ISSN 0005-1179 (print), ISSN 1608-3032 (online), Automation and Remote Control, 2025, Vol. 86, No. 2, pp. 165–190.

CONTROL IN TECHNICAL SYSTEMS

Visual Servoing for Deformable Objects with Pre-Planned Trajectory-Guided Geometric Primitives

D. D. Yasser[*,a], **S. S. Farah**[*,b], **and M. M. Abed**[**,c]

[*] *LAT, Laboratoire d'Automatique de Tlemcen, Universit de Tlemcen, Tlemcen, 13000, Algeria*
[**] *GBM, Laboratoire de Genie Biomedical, Universit de Tlemcen, Tlemcen, 13000, Algeria*
e-mail: [a] *derrar.yasser@gmail.com,* [b] *saidifarah.mimouni@gmail.com,* [c] *abed.malti@gmail.com*

Received February 13, 2024
Revised August 15, 2023
Accepted August 26, 2024

Abstract—This paper presents a novel technique for the rigid alignment of a perspective camera to the deformations of a non-rigid object, using an adaptive visual servoing approach. Unlike existing methods, the proposed approach does not rely on any parametric or model-based priors regarding the deformation. Assuming the availability of a pre-planned camera trajectory observing the non-rigid object, the method aims to align this trajectory during the execution phase, utilizing only the most relevant landmarks as prior information. Importantly, the approach does not depend on any parametric or non-parametric models of the underlying deformation physics. Instead, it is formulated as a tracking problem embedded within an optimal visual control framework. This tracking process involves the visual servoing of geometric features from the deformable object, bridging the gap between the planning and execution stages. The optimal visual control is defined using a weighted least-squares criterion, which minimizes the distance between the reference features and those observed in real-time. The weights are time-dependent, smooth functions that encode the relevance of the visible object features. The experimental results demonstrate the method?s ability to adapt to various pre-planned trajectories and different types of deformations, without requiring prior knowledge, while also exhibiting resilience to noise in the detection of image features.

Keywords: visual servoing, deforming objects, motion planning, trajectory tracking

DOI: 10.31857/S0005117925020057

1. INTRODUCTION

Visual servoing refers to the use of computer vision data, acquired from one or more cameras, to control their movement. Unlike vision-based tracking problems with rigid objects, for which a certain maturity has been reached, tracking by visual servoing on non-rigid objects is still a challenging problem. It has aroused much interest in recent years in the computer vision and robotics communities [1]. Many potential applications are targeted, in fields such as augmented reality, medical imaging, robotic manipulation, by handling a huge variety of objects: tissues, paper, rubber, viscous fluids, cables, food, organs, etc [2].

One characteristic of deformable object tracking is that the object shape changes while being tracked. Recent methods proposed a hybrid visual servoing decoupling the translational velocity in Z-axis and three rotational velocities. Such method has improved the performance of classical HVS (Hybrid Visual Servoing) in both image-plane and task space [3].

However, decoupling camera degrees of freedom put restrictions on the correction of camera motion and can lead to non-smooth tracking of the object's deformation. Tracking the geometry of deformable objects such as rope and cloth is difficult due to the continuous nature of the object

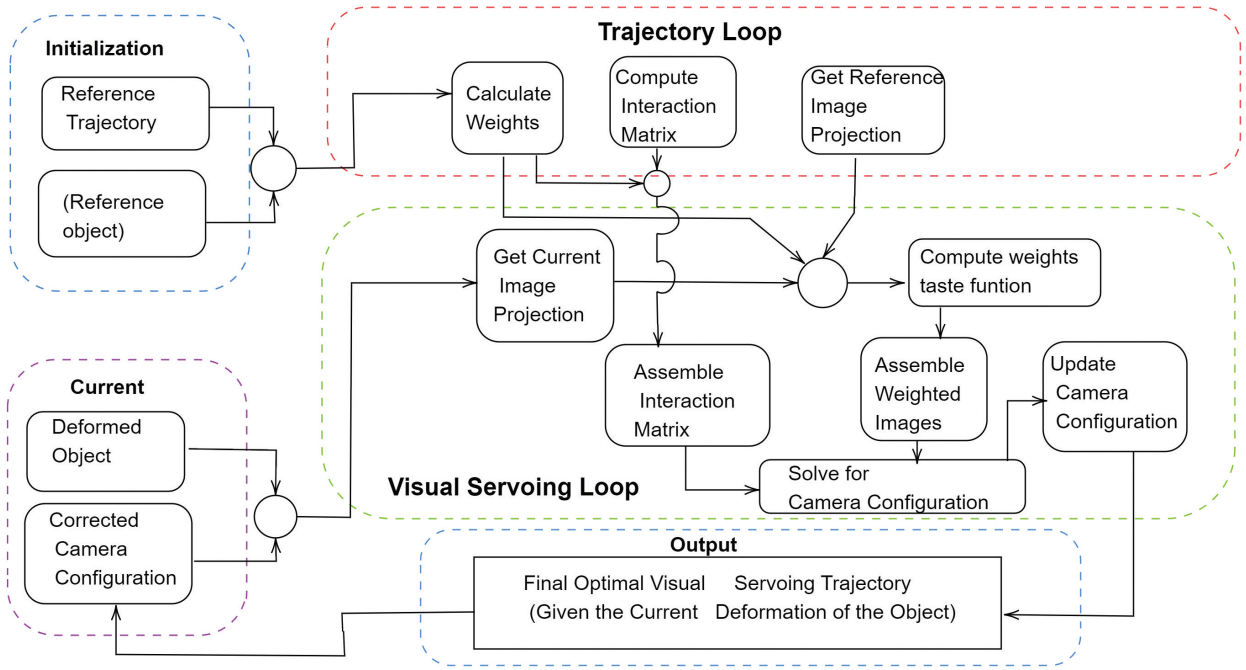

Fig. 1. Flowchart of the triangle-mesh-based visual servoing algorithm for deformable objects.

(i.e. an infinite number of degrees of freedom) [4, 5]. Most current method on tracking by visual servoing requires both the knowledge of physical properties of the object and real-time monitoring of the deformations. Several studies have focused on the monitoring of deformations in real time but not simultaneously with the control of the deformation [6]. These methods are highly parametric and require specific fine tuning that is strongly dependent on the physics of deformations. In this work, we propose a robust approach to track object's deformation by an optimal visual servoing. We assume having a pre-planned reference camera trajectory viewing the targeted object. We assume that this object is described within a mesh composed of set of triangular geometric primitives. We do not assume any prior knowledge or information on the type or physics of deformation. We used the liver as a reference deformable object. The addressed problem is stated as follows: Let us consider a camera following a reference trajectory and viewing a liver. If the latter is deformed, then how should one at best adapt the trajectory of the former. In the context of robotized laparoscopy, this is a standard problem and is usually put at the responsibility of an operator who manipulates the camera (the laparoscope) with a joystick according to the surgeon recommendation. This task can be fully automatized given the resolution of two main issues: First, an appropriate liver tracking which allows one to follow $2D$ features on the liver. Second, an automation criterion that allows one to adapt to deformations and modify the camera trajectory accordingly.

This paper focuses on proposing a novel strategy to tackle the second problem.

This diagram 1 illustrates the flow of a triangle-mesh-based visual servoing algorithm used for controlling a camera's motion when observing a deformable object. The process begins by initializing the reference trajectory and mesh data, followed by iterating through each point on the trajectory. For each point, the algorithm processes each triangle in the object's mesh, calculating key parameters such as weights and interaction matrices. A visual servoing loop then iteratively adjusts the camera's position and orientation to minimize the error between the current image of the deformed object and the reference image, ultimately producing an optimized trajectory that accounts for the object's deformation.

The contributions of our work can be summarized as follows

(1) Define a camera-based motion planning to compensate non-rigid object deformations with rigid camera motion. This definition allows us to formalize the viewing interaction between the deformable object and the camera with a set of camera-triangle primitives. These triangles are the geometric elements of the object's mesh.
(2) Associate a continuous weight function to every pair of camera-triangle primitive. The continuity is along the camera-trajectory and allows us to ensure smooth tracking.
(3) Solve the visual servoing and tracking task in an optimal visual control framework. This proposed method does not assume any prior on the object's deformation. The aforementioned weight functions encode the relevance of the camera-triangle primitives. The most important target related primitives have higher weights and allows us to find the optimal camera motion to compensate non-rigid deformations.
(4) Experimental validation demonstrates the robustness of the approach regarding the noise in image detection, the variability in trajectories and type of deformations.

This paper is organized as follows. Section 2 presents the related state-of-the-art works. Section 3 introduces the fundamental tools for our modeling. Section 4 describes the visual servoing with optimal visual control and tracking. Section 5 presents the experimental results and comments. Section 6 concludes the paper and draws a sketch of future works.

2. RELATED WORK

Tracking and visual servoing on non rigid objects are recent open problems. Many approaches propose to use physics-based or data-based mechanical models embedded into the classic algorithms. These methods are highly parametric and are fine-tuned for every specific type of deformable objects. In this paper, we propose a generic approach that enables rigid camera visual servoing to fit as much as possible visual features of objects subjected to non-rigid deformation. In this section, we review the literature on visual servoing, visual tracking and image alignment.

2.1. Visual Servoing on Rigid Objects

Classic visual servoing is a method of controlling the movement of a robot using real-time feedback from vision sensors [7–9]. Visual servoing control has two basic approaches: Position-based visual servoing and image-based visual servoing [10, 11]. In this paper, we use the position-based visual control since we assume tracking $2D$ features on the target deformable object. The relationship between the camera and the object is represented by an interaction matrix. This matrix can be determined in the image space by points, lines or ellipses and moments [12, 13]. In this work, point-based interaction matrix is used. It is used as an elementary building block for triangle-based interaction matrices. The triangles being the primitives that compose the surface of the deformable shape. Being an old problem, the literature of visual servoing on rigid objects is extensive and cannot be covered in this paragraph. We cover here, some main results of these works. Janabi et al. [14] used a reduced set of holes, circles and wedges as characteristics for their availability in many industrial parts and for their easy and robust extraction. Chaumette [15] used the moments calculated by image segmentation to determine the analytical shape of the interaction matrix. Molnar et al. [16] offered a marker-based visual servo technique for automated camera positioning in the case of Robot-Assist minimally invasive surgery. Their approach is to keep the followed surgical instrument within the limits of the camera image. This is done by constantly adjusting the endoscopic camera manipulation pose instead of keeping the instrument in the center of the image. In addition, the interaction matrix can be determined by a hybrid visual servoing which combines image-based visual servoing and position-based visual servoing. Iheb [17] proposed a real-time and inversion-free control strategy on the basis of sampling-based model predictive control algorithm for both image-based and position-based visual techniques. They considered system

constraints and uncertainties parameters associated with the robot and the camera measurement. In this paper, we do not consider model uncertainties and we do not use a predictive control scheme. Instead, we use an optimal visual control framework to compensate for both noise and deformations. We propose to use a continuous weight functions on the triangle primitive to account for their relevance while tracking the reference trajectory.

2.2. Visual Servoing on Non-Rigid Objects

Robotic manipulation of non-rigid objects is a difficult task due to the deformations that occur which add several degrees of freedom to the initial problem with rigid objects. Among deformations we can cite shearing, scaling, stretching, torsion, compression, etc. Human organs are relevant cases of non-rigid objects with a variety of shapes and deforming behaviors [18]. Using such objects with visual servoing approaches gives solutions to automate problems in medical and surgery contexts. For instance, augmented reality to overlay MRI (Magnetic Resonance Imaging) images on live laparoscopic's stream [2], robotized laparoscopy to guide the laparoscope, etc. Hu et al. [19] presented a controller based on a deep neural network to control the position and shape of deformable objects with unknown deformation properties. They treated the nonlinear properties of the deformable objects and used a multilayer neural network to model the mapping function. In this work, we do not use any prior model on the deformation of the organ. We use a a set of continuous weight functions that allows us to account for relevant regions to focus on. Jia et al. [20] presented a histogram of oriented wrinkles to describe the variation in shape of a highly deformable object. The characteristics of the deformable object are calculated by applying Gabor filters and extracting the high and low frequency components. They precomputed the visual feedback using an offline training phase that stores a correspondence between these visual characteristics and the speed of the end effector. Our approach uses an optimal visual control setup which offers in-line camera correction without requiring to acquire data or train neural networks. Hu et al. [21] designed a servo algorithm that can learn a nonlinear deformation function along with the manipulation process. They used Gaussian process regression to model and learn the deformation parameters of a soft object. In this paper, we propose a generic approach that enables rigid camera visual servoing to adapt as much as possible visual features of objects subjected to non-rigid deformation. The proposed method uses a triangular mesh representation of the visualized object. It computes a relative camera rigid repositioning which fit the most relevant triangular primitives without fully ignoring the remainders.

