

Shape Control of Elastic Objects Based on Implicit Sensorimotor Models and Data-Driven Geometric Features

Wanyu Ma¹, Jihong Zhu² and David Navarro-Alarcon¹

¹The Hong Kong Polytechnic University, Hong Kong

²TU Delft and Honda Research Institute, Netherlands
wanyu.ma@connect.polyu.hk

Abstract. This paper proposes a general approach to design automatic controls to manipulate elastic objects into desired shapes. The object’s geometric model is defined as the shape feature based on the specific task to globally describe the deformation. Raw visual feedback data is processed using classic regression methods to identify parameters of data-driven geometric models in real-time. Our proposed method is able to analytically compute a pose-shape Jacobian matrix based on implicit functions. This model is then used to derive a shape servoing controller. To validate the proposed method, we report a detailed experimental study with robotic manipulators deforming an elastic rod.

Keywords: Robotics, visual servoing, deformable objects, shape control

1 Introduction

The automatic manipulation of rigid objects is one of the canonical problems in robot control; It has been extensively studied in the literature for more than five decades and — to a great extent — is considered a solved problem. In recent years, the manipulation of soft deformable objects has attracted the attention of many robotic researchers, mostly due to its multiple potential applications, e.g. palpation of tissues, shaping food materials, handling cables, manipulating fabrics, etc.

One of the major distinction between manipulation of rigid and deformable objects is the latter’s shape changes during manipulation. Research thus have been focused on controlling the shape of deformable objects. The deformation is often introduced by force on the objects. Therefore, at the very beginning, most researchers preferred to model the deformable object based on accurate physical mechanisms. The most popular methods are the mass-spring-damping model [1] and finite element model [2], which require to estimate the elastic parameters, such as Young’s modulus and Poisson’s ratio. However, it is impossible to exactly analyze force and deformation and estimate physical parameters for each object to be manipulated since the soft object may be non-homogeneous and the properties at one point of the object could even change over time or the contact with the environment.

Instead of considering force-based modeling, which often requires prior knowledge of the object’s deformation property, later research use vision to perceive

the shape as a feedback for deformation control. The area was termed shape servoing [3]. Shape servoing controls the manipulator for deforming the soft object from the current to the desired shape based on visual feedback.

Image data is often high dimensional hence not directly usable for control. Instead, we represent the shape with feature vectors [4]. One of the key research question in shape servoing is feature selections. Many features has been proposed for shape control, such as points [5], angles [6], curvature [7], eigenspace [8], catenary [9], etc. Shape feature with fewer variables improves the controllability of the deformation. Besides, learning or estimating techniques [10] can obtain the relationship between deformation and robotic pose. Although these methods don't require physical models, collecting data and training, repeated with the object changing in each specific task, highly increases workload. Hence, this work utilizes continuous geometry, such as curve or surface, to globally and analytically describe the deformable object, and builds up the mapping of deformation and robotic pose based on geometric relationship. The parameters of the geometric mapping are defined as the shape feature which is online identified without prior knowledge of objects' deformation property.

On the sensing side, most of works mentioned above utilize 2D images [11], as the feedback. However, 2D feedback will lose one dimension of information resulting in distortion when expressing the real physical world. With the development of the depth camera and the relevant processing methods, the application of 3D data starts to show its advantages. Therefore, more researchers attempt to use 3D data (mostly point cloud) to construct the 3D surface model of the soft object [12]. While, the unorganized raw feedback data from an RGB-D sensor are not able to tell the relationship of one point in the real world with the others, for example, if they belong to one object, or if they are neighbors. Hence, the classic point cloud processing approach is building topology geometric mesh to simulate and reconstruct their physical relationship [13]. Although it can obtain a precise surface structure, it results in heavy computing load and redundant information since sometimes the deformation only leads to slight changes in a topological mesh. Therefore, processing data using the topology model is not a good choice to design controllers which require updating the shape very quickly.

This paper propose a novel framework in shape servoing which makes contributions on both feature selection and 3D sensing: 1) We propose a 3D feature for shape servoing based on a continuous geometric model, and subsequently, using implicit function theorem to obtain an analytical Jacobian matrix; 2) On the sensing side, we process high dimensional feedback in real-time by dealing with unorganized raw visual feedback with less computational demanding.

The rest of this paper is organized as follows. Section 2 presents a general approach. Section 3 elaborates the mechanism of the proposed approach based on a specific case. Section 4 analyses and verifies our approach by robotic experiments. Finally, the conclusion is given in Section 5.

2 Methodology

In this section, an overview of the framework is given at first. Then, the detailed skills are later elaborated.

2.1 Overview

First, we determine the research target, that is, the type of soft object, the feedback data from the sensor, the desired result, etc. Then, according to this target, select a suitable geometric model to approximate the shape of the object. For example, when we study a linear deformable object, we could use a conic section on a spatial plane if the soft object is bent and use a helix if the soft object is twisted.

Second, we analyze the geometric relationship between the soft object and the pose of the robotic manipulator relying on the selected shape feature. Then, compute the analytical pose-shape Jacobian matrix by taking the partial derivatives of the pose of the manipulator with respect to the shape feature parameters. Since implicit functions are the most likely used to describe the geometric relationship, the implicit function theorem is introduced to get the pose-shape Jacobian matrix.

