Robotics and Autonomous Systems 88 (2017) 187–201

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Tracking elastic deformable objects with an RGB-D sensor for a pizza chef robot

Antoine Petit *, Vincenzo Lippiello, Giuseppe Andrea Fontanelli, Bruno Siciliano

DIETI, Università degli Studi di Napoli Federico II, Italy

HIGHLIGHTS

- Track in real-time a 3D textureless deformable object with an RGB-D sensor.
- Modeling large elastic deformations based on FEM.
- Efficient registration using nearest neighbors correspondences and contours.
- Experimented on a humanoid robotic platform for manipulation.

ARTICLE INFO

Article history:
Received 30 July 2015
Received in revised form 20 May 2016
Accepted 25 August 2016
Available online 13 September 2016

Keywords:
Perception
Deformable object modeling
Registration
Robotic manipulation

ABSTRACT

This paper presents a method for tracking a 3D textureless object which undergoes elastic deformations, using the point cloud data provided by an RGB-D sensor and in real-time. This solution is expected to be useful for enhanced manipulation of humanoid robotic systems, especially in the case of pizza dough to be ideally manipulated by a pizza chef robot. Our tracking framework relies on a prior visual segmentation of the object in the image. The segmented point cloud is registered first in a rigid manner and then by non-rigidly fitting the mesh, based on the Finite Element Method to model elasticity, and on geometrical point-to-point correspondences to compute external forces exerted on the mesh. The system has been evaluated on synthetic and real data, and by integrating it into manipulation experiments on the RoDyMan[1] humanoid robotic platform.

1. Introduction

Unlike vision-based tracking problems with rigid objects, for which a certain maturity has been reached, perception for non-rigid objects is still a challenging problem. It has aroused much interest in recent years in the computer vision, computer graphics [1,2] and robotics communities [3]. Many potential applications would indeed be 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.

This study was conducted in the context of the RoDyMan project, consisting of a unified framework for robotic dynamic manipulation of deformable objects. As seen in Fig. 1, a demonstration scenario is the humanoid dual-arm/hand manipulation of the pizza dough, in an authentic manner, which requires complex dynamic manipulation tasks of a deformable object, tasks which are currently unfeasible with the prototypes available in the robotics research community. Research has increasingly focused on robots involved in food manipulation tasks in recent years [4–7]. Although, some progress has been made in the food processing industries, this project tackles food manipulation in an original and creative manner through the development of a humanoid robot involved in culinary traditions and rituals, similar in spirit to the work of Higashimori et al. [8]. This paper addresses a central issue within the project which is perception, with the aim of providing to the robot controller accurate, robust and real-time sensing of the manipulated deformable object.

With respect to rigid objects, the problem of dealing with deformations in a perception system poses several additional challenges such as modeling the properties of the considered material, and fitting this model with the vision and/or range data. This registration problem also involves critical real-time concerns, which are

* Corresponding author.
E-mail addresses: antoine.petit@unina.it (A. Petit), vincenzo.lippiello@unina.it (V. Lippiello), g.a.fontanelli@gmail.com (G.A. Fontanelli), bruno.siciliano@unina.it (B. Siciliano).

[1] http://www.rodyman.eu/. The research leading to these results has been supported by the RoDyMan project, which has received funding from the European Research Council (FP7 IDEAS) under Advanced Grant agreement number 320992.

http://dx.doi.org/10.1016/j.robot.2016.08.023
0921-8890/

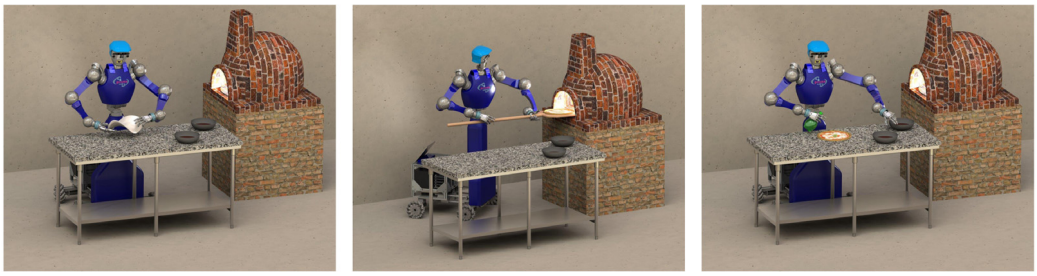


Fig. 1. Artistic views of the RoDyMan robotic platform and the pizza making process.

especially required for robotic dynamic manipulation. Although numerous studies have proposed efficient real-time techniques to handle 3D surfaces (paper, clothes) which undergo isometric or slightly elastic deformations, a large open field remains when considering larger elastic deformations. The aim of this paper is thus to propose a real-time tracking system able to handle elastic objects, potentially textureless, by tracking large deformations and fast rigid motions, using visual and range data provided by an RGB-D sensor.

To cope with deformations, our approach involves a physical modeling of the considered object, by relying on a Finite Element Method (FEM). The considerable progresses recently made within the computer graphics and medical simulation domains have enabled real-time performance for processing such models. As demonstrated in this paper, our whole system is able to run fast at around 35 frame per seconds. Another goal has been to integrate this registration method within a manipulation control task on the RoDyMan robotic platform, using simple planning and control features.

The remainder of the paper is organized as follows. After a review of the related state-of-the-art in Section 2, Section 3 describes the prior visual segmentation method of the RGB-D data. In Section 4 the physical model of the object considered here is introduced, Section 5 explains how the segmented point cloud data is processed and matched with the model to perform registration. Finally, some experimental results are presented in Section 6, in which a focus on the RoDyMan humanoid robotic platform is provided, with a line of robotic manipulation experiments involving the RoDyMan platform.

2. Related works and motivations

In the literature, the various approaches proposed to register deformable objects, using vision and/or range data, could be classified according to the underlying model of the considered object and its physical realism. Let us first clarify our scope and distinguish it from non-rigid reconstruction methods for which at each frame provided by the vision/range sensor, a single mesh is reconstructed, as in [9,10]. Instead, the goal is here to continuously estimate the rigid transformations and the deformations undergone by the object, modeled by a known mesh.