2.3. Visual Tracking of Non-Rigid Objects

Tracking deformation has been studied in several works [22–25]. Chen et al. [26] presented a two-step object tracking method. They used the kernel-based method to efficiently locate the object under complex conditions with camera movement. For precision, they improved the tracking accuracy by using the contour-based method to follow the object's contour precisely after locating the target. Cao et al. [27] assumed that the extension of an object would deform from a reference by moving certain control points of the latter towards those of the old one. In our method, we do not consider any model of the deformed object. Our motion compensation is based on relevant image feature differences. Joo et al. [28] presented a generative body deformation model which has the capacity to express the movement of every principal part of the object. Royer et al [29] presented a novel approach for tracking a deformable anatomical target within 3D ultrasound volumes. This method is able to estimate deformations caused by the physiological motions of the patient. The displacements of moving structures are estimated from an intensity-based approach combined with a physically-based model and has therefore the advantage to be less sensitive to the image noise. Our method does not process a 3D reconstruction of the object and focuses only on the best move of the camera to compensate deformations and ultimately noise. Royer et al [30]

extended their former work by proposing a 3D tracking method which is regularized by a statistical motion model obtained from biomechanical modeling. However, their method requires manually identifying certain points of the target object (here a prostate) on each ultrasound frame in order to drive the model. Kajihara et al. [31] developed a system that tracks the trajectory of a line drawn on a flexible object. They estimated the tracking performance with speed and precision in the presence of many uncertainties based on dynamic compensation. In the proposed approach, there is no pattern that is drawn on the target object. Instead, a set of weight functions are pre-defined on pre-selected regions of the deformable objects. These regions are defined within a set of object's mesh triangles.

2.4. Image Alignment

Image alignment is relative to the alignment of two or more images of the same scene on a common spatial axis. It plays an important role in computer vision and graphics, such as structure from motion, 3D reconstruction, motion tracking and recovery [32, 33]. The problem we are addressing can be seen as an image alignment of a deformable scene. Indeed, the camera motion compensation that is calculated by our method can also be used to align the current image with the reference image from the pre-defined trajectory. Shingo et al. [34] proposed a technique to align an end-edge of flexible sheet object to a specified line segment in 3D space. They made online estimation for a relative pose between the end of the edge and the robot hand in the visual servoing control laws. Our method relies on feature points instead of line features. Xi et al. [35] focused on parametric and non parametric alignment method which have complementary strength. They proposed a feature based parametric alignment using one or more homographies followed by non-parametric fine pixel wise alignment. Our method is non-parametric regarding to the deformation of the object. Dong et al. [36] proposed the spectral-spatial weighted kernel manifold embedded distribution alignment method for the classification of remote sensing images. They used a filter to express the mean spectrum of the neighboring pixel samples from each pixel sample. In this work, we consider the 2D feature points as being known. Mathiassen et al. [37] proposed a visual servoing method to move the ultrasound probe, using a robot to align the image plane of the probe with the needle. The method segments the needle and updates a set of visual features based on a model of the needle. A state machine is used to keep track of the alignment process, and different visual features are used to control the probe in the different states. In this paper, we consider a perspective camera model. The proposed method can be applied to ultrasound sensors however it will be necessary to modify the projection model that is used in this paper. Mura et al. [38] presented a vision-based haptic feedback system with the aim to assist the movement of an endoscopic device during capsule endoscopy guidance. It allows the user to control the movement of the capsule along the generated path. The haptic module also helps the operator by transforming the 3D maps and the relative paths into a guiding virtual force. Measuring the current relative distance between the user input and the maps boundaries, the haptic guidance module will check if the user is moving away or toward the colonic walls and will generate a feedback force with the aim to assist the operator during the navigation procedure. The user will also sense an attractive virtual feedback force toward the generated path that will help the user in the navigation. Our method can be applied to feed the haptic module with the necessary correction. Indeed, instead of compensating the trajectory of the camera, one correct the trajectory of the capsule by controlling the haptic joystick.

3. VISUAL PRIMITIVES AND INTERACTION MODELING

3.1. Definitions

3.1.1. Triangle. Let us consider a triangle T as a basic geometric primitive composed of 3 points in the 3D workspace. Let $L = \mathbf{R}^9$ be the configuration space of this primitive. We denote by

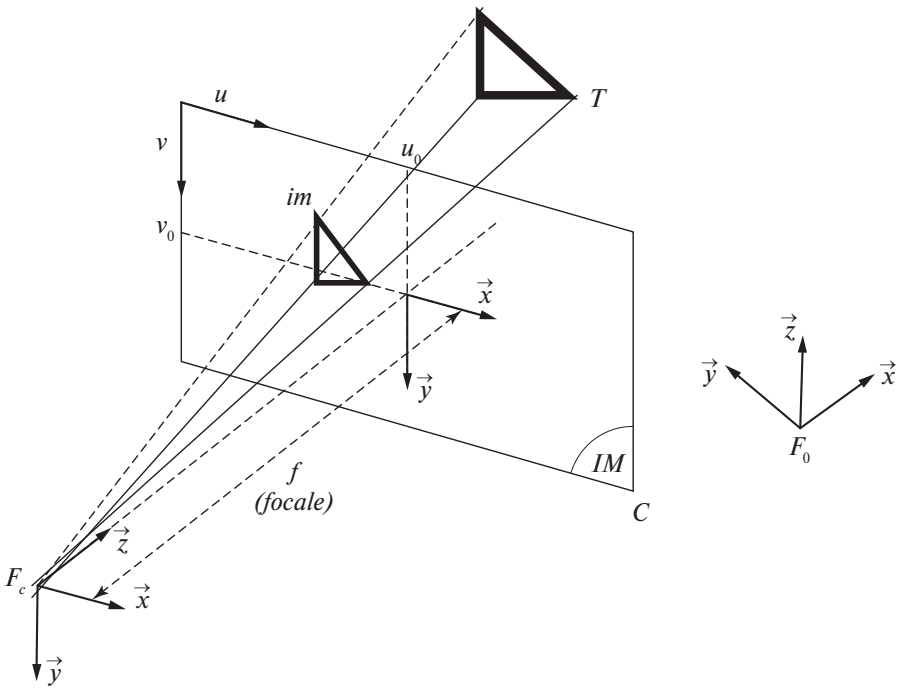

Fig. 2. Perspective projection of a triangle on the image plane of a camera.

$l = (X_1, X_2, X_3, Y_1, Y_2, Y_3, Z_1, Z_2, Z_3)^\top$ the configuration of T in L. Where $(X_i, Y_i, Z_i)^\top$, $1 \leqslant i \leqslant 3$ being the coordinates of the three vertices composing triangle T. Index i represents the number of the vertex in the considered triangle.

3.1.2. Camera. Let us consider C as a camera that maps one or several triangular primitives to a 2D feature in the image space. Let $C = \mathbf{SE}(3)$ be the configuration space of this sensor. We denote by c the configuration of the camera in the space C.

3.1.3. Sensing a triangle. The perception of a triangle T by a camera C can be characterized by a continuous mapping:

$$\Pi : C \times T \to I_{c,l}, \tag{1}$$

$$(c, l) \mapsto \Pi(c, l). \tag{2}$$

That associates to each configuration of the camera and the triangle a feature in the image space $I_{c,l} \subset \mathbf{R}^6$. The image space being composed of the projection of the triplet of points describing the triangle of interest (see Fig. 2). Remark that Π is usually defined over a subset of $C \times T$ corresponding to configurations pairs of the camera and the triangle that must be in the field of view of the camera. In the case of a perspective camera, Π can be written as

$$\Pi(c, l) = \Big(x_1 = X_1/Z_1,\ \ x_2 = X_2/Z_2,\ \ x_3 = X_3/Z_3,\ \ y_1 = Y_1/Z_1,\ \ y_2 = Y_2/Z_2,\ \ y_3 = Y_3/Z_3\Big)^\top. \tag{3}$$

3.2. Localization and Visual Servoing

3.2.1. Localization Equation. Let us consider a camera at a configuration $c \in C$. Let us consider m triangles $T_1, \ldots, T_m$ of known configurations $l_1, \ldots, l_m \in L$, where each of them is visible from a camera C. Therefore, each pair (C, T_i) of camera-triangle gives rise to a localization equation system:

$$\Pi(c, l_i) = im_i. \tag{4}$$

Where $im_i \in I_{c,l_i}$ is the projection of T_i in the image space. im_i is measured and can thus be supposed as known. It can be denoted as $im_i = (x_1, x_2, x_3, y_1, y_2, y_3)^\top$. The knowledge of c and l_i is addressed in the next section.

3.2.2. Visual Servoing About a Reference Configuration. If the camera is expected to follow a reference trajectory viewing a reference object that can deform, then the result of equation (4) is expected to be in the neighborhood of a reference configuration of the camera and a reference configuration of the object. If we neglect the remainder of the linearization, then equation (4) can be linearized as follows:

$$\frac{\partial \Pi}{\partial c}(c_0, l_{0i})(c - c_0) + \frac{\partial \Pi}{\partial l}(c_0, l_{0i})(l_i - l_{0i}) = im_i - im_{0i}. \tag{5}$$

Where c_0 is the reference configuration of the camera on the pre-defined trajectory. l_{0i} is the expected reference configuration of the viewed triangle with reference features at im_{0i} in the image space I_{c,l_i}. At execution time, both of these reference configurations can be different from runtime configurations for two main reasons: Camera drift and object deformation. These consideration are valid if one assume that the localization is ran at high frequency [39] (faster than both the camera control loop and the dynamic of object's deformation). im_i and im_{0i}, respectively, represent the image of triangle T_i in sensor C and the expected image, i.e., the image that would be seen from configuration c_0 if there was no camera drift and no object deformation.

$\frac{\partial \Pi}{\partial c}$ is the Jacobian matrix of order 6 representing the variation of the image with respect to the variation of the camera configuration. This Jacobian matrix derives from the geometric properties of the camera and triangle. It represents the so called interaction matrix in the classic visual servoing terminology. $\frac{\partial \Pi}{\partial l}$ is the Jacobian matrix of order 6×9 representing the variation of the image with respect to the variation of the triangle configuration. This Jacobian is the zero-matrix in the case of non-deforming objects. Most of the previous works have modeled this matrix either with physics-based equations of deformations or with data-driven machine learning algorithms [40–42]. In this work, we use an optimal visual control framework to embed the control loop of the visual servoing algorithm. This approach allows us to account for the deformation behavior without having to numerically calculate or regress the deformations. Let us first focus on the analytical formulation of the interaction matrix. It is well known that for a single point at 3D corrdinates (X, Y, Z) perceived at 2D coordinates (x, y), the interaction matrix that relates variations of the camera pose to variations of the 2D projected coordinates is written as [15]:

$$L(x, y, Z) = \begin{bmatrix} L_u(x, y, Z) \\ L_v(x, y, Z) \end{bmatrix} = \begin{bmatrix} \frac{1}{Z} & 0 & \frac{-y}{Z} & -xy & 1 + x^2 & -y \\ 0 & \frac{1}{Z} & \frac{-y}{Z} & -(1 + y^2) & xy & x \end{bmatrix}. \tag{6}$$

The camera intrinsics are assumed to be known and undone from pixel coordinates. If we consider a triangle T at reference coordinates $l_0 = (X_1^0, X_2^0, X_3^0, Y_1^0, Y_2^0, Y_3^0, Z_1^0, Z_2^0, Z_3^0)^\top$ (see for instance Fig. 2), we represent the associated interaction matrix as

$$\frac{\partial \Pi}{\partial c}(c_0, l_{0i}) = \begin{bmatrix} L_u(x_1^{0i}, y_1^{0i}, Z_1^{0i}) \\ L_u(x_2^{0i}, y_2^{0i}, Z_2^{0i}) \\ L_u(x_3^{0i}, y_3^{0i}, Z_3^{0i}) \\ L_v(x_1^{0i}, y_1^{0i}, Z_1^{0i}) \\ L_v(x_2^{0i}, y_2^{0i}, Z_2^{0i}) \\ L_v(x_3^{0i}, y_3^{0i}, Z_3^{0i}) \end{bmatrix}. \tag{7}$$

Where $im^0 = (x_1^0, x_2^0, x_3^0, y_1^0, y_2^0, y_3^0, z_1^0, z_2^0, z_3^0)^\top$ is the reference projection of the triangle onto the camera plane (focal length and camera center are considered as known and undone.). We invite the reader to refer to appendix A.1 for more details on the definition of c and $\frac{\partial \Pi}{\partial c}(c_0, l_{0i})$.