Finally, we design a velocity-based controller in the task space using the obtained analytical pose-shape Jacobian matrix, for this matrix describes the velocity relationship.

After the analysis is done, the control law is able to be used to update the control command for the manipulator in each loop. Once the control process starts, the raw data should be online fitted to the selected geometric model to identify shape feature parameters. This is a typical regression problem, which could be either linear or nonlinear. The frequent solutions are the Least Square Method (LSM), Gradient Descent, Newton's method, Quasi-Newton Methods, Conjugate Gradient, etc.

2.2 Online Identification of Shape Feature

In this section, the regression method is used to fit raw data to a data-driven geometric model identifying shape features in every control loop. We choose LSM for it is simple and fast. LSM [14] is a classical linear regression algorithm whose core is to identify the parameters of the model to minimize the sum of squared residual defined as the difference between the observed value by sample and the predicted value by model.

Let $\mathbf{S} = [\mathbf{s}_1, \dots, \mathbf{s}_N] \in \mathbb{R}^{n \times N}$ denote the unorganized raw feedback from sensors where \mathbf{s}_i is a single n -dimensional element of data. Assume there are N elements in set \mathbf{S} used for parameter identification. Denote m -dimensional parameters vector of shape feature as $\mathbf{y} \in \mathbb{R}^m$ which is online identified during the control process. The mapping between feedback \mathbf{s} and shape feature parameter \mathbf{y} satisfies $\mathbf{f}([\mathbf{s}^T, \mathbf{y}^T]^T) = \mathbf{0}_l$, where \mathbf{f} is an implicit function set comprising l functions $\mathbf{f} = \{f_1, \dots, f_l\}$, and $f_i (i = 1 \dots l)$ is a twice continuously differentiable function. The fitting process is an unconstrained optimization problem

$$\min_{\mathbf{y}} \left\{ \sum_{i=1}^l f_i^2([\mathbf{s}^T, \mathbf{y}^T]^T) : \mathbf{s} \in \mathbb{R}^n, \mathbf{y} \in \mathbb{R}^m \right\}. \quad (1)$$

2.3 Derivation of Analytical Jacobian Matrix

Jacobian matrix is a tool to describe the velocity relationship which is obtained from the first-order partial derivatives of the displacement mapping. Tradition-

ally, the mapping should be explicit, while it is hard to obtain for the selected geometric model. Hence, to derive a Jacobian matrix based on implicit mapping, we firstly introduce implicit function theorem [15, 16] as follows:

Lemma 1 (Implicit function theorem). *Let $\mathbf{h}([\mathbf{x}^T, \mathbf{y}^T]^T) = \mathbf{0} : \mathbb{R}^{q+m} \rightarrow \mathbb{R}^m$ be a continuously differentiable function of two sets of variables, $\mathbf{x} \in \mathbb{R}^q$ and $\mathbf{y} \in \mathbb{R}^m$. If the Jacobian matrix $\mathbf{J}_{\mathbf{h}, \mathbf{x}} = \frac{\partial \mathbf{h}}{\partial \mathbf{x}} \in \mathbb{R}^{(q+m) \times q}$ is invertible, the Jacobian matrix of \mathbf{x} with respect to \mathbf{y} is given by the matrix product*

$$\mathbf{J}_{\mathbf{x}, \mathbf{y}} = \frac{\partial \mathbf{x}}{\partial \mathbf{y}} = - \left[\frac{\partial \mathbf{h}}{\partial \mathbf{x}} \right]^{-1} \frac{\partial \mathbf{h}}{\partial \mathbf{y}} \in \mathbb{R}^{q \times m} \quad (2)$$

Then, let $\mathbf{x} \in \mathbb{R}^q$ denote the pose of the robot end-effector, which is the feedback from the robotic manipulator. The analytical pose-shape Jacobian matrix $\mathbf{J}_S \in \mathbb{R}^{q \times m}$ is defined as:

$$\dot{\mathbf{x}} = \mathbf{J}_S \dot{\mathbf{y}} \quad (3)$$

To obtain the Jacobian matrix \mathbf{J}_S , geometric relation between shape feature \mathbf{y} and the pose of the end-effector \mathbf{x} should be developed:

$$\mathbf{h}([\mathbf{y}^T, \mathbf{x}^T]^T) = \mathbf{0}_p \quad (4)$$

and \mathbf{h} has p functions $\mathbf{h} = \{h_1, \dots, h_p\}$ where each $h_i (i = 1 \dots p)$ is continuously differentiable, probably either a linear function or a nonlinear function. Then, we make derivative of Eq.(4):

$$\mathbf{h}' = [\mathbf{J}_1 \ \mathbf{J}_2] \begin{bmatrix} \dot{\mathbf{y}} \\ \dot{\mathbf{x}} \end{bmatrix} = \mathbf{0}_p, \text{ where } \mathbf{J}_1 = \frac{\partial \mathbf{h}}{\partial \mathbf{y}} \in \mathbb{R}^{p \times m}, \ \mathbf{J}_2 = \frac{\partial \mathbf{h}}{\partial \mathbf{x}} \in \mathbb{R}^{p \times q} \quad (5)$$