2.1. Registration using implicit physical modeling

Based on implicit physical models, approaches in [11–13] use a 1D parametric curve or 2D splines models (B-splines, Radial Basis Functions) to track deformable objects in monocular images. This class of methods relies on the minimization of an energy function involving an external energy term related to some image features, and an internal energy term regularizing curvature, bending or twisting, compelling the model to vary smoothly. Adapting these techniques to register with 3D shapes or surfaces in monocular images is much more complex, since 3D deformations can imply ambiguous 2D transformations, resulting in an underconstrained problem. A first attempt by Terzopoulos et al. [14], relying on 3D splines and inspired by Kass et al. [11], densely processes gradient features, to compute the data energy term. Less ambiguous feature-based approaches [15] have been preferred and additional constraints are often added to solve ambiguities. With point cloud data, methods in [16,17] employ an RGB-D sensor to register the acquired point cloud to a surface mesh by minimizing an error function accounting for geometric or direct depth and color errors, and a stretching penalty function for the mesh. By means of a NURBS parametrization [16] or an optimized GPU implementation [17], real-time performance can be achieved. Although these two systems have shown promising and impressive results, they are still bounded to isometric or slightly elastic deformations, by means of regularization functions proportional to squared distances between nodes of the mesh, whereas we wish to model elastic in more physically realistic manner, to handle volumetric effects and larger strains. Another limitation of these methods is that they process mesh to input point cloud correspondences in their data error functions, and are thus sensitive to missing data, or unobserved areas of the considered object due to occlusions. We consider in this paper also correspondences from the input point cloud to the mesh, through the use of a segmentation method to restrict the input point cloud to the observed areas of the object, and based on these correspondences, the occluded or unobserved areas would not affect registration.

We can also mention template free non-rigid reconstruction methods for which at each frame provided by the vision/range sensor, a single mesh of the sensed scene is reconstructed and tracked, as in [9,10,18,19]. However, these dense and exhaustive methods, besides often failing fulfilling real-time constraints, are limited to isometric slightly elastic deformations.

2.2. Registration using explicit physical modeling

Instead, another formulation of the problem relies on physics-based deformable models to perform registration, by modeling more explicitly elasticity. With respect to implicit methods, other sorts (such as non-linear elasticity) and magnitudes of deformations can be treated, inferring more consistently shape and/or volumetric regularization. Statistically, the solution can be determined, by setting internal and external forces equal or, equivalently, minimizing energy functions. Physics-based methods include discrete mass–spring–damper systems [20–22], or more explicit approaches relying on the Finite Element Method (FEM), based on continuum mechanics. In [22], based on a mass–spring–damper system, 3D–3D correspondences, determined through a probabilistic inference, enable the computation of the external forces applied to the mesh. However mass–spring systems are limited in terms of magnitude of elastic deformations, and do not guarantee certain mechanical properties such as volume preservation, failing to handle volumetric effects. First attempts for registration

employing the FEM for 3D surfaces in [23,24] used linear elasticity FEM models. More recently in [25], registration in monocular images is addressed by designing a stretching/shrinking energy using continuous mechanical constraints on 2D elements assuming linear elasticity, and some 3D boundary conditions. Haouchine et al. [26] uses a linear tetrahedral co-rotational FEM model, coping with larger elastic deformations, external forces being related to correspondences between tracked 3D feature points mapped to the 3D mesh by means of a stereo camera system. This system has been then extended in [27] by involving a non-linear St-Venant Kirchoff FEM model to efficiently handle hyperelastic behaviors, registration relying on monocular keypoint based external forces. To the best of our knowledge, this latter method proposes the most realistic physical elastic model within a real-time vision-based tracking system. However, both approaches [26,27] do not address issues such as capturing rigid motions and require textured materials.

2.3. Motivations and contributions

Since our system would attempt to handle large deformations and elastic volumetric strains, a realistic mechanical model, based on the FEM, has been adopted. Besides, for potential robotic dynamic manipulation applications, an explicit physical modeling would enable the reliable computation and prediction of internal forces undergone by the object and thus to perform proper force control tasks. The recent suitability of these models for real-time applications, as demonstrated by promising approaches [21,22,26,27], has confirmed our choice. We assume the prior knowledge of a consistent mesh (which could be automatically reconstructed offline) and of the material properties (through the Young modulus and the Poisson ratio). Robustness concerns, regarding for instance textureless objects, have lead us to rely on an RGB-D sensor.

Among the methods having the closest goals, motivations and constraints to ours, based on RGB-D or point cloud data, we can mention [16,22,26,17]. With respect to them, several contributions are proposed, such as handling various large deformations like elastic ones while ensuring physical consistency, handling rigid motions, occlusions, and addressing all these tasks in real-time (35 fps).

2.4. Overview of the system

As also represented in Fig. 2, our frame-by-frame tracking system can be outlined as follows:

Input: the known 3D volumetric mesh of the object, a given RGB-D data, and assuming a fair registration at the previous time step.

1. Visual segmentation of the considered object, with a graph cut-based approach ensuring temporal coherence.
2. Using the resulted segmented point cloud, perform a rigid Iterative Closest Point (ICP) to estimate a rigid transformation from the point cloud to the mesh.
3. Using the resulting segmented point cloud, compute external linear elastic forces exerted on the vertices of the mesh from the point cloud to the mesh and conversely, based on closest point correspondences.
4. Numerical resolution of mechanical equations to compute the deformations, based on a tetrahedral linear co-rotational FEM model.

3. Segmentation

In this work we advocate the use of a prior visual segmentation step in order to restrict each successive acquired point cloud to the object of interest (see Section 5.1 for a more detailed justification).

3.1. Grabcut segmentation

We rely here on the efficient and widespread *GrabCut* method [28], based on graph cuts. In its original formulation, the *Grabcut* algorithm addresses the visual bilayer segmentation task as an energy minimization problem, based on statistical models of the foreground (the object) and the background.