3.2.3. Weighted Triangle-based Visual Servoing on Rigid Objects. If we consider the object as being rigid, then $l_i = l_{0i}$ for all the m triangles and equation (5) simplifies to:

$$\frac{\partial \Pi}{\partial c}(c_0, l_{0i})(c - c_0) = im_i - im_{0i}. \tag{8}$$

Classic visual servoing approaches on rigid objects consist in solving for the camera pose c such an equation for the whole set of triangles. This equation defines and solves the local task that is to fit the current triangle's view to the expected one. We can further assign real positive weights to the object's triangles that are relevant to the current task. The above equation turns to be

$$w_i \frac{\partial \Pi}{\partial c}(c_0, l_{0i})(c - c_0) = w_i(im_i - im_{0i}). \tag{9}$$

Here w_i is the assigned weight to triangle L_i. Considering the whole set of triangles composing the objects with their associated set of weights, we build the following system of equations

$$W(c - c_0) = IM - IM_0. \tag{10}$$

Where

$$W = \begin{pmatrix} w_0 \dfrac{\partial \Pi}{\partial c}(c, l_0) \\ \vdots \\ w_m \dfrac{\partial \Pi}{\partial C}(c, l_m) \end{pmatrix}, \tag{11}$$

$$IM = \begin{pmatrix} w_1 im_1 \\ \vdots \\ w_m im_m \end{pmatrix} \quad \text{and} \quad IM_0 = \begin{pmatrix} w_1 im_{01} \\ \vdots \\ w_m im_{0m} \end{pmatrix}. \tag{12}$$

To solve equation (10) for the camera pose in the rigid object case many consideration can be taken. For instance, the noise in the measurement of the triangle positions in the image, the inaccuracy of the 3D object shape that was scanned or modeled beforehand, etc. If the number of triangles is too large, the system of equations (10) has no exact solution and the servoing task is said to be over-constrained. In the contrary, if the system of equations (10) is underdetermined, there are an infinite number of possible camera poses that satisfies it. In this case, considerations like minimum norm solution can be taken to solve the system 10 as it was done in [39]. In the next section, we describe our third contribution which is embedding the weighted triangle-based visual servoing formalism in an optimal visual control framework for non-rigid objects. This embedding allows us to calculate the correction to impose to the camera motion while tracking a reference trajectory on a deforming object. The proposed approach does not consider any prior knowledge or parametric model that describes the deformation of the target object.

4. SERVOING ON DEFORMABLE OBJECTS WITH AN OPLIMAL VISUAL CONTROL

In 3D reconstruction of deformable objects from monocular views, the assumption of considering that any deformed shape lies at the minimal stretching/compressing energy has proven to be effective [43–46]. In our work, we do not aim to infer the 3D shape from the 2D view. We only need to compensate the deformation with a camera motion. Thus we define such motion as

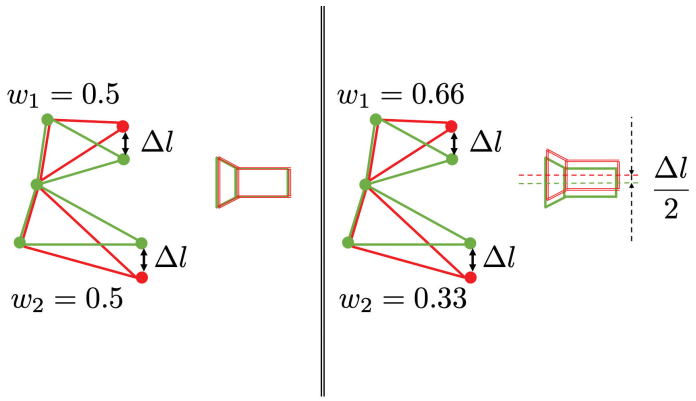

Fig. 3. Weight effect illustration. In this figure, we consider a reference camera in green viewing two reference triangles in green color. After deformation, the red vertices move in opposite directions by Δl along the vertical axis. This shift causes a non-rigid deformation of the two triangles. We consider here two scenarios: Left figure considers equal weights for both triangles. In this case, the correction with equation (14) does not produce any motion since the motion of the vertices compensates each other and the triangle have the same weight. Right figure considers the top triangle with a weight twice the value of the bottom one. In this case, equation (14) produces a vertical shift with half of Δl. If $w_2 = 0$ then the correction would be a vertical shift to the top within Δl.

being the one for which the stretching/compressing energy is minimal. Using equation (5), this statement comes to minimizing the deformation seen through the perspective projection, which can be formalized as follows

$$\begin{aligned} \hat{c} =& \underset{\tilde{c}\in C}{\operatorname{argmin}} \frac{1}{2}\sum_{i=1}^{m} \left\| \frac{\partial \Pi}{\partial l}(c_0, l_{0i})(l_i - l_{0i}) \right\|_2^2, \\ \text{s.t. } & \frac{\partial \Pi}{\partial l}(c_0, l_{0i})(l_i - l_{0i}) + \frac{\partial \Pi}{\partial c}(c_0, l_{0i})(\tilde{c} - c_0) = (im_i - im_{0i}). \end{aligned} \tag{13}$$

In our work, we aim at giving more importance to some spatial deformation than others. This goal is formalized through the association of real positive weight values to triangular primitives. Thus the above criterion can be rewritten as

$$\begin{aligned} \hat{c} =& \underset{\tilde{c}\in C}{\operatorname{argmin}} \frac{1}{2}\sum_{i=1}^{m} \left\| w_i \frac{\partial \Pi}{\partial l}(c_0, l_{0i})(l_i - l_{0i}) \right\|_2^2, \\ \text{s.t. } & w_i \frac{\partial \Pi}{\partial l}(c_0, l_{0i})(l_i - l_{0i}) = -w_i \frac{\partial \Pi}{\partial c}(c_0, l_{0i})(\tilde{c} - c_0) + w_i(im_i - im_{0i}). \end{aligned} \tag{14}$$

This criterion allows us to estimate a rigid camera motion $\hat{c}$ which minimizes the view between reference and current deformed shape. The weights allows us to give more importance to some regions above others as illustrated in Fig. 3. The following expression is used to compute the correction of the camera pose.

Proposition 1. *Let us consider a camera at some reference configuration $c_0 \in C$. The camera is viewing a deformed shape that is assumed to be at its minimal stretching/compressing energy. The rigid camera motion compensation $\hat{c}$, which satisfies criterion of equation* (14)*, can be calculated as*

$$\hat{c} = c_0 + W^{+}(IM - IM_0), \text{ if } W \text{ is full column rank}, \tag{15}$$

$$\hat{c} = c_0 + W^{\top}(W\,W^{\top})^{-1}(IM - IM_0), \text{ else and if } W \text{ is full row rank}. \tag{16}$$

W^{+} being the pseudo-inverse matrix of W.

Proof. Taking into account equation (14) and using the linearisation equation (5), we write the optimal criterion as

$$\hat{c} = \underset{\tilde{c}\in C}{\operatorname{argmin}} \frac{1}{2} \sum_{i=1}^{m} \left\| w_i \frac{\partial \Pi}{\partial c}(c_0, l_{0i})(c - c_0) - w_i(im_i - im_{0i}) \right\|_2^2. \tag{17}$$

This is exactly the weighted sum of the squares of the Euclidean norms of the differences between each image that would be seen from $c \in C$ and the image seen at run time. This minimization can be rewritten in a concatenated formula as

$$\hat{c} = \underset{\tilde{c}\in C}{\operatorname{argmin}} \frac{1}{2} \|W(\tilde{c} - c_0) - (IM - IM_0)\|_2^2. \tag{18}$$

If W is full column rank, then solving for c this weighted least-squares problem gives the following solution

$$\hat{c} = c_0 + (W^\top W)^{-1} W^\top (IM - IM_0). \tag{19}$$

In our work, we consider that at every instant time there are sufficient visible triangles to compute the correction for the 6 degrees of freedom of the camera. For this purpose, we require at least 2 triangles to be visible to avoid ambiguous situations [45]. Given a mesh of N points, and a camera tracking full regions of this mesh, it is fair to consider this condition being verified in most of the time. In some edge cases where this might not be verified (for instance the camera is too close to the object), we setup a safe test for the number of visible triangles which allows us to compute the solution of minimum norm. If W is full row rank, then the camera correction is computed as the minimum norm of camera motion satisfying the following criterion [47]

$$\hat{c} = \underset{\tilde{c}\in C}{\operatorname{argmin}} \frac{1}{2} \|\tilde{c} - c_0\|_2 \text{ s.t. } W(\tilde{c} - c_0) = (IM - IM_0). \tag{20}$$

Given that W is full row rank, the solution of this under-determined least-squares problem is given as

$$\hat{c} = c_0 + W^\top (W W^\top)^{-1} (IM - IM_0). \tag{21}$$

In some isolated degenerate cases where W is neither full column rank nor full row rank, the camera is not corrected from the reference configuration and we keep $\hat{c} = c_0$.

The developments conducted in this section can be summarized as follows. Rigid compensation with camera motion of a nonrigid object motion consists of solving a system of equations that relate the configuration of the camera with the images of the object's triangles. If the system is over-constrained, the compensation consists of finding a configuration that minimizes a weighted sum of residues. If there is no deformation and the triangles are all rigid, the choice of weights will have no effect on the result (ignoring noise in vertex detection). If the object's triangles deform, the choice of weights will have an effect on the corrected configuration $\hat{c}$ as illustrated in Fig. 3. For this reason, we propose to use these weights as a tool for planning triangle mesh-based motions for deformable objects.

4.1. Triangle-Mesh-Based Motion for Visual Servoing on Deformable Objects

Definition 1. Let us consider a camera following a reference trajectory and viewing a deformable object meshed with m traingles $T_1, \ldots, T_m$. A triangle-mesh-based motion is composed of two main components:

(i) A reference trajectory

$$\gamma : [0,1] \to C : \ s \mapsto \gamma(s), \tag{22}$$

where $[0,1]$ is a normalized interval of the trajectory temporal parameter s.

(ii) m continuous positive real-valued functions $w_1, \ldots, w_m$:

$$w_i : [0,1] \to R^+ : s \mapsto w_j(s), \tag{23}$$

such that $w_i \ (s) = 0$ for any s such that T_i is not visible by camera C when the camera is at configuration $\gamma(s)$. Continuity of w_i is required to avoid undesirable jumps during the correction of the camera trajectory [39].

The above triangle-mesh-based motion for the rigid motion compensation of the camera pose consists in correcting the current configuration of the camera within the closed-loop control task by computing the correction from Proposition 1 about the reference configuration $\gamma(s)$ and with values of the weights $w_i(s)$ for abscissa s along the trajectory. The correction formula can be written for a given abscissa s as

1) If $W(s)$ is full column rank

$$\hat{c}(s) = \gamma(s) + (W(s)^\top W(s))^{-1} W(s)^\top (IM(s) - IM_0(s)). \tag{24}$$

2) If $W(s)$ is not full column rank but is full row rank

$$\hat{c}(s) = \gamma(s) + W(s)^\top (W(s)\, W(s)^\top)^{-1} (IM(s) - IM_0(s)). \tag{25}$$

3) Else

$$\hat{c}(s) = \gamma(s). \tag{26}$$

Where $W(s)$, $IM(s)$ and $IM_0(s)$ are extensions of the notations in equations (11) and (12) when the reference camera configuration follows a reference trajectory $c_0(s) = \gamma(s), \ s \in [0,1]$. The case 3 of equation (26) is an edge case and almost never occurs. It is used as a safety case if ever W is neither full column rank nor full row rank. The above formulas of corrected trajectory $\hat{c}(s)$ from equations (24)–(26) can be interpreted as the one satisfying an optimal visual alignment and represents the minimum of the integral along the whole trajectory of the difference between reference view and current deformed view as follows

$$J = \int_0^1 \frac{1}{2} \| W(c(s) - \gamma(s)) - (IM(s) - IM_0(s)) \|_2^2 \ ds. \tag{27}$$

In order to have smooth corrections it is necessary to have continuons weights [39]. This is the case if for instance the weights are proportional to the 2D area of the viewed triangle in the image plane (see the experimental Section A.2 for details on their usage and their computation). If the object is an organ and the task consists in compensating the deformations viewed by a laparoscope, then ideally the weights can be set manually by a technician before surgery. Indeed, a surgeon with the help of technician can point the main target area of an organ to be kept steady when seen by the laparoscope. The technician can then set higher weights to those area compared to others. Such semi-automatic process can be done at the surgery planning.