According to Eq.(5) and Lemma 1, we yield $\mathbf{J}_1 \dot{\mathbf{y}} + \mathbf{J}_2 \dot{\mathbf{x}} = \mathbf{0}_p \Rightarrow \dot{\mathbf{x}} = -\mathbf{J}_2^\dagger \mathbf{J}_1 \dot{\mathbf{y}}$. Therefore, we obtain the Jacobian matrix \mathbf{J}_S as follows:

$$\mathbf{J}_S = -\mathbf{J}_2^\dagger \mathbf{J}_1 \quad (6)$$

Note that, in this paper, we use pseudoinverse trick to take inverse of a matrix since the matrix is probably not a full-rank matrix.

2.4 Designing Controller

The desired shape feature is denoted as \mathbf{y}_d . The proposed shape servoing controller aims at minimizing $\mathbf{e} = \mathbf{y} - \mathbf{y}_d$ through robotic motion [17, 18]. Given the desired shape \mathbf{y}_d and its deformation velocity $\dot{\mathbf{y}}_d$, a task space controller can be designed based on analytical Jacobian matrix \mathbf{J}_S and online updating shape-feature parameters \mathbf{y} . The desired velocity of the manipulator end-effector can be obtained by

$$\dot{\mathbf{x}} = \mathbf{J}_S (\dot{\mathbf{y}}_d - \mathbf{K}(\mathbf{y} - \mathbf{y}_d)) \quad (7)$$

where \mathbf{K} is a positive definite gain matrix. The desired end-effector trajectory \mathbf{x} is obtained by numerical integration.

To proof the stability, we select a Lyapunov function $V = \frac{1}{2} \mathbf{e}^T \mathbf{e}$, where $\mathbf{e} = \mathbf{y} - \mathbf{y}_d$. Take the derivative of V to yield $\dot{V} = \mathbf{e}^T \dot{\mathbf{e}}$, where $\dot{\mathbf{e}} = \dot{\mathbf{y}} - \dot{\mathbf{y}}_d$. Submit Eq.(7) in it, we yield $\dot{\mathbf{e}} = \mathbf{J}_S^\dagger \dot{\mathbf{x}} - [\mathbf{J}_S^\dagger \dot{\mathbf{x}} + \mathbf{K}(\mathbf{y} - \mathbf{y}_d)] = -\mathbf{K}\mathbf{e}$. Then, submit this $\dot{\mathbf{e}}$ into \dot{V} , the derivative becomes $\dot{V} = -\mathbf{e}^T \mathbf{K} \mathbf{e} \leq 0$. Finally, the global stability of the controller is proofed.

3 Case of Study: Bending Based on Spatial Arc

To demonstrate how the general approach works, a specific deformable object manipulation task is going to be solved under the proposed framework.

3.1 Analysis of the Task

We select a linear elastic deformable object as the research target. A vision-based shape servoing controller will be designed to bend the object into the desired curvature and position defined by a continuous geometric curve. The setup consisting of an RGB-D camera and a robotic arm will be used to conduct the experiment, of which, the conceptual demonstration is shown in Fig.1.

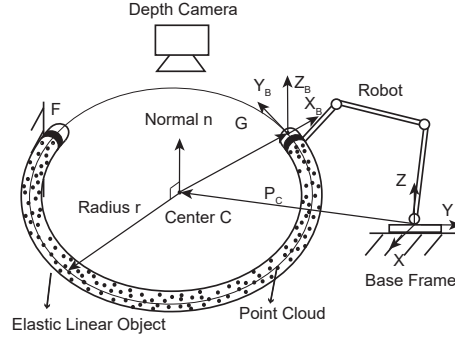


Fig. 1. Conceptual illustration of the setup and the shape feature (spatial arc).

For the deformable object, one end tip is fixed at a known point F and the other tip is grasped at point G by a robot arm without relative motion. Both F and G are on the surface of the deformable object. The feedback provided by the robotic manipulator includes three-axial Cartesian coordinates and Euler angles under the rotation sequence Z-Y-Z, which describe respectively position and orientation of the end-effector, that is, $q = 6$. Denote Euler angles as $[x_1, x_2, x_3]^T$ and Cartesian position as $[x_4, x_5, x_6]^T$, forming 6D robotic pose $\mathbf{x} = [x_1, x_2, x_3, x_4, x_5, x_6]^T$. It is assumed that the deformable object in this work has elastic performance maintaining its shape against the manipulation of the robot arm, and gravity, occlusion, overlapping, crossing, twisting are not considered.

Here gives some denotations. The vector $\mathbf{p}_A = [x_A, y_A, z_A]^T$ represents the position of point A with respect to the robot base frame. The vector $\mathbf{d}_A = [d_{Ax}, d_{Ay}, d_{Az}]^T = \mathbf{p}_G - \mathbf{p}_A$ represents the vector from point A to the robot end-effector, that is, the grasped point G in this paper.