For an input image $\mathbf{I}$, we denote by $\alpha = \{\alpha_i\}_{i=1}^{N}$ the set of the unknown binary labels of the set of pixels ($\alpha_i = 0$ for the background pixels, $\alpha_i = 1$ for the foreground). Estimating the values $\widehat{\alpha}$ of the labels can be formulated as the minimization of an energy-based Markov Random Field objective function $E(\alpha)$, with respect to α:

$$\widehat{\alpha} = arg\min_{\alpha} E(\alpha) \tag{1}$$

$$\text{with} \quad E(\alpha) = E_{data}(\alpha) + \gamma E_{smooth}(\alpha) \tag{2}$$

$$\text{and} \quad E_{data}(\alpha) = \sum_i U_i(\alpha_i) \tag{3}$$

E_{data} is the data energy term, with $U_i(\alpha_i)$ a unitary term accounting for the observation probability $p(\mathbf{p}_i \mid \alpha_i)$ for a pixel $\mathbf{p}_i$ to belong to the foreground or to the background, based on some image "data" (intensity, color, location, ...) observed on the pixel, using the statistical models built for the background and the foreground. More formally, we have $U_i(\alpha_i) = -\log(p(\mathbf{p}_i \mid \alpha_i))$. E_{smooth} is the smoothness energy term whose goal is to favor smoothness, or spatial coherence within the pixels.

In order to compute the optimal solution of this energy minimization problem and determine $\widehat{\alpha}$, a *graph cuts* minimization algorithm [29] is employed, providing us with a segmented frame $\mathbf{I}^s$.

Statistical models for the data energy function are Gaussian Mixture Models (GMM) based on color distributions, learned for both the foreground and background layers, which are initially determined by the user through a bounding box manually defined around the foreground on the initial image. Besides, pixels outside this bounding box are definitely assigned to the background layer ($U_i(\alpha_i = 0) = inf$), whereas inside their label is unknown, so that energy minimization only has effects inside the bounding box.

3.2. Temporal coherence and real-time issues

Once the initial image is segmented through user interaction, the following frames are treated similarly, except that instead of segmenting the whole frame, the area to effectively segment is updated frame-by-frame, around the silhouette contour of the segmented area, providing temporal coherence in the segmentation process. More precisely and as shown in Fig. 3, the silhouette contour of the previous segmented foreground is extracted, and the distance transform is computed over it, providing a signed distance map $\mathbf{d}$ to these contours (negative outside, positive inside). According to a fixed threshold d_t on this distance map, we define a narrow strip around the contour ($|\mathbf{d}_i| < d_t$, gray area), in which the label of a pixel $\mathbf{p}_i$ is unknown), with $U_i(\alpha_i) = -\log(p(\mathbf{p}_i|\alpha_i))$, whereas it is definitely assigned to the foreground on the inner side of the strip ($\mathbf{d}_i < -d_t$, white area), by setting $U_i(\alpha_i = 1) = inf$, and to the background otherwise ($\mathbf{d}_i > d_t$, black area), by setting in this case $U_i(\alpha_i = 0) = inf$. In this manner, temporal consistency is ensured, since energy minimization is only effective within this strip, in the vicinity of the previous segmentation boundary, avoiding some outliers outside or inside, and reducing significantly computations. Let us note that the GMMs are determined initially, making this segmentation process valid for sequences for which color distributions of both the foreground and the background are assumed quite constant, which is a fair assumption for the robotic manipulation applications considered in this paper.

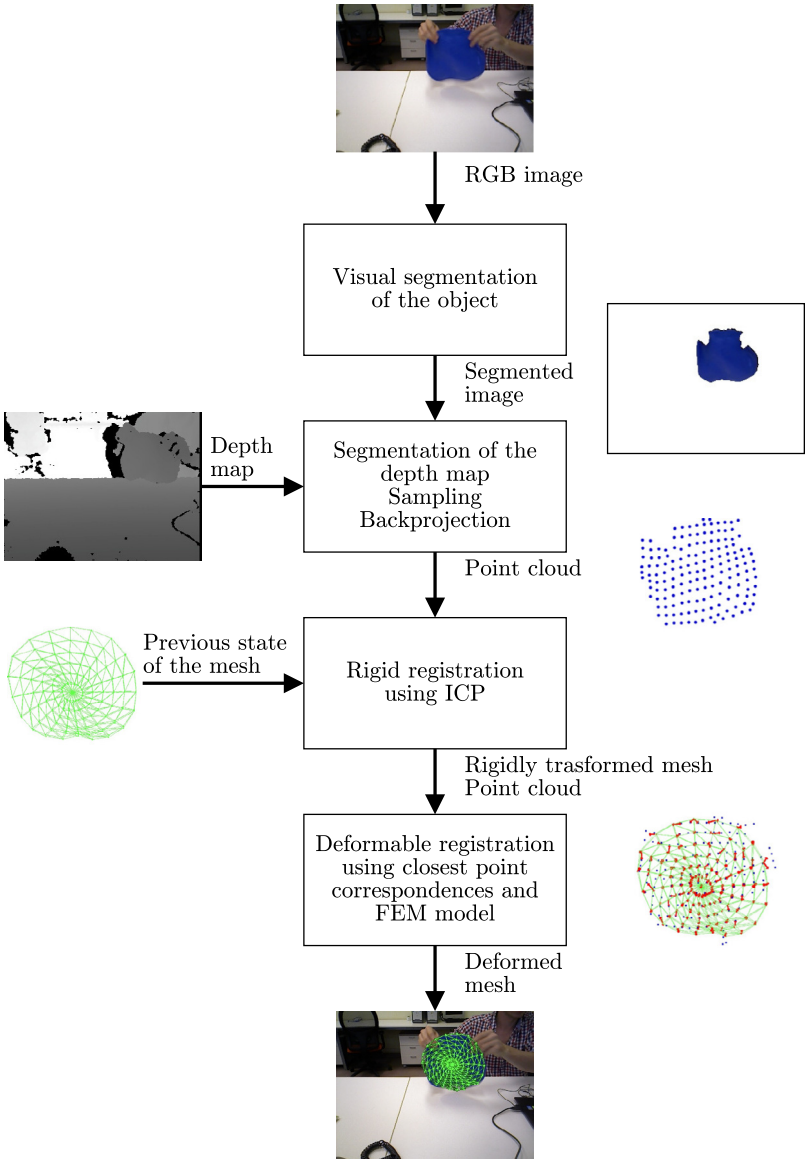


Fig. 2. Overview of our approach for deformable object tracking.