5. PROPOSED ALGORITHM AND IMPLEMENTATION DETAILS

Algorithm 1 Triangle-Mesh-Based Motion for Visual Servoing on Deformable Object

Require: Reference trajectory $\gamma : [0,1] \to C : s \mapsto \gamma(s)$;
Require: Deformable object meshed with m triangles $T_1, \ldots, T_m$;
Require: step_size for sampling reference trajectory;
Require: max_iter for maximum number of iterations;
Require: ϵ_{im} stopping error on image difference;
1: **Initialization:**
2: Load reference trajectory γ for the camera;
3: Load data mesh of the reference object;
4: Load data mesh of the same object in its deformed state;
5: **Trajectory Loop:**
6: **for** $s \in [0,1]$ **with** step_size **do**
7: Get current camera configuration $c(s) \leftarrow \gamma(s)$;
8: **Triangle-Mesh Loop:**
9: **for** each triangle i from 1 to m **do**
10: Get current weight w_i by applying formulas in equations (B14) and (B15);
11: Get current interaction matrix by applying formula of equation (7);
12: Get the reference image projection im_i^0 of the triangular primitive;
13: Compute the current task function of the triangle using equation (10);
14: **end for**
15: $error \leftarrow \epsilon_{im} + 1$;
16: $iter \leftarrow 0$;
17: **Visual Servoing Loop:**
18: **while** $iter < max\_iter$ **or** $error > \epsilon_{im}$ **do**
19: Get the current image projection primitive im_i from the current camera pose;
20: Assemble object weighted interaction matrix W using equation (11);
21: Assemble reference and current object weighted image IM, IM_0 using equation (12);
22: Solve the optimal servoing function using equations (24), (25), (26);
23: Use the obtained $\hat{c}(s)$ to update the current camera configuration;
24: $iter \leftarrow iter + 1$;
25: $error \leftarrow \|W(\hat{c}(s) - c(s)) - (IM - IM_0)\|_2$;
26: **end while**
27: **end for**
28: **Output:**
29: Final optimal visual servoing trajectory given the current deformation of the object;

The visual servoing Algorithm 1 described in this section uses a deformable object's mesh to achieve precise camera positioning relative to the object, even as the object deforms. The key steps are:

(1) The trajectory loop (lines 5:27) iterates through the reference camera trajectory, and for each camera pose, the algorithm computes:
   (a) for each triangle, using the reference object, the weights, the reference image projection, and the interaction matrix;
   (b) an overall weighted interaction matrix and reference and current object weighted images;
(2) The visual servoing loop (lines 17:26) then iteratively solves for the optimal camera pose update that minimizes the difference between the reference and current object weighted images, updating the camera pose accordingly.

This approach allows the visual servoing to adapt to deformations of the object, using the mesh representation to correct the camera trajectory accordingly.

This algorithm was implemented using Matlab version a2015b. It was run on a laptop computer with an Intel Core (TM) i5-4200U CPU processor and 6 GB of RAM. The object model used was

a triangle mesh with 10 000 vertices and 20 000 faces. The reference trajectory was sampled at a step _size of 10 Hz. The maximum number of iterations, max _iter, was set to 100, and the stopping error, ϵ_{im}, was set to 0.01 pixels. The time complexity is of $O(N \times m^4 \text{max\_iter})$ order and the space complexity is of $O(m^2)$ order.

The overall algorithm runs at an average speed of 40 Hz, enabling real-time visual servoing of the deformable object.

This algorithm was tested across various scenarios, including multiple object types, reference trajectories, deformation modes, and levels of image noise. The following experimental section describes the results obtained in these diverse settings.

6. RESULTS ON SIMULATED DATA

This section shows several representative examples of our verification tests with simulated data using two deformable objects: a planar object that is deformed to a bump-like shape and a model of human liver that is deformed on its left and right lobes. We ran experiments on both linear and nonlinear deformations. We tested our method on two geometric trajectories: Linear and circular. To validate the robustness of our approach, we simulates noise detection on the 2D mesh vertices. We used a perspective camera with a focal length of 1500. We ran our simulations on a Core (TM) i5-4200U CPU processor and 6 GB of RAM and Matlab2015a. In the shown figures, distances are in meters, translation speeds of the camera is in meter per second and rotation speeds are in radian per seconds. The errors in image space are in pixels.

6.1. Results on a Planar Object and Linear Geometric Path

Figure 4 represents the target object in 3D space and its projection on the camera. The object is composed of 200 triangles as shown in Fig. 5. During deformation, the camera trajectory is used as a reference constraint to ensure that the target features remain within and adjust to the expected

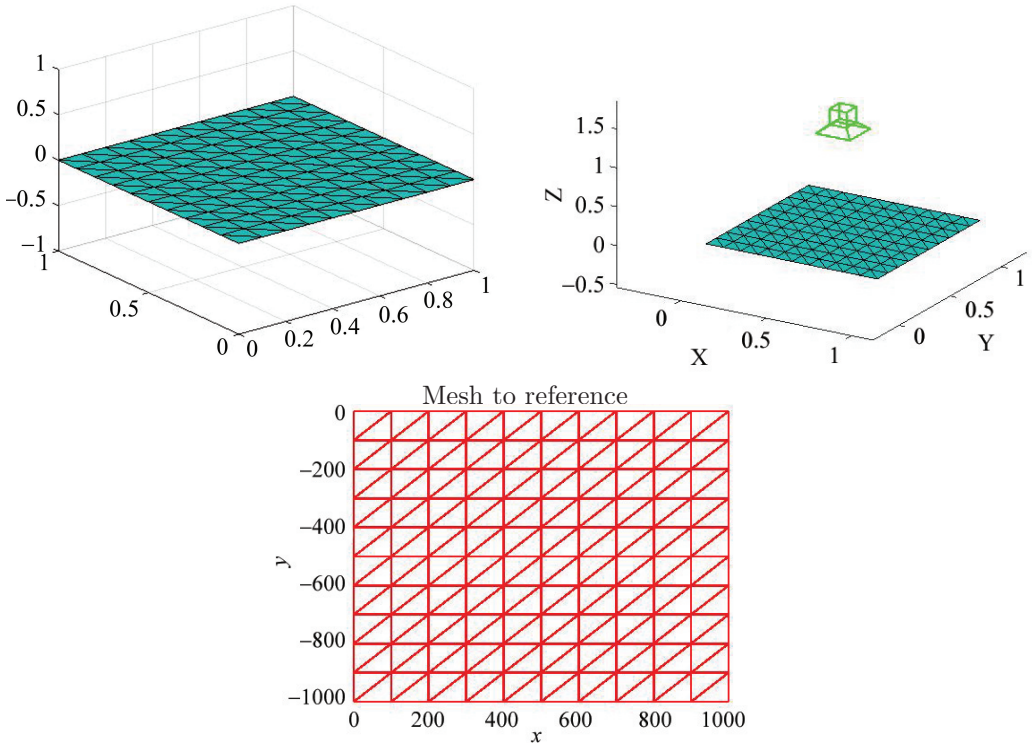

Fig. 4. Top right: The reference object is a planar triangular 3D mesh. Top left: The reference object and the camera configuration in 3D space. Bottom: The viewed planar mesh in the camera plane.

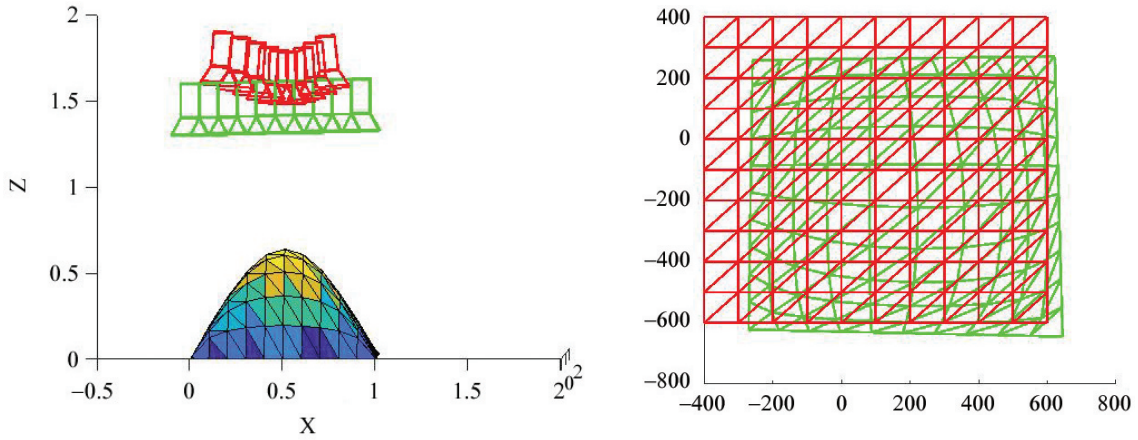

Fig. 5. Left: Deformed reference object with reference (green) and deformed (red) camera trajectories. The reference camera follows a linear planned trajectory. Green cameras is the reference trajectory. Red Cameras is the corrected trajectory. Here all weights are equal to one. Right: Reference viewed object (red) and deformed viewed object (green). The corrected trajectory tries to fit as much as possible the reference view of the object.

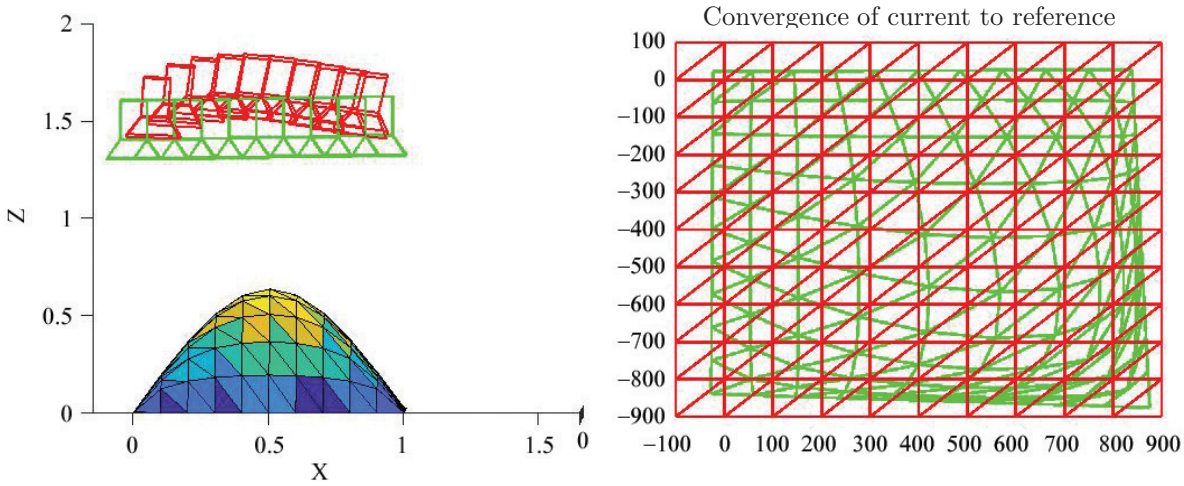

Fig. 6. Left: The deformed 3D plane object with the camera which follows a linear planned trajectory. Green cameras is the reference trajectory. Red Cameras is the corrected trajectory. Here all weights are equal to one except the weights associated to the triangles on the top hill (yellow triangles). They are set to 4. Right: Reference viewed object (red) and deformed viewed object (green). The corrected trajectory tries to fit as much as possible the reference view of the object.

view dynamically. In Figs. 6, we control movement of the camera which depends on weights along the path. In Figs. 5 and 6, yellow, orange, green and blue regions represent respectively the heatmap of deformation going from high to low deformations. In this experiment we allocated continuous constant weights. The weights of yellow region are the highest with continuous constant weights equal to 4. The remaining regions have continuous constant weights equal to 1. There is no upper or lower limit to the choice of weights. Only relative bigger/smaller values matter to modulate more/less tracking focus on a particular area. Regions which are relevant at run time have bigger weight values to enforce the tracking task to better follow those deformed parts.

6.2. Results on a Simulated Model of a Human Liver

The Liver is the largest organ in the human body. It belongs to the digestive system and provides many vital functions to the body. In our work, we use the liver mesh to test the performance of

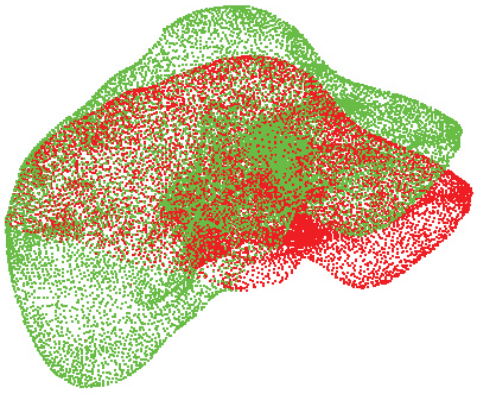

Fig. 7. Liver before and after nonlinear elastic deformation. Red Liver represents the organ at rest. Green Liver represents the organ after nonlinear deformation.