The selected shape feature is spatial arc, whose description is a part of a spatial circle composed of radius, center, and normal vector of the plane where the circle is, respectively denoted as r , $\mathbf{p}_C = [x_C, y_C, z_C]^T$, and $\mathbf{n} = [n_x, n_y, n_z]^T$. Hence, the shape feature is reformed as a parameter vector $\mathbf{y} = [r, \mathbf{p}_C^T, \mathbf{n}^T]^T \in \mathbb{R}^7$. The normal vector is limited to a unit vector. Fig.1 demonstrates the component of a 3D circle and how the spatial conic section represents the deformable object based on the setup.

3.2 Analytical Jacobian Matrix

We separately analyze the orientation Jacobian matrix and position Jacobian matrix when we compute analytical Jacobian matrix \mathbf{J}_S between the deformation velocity $\dot{\mathbf{y}}$ and the end-effector velocity $\dot{\mathbf{x}}$. It is reasonable because we usually decouple the geometric relationship from orientation and position. The orientation Jacobian matrix and position Jacobian matrix are denoted by \mathbf{J}_{SO} and \mathbf{J}_{SP} , forming the pose-shape Jacobian matrix $\mathbf{J}_S = [\mathbf{J}_{SO}^T, \mathbf{J}_{SP}^T]^T$.

To derive the mapping of pose and deformation in terms of orientation, we define two frames, namely, the body frame of the deformable object and the end-effector frame, respectively denoted as \mathbf{F}_B and \mathbf{F}_E . Corresponding rotation matrices \mathbf{R}_B is computed by shape features \mathbf{y} , and \mathbf{R}_E is computed by robot pose \mathbf{x} . The body frame is represented by $\{\mathbf{F}_B\} = \{G, \mathbf{X}_B, \mathbf{Y}_B, \mathbf{Z}_B\}$.

Since there is no relative motion between the end-effector and the grasped object, we assume that \mathbf{F}_B has a constant transformation with respect to \mathbf{F}_E , namely, $\mathbf{R}_E = {}^E\mathbf{R}_B \mathbf{R}_B$, where the constant orientation relationship ${}^E\mathbf{R}_B$ can be initialized by ${}^E\mathbf{R}_B = \mathbf{R}_{E0} \mathbf{R}_{B0}^T$.

To describe the orientation of the deformable object, the body frame $\{\mathbf{F}_B\}$ should be defined, where origin is the grasping point G , axis \mathbf{X}_B is a unit vector along vector \mathbf{d}_C , axis \mathbf{Z}_B is a unit vector along the normal \mathbf{n} , and axis \mathbf{Y}_B is determined by right-hand rule. That is, $\mathbf{X}_B = \frac{\mathbf{d}_C}{\|\mathbf{d}_C\|}$, $\mathbf{Z}_B = \mathbf{n}$, $\mathbf{Y}_B = \mathbf{Z}_B \times \mathbf{X}_B$. The orientation of the body frame $\{\mathbf{F}_B\}$ is described by rotation matrix $\mathbf{R}_B = \begin{bmatrix} \frac{\mathbf{d}_C}{\|\mathbf{d}_C\|}, \frac{\mathbf{n} \times \mathbf{d}_C}{\|\mathbf{d}_C\|}, \mathbf{n} \end{bmatrix}$. The rotation matrix \mathbf{R}_E can be obtained by Euler angles under the rotation sequence Z-Y-Z:

$$\mathbf{R}_E = \mathbf{R}_Z(x_1) \mathbf{R}_Y(x_2) \mathbf{R}_Z(x_3) = \begin{bmatrix} c_1 c_2 c_3 - s_1 s_3 & -c_1 c_2 s_3 - s_1 c_3 & c_1 s_2 \\ s_1 c_2 c_3 + c_1 s_3 & -s_1 c_2 s_3 + c_1 c_3 & s_1 s_2 \\ -s_2 c_3 & s_2 s_3 & c_2 \end{bmatrix} \quad (8)$$

and $s_1, c_1, s_2, c_2, s_3, c_3$ represent respectively $\sin x_1, \cos x_1, \sin x_2, \cos x_2, \sin x_3, \cos x_3$. Reshaping \mathbf{X}_B and \mathbf{Z}_B into a vector with approximating $\|\mathbf{d}_C\| = r$ yields $\mathbf{V}_1 = \left[\frac{\mathbf{d}_C}{r}, \mathbf{n} \right]^T$. Reshaping the corresponding X axis and Z axis of \mathbf{R}_E into a vector yields $\mathbf{V}_2 = [c_1 c_2 c_3 - s_1 s_3, s_1 c_2 c_3 + c_1 s_3, -s_2 c_3, c_1 s_2, s_1 s_2, c_2]^T$. Then, the orientation mapping is:

$$\mathbf{h}_O = {}^E\mathbf{T}_B \mathbf{V}_1 - \mathbf{V}_2 = \mathbf{0}, \quad {}^E\mathbf{T}_B = \begin{bmatrix} {}^E\mathbf{R}_B & \mathbf{0}_{3 \times 3} \\ \mathbf{0}_{3 \times 3} & {}^E\mathbf{R}_B \end{bmatrix} \quad (9)$$