(a) Original frame. (b) Segmentation. (c) Distance map. (d) Trimap.

Fig. 3. Temporal consistency for segmentation. Segmentation will be effective on the strip (gray area on (d)) around the contour of the previous segmented frame (b), through the distance map to the contour (c).

4. Deformable object modeling with the finite element method

Since we deal with objects which may undergo large elastic deformations, a major issue lies in the definition of a relevant physical model. The Finite Element Method (FEM) provides a realistic physical model, by relying on continuum mechanics, instead of finite differences for mass–spring systems for instance. For an exhaustive description, the reader can refer to [30]. It consists in tessellating the deformable object into a mesh made of elements. We rely here on a volumetric linear FEM approach with tetrahedral elements, which are preferred over other topologies such as triangles or hexahedrons, due to their modeling accuracy (w.r.t to triangles for instance), while being computationally efficient, and convenient for meshing volumes with topologies of any complexity (w.r.t to hexahedrons). The deformation field $\mathbf{u}_e$ over an element e is then approximated as a continuous interpolation of the displacement

vectors $\hat{\mathbf{u}}_e$ of its four vertices, through the 3×12 matrix $\mathbf{N}_e(\mathbf{x})$, which contains the polynomial basis functions of the element:

$$\mathbf{u}_e(\mathbf{x}) = \mathbf{N}_e(\mathbf{x})\hat{\mathbf{u}}_e, \tag{4}$$

with being $\mathbf{x}$ a point in the element, $\hat{\mathbf{u}}_e = \mathbf{x}_e - \mathbf{x}_{e,0}$, $\mathbf{x}_e$ and $\mathbf{x}_{e,0}$ respectively the deformed and initial world coordinates of the four vertices of e.

4.1. Modeling elastic deformations

In order to model deformations and elasticity, we resort to the infinitesimal strain theory [30] to compute the Cauchy's linear strain tensor $\boldsymbol{\epsilon}_e$ within the tetrahedron. It can be linearly expressed with respect to $\hat{\mathbf{u}}_e$:

$$\boldsymbol{\epsilon}_e = \mathbf{L}_e\hat{\mathbf{u}}_e, \tag{5}$$

with $\mathbf{L}_e$ a constant 6×12 matrix. In order to relate the stress with the strain, we then rely on Hooke's law linear elasticity theory for a continuous isotropic material, which leads us to write the infinitesimal stress tensor $\boldsymbol{\sigma}_e$ in the element e as:

$$\boldsymbol{\sigma}_e = \mathbf{C}_e\boldsymbol{\epsilon}_e, \tag{6}$$

where $\mathbf{C}_e$ is a 6×6 symmetric matrix depending on two elastic parameters of the material, the Young modulus E and the Poisson ratio ν. Using Eq. (5) we have:

$$\boldsymbol{\sigma}_e = \mathbf{C}_e\mathbf{L}_e\hat{\mathbf{u}}_e. \tag{7}$$

The strain energy in e can be computed as a function of $\boldsymbol{\sigma}_e$, $\boldsymbol{\epsilon}_e$ and the volume V_e of the element, and can then be derived to determine the internal elastic forces $\mathbf{f}_e$ exerted on the four vertices of e. Based on Eqs. (5) and (7), $\mathbf{f}_e$ can be linearly related to the displacements $\hat{\mathbf{u}}_e$ of the vertices:

$$\mathbf{f}_e = \mathbf{K}_e\hat{\mathbf{u}}_e = V_e\mathbf{L}_e^T\boldsymbol{\sigma}_e \tag{8}$$

with $\mathbf{K}_e = V_e\mathbf{L}_e^T\mathbf{C}_e\mathbf{L}_e$ being the 12×12 stiffness matrix of e. Although it is insensitive to translation transformations, the model, by using an infinitesimal approximation of the strain tensor, giving a constant $\mathbf{K}_e$ linearizing the elastic forces, is however inaccurate when modeling large rotations of the elements, the non-linear effects leading to non-zero summations of the forces and causing for instance unexpected growth of volume. A work-around consists in the co-rotational approach proposed in [31–33], used for registration purposes in [26], which is a good compromise between the ability to model large elastic deformations, and computational efficiency. The displacement of an element can be decomposed into a rigid rotation $\mathbf{R}_e$ and a pure deformation. As suggested in [33], we use a polar decomposition on the transformation matrix between the current and initial state of the vertices to extract $\mathbf{R}_e$. Then the stiffness matrix $\mathbf{K}_e$ can be warped with respect to this rotation, so as to accommodate rotation transformations, giving:

$$\mathbf{f}_e = \mathbf{R}_e\mathbf{K}_e\hat{\mathbf{u}}_e^r = \mathbf{R}_e\mathbf{K}_e(\mathbf{R}_e^{-1}\mathbf{x}_e - \mathbf{x}_{e,0}), \tag{9}$$

with being $\hat{\mathbf{u}}_e^r = \mathbf{R}_e^{-1}\mathbf{x}_e - \mathbf{x}_{e,0}$, with $\mathbf{R}_e^{-1}\mathbf{x}_e$ the back rotated deformed coordinates of the vertices of e, to an unrotated frame, the forces $\mathbf{K}_e\hat{\mathbf{u}}_e^r$ being then rotated to the current deformed element through the multiplication by $\mathbf{R}_e$. In this way, the overall forces on the whole mesh can be summed to zero, while computational efficiency is ensured since $\mathbf{K}_e$ can be computed in advance, in contrast to non-linear FEM approaches.

5. Registration with point cloud data

Our deformable registration problem consists in fitting the point cloud data provided by an RGB-D sensor with the known tetrahedral mesh. The basic idea is to derive external forces exerted by the point cloud on the mesh and to integrate these forces, along with the internal forces computed using the physical model presented in Section 4, into a numerical solver solving the resulting mechanical equations.