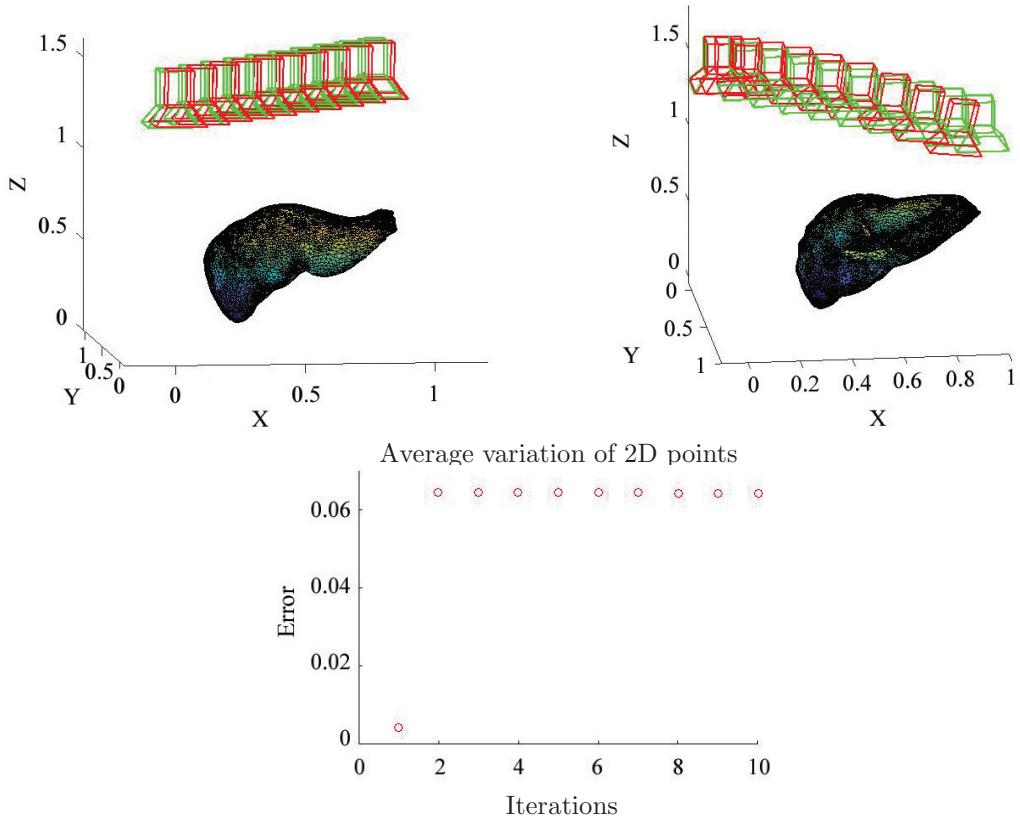

Fig. 8. Liver after a linear elastic deformation and camera follows a reference straight trajectory. Results of test (1) are shown. Left shows The reference trajectory. Right shows a side view of the trajectory after correction. Green cameras is the reference trajectory. Red Cameras is the corrected trajectory. Bottom shows the variation of pixel points during the correction of the camera pose. Green cameras is the reference trajectory.

our method. To deform the liver we used a Mooney-Rivilin hyper-elastic physical model and used the approach proposed by Saidi et al. work [48] to speed-up the calculation of the nonlinear elastic deformations. The liver mesh contains 130 382 triangle and 12 226 vertex (see Fig. 7 for an illustration). We ran experiments on two types of deformations: Linear and nonlinear. For the same mesh of the liver, we moved a selected deformed part of 825 by 6 mm in the z-axis, which is considered as a linear deformation. In a second deformation, we moved the same 825 vertices by 22 mm, which is considered as a non-linear deformation.

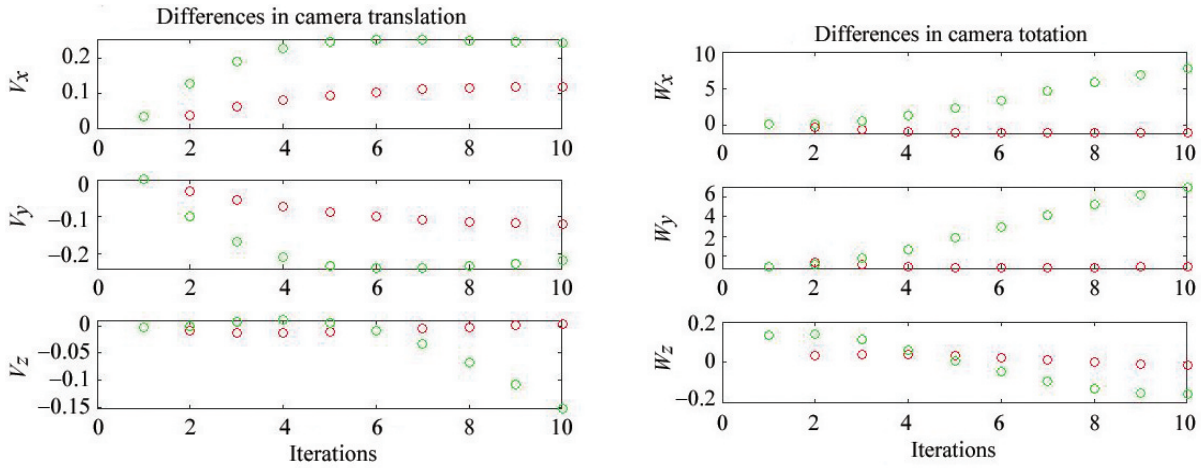

Fig. 9. Analysis of the performance of the camera speed when tracking a deformed object with linear deformation. Left figure represents camera speed in translation. Right figure represents camera speed in rotation. Red dots represent camera speed following strategy of test (1). Green dots represent camera speed following strategy of test (2). ω_y and ω_z are similar for both tests.

6.2.1. Results with linear deformation of the liver. For the linear deformation, we used a command to control the movement of the camera which was following a reference horizontal straight line trajectory as shown in Fig. 8. We ran two tests: Test (1) where all the weights are constants and equal to 1 for all the triangles. Test (2) where the weights vary according to the amount of viewed deformation. This amount is computed as the ratio of the area of current deformed triangle above the area of the reference visible triangle (see Section A.2 for more details). The areas are computed on the visible triangles in the 2D image (reference and current deformed). This strategy allows us to give more importance to deformed regions in comparison to other areas less or non-deformed. The obtained result shows us that the camera has followed the deformation of the liver without losing the imposed trajectory (test (2)). In test (1), the trajectory of the camera has changed barely and uniformly since the weights of all triangles were equal. In test (2) the camera has updated significantly its trajectory around the deformed area while keeping a trajectory close to the reference one far from the area of deformation. Figure 9 shows the performance of the camera control analysis during the dynamic tracking. It displays the translation and rotation of the camera during the tracking of the liver deformation. The rotation speeds ω_y and ω_z are similar for both tests. The translation amounts in y and z axes are also almost similar. The noticeable difference is in translation and rotation along x-axis which corresponds to the major axis of deformation.

6.2.2. Results with nonlinear deformation of the liver. In this experiment we ran the same configuration of test (1) and test (2) setups as previously. Here we consider a nonlinear deformation with a maximum of 6 mm displacement of the left liver lobe as shown in Fig. 7. The trajectory of the camera is a straight horizontal line represented by green, and the corrected view represented by red color. Figure 10-left shows a corrected camera trajectory in red which was obtained with all the weights being set to 1. Figure 10-right shows a corrected camera trajectory in red which was obtained with the weights of the most deformed regions are equal to 5 and the least deformed or non-deformed has a weight of 1. The curvature of the corrected trajectory is more important in the case of test (2) (right figure). The nonlinear deformation of the triangles in the image is kept as close as possible to their areas in the reference image. The weight functions allow us to take the most relevant deformed triangles from the liver and adjust the amount of correction accordingly. We notice through the Fig. 10 on the right that the camera path is most affected. This is consistent with the nonlinear deformed triangles.

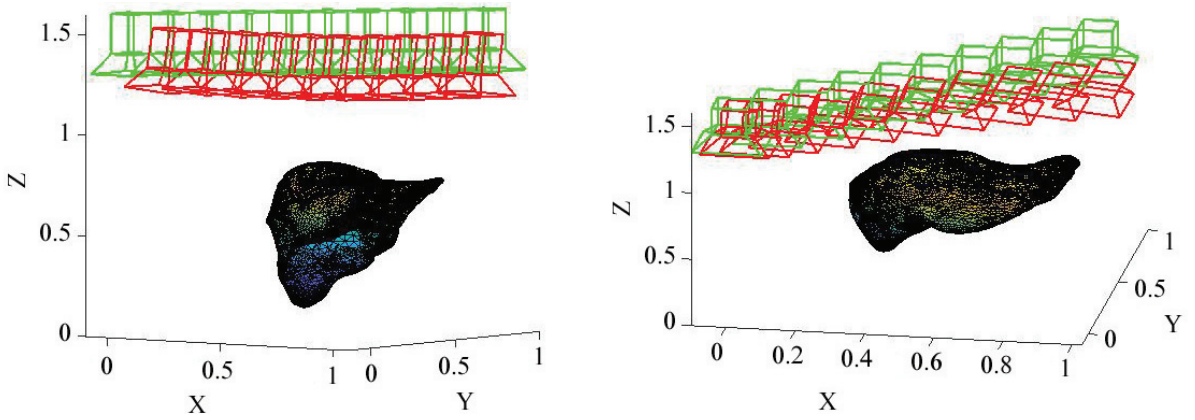

Fig. 10. Liver after a nonlinear elastic deformation and camera follows a reference straight trajectory. Results of test (2) are shown. Left shows side view view. Right shows another side view. Green cameras is the reference trajectory. Red Cameras is the corrected trajectory.

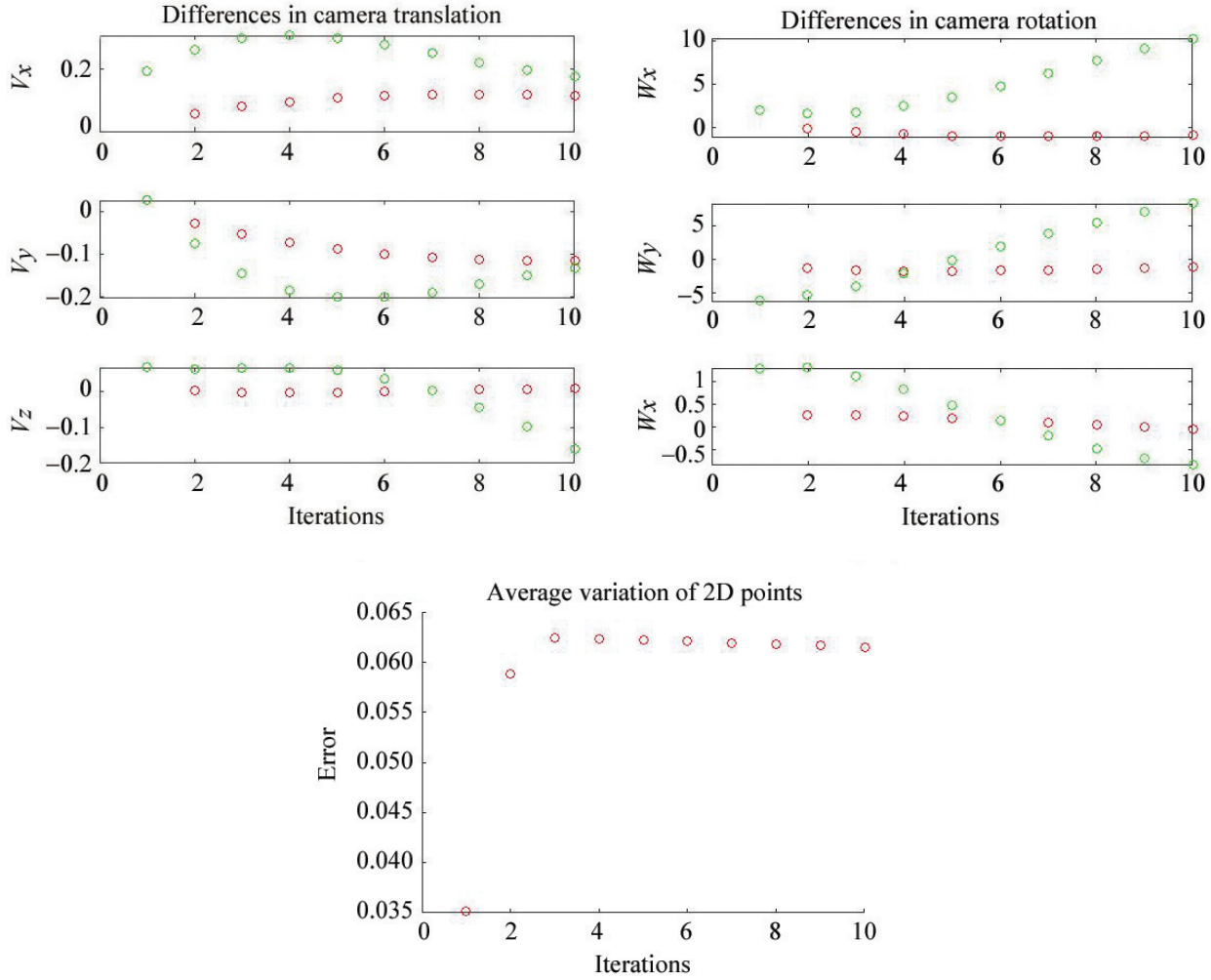

Fig. 11. Analysis of the performance of the camera speed when tracking a deformed object with nonlinear deformation. Left figure represents camera speed in translation. Right figure represents camera speed in rotation. Red dots represent camera speed following strategy of test (1). Green dots represent camera speed following strategy of test (2).