Therefore, the orientation Jacobian matrix is $\mathbf{J}_{SO} = -\mathbf{J}_2^\dagger \mathbf{J}_1 \in \mathbb{R}^{3 \times 7}$, where the Jacobian matrices $\mathbf{J}_1 = {}^E \mathbf{T}_B \tilde{\mathbf{J}}_1$ and \mathbf{J}_2 are:

$$\tilde{\mathbf{J}}_1 = \begin{bmatrix} -\frac{d_{Cx}}{r^2} & -\frac{1}{r} & 0 & 0 & 0 & 0 & 0 \\ -\frac{d_{Cy}}{r^2} & 0 & -\frac{1}{r} & 0 & 0 & 0 & 0 \\ -\frac{d_{Cz}}{r^2} & 0 & 0 & -\frac{1}{r} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{J}_2 = \begin{bmatrix} s_1 c_2 c_3 + c_1 s_3 & c_1 s_2 c_3 & c_1 c_2 s_3 + s_1 c_3 \\ -c_1 c_2 c_3 + s_1 s_3 & s_1 s_2 c_3 & s_1 c_2 s_3 - c_1 c_3 \\ 0 & c_2 c_3 & -s_2 s_3 \\ s_1 s_2 & -c_1 c_2 & 0 \\ -c_1 s_2 & -s_1 c_2 & 0 \\ 0 & s_2 & 0 \end{bmatrix} \quad (10)$$

Since there is no relative motion between the end-effector and the grasped object, we assume that the position of the grasped point is equal to the feedback position of the robot, namely, $\mathbf{p}_G = [x_4, x_5, x_6]^\top$.

According to the geometric information, the relationship between end-effector position and the shape feature $\mathbf{h}_P = \mathbf{0}$ can be described as follows:

$$\mathbf{h}_P = \begin{cases} \arccos \frac{\overrightarrow{CF} \cdot \mathbf{d}_C}{r^2} - \theta = 0 \\ \mathbf{n} \cdot \mathbf{d}_C = 0 \\ \mathbf{d}_C \cdot \mathbf{d}_C - r^2 = 0 \end{cases} \quad (11)$$

in which, θ is the angle swiping from \overrightarrow{CF} to \mathbf{d}_C through the object.

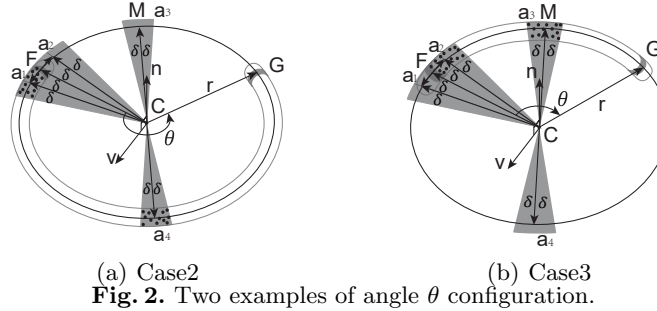


Fig. 2. Two examples of angle θ configuration.

Since the length of the manipulated deformable object is constant, the arc length L of the identified curve doesn't change either, which should limit and guide the robot moving. For example, as the configuration is shown in Fig.2(a), if we want the radius decreasing, the robot should move counterclockwise while if the configuration is like Fig.2(b), the robot should clockwise move to decrease the radius, which means, though the angle θ is the same in two configurations, the robot motion is different. In this way, it is necessary to determine the exact geometric relationship under the limitation of the arc length.

Firstly, note that, since the setup is predesignated, the manipulated object has a limited and fixed workspace. Thus, during the process of manipulation, we can set the normal vector \mathbf{n} always pointing to one side of the plane, though two vectors along opposite directions are both able to describe a plane. Then, we define a vector $\mathbf{v} = \mathbf{n} \times \overrightarrow{CF}$ and a small positive angle δ . $\mathbf{a}_1 = \overrightarrow{CF} + \delta \mathbf{v}$, $\mathbf{a}_2 = \overrightarrow{CF} - \delta \mathbf{v}$ approximately represent the vectors forming a small angle δ with \overrightarrow{CF} counterclockwise and clockwise. $\mathbf{a}_3 = \overrightarrow{CF} + \mathbf{d}_C$ is the midline of the acute angle between \overrightarrow{CF} and \mathbf{d}_C , and $\mathbf{a}_4 = -\mathbf{a}_3$ is the midline of the obtuse angle.

Surrounding $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_4$, generate four areas of feedback points. All vectors, starting from the center C pointing to the feedback point in the area, form an angle less than δ with $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_4$ respectively (as the shadows in Fig.2). Then, denote the number of the feedback points which is within the defined areas using N_1, N_2, N_3, N_4 , by computing the angles between the vectors. This detection technique to determine θ and initial L is demonstrated by Table 1. r_0 , C_0 , G_0 , and F_0 denote initial states of radius, center, grasped point, and fixed point.