In this work, these external forces are computed based on geometrical point-to-point correspondences between the point cloud and the mesh, relaxing the assumption of having a textured object [26] or an object with a rough surface, for which 2D or 3D keypoints can be extracted and matched. We assume that the mesh is available (manually designed here) and correctly initialized. Let us however note that off-line automatic reconstruction and meshing techniques could be considered to build the mesh and initialization could be addressed through some learning and recognition of spin images or local 3D features. The resolution of the mesh is chosen as a compromise between modeling accuracy and real-time performances, while being similar to the resolution of the extracted and sampled point cloud. Besides, the Young modulus and Poisson ratio of the considered material are assumed to be known.

5.1. Segmented and sampled point cloud

As introduced in Section 3, we use the acquired RGB image sequence to visually segment the object of interest from its background and occlusions. Since we do not rely on some distinctive visual features, the point cloud provided by the depth sensor is indeed restricted to the considered object, so as to avoid ambiguities, in the matching process with the background or with occluding shapes, and to be able to process correspondences from the input point cloud to the mesh. At frame k, using the segmented image $\mathbf{I}^s$, a segmented depth map $\mathbf{D}^s$ is obtained by aligning and intersecting the original input depth map $\mathbf{D}$ with the segmented area in $\mathbf{I}^s$. Then by back-projecting $\mathbf{D}^s$ in the sensor frame, the desired segmented point cloud $\mathbf{Y} = \{\mathbf{y}_j\}_{j=1}^{n_\mathbf{Y}}$, with $\mathbf{y}_j$ a 3D point in the sensor frame, is determined. For computational concerns, we limit the size of $\mathbf{Y}$ by first sampling $\mathbf{D}^s$ on a regular grid in the image plane.

5.2. Rigid iterative closest point

A first step in our method is to register the observed segmented point cloud $\mathbf{Y}$ in terms of rigid translation and rotation transformations, initially considering the mesh of the object as rigid. Let us first define $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{n_\mathbf{X}}$ the set of vertices of the mesh, initially in their previous computed states, for frame $k-1$. We suggest a classical rigid ICP algorithm [34] between $\mathbf{Y}$ and the vertices of the visible surface $\mathbf{X}_V$ of the mesh transformed with respect to the previous RGB-D data. $\mathbf{X}_V$ is determined by performing a visibility test on the rendered depth map of the projected 3D mesh of the object, for frame $k-1$. Through this procedure, which converges rapidly, fast rigid motions can be tracked and a fair initialization for the non-rigid process can be obtained.

5.3. Deformable iterative closest point

In order to register the segmented point cloud with the mesh in a non-rigid manner, we suggest an ICP-like procedure, given the sets $\mathbf{X}$ and $\mathbf{X}_V$ updated by the rigid transformation estimated as described above.

5.3.1. Nearest neighbor correspondences

By means of Kd-trees searches, nearest neighbor correspondences are determined, both from the segmented point cloud to the visible surface of the mesh and from the visible surface of the mesh to the segmented point cloud. This step provides us with the sets of nearest neighbors $\mathbf{N}_{\mathbf{X}_V} = \{\mathbf{nn}^\mathbf{Y}(\mathbf{x}_i)\}_{i=1}^{n_{\mathbf{X}_V}}$ and $\mathbf{N}_\mathbf{Y} = \{\mathbf{nn}^{\mathbf{X}_V}(\mathbf{y}_j)\}_{j=1}^{n_\mathbf{Y}}$ respectively in $\mathbf{Y}$ for $\mathbf{X}_V$, with the 1-NN function $\mathbf{nn}^\mathbf{Y}$, and in $\mathbf{X}_V$ for $\mathbf{Y}$, with the 1-NN function $\mathbf{nn}^{\mathbf{X}_V}$.

Both sets of correspondences are employed since relying on the sole geometrical proximity may lead to inconsistent matches using

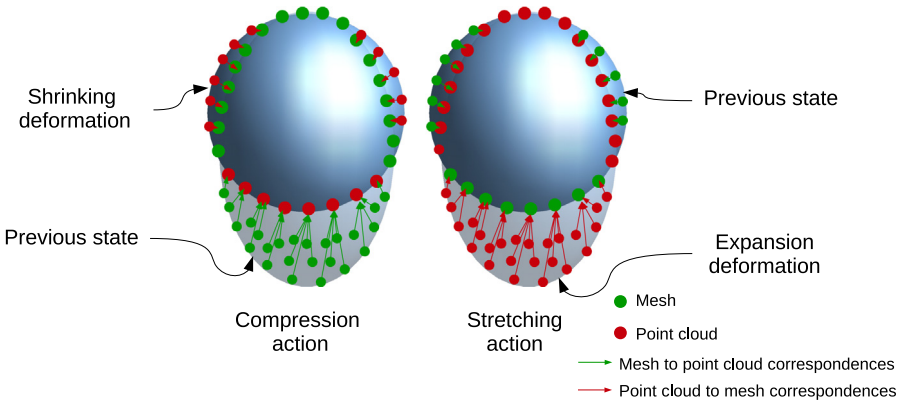


Fig. 4. Point cloud to mesh and mesh to point cloud correspondences. With shrinking deformations, mostly mesh to point cloud correspondences are able to attract the mesh towards the compressed point cloud, and conversely for expansion deformations. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

single point-to-point matches. Indeed, as illustrated in Fig. 4, from the segmented point cloud to the mesh, correspondences enable to track for instance expansion deformations under stretching forces, for which the observed segmented point cloud $\mathbf{Y}$ would spread over the visible surface of the mesh $\mathbf{X}_V$. The extended areas of $\mathbf{Y}$ with respect to $\mathbf{X}_V$ can be matched with the outer areas of $\mathbf{X}_V$ (red arrows on the right side of Fig. 4). These correspondences also enable to deal with occlusions and segmentation errors since the corresponding unobserved areas of the object would not affect the underlying areas of $\mathbf{X}_V$. Conversely, from $\mathbf{X}_V$ to $\mathbf{Y}$, correspondences are instead more suited to track shrinking deformations under compression actions, the outer areas of $\mathbf{X}_V$ being coherently matched with the outer areas of the observed point cloud $\mathbf{Y}$ of the compressed object (green arrows on the left side of Fig. 4). As a drawback, unobserved areas (occlusions, segmentation errors) would affect the underlying areas $\mathbf{X}_V$ which would match with the closest areas of $\mathbf{Y}$.