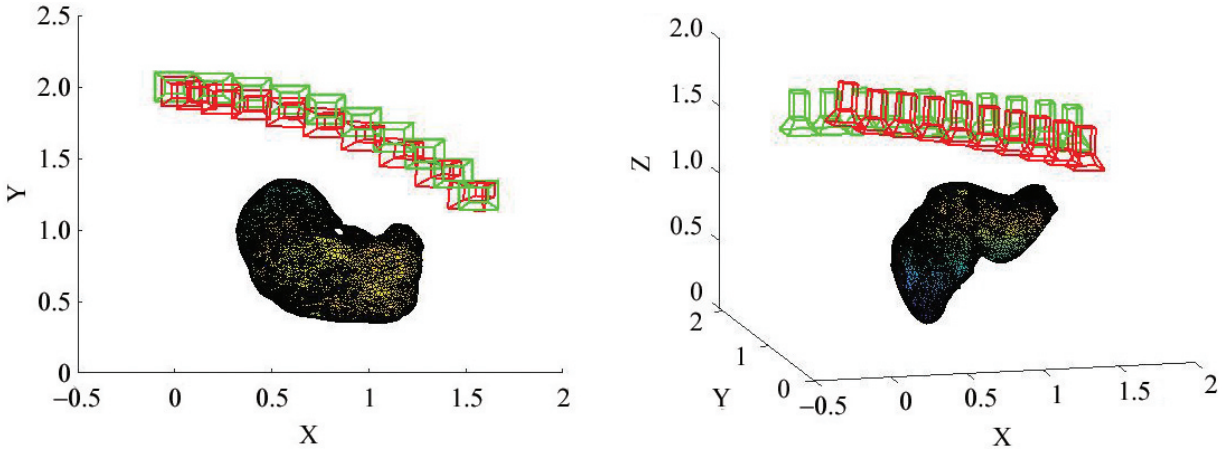

Fig. 12. Liver after a nonlinear elastic deformation and camera follows a reference curvilinear trajectory. Results of test (1) are shown. Left shows top view. Right shows a side view. Green cameras is the reference trajectory. Red Cameras is the corrected trajectory.

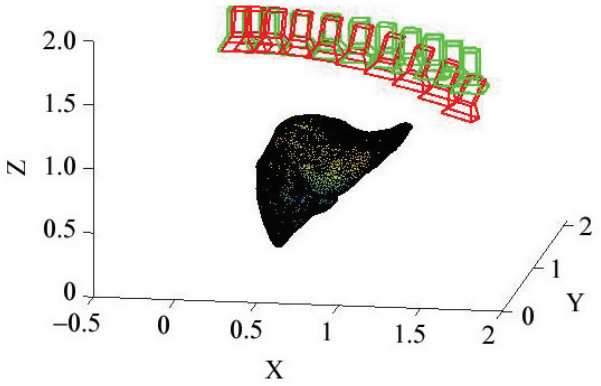

Fig. 13. Liver after a nonlinear elastic deformation and camera follows a reference curvilinear trajectory. Results of test (2) are shown. Green cameras is the reference trajectory. Red Cameras is the corrected trajectory.

We can observe that the convergence speed is quite fast for the approach (after 10 iterations). The result of the computation of the correction of the camera configuration is displayed in Fig. 11.

6.2.3. Results with nonlinear deformation and curvilinear reference trajectory. For a non-linear trajectory, we performed a test on a curvilinear reference trajectory plane with the following parameters: $\theta = (0 : 0.01 : \frac{\pi}{2})$, $c_x = R * \cos(\theta)$, $c_y = R * \cos(\theta)$, $c_z = 1$. The angle of the camera is kept constant during this trajectory pointing down to the negative z-axis. We used this reference camera configuration in equation (22). We tested our control strategy with the two test cited in the Section 6.2.1 (test (1) and test (2)). For test (1), we observed that the camera followed the trajectory as is showed in the Fig. 12 in the right. For test (2), we made a test with the same control strategy and curvilinear trajectory but we changed the weight of the triangles where we gave an importance for the triangles that deform a constant value of weight between 1 and 5 as is showed in the Fig. 13. The result obtained shows us that test (2) is better than the test (1). In test (1), the camera followed the trajectory but the deformation was not well captured. Test 2 succeeded in following the trajectory with a precise deformation capture. In addition, the result of the speed translation and rotation of the camera tracking was better in test (2) than the test (1) as it is shown in Figs. 17. Also, we noted that even in the pixel variation there is a difference which test (1) has a variation of 0.06 to 0.1. and test (2) has a variation of 0 to 0.5.

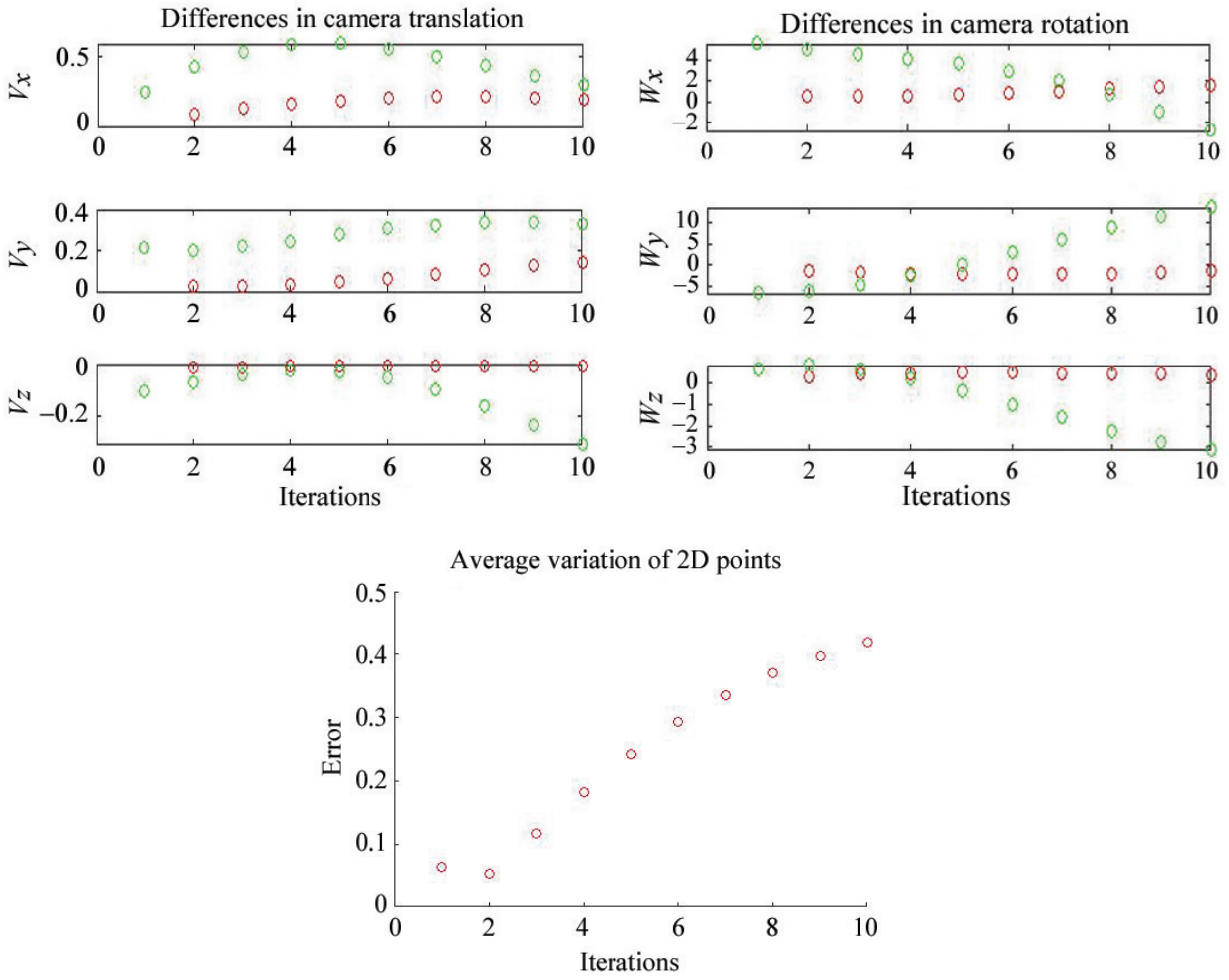

Fig. 14. Analysis of the performance of the camera speed when tracking a deformed object with nonlinear deformation and following a curvilinear trajectory. Left figure represents camera speed in translation. Right figure represents camera speed in rotation. Red dots represent camera speed following strategy of test (1). Green dots represent camera speed following strategy of test (2).

6.3. Robustness to noise in 2D vertex detection

Our method aims at making the camera motion control more robust. Therefore, we tried to mimic the real image measurements perturbations with a Gaussian random noise added to equation (4). This noise is centered with a standard deviation of 1 pixel. It is added to the coordinates of the image im_i, $1 \leqslant i \leqslant m$. We performed a test on a nonlinear deformation with a comparison of both experimental setups (test (1) and test (2)). We observed that for test (1), the corrected trajectory was not as satisfactory when compared to the same setup in the noise-free case (see Fig. 15 for an illustration). For test (2), the result was more consistent with the one obtained in the noise-free case as can be seen in Fig. 16. So we can say that allocating appropriate weights to relevant deformed regions is not influenced by the perturbations while tracking and control the camera trajectory. Figure 17 show the differences speed between the cameras translation, rotation and pixel point variation. The green color is for test (1) and the red color is for test (2). We noticed that test (1) is slow in translation and rotation compared to test (2).

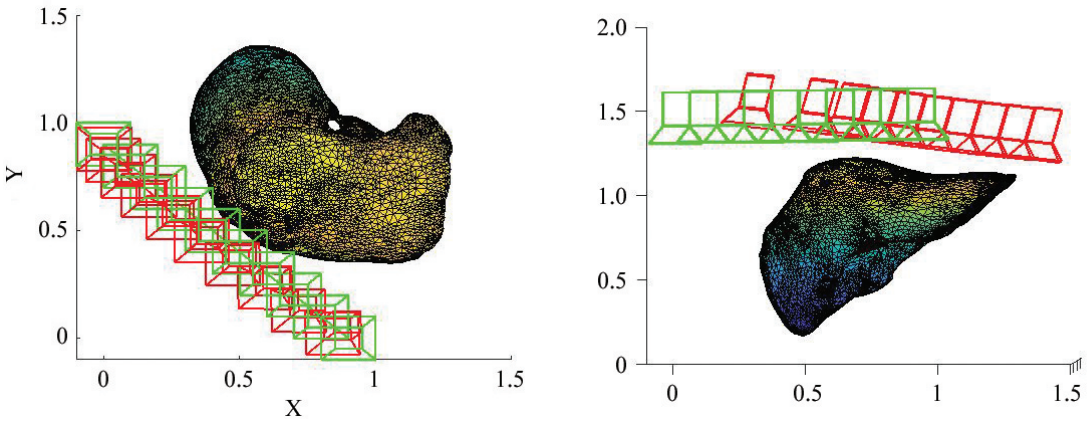

Fig. 15. Liver after a nonlinear elastic deformation and camera follows a reference curvilinear trajectory. Results of test (1) with noisy data are shown. Left shows top view. Right shows a side view. Green cameras is the reference trajectory. Red Cameras is the corrected trajectory.

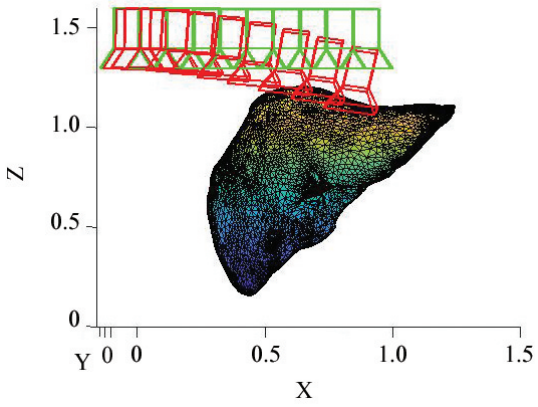

Fig. 16. Liver after a nonlinear elastic deformation and camera follows a reference curvilinear trajectory. Results of test (2) with noisy data are shown. Left shows top view. Right shows a side view. Green cameras is the reference trajectory. Red Cameras is the corrected trajectory.

6.4. Discussions and comments

The importance of the weights is clearly shown in Figs. 5 and 6. On one hand, we can see on Fig. 6-left how the reference trajectory is bent mostly when it comes close to the most deformed area. This behavior is obviously not observed when all the weights are equal to each other. On the other hand, we can see how the view of the deformed region is closer to the reference view in Fig. 6-right when compared to Fig. 5-right. The importance of the optimal visual control framework and its robustness can be seen in the results obtained in Figs. 8 to 17. These figure emphasize the reliability of the proposed approach to the type of deformation (linear/nonlinear), the type of trajectories (geometrically linear/nonlinear) and the noise in 2D detection. We have shown that at every scenario of these experiments, the proposed method allows us to compute a rigid camera correction. Given that the most relevant triangles of the deformable area are given high weight values, the reference trajectory is bent to adapt mostly to those areas no matter are the aforementioned categories.