	$N_1 \& N_2$	$N_3 \& N_4$	θ	L
Case1	$N_1 > N_2$	$N_3 > N_4$	$\frac{L}{r}$	$r_0 \cdot \arccos \frac{\overrightarrow{C_0 F_0} \cdot \overrightarrow{C_0 G_0}}{r_0^2}$
Case2	$N_1 > N_2$	$N_3 < N_4$	$2\pi - \frac{L}{r}$	$2\pi r_0 - r_0 \cdot \arccos \frac{\overrightarrow{C_0 F_0} \cdot \overrightarrow{C_0 G_0}}{r_0^2}$
Case3	$N_1 < N_2$	$N_3 > N_4$	$-\frac{L}{r}$	$r_0 \cdot \arccos \frac{\overrightarrow{C_0 F_0} \cdot \overrightarrow{C_0 G_0}}{r_0^2}$
Case4	$N_1 < N_2$	$N_3 < N_4$	$-2\pi + \frac{L}{r}$	$2\pi r_0 - r_0 \cdot \arccos \frac{\overrightarrow{C_0 F_0} \cdot \overrightarrow{C_0 G_0}}{r_0^2}$

Table 1. Detection technique to determine θ and initialize L

Then, we derive the pose-shape Jacobian matrix for position $\mathbf{J}_{SP} = -\mathbf{J}_2^\dagger \mathbf{J}_1 \in \mathbb{R}^{3 \times 7}$. Since there are four cases in Table 1, \mathbf{J}_2 and \mathbf{J}_1 also have four cases respectively. Here we give the solution for Case2 by computing the derivatives of \mathbf{h}_P (the other three cases should have similar solutions):

$$\mathbf{J}_1 = \begin{bmatrix} \frac{-2r\gamma\eta+L}{r^2} & -\gamma (\overrightarrow{CF}^T + \mathbf{d}_C^T) & \mathbf{0}_{1 \times 3} \\ 0 & -\mathbf{n}^T & \mathbf{d}_C^T \\ -2r & -2\mathbf{d}_C^T & \mathbf{0}_{1 \times 3} \end{bmatrix}, \quad \mathbf{J}_2 = [\gamma \overrightarrow{CF} \quad \mathbf{n} \quad 2\mathbf{d}_C]^T \quad (12)$$

where $\gamma = \frac{-1}{\sqrt{1-\eta^2}}$, and $\eta = \frac{\overrightarrow{CF} \cdot \mathbf{d}_C}{r^2}$.

4 Experiments and Results

In this section, we conduct experiments to validate our approach. We aim to achieve a converging controller to control the manipulator bend of a linear elastic soft object (elastic rod) into the desired shape. The real setup consists of an Intel RealSense D415 depth camera and a UR3 robot arm, where the depth camera is settled with the system in an eye-to-hand manner. The calibration is needed before the experiment.

Firstly, to validate the algorithm of shape feature identification, four point-clouds are collected under the different configurations of the manipulated object. The collected data is 10 times downsampled so that every point cloud has about 200 points. A filter is used to reduce the noise of raw visual feedback. Then, LSM is used to compute the shape feature parameters.

The identification results are visualized as Fig.3 which are observed from the same view. The green points are feedback point cloud. The red plane represents the orientation of the normal vector. The identified center and radius of the geometric model generate a complete circle which is drawn with a black solid line. The regression stops around 20 ms every loop, which can achieve real-time computation.

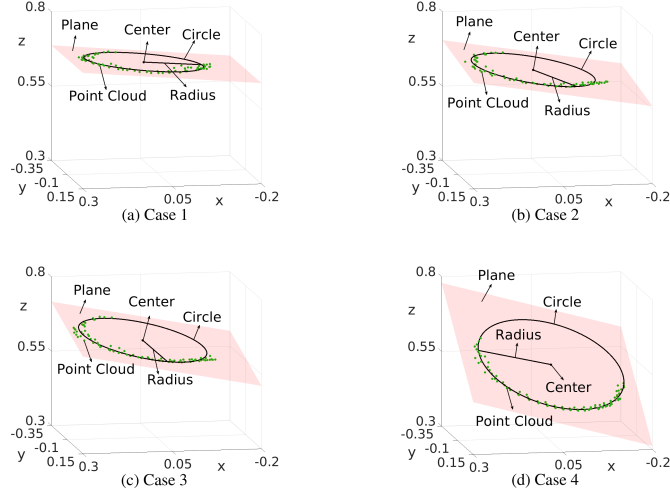


Fig. 3. The visualization of shape feature identification results.

Table 2 shows the corresponding statistic indexes including residual mean and residual variance of the regression. Each point's residual error is computed by $\sum_{i=1}^l f_i^2([\mathbf{s}^T, \mathbf{y}^T]^T)$ according to Eq.1, where \mathbf{s} is submitted by a feedback point and \mathbf{y} is submitted by the identified shape feature. All residual errors fall within 3 standard deviations of the mean, which indicates that residual errors follow a normal distribution according to the three-sigma rule so that the identification results are acceptable.

	Case 1	Case 2	Case 3	Case 4
Residual Mean	0.2385	0.2090	0.1897	0.0927
Residual Variance	0.0014	0.0034	0.0053	0.0045

Table 2. Residual mean and variance of arc fitting results.