As described hereafter, a trade-off has to be found between these two sets of correspondences, whether the application deals with stretching or compression actions on the object, and whether occlusions or segmentation errors are to be dealt with.

5.3.2. Computation of external forces

Based on the two sets of mesh-to-point cloud and point cloud-to-mesh correspondences, given by $\mathbf{N}_{\mathbf{X}_V}$ and $\mathbf{N}_{\mathbf{Y}}$, an external elastic force $\mathbf{f}_{ext}$ exerted on each $\mathbf{x}_i$ in $\mathbf{X}_V$, can be computed as follows:

$$\mathbf{f}_{ext}(\mathbf{x}_i) = k_{ext}(\mathbf{x}_i - \mathbf{y}_i^f) \tag{10}$$

with

$$\mathbf{y}_i^f = \begin{cases} \lambda \mathbf{nn}^{\mathbf{Y}}(\mathbf{x}_i) + (1-\lambda)\dfrac{1}{n_{\mathbf{N}^i}} \displaystyle\sum_{\mathbf{y}_j \in \mathbf{N}_{\mathbf{Y}}^i} \mathbf{y}_j \\ \quad \text{if } n_{\mathbf{N}^i} > 0 \\ \lambda \mathbf{nn}^{\mathbf{Y}}(\mathbf{x}_i) + (1-\lambda)\mathbf{x}_i \\ \quad \text{if } n_{\mathbf{N}^i} = 0 \end{cases} \tag{11}$$

where $\mathbf{N}_{\mathbf{Y}}^i = \{\mathbf{y}_j \in \mathbf{Y} | \mathbf{nn}^{\mathbf{X}_V}(\mathbf{y}_j) = \mathbf{x}_i\}$ is the set of points in $\mathbf{Y}$ whose nearest neighbors are $\mathbf{x}_i$. $\mathbf{y}_i^f$ acts as a virtual point attracting $\mathbf{x}_i$. The component of $\mathbf{y}_i^f$ resulting from the multiple point cloud to mesh correspondences is scaled by $\frac{1}{n_{\mathbf{N}^i}}$, with $n_{\mathbf{N}^i}$ the size of $\mathbf{N}_{\mathbf{Y}}^i$, to be homogeneous with the component resulting from the single mesh to point correspondence. The fixed scalar λ tunes the balance between the mesh-to-point cloud and point cloud-to-mesh correspondences, as a trade-off between the stretching or compression actions to be tracked, as suggested above (Section 5.3.1). If $\mathbf{N}_{\mathbf{Y}}^i$ is empty, the missing point cloud-to-mesh correspondences are replaced by a self-contribution for the vertex $\mathbf{x}_i$, compelling it to remain at its current position. k_{ext} is the stiffness of these external spring elastic forces. It can be set accordingly to the Young modulus, using the equivalence between a mass spring model and a linear FEM model, as described in Section 6.1. However modeling this way the external forces, which with vision data only affects the visible vertices, using equivalent springs, can be coarse. In Fig. 5, the vectors $\mathbf{x}_i - \mathbf{y}_i^f$ are displayed, from each $\mathbf{x}_i$. Some outliers in the point cloud may result in aberrant correspondences and thus aberrant forces exerted on some vertices. A simple solution has been to discard points in the point cloud whose point-to-point distances with their nearest neighbors in the mesh are above a certain threshold with respect to the mean value and the standard deviation of the whole set of point-to-point distances. In this case, for the considered vertices $\mathbf{x}_i$, we have $\mathbf{f}_{ext}(\mathbf{x}_i) = 0$. Finally, regarding points $\mathbf{x}_i$ in $\mathbf{X}$ which are not visible, we also set $\mathbf{f}_{ext}(\mathbf{x}_i) = 0$. The whole set of forces is finally concatenated in a vector $\mathbf{f}_{ext}$ of size n_X.

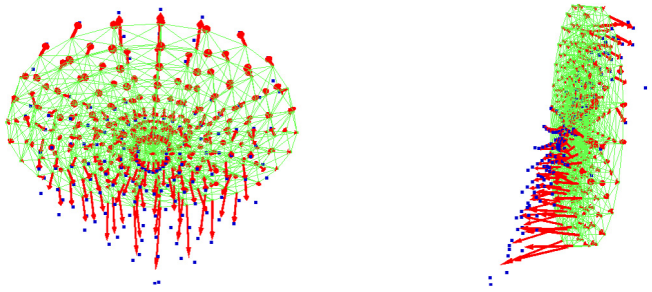


Fig. 5. External forces exerted on the vertices of the mesh, with $k_{ext} = 1\,\mathrm{N\,m^{-1}}$ and $\lambda = 0.7$.

5.3.3. Weighting forces using contours

A limitation of this method lies in tracking large elastic deformations, due to stretching efforts for instance. In this case, since correspondences are established based on 3D geometry, only vertices lying on the occluding contour of the mesh are attracted to the extended area in the point cloud. As a consequence, forces attracting the contours are weak. We propose to emphasize them by weighting the vertices of the visible surface of the mesh, given their distance to the occluding contour of the projected mesh. Based on the depth map $\mathbf{d}^M$ of the projected mesh, we compute the distance map of the occluding contour of the mesh. Then, the weight w_i for the vertex $\mathbf{x}_i$ is computed as follows:

$$w_i \propto e^{-\frac{d_i^M}{\sigma}} \tag{12}$$

where d_i^M is the distance from $\mathbf{x}_i$ to the nearest contour of the projected mesh, σ is a parameter which is empirically set; w_i is normalized so that we get an observation probability. Finally, forces are computed the following way:

$$\mathbf{f}_{ext}(\mathbf{x}_i) = w_i k_{ext}(\mathbf{x}_i - \mathbf{y}_i^f). \tag{13}$$