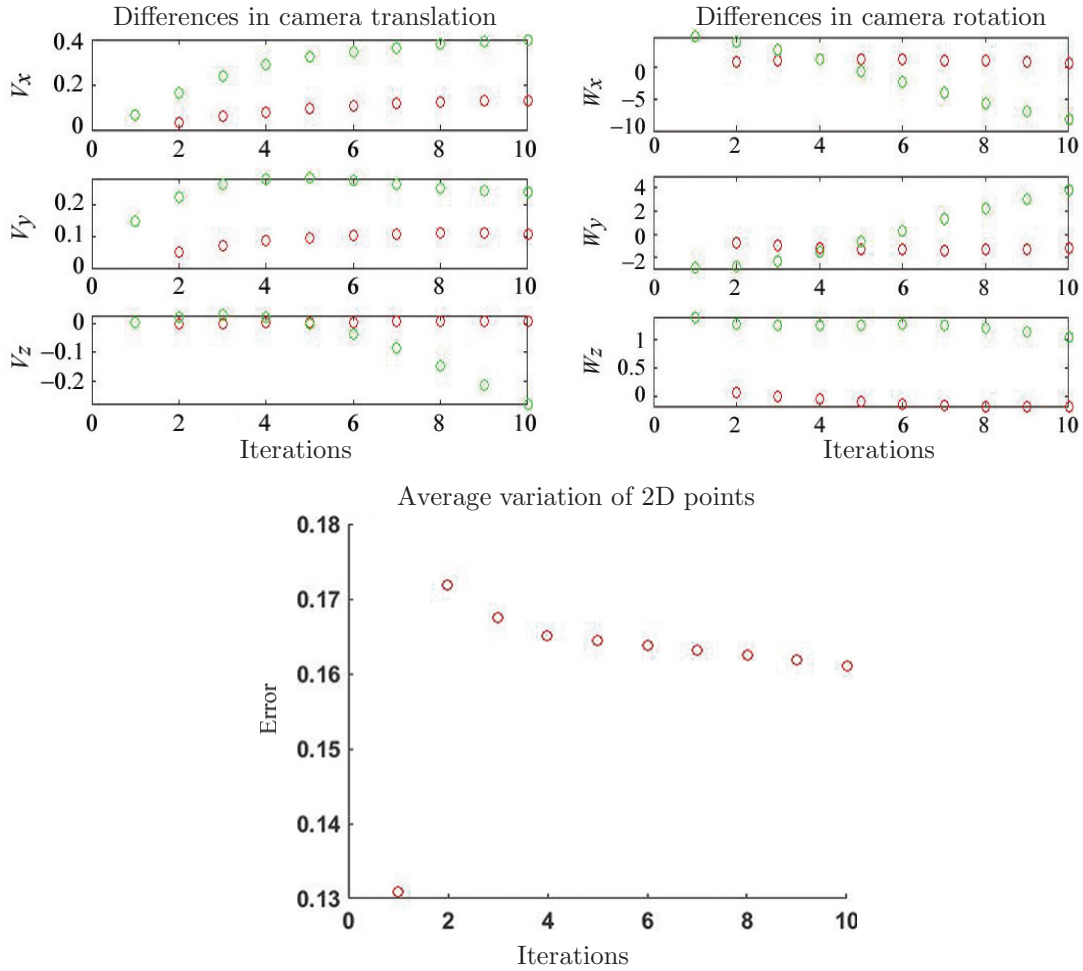

Fig. 17. Analysis of the performance of the camera speed when tracking a deformed object with nonlinear deformation and noisy data. Left figure represents camera speed in translation. Right figure represents camera speed in rotation. Red dots represent camera speed following strategy of test (1). Green dots represent camera speed following strategy of test (2).

7. CONCLUSION

This paper presents a nonparameteric approach to visually compensate nonrigid object deformation with rigid camera motion. This compensation is processed while the camera is following a reference trajectory thanks to continuous weight functions. These weights ensure to focus on the most relevant deforming areas while stabilizing the view at run-time according to the reference view. We embed our compensation strategy in an optimal visual control setup. We experimentally proved that our approach is robust to linear/nonlinear deformations, to linear/nonlinear geometric paths and to noise in 2D detection. We emphasized the fact that our method do not require information on the type of object, deformation or trajectory.

This work allows us to setup strong formal basis for a prospective real application in a laparosurgery context. The surgeon can plan the intervention with the help of a technician giving large weights to relevant area of the target organ and describing a reference geometric path for the laparoscope. During surgery, the motion compensation can be done by a robot holding the laparoscope using our method. Based on state-of-the art approaches of 2D landmark detection in laparoscopy, we will implement a realtime algorithm to detect and build 2D mesh with vertices. This will allows us to fully automate the processed trajectory tracking and visual servoing.

APPENDIX

A.1. COMPUTATION OF THE INTERACTION MATRIX

Given a 3D vertex with coordinates $(X, Y, Z)^\top$ in the camera frame then its 2D perspective projection onto the camera plane is given as

$$x = X/Z, \tag{A.1}$$

$$y = Y/Z. \tag{A.2}$$

Where $im = (x, y)^\top$ is the image coordinate of the 3D point without considering the focal length and the principal point (in our work we consider a calibrated camera of known intrinsics). The time derivative of the above projection about a reference 3D vertex coordinate (X_0, Y_0, Z_0) is given as

$$\dot{x} = \dot{X}/Z - \dot{Z}x/Z, \tag{A.3}$$

$$\dot{y} = \dot{y}/Z - \dot{Z}y/Z. \tag{A.4}$$

Let us consider the translation velocity of the cameras as $v = (v_x, v_y, v_z)^\top$ and its rotation velocity as $\omega = (\omega_x, \omega_y, \omega_z)^\top$ such that the total vector of camera velocity is written as $\dot{c} = (v^\top, \omega^\top)^\top$. The formula relating the velocity of the 3D point to the velocities of the camera can written with the following formula from the classic work by [49]:

$$\dot{X} = -v_x - \omega_y Z + \omega_z Y, \tag{A.5}$$

$$\dot{Y} = -v_y - \omega_z X + \omega_x Z, \tag{A.6}$$

$$\dot{Z} = -v_z - \omega_x Y + \omega_y X. \tag{A.7}$$

Replacing the 3D velocity expression of equations (A.5)–(A.7) into the expressions of 2D velocity (A.3)–(A.4)

$$\dot{x} = -v_x/Z + xv_z/Z + +xy\omega_x - (1 + x^2)\omega_y + y\omega_z, \tag{A.8}$$

$$\dot{y} = -v_y/Z + yv_z/Z + (1 + y^2)\omega_x - xy\omega_y - x\omega_z. \tag{A.9}$$

Rearranging the terms and using the notation of the interaction matrix from equation (6), we can rewrite the above system as

$$(\dot{x}, \dot{y})^\top = L(x, y, Z)\dot{c}. \tag{A.10}$$

If we consider a high enough sampling period T (at least 30 fps), then the above formula can be discretized as follows

$$(x - x_0, y - y_0)^\top/T = L(x, y, Z)(c - c_0)/T. \tag{A.11}$$

Where c is represented by the translation position of the camera center and the Rodriguez vector of rotation

$$c = (t_x, t_y, t_z, \theta_x, \theta_y, \theta_z)^\top. \tag{A.12}$$

Where (t_x, t_y, t_z) are the 3D position coordinates and $\theta = \sqrt{\theta_x^2 + \theta_y^2 + \theta_z^2}$ is the angle of rotation and $(\theta_x/\theta, \theta_y/\theta, \theta_z/\theta)$ is the unit axis of rotation [50]. $c_0 = (t_x^0, t_y^0, t_z^0, \theta_x^0, \theta_y^0, \theta_z^0)^\top$ is a reference camera configuration close to c. Simplifying equation (A.11) by the sampling period on both sides gives

$$(x - x_0, y - y_0)^\top = L(x, y, Z)(c - c_0). \tag{A.13}$$

This mapping is known as the interaction matrix which relates close variations of the camera velocity to close variations of the 2D projected points. Let us consider the mapping Π that is defined in equation (3) which assumes the projection of a 3D triangle as a compound 3 vertices geometric primitive. The first order Taylor expansion of equation (4) about a reference camera configuration c_0 close to a given camera configuration c is provided in equation (5). The term $\frac{\partial \Pi}{\partial c}$ is the interaction matrix that relates the variation of the variation of camera configuration to the 2D projection of the 3 vertices composing the triangle. Thus the analytic expression of $\frac{\partial \Pi}{\partial c}$ is obtained by vertically concatenating the matrix provided in equation (A.10). Thus matrix $\frac{\partial \Pi}{\partial c}$ is 6 columns. It has 6 rows since every vertex' triangle gives 2 equations and every triangle has three of them.

A.2. EXAMPLE OF WEIGHT COMPUTATION WITH VISIBLE AREAS OF TRIANGLES

There is no optimal or exact way of computing the weight functions [39]. However, there many approaches that can be taken to build continuous weight functions. The simplest one is to make them constant as was done the protocol test (1) in our experiment. Another approach associate them to the amount of deformation to take into account deformed regions more than others. In this case, let us assume that we have one undeformed 3D triangle viewed by a camera along a reference trajectory $\gamma(s)$. The 2D projection of this triangle onto the reference camera is continuous along $\gamma(s)$ since it is a static object seen by a continuously moving camera (see Fig. 18). The 2D location of the projected triangle can be written as

$$im_0(s) = \left(x_1^0(s), x_2^0(s), x_3^0(s), y_1^0(s), y_2^0(s), y_3^0(s)\right)^\top. \tag{A.14}$$

The area of the projected triangle along $\gamma(s)$ is also continuous and can be computed as every s as

$$a_0(s) = \frac{1}{2}\det \begin{pmatrix} x_1^0(s) & y_1^0(s) & 1 \\ x_2^0(s) & y_2^0(s) & 1 \\ x_3^0(s) & y_3^0(s) & 1 \end{pmatrix}. \tag{A.15}$$

Where det() denotes the determinant of squared matrices. If we consider $a(s)$ as the area of the projected triangle onto the camera at runtime, then the formula of computing the weight function

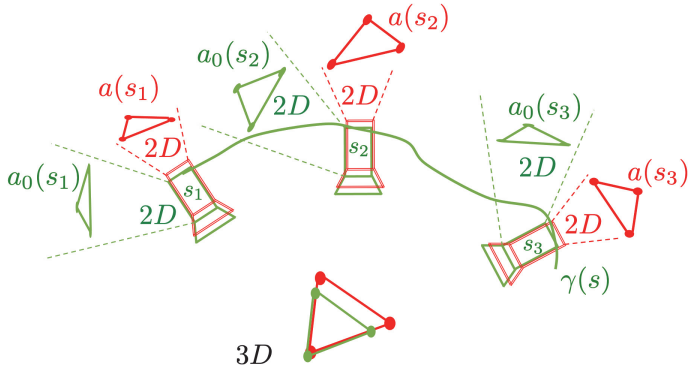

Fig. 18. Example of weight computation based on the area of 2D triangles. In this figure, we consider a reference camera in green viewing one reference triangle in green along a reference trajectory $\gamma(s)$, $0 \leqslant s \leqslant 1$. After deformation, the new deformed triangle is showed in 3D space with red color. The corrected camera along the trajectory is displayed in red color. We picked-up three samples of the trajectory s_1, s_2 and s_3 to depict three couple of views of both the reference triangle from reference camera and deformed triangle from corrected camera pose. As can be seen in the 2D image planes of both cameras, the area of the viewed triangles change continuously along the trajectory. Here the weight is computed as $w(s) = a_0(s)\text{abs}(a_0(s) - a(s))/(1 + a_0(s))$.

can be computed as

$$w(s) = a_0(s)\frac{\text{abs}(a_0(s) - a(s))}{(1 + a_0(s))}. \tag{A.16}$$

Where abs() stands for the absolute value of real numbers. The above defined $w(s)$ is continuous as composed of continuous functions ($a(s)$ and $a_0(s)$). It is positive at every s for $0 \leqslant s \leqslant 1$. When the triangle is not visible in the reference view, its area $a_0(s)$ is null, which makes its weight also zero. The more the triangle is deformed the bigger is $\text{abs}(a_0(s) - a(s))$ which allows us to account more for deformed regions.