After the parameters identification, the Jacobian matrix should be analyzed. In this paper, the analytical pose-shape Jacobian matrix is evaluated. According to Eq.3, given a set of shape feature differential, the pose differential can be computed, which directs the movement of the end-effector. Therefore, the ideal Jacobian matrix should make the computed pose differential using the feedback shape features has the same sign as the corresponding real feedback pose. Based on this analysis, we design a movement that the robot end-effector moves along a path without control, during which process, shape features and robotic poses are collected. Figure 4 shows the comparison of the feedback poses plots(in blue thick line) and computed poses plots(in red thin line), of which, the left three subfigures display Euler angles under the rotation sequence Z-Y-Z denoted by rz_1, ry_2, rz_3, respectively, and the right three subfigures display three-axial Cartesian coordinates denoted by x, y, z, respectively. All plotted data have been normalized into the range $[-1, 1]$.

Figure 4 demonstrates that mostly the computed poses have the same sign as the feedback posed, which validates the proposed algorithm. Note that nearly half part of the z-axial coordinate has the opposite sign. This is because of the z-axial coordinate changes very small during the designed movement, which make the computed pose more like an oscillation around zero. Furthermore, relatively

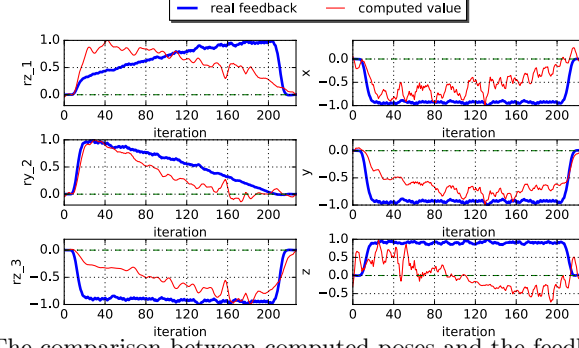


Fig. 4. The comparison between computed poses and the feedback poses.

small motion contributes little to the controller, of which the sign is not that important. So the computed plot going through zero is reasonable.

With the proposed analytical Jacobian matrix proved to validate, the experiment that robot manipulates the soft object into the desired shape under the designed shape servoing controller is conducted. We select all seven shape feature elements as the control target, including radius, center, and the normal vector of the spatial plane where the circle is. The desired shape is $\mathbf{y}_d = [0.209, -0.246, 0.359, -0.179, -0.207, 0.935, 0.289]^T$ and the initial state is $\mathbf{y}_0 = [0.159, -0.207, 0.444, -0.178, 0.067, 0.998, -0.012]^T$.

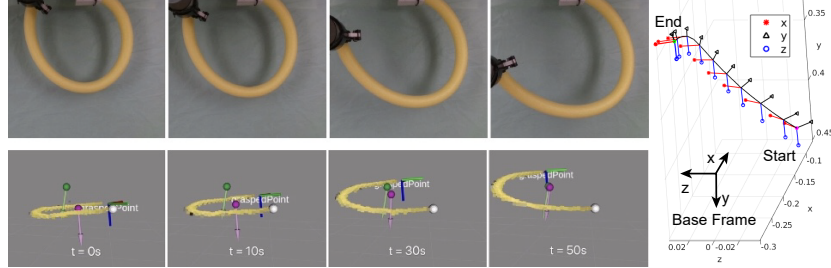


Fig. 5. The control process on physical setup and simulating visualization, and the trajectories of robotic poses.

Figure 5 demonstrates the control process (moments of $t=0s, 10s, 30s, 50s$ are selected) on the physical setup (top view) and simulating visualization (side view). The downside subfigures show the feedback point cloud under control. The white sphere is the fixed point of the deformable object. The frame located at the other tip (grasped point) is the defined body frame. The combination of the purple sphere and arrow represents the updating current center and normal vector. The combination of the green sphere and arrow represents the designed target center and normal vector. The movement that the purple combination tracks to the green combination illustrate the deformation process under the shape servoing controller. The right-side plot in Fig. 5 records the trajectory of robotic poses under the robot base frame during the experiment. The frames along the trajectory are robotic end-effector orientation.

Figure 6 shows shape errors plots recorded during the movement of the robot. The left three visualized errors are defined as $|r - r_d|$, $\arccos \mathbf{n} \cdot \mathbf{n}_d$, and $\|\mathbf{p}_C - \mathbf{p}_{Cd}\|$, and the right three visualized errors are respectively three coordinates of center, defined as $|x_C - x_{Cd}|$, $|y_C - y_{Cd}|$, $|z_C - z_{Cd}|$. It shows that shape errors converge in 50s.

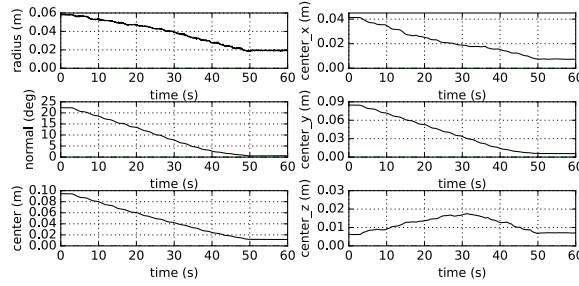


Fig. 6. The visualization of shape errors.