5.3.4. Numerical solver to compute the deformations

Estimating the deformations of the mesh consists in solving a dynamic system of ordinary differential equations involving the internal and external forces, based on Lagrangian dynamics:

$$\mathbf{M}\ddot{\mathbf{x}} + \mathbf{C}\dot{\mathbf{x}} + \mathbf{f} = \mathbf{f}_{ext} \tag{14}$$

$$\text{with} \quad \mathbf{f} = \mathbf{K}'\mathbf{x} + \mathbf{f}_0 \tag{15}$$

where $\mathbf{x}$ is a $n_\mathbf{X}$ vector containing the positions of the vertices in $\mathbf{X}$, $\mathbf{M}$ and $\mathbf{C}$ is the $n_\mathbf{X}\times n_\mathbf{X}$ mass and damping matrices, $\mathbf{K}'$ the $n_\mathbf{X}\times n_\mathbf{X}$ global stiffness matrix which sums the $n_\mathbf{X}\times n_\mathbf{X}$ element-wise *rotated* stiffness matrices $\mathbf{K}'_e = \mathbf{R}_e\mathbf{K}_e\mathbf{R}_e^{-1}$, written with respect to whole set of vertices, and $\mathbf{f}_0$ the corresponding global offset summing the element-wise ones $\mathbf{R}_e\mathbf{K}_e\mathbf{x}_{e,0}$. An Euler implicit integration scheme is used to solve the system, along with a conjugate gradient method. $\mathbf{X}$ can then be updated using the resulting estimated positions $\mathbf{x}$ of the vertices of the mesh.

5.4. Registration algorithm

Algorithm 1 summarizes the whole process presented above.

Data: Tetrahedral mesh of the object, RGB-D data
Result: Fitting the mesh with the RGB-D data
Initialization of the vertices $\mathbf{X}$ in the mesh;
Initialization of segmentation;
for *each new* $\mathbf{I}$ *and* $\mathbf{D}$ **do**
  Segmentation in $\mathbf{I}$, providing $\mathbf{D}^s$;
  Sampling and back-projection, giving $\mathbf{Y}$;
  Determine $\mathbf{X}_V$;
  Rigid ICP between $\mathbf{X}_V$ and $\mathbf{Y}$;
  Rigidly transform $\mathbf{X}$;
  Non-rigid process:
  3D correspondences from $\mathbf{Y}$ to $\mathbf{X}_V$;
  3D correspondences from $\mathbf{X}_V$ to $\mathbf{Y}$;
  Computation of external forces $\mathbf{f}_{ext}$;
  In SOFA simulator:
  Computation internal forces $\mathbf{f}$;
  Computation of the new positions $\mathbf{x}$:
  by solving $\mathbf{M}\ddot{\mathbf{x}} + \mathbf{C}\dot{\mathbf{x}} + \mathbf{f} = \mathbf{f}_{ext}$ (Euler implicit + conjugate gradient);
end

Algorithm 1: Registration process

6. Experimental results

In order to evaluate the performance of our method and contributions, some experimental results are shown in this section, on some computer-generated data, and on some real data. Various objects, deformations and conditions are tested. For the non-rigid registration phase, we have employed the Simulation Open Framework Architecture (SOFA) simulator [35], which enables to deal with various physical models and to evolve simulations in real-time.

6.1. Results for tracking on synthetic data

Relying on the SOFA framework, we have first generated a sequence involving the deformations of a cylindrical elastic object, modeled by the FEM co-rotational approach. The sequence involves the elastic deformations of a cylindrical elastic object, modeled by the FEM co-rotational approach. It has a Young Modulus of $E = 800$ Pa and Poisson ratio of $\nu = 0.3$. Based on a manually designed cylindrical surface triangular mesh with radius/height dimensions of 0.11 m×0.02 m, the volumetric tetrahedral mesh was generated using the 3D Delaunay triangulation, through the CGAL library.[2] A disordered placement of the nodes has been here adopted on this simple cylindrical geometric topology, not to lose generality with respect to an arbitrary object topology, for which a 3D Delaunay triangulation would be preferred as a mesh generation routine. The resulting mesh is made of 1369 elements and 497 vertices, and is featured in Fig. 6. In order to simulate deformations and generate the considered sequence, an elastic stretching force is applied in the $-Z$ direction (see Fig. 7), on one point on the border of the object (point 1), few other points being fixed on the opposite border (points 4, 5, 6), and two compression forces being applied along Y and $-Y$ (points 2, 3). The applied forces result in a fast elongation deformation of the object, with a maximum elongation above 50%, and in an important bending deformation along Y, as it can be seen in Fig. 9. For the tracking phase, segmentation aspects are not considered in these experiments. We only process the visible vertices of the rendered object in the sequence, and as a ground truth, the positions of the whole set of points are stored for evaluation. Results can be visually observed in Fig. 7, featuring the original target (red) and the tracking 3D mesh (blue), and in Fig. 8 the 3D errors between the vertices of the registered mesh and the corresponding points in the point cloud are plotted (see also the attached video).

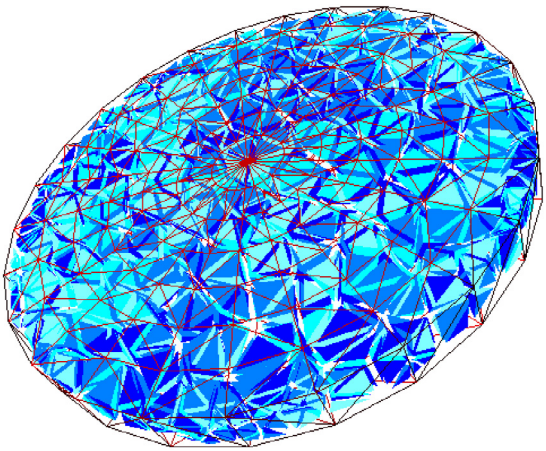


Fig. 6. Processed mesh.

The following models and methods have been compared:

- Volumetric mass–spring model
- Standard FEM model
- Co-rotational FEM model
- Co-rotational FEM model along with contour weighting (CW)

On these results, we set the Young modulus and Poisson coefficient for the FEM approaches equal to the values assigned for the simulated object ($E = 800$ Pa, $\nu = 0.3$), and we set $\lambda = 0.5$, balancing point cloud to mesh and mesh to point cloud forces, and we set $k_{ext} = 10\ \mathrm{N\,m^{-1}}$ for the stiffness of the external forces, which appeared to provide satisfying results. For the mass spring model, we consider a volumetric mass spring model on

[2] http://www.cgal.org.