REFERENCES

1. Petit, A., Lippiello, V., Fontanelli, G.A., and Siciliano, B., Tracking elastic deformable objects with an RGB-d sensor for a pizza chef robot, *Robotics and Autonomous Systems*, 2017, vol. 88, pp. 187–201. https://doi.org/10.1016/j.robot.2016.08.023. Accessed 2021-06-17.
2. Haouchine, N., Dequidt, J., Peterlik, I., Kerrien, E., Berger, M.-O., and Cotin, S., Image-guided simulation of heterogeneous tissue deformation for augmented reality during hepatic surgery, *IEEE ISMAR*, 2013, pp. 199–208. https://doi.org/10.1109/ISMAR.2013.6671780
3. Rastegarpanah, A., Aflakian, A., and Stolkin, R., Optimized hybrid decoupled visual servoing with supervised learning, *Proc. Inst. Mech. Eng., Part I*, 2021, vol. 236, pp. 09596518211028379. https://doi.org/10.1177/09596518211028379. Accessed 2021-07-21.
4. Chi, C. and Berenson, D., Occlusion-robust deformable object tracking without physics simulation, *IEEE/RSJ IROS*, Macau, China, 2019, pp. 6443–6450. https://arxiv.org/abs/2101.007332101.00733. https://doi.org/10.1109/IROS40897.2019.8967827. Accessed 2021-06-29.
5. Lagneau, R., Shape Control of Deformable Objects by Adaptive Visual Servoing, *INSA de Rennes*, 2020. https://tel.archives-ouvertes.fr/tel-03087518. Accessed 2021-06-17.
6. Lagneau, R., Krupa, A., and Marchal, M., Active Deformation through Visual Servoing of Soft Objects, *IEEE ICRA*, Paris, France, 2020, pp. 8978–8984. https://doi.org/10.1109/ICRA40945.2020.9197506
7. Chaumette, F. and Ikeuchi, K. (ed.), Visual Servoing, *Computer Vision: A Reference Guide*, Springer, Boston, MA, 2014, pp. 869–874. https://doi.org/10.1007/978-0-387-31439-6_281
8. Tahri, O. and Chaumette, F., Image moments: generic descriptors for decoupled image-based visual servo, *IEEE ICRA*, 2004, vol. 2, pp. 1185–11902. https://doi.org/10.1109/ROBOT.2004.1307985
9. Agravante, D.J., Claudio, G., Spindler, F., and Chaumette, F., Visual servoing in an optimization framework for the whole-body control of humanoid robots, *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 608–615. https://doi.org/10.1109/LRA.2016.2645512
10. Ren, X., Li, H., and Li, Y., Image-based visual servoing control of robot manipulators using hybrid algorithm with feature constraints. *IEEE Access*, 2020, vol. 8, pp. 223495–223508. https://doi.org/10.1109/ACCESS.2020.3042207
11. Wang, T., Wang, W., and Wei, F., An overview of control strategy and trajectory planning of visual servoing / Fei, M., Li, K., Yang, Z., Niu, Q., and Li, X. (eds.) *Recent Featured Applications of Artificial Intelligence Methods. LSMS 2020 and ICSEE 2020 Workshops*, Springer, Singapore, 2020, pp. 358–370.
12. Corke, P.I. and Hutchinson, S.A., A new partitioned approach to image-based visual servo control, *IEEE Transactions on Robotics and Automation*, 2001, vol. 17, no. 4, pp. 507–515. https://doi.org/10.1109/70.954764.
13. Gans, N.R., Corke, P.I., and Hutchinson, S.A., Performance Tests for Visual Servo Control Systems, with Application to Partitioned Approaches to Visual Servo Control, *The International Journal of Robotics Research*, 2003, vol. 22, no. 10–11, pp. 955–981. https://doi.org/10.1177/027836490302210011. Accessed 2021-03-07.
14. Janabi-Sharifi, F. and Wilson, W.J., Automatic selection of image features for visual servoing, *IEEE Transactions on Robotics and Automation*, vol. 13, no. 6, pp. 890–903. https://doi.org/10.1109/70.650168
15. Chaumette, F., Image moments: a general and useful set of features for visual servoing, *IEEE Transactions on Robotics*, 2004, vol. 20, no. 4, pp. 713–723. https://doi.org/10.1109/TRO.2004.829463

16. Molná, C., Nagy, T.D., Elek, R.N., and Haidegger, T., Visual servoing-based camera control for the da vinci surgical system, *2020 IEEE 18th International Symposium on Intelligent Systems and Informatics (SISY)*, Subotica, Serbia, 2020, pp. 107–112. https://doi.org/10.1109/SISY50555.2020.9217086

17. Mohamed, I., MPPI-VS: Sampling-Based Model Predictive Control Strategy for Constrained Image-Based and Position-Based Visual Servoing, 2021. https://doi.org/10.48550/arXiv.2104.04925

18. Lagneau, R., Krupa, A., and Marchal, M., Active deformation through visual servoing of soft objects, *2020 IEEE International Conference on Robotics and Automation (ICRA)* Paris, France, 2020, pp. 8978–8984. https://doi.org/10.1109/ICRA40945.2020.9197506

19. Hu, Z., Han, T., Sun, P., Pan, J., and Manocha, D., 3-d deformable object manipulation using deep neural networks, *IEEE Robotics and Automation Letters*, 2019, vol. 4, no. 4, pp. 4255–4261. https://doi.org/10.1109/LRA.2019.2930476

20. Jia, B., Hu, Z., Pan, J., and Manocha, D., Manipulating highly deformable materials using a visual feedback dictionary, *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, Brisbane, QLD, Australia, 2018, pp. 239–246. https://doi.org/10.1109/ICRA.2018.8461264

21. Hu, Z., Sun, P., and Pan, J., Three-dimensional deformable object manipulation using fast online gaussian process regression, *IEEE Robotics and Automation Letters*, 2018, vol. 3, no. 2, pp. 979–986. https://doi.org/10.1109/LRA.2018.2793339

22. Zhu, J., Vision-based robotic manipulation of deformable linear objects, *PhD thesis*, Université Montpellier, 2020.

23. Chen, Z., Li, S., Zhang, N., Hao, Y., and Zhang, X., Eye-to-hand robotic visual tracking based on template matching on fpgas, *IEEE Access*, 2019, vol. 7, pp. 88870–88880. https://doi.org/10.1109/ACCESS.2019.2926807

24. Staneva, V. and Younes, L., Modeling and estimation of shape deformation for topology-preserving object tracking, *SIAM Journal on Imaging Sciences*, 2014, vol. 7, no. 1, pp. 427–455. https://doi.org/10.1137/130919714

25. Hu, Y., Carter, T.J., Ahmed, H.U., Emberton, M., Allen, C., Hawkes, D.J., and Barratt, D.C., Modelling prostate motion for data fusion during image-guided interventions, *IEEE Transactions on Medical Imaging*, 2011, vol. 30, no. 11, pp. 1887–1900. https://doi.org/10.1109/TMI.2011.2158235

26. Chen, Q., Sun, Q.-S., Heng, P.A., and Xia, D.-S., Two-stage object tracking method based on kernel and active contour, *IEEE Transactions on Circuits and Systems for Video Technology*, 2010, vol. 20, no. 4, pp. 605–609. https://doi.org/10.1109/TCSVT.2010.2041819

27. Cao, X., Lan, J., and Rong Li, X., Extension-deformation approach to extended object tracking, *2016 19th International Conference on Information Fusion (FUSION)*, Germany, 2016, pp. 1185–1192.

28. Joo, H., Simon, T., and Sheikh, Y., Total capture: A 3d deformation model for tracking faces, hands, and bodies, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Proc.*, 2018.

29. Royer, L., Marchal, M., Le Bras, A., Dardenne, G., and Krupa, A., Real-time tracking of deformable target in 3d ultrasound images, *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 2430–2435. https://doi.org/10.1109/ICRA.2015.7139523

30. Royer, L., Krupa, A., Dardenne, G., Bras, A.L., Marchand, E., and Marchal, M., Real-time target tracking of soft tissues in 3d ultrasound images based on robust visual information and mechanical simulation, *Medical Image Analysis*, 2017, vol. 35, pp. 582–598. https://doi.org/10.1016/j.media.2016.09.004

31. Kajihara, K., Huang, S., Bergström, N., Yamakawa, Y., and Ishikawa, M., Tracking of trajectory with dynamic deformation based on dynamic compensation concept, *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Macau, Macao, 2017, pp. 1979–1984. https://doi.org/10.1109/ROBIO.2017.8324709

32. Zhou, H., Ma, J., Tan, C.C., Zhang, Y., and Ling, H., Cross-weather image alignment via latent generative model with intensity consistency, *IEEE Transactions on Image Processing*, 2020, vol. 29, pp. 5216–5228. https://doi.org/10.1109/TIP.2020.2980210

33. Toriya, H., Dewan, A., and Kitahara, I., Sar2opt: Image alignment between multi-modal images using generative adversarial networks, *IGARSS 2019 – 2019 IEEE International Geoscience and Remote Sensing Symposium*, 2019, pp. 923–926. https://doi.org/10.1109/IGARSS.2019.8898605

34. Kagami, S., Omi, K., and Hashimoto, K., Alignment of a flexible sheet object with position-based and image-based visual servoing, *Advanced Robotics*, 2016, vol. 30, no. 15, pp. 965–978. https://doi.org/10.1080/01691864.2016.1183518

35. Shen, Xi, Darmon, F., Efros, A., and Aubry M., Ransac-flow: Generic two-stage image alignment, *Computer Vision-ECCV 2020*, 2020, vol. 12349, pp. 618–637. https://doi.org10.1007/978-3-030-58548-8_36

36. Dong, Y., Liang, T., Zhang, Y., and Du, B., Spectral spatial weighted kernel manifold embedded distribution alignment for remote sensing image classification, *IEEE Transactions on Cybernetics*, 2021, vol. 51, no. 6, pp. 3185–3197. https://doi.org/10.1109/TCYB.2020.3004263

37. Mathiassen, K., Glette, K., and Elle, O.J., Visual servoing of a medical ultrasound probe for needle insertion, *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp 3426–3433. https://doi.org/10.1109/ICRA.2016.7487520

38. Mura, M., Abu-Kheil, Y., Ciuti, G., Visentini-Scarzanella, M., Menciassi, A., Dario, P., Dias, J., and Seneviratne, L., Vision-based haptic feedback for capsule endoscopy navigation: a proof of concept, *Journal of Micro-Bio Robotics*, vol. 11, pp. 35–45. https://doi.org/10.1007/s12213-016-0090-2. Accessed 2021-03-23.

39. Malti, A., Tax, M., and Lamiraux, F., A general framework for planning landmark-based motions for mobile robots, *Advanced Robotics*, 2011, vol. 25, no. 11–12, pp. 1427–1450. https://doi.org/10.1163/016918611X579457

40. Sengupta, A., Krupa, A., and Marchand, E., Visual Tracking of Deforming Objects Using Physics-based Models, *2021 IEEE International Conference on Robotics and Automation (ICRA)*, Xi'an, China, 2021, pp. 14178—14184. https://doi.org/10.1109/ICRA48506.2021.9561401

41. Feng, X., Mei, W., and Hu, D., A Review of Visual Tracking with Deep Learning, *AIIE 2016 Proc.*, 2016, pp. 231–234. https://doi.org/10.2991/aiie-16.2016.54

42. Marvasti-Zadeh, S.M., Cheng, L., Ghanei-Yakhdan, H., and Kasaei, S., Deep Learning for Visual Tracking: A Comprehensive Survey, *IEEE Transactions on Intelligent Transportation Systems*, 2021, vol. 23, no. 5, pp. 3943–3968. https://doi.org/10.1109/TITS.2020.3046478

43. Malti, A., Hartley, R., Bartoli, A., and Kim, J.-H., Monocular template-based 3d reconstruction of extensible surfaces with local linear elasticity, *2013 IEEE Conference on Computer Vision and Pattern Recognition*, Portland, OR, USA, 2013, pp. 1522–1529, https://doi.org/10.1109/CVPR.2013.200

44. Malti, A., Bartoli, A., and Hartley, R., A linear least-squares solution to elastic shape-from-template, *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA, 2015, pp. 1629–1637. https://doi.org/10.1109/CVPR.2015.7298771

45. Malti, A. and Herzet, C., Elastic shape-from-template with spatially sparse deforming forces, *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 2017, pp. 143–151. https://doi.org/10.1109/CVPR.2017.23

46. Casillas-Perez, D., Pizarro, D., Fuentes-Jimenez, D., Mazo, M., and Bartoli, A., Equiareal Shape-from-Template, *Journal of Mathematical Imaging and Vision*, 2019, vol. 61, no. 5, pp. 607–626.

47. Ben-Israel, A. and Greville, T.N.E., *Generalized Inverses Theory and Applications*, New York: Springer, 2003. Google-Books-ID: o_zXUXaqGU8C.

48. Saidi, F. and Malti, A., Fast and accurate nonlinear hyper-elastic deformation with a posteriori numerical verification of the convergence of solution: Application to the simulation of liver deformation, *Int. J. Numer. Meth. Biomed. Engng.*, 2021, 37:e3444. https://doi.org/10.1002/cnm.3444. Accessed 2021-09-28.

49. Chaumette, F. and Hutchinson, S., Visual servo control. i. basic approaches, *IEEE Robotics Automation Magazine*, vol. 13, no. 4, pp. 82–90. https://doi.org/10.1109/MRA.2006.250573

50. Murray, R.M., Li, Z., and Sastry, S.S., *A Mathematical Introduction to Robotic Manipulation*, Boca Raton, CRC Press, 1994. https://doi.org/10.1201/9781315136370

This paper was recommended for publication by D.V. Vinogradov, a member of the Editorial Board