The stable errors in Fig.6 are less than 2 centimeters or degrees which is an acceptable magnitude. First, the point cloud has large noise while we did not use a complicated algorithm to obtain more accurate feedback data in order to reduce the calculation and achieve a real-time algorithm. Second, the depth camera is fixed and only obtains the point cloud from the exposed side of the manipulated object. So that the observing view is changing along with the deformation of the elastic rod whose radius of the cross-section (about 4 cm) can lead to a bias. Furthermore, on one hand, the defined geometric model is too ideal to approximate the whole object. On the other hand, in our study, the deformation serves the required motion, that is, bending in this experiment, so the absolutely accurate description of shape is not necessary.

5 Conclusion

In this work, we proposed a general approach to design a shape servoing controller for manipulating the deformable object into the desired shape. To demonstrate and validate the proposed approach, we designed a specific task of bending a elastic rod into the desired curvature with a point cloud as the sensory feedback. First, a continuous geometric model (spatial arc) is defined as the shape feature to globally describe deformation under bending. Second, use LSM to identify the parameters of the defined shape feature in real-time after filtering the raw visual feedback data. The statistic analysis indicates that the identification results are acceptable. Then, derive the analytical pose-shape Jacobian matrix based on implicit functions of the mapping of object deformation and robotic pose. The comparison between the computed pose and the real feedback pose during the open-loop movement of the robot proves the validation of the Jacobian matrix. Finally, the shape servoing controller based on velocity and the derived pose-shape Jacobian matrix enables the robot to manipulate the deformable object into the desired shape and achieve the desired motion in the task. The shape errors converge to acceptable stable errors. To sum up, both theory and experiment validate that the proposed approach can generally help design shape servoing controller based on the data-driven implicit model and achieve real-time control.

Acknowledgements. This work is supported in part by the Germany/Hong Kong Joint Research Scheme sponsored by the Research Grants Council of Hong Kong and the German Academic Exchange Service under grant G-PolyU507/18

References

1. M. Kimura, Y. Sugiyama, S. Tomokuni, and S. Hirai, “Constructing rheologically deformable virtual objects,” in *IEEE Int. Conf. on Robotics and Automation*, vol. 3, 2003, pp. 3737–3743.
2. A. Petit, F. Ficuciello, G. A. Fontanelli, L. Villani, and B. Siciliano, “Using physical modeling and rgb-d registration for contact force sensing on deformable objects,” 2017.
3. 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*, vol. 34, no. 1, pp. 272–279, 2017.
4. F. Chaumette and S. Hutchinson, “Visual servo control. i. basic approaches,” *IEEE Robotics & Automation Magazine*, vol. 13, no. 4, pp. 82–90, 2006.
5. Z. Wang, X. Li, D. Navarro-Alarcon, and Y.-h. Liu, “A unified controller for region-reaching and deforming of soft objects,” in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2018, pp. 472–478.
6. 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 Trans. on Robotics*, vol. 32, no. 2, pp. 429–441, 2016.
7. A. R. Fugl, A. Jordt, H. G. Petersen, M. Willatzen, and R. Koch, “Simultaneous estimation of material properties and pose for deformable objects from depth and color images,” in *Joint DAGM and OAGM Symposium*, 2012, pp. 165–174.
8. J. Zhu, D. Navarro-Alarcon, R. Passama, and A. Cherubini, “Vision-based manipulation of deformable and rigid objects using subspace projections of 2d contours,” 2020.
9. M. Laranjeira, C. Dune, and V. Hugel, “Catenary-based visual servoing for tether shape control between underwater vehicles,” *Ocean Engineering*, vol. 200, 2020.
10. A. Nair, D. Chen, P. Agrawal, P. Isola, P. Abbeel, J. Malik, and S. Levine, “Combining self-supervised learning and imitation for vision-based rope manipulation,” in *2017 IEEE Int. Conf. on Robotics and Automation*, 2017, pp. 2146–2153.
11. P. Güller, K. Pauwels, A. Pieropan, H. Kjellström, and D. Kragic, “Estimating the deformability of elastic materials using optical flow and position-based dynamics,” in *IEEE Int. Conf. on Humanoid Robots*, 2015, pp. 965–971.
12. 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.
13. J. Huang and C.-H. Menq, “Combinatorial manifold mesh reconstruction and optimization from unorganized points with arbitrary topology,” *Computer-Aided Design*, vol. 34, no. 2, pp. 149–165, 2002.
14. O. Axelsson, “A generalized conjugate gradient, least square method,” *Numerische Mathematik*, vol. 51, no. 2, pp. 209–227, 1987.
15. K. Jittorntrum, “An implicit function theorem,” *Journal of Optimization Theory and Applications*, vol. 25, no. 4, pp. 575–577, 1978.
16. S. Ha, S. Coros, A. Alspach, J. Kim, and K. Yamane, “Joint optimization of robot design and motion parameters using the implicit function theorem,” in *Robotics: Science and systems*, 2017.
17. S. Hutchinson and F. Chaumette, “Visual servo control, part i: Basic approaches,” *IEEE Robotics and Automation Magazine*, vol. 13, no. 4, pp. 82–90, 2006.
18. F. Chaumette and S. Hutchinson, “Visual servo control. ii. advanced approaches,” *IEEE Robotics & Automation Magazine*, vol. 14, no. 1, pp. 109–118, 2007.