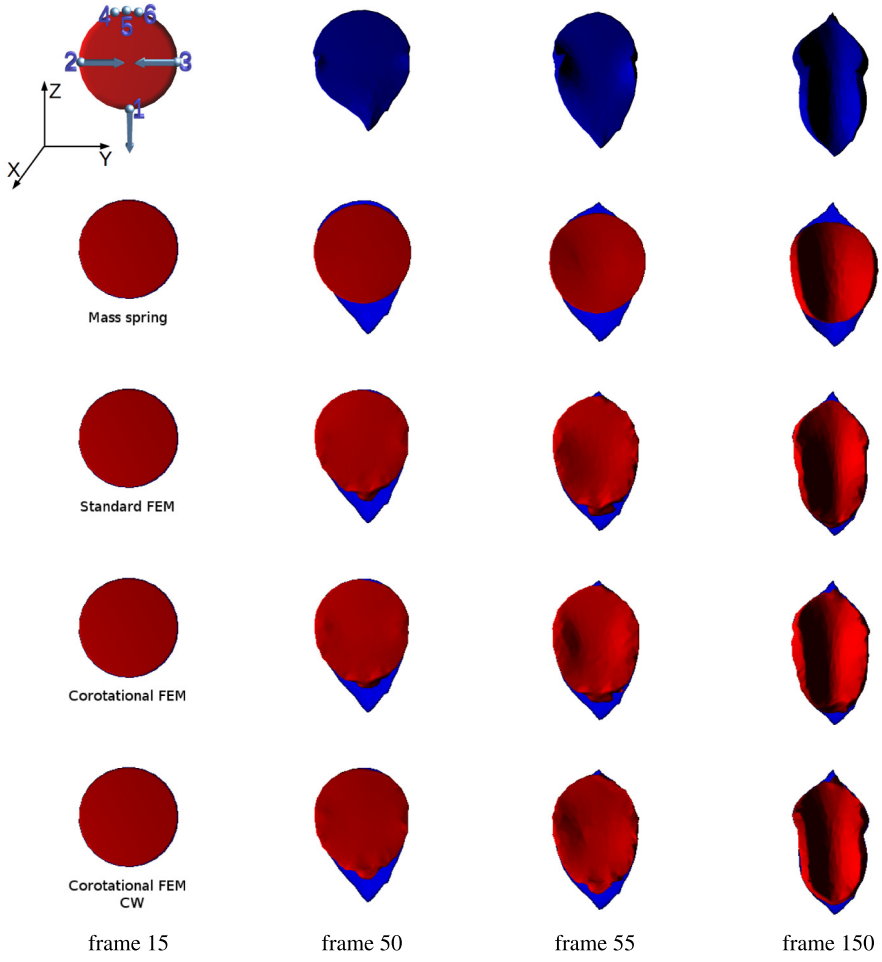


Fig. 7. Results of the deformable tracking process. On the first row is featured the ground truth, on the second the tracking with the equivalent mass spring system, the third linear FEM, the fourth with the co-rotational FEM and the fifth adding contour weighting. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

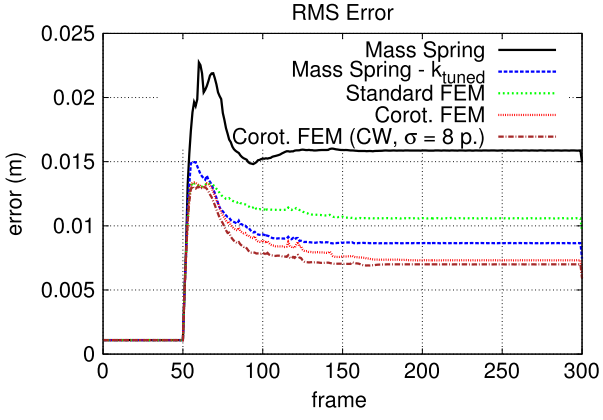


Fig. 8. Errors of the deformable tracking process, with the different tested approaches, the mass spring system with the equivalent stiffness, the mass spring system with tuned stiffness (k_{tuned}), the standard FEM, the co-rotational FEM and the co-rotational FEM with contour weighting (with $\sigma = 8$ pixels).

the tetrahedrons of the generated mesh, by attaching springs to connected vertices.

Determining the spring parameters of a mass–spring model equivalent to the FEM model in terms of elasticity behavior is not trivial, especially when dealing with unordered and irregular tetrahedrons, as pointed out by [36,37]. We use the analytical expression proposed by [37] to derive the stiffness $k_m(i, j)$ of an equivalent spring between connected vertices i and j of the tetrahedral mesh:

$$k_m(i,j) = \sum_e \frac{2\sqrt{2}}{25} \left(V_e \frac{12}{\sqrt{2}} \right)^{\frac{1}{3}} E \tag{16}$$

with the sum over the elements adjacent to the edge (i, j). The resulting stiffnesses are obviously too large to register the investigated deformations. The performance of a mass spring model with spring stiffnesses scaled by 0.05 (defined as k_{tuned}), which appeared to fit best on this sequence, is thus also shown.

We observe that the equivalent mass spring model is clearly outperformed by the FEM approaches and by the mass spring model with a manually tuned stiffness. This latter performs worse than the FEM models on the first frames when triggering the deformations (frame 50), due to the stretching deformations which are severe over this phase. However the standard linear FEM tends then to fail tracking the large bending deformations which occur until steady state, enhancing the sensitivity of this approach regarding rotation transformations within elements, as it can also be observed in Fig. 9. On the whole sequence, the benefit of using a co-rotated approach and the advantage of using the contour weighting function (with $\sigma = 8$ pixels) to track large bending and stretching deformations can thus be particularly stressed out. In order to evaluate the sensitivity with respect to tuning parameters, in Fig. 10 are represented the errors for different external stiffnesses

