92, 2016.

[23] Z. Levi and C. Gotsman, "Smooth rotation enhanced as-rigid-as-possible mesh animation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 21, no. 2, pp. 264–277, 2015.

[24] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel, "Laplacian surface editing," in *Proc. 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, 2004, pp. 175–184.

[25] S. Duenser, J. Bern, R. Poranne, and S. Coros, "Interactive robotic manipulation of elastic objects," in *2018 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2018, pp. 3476–3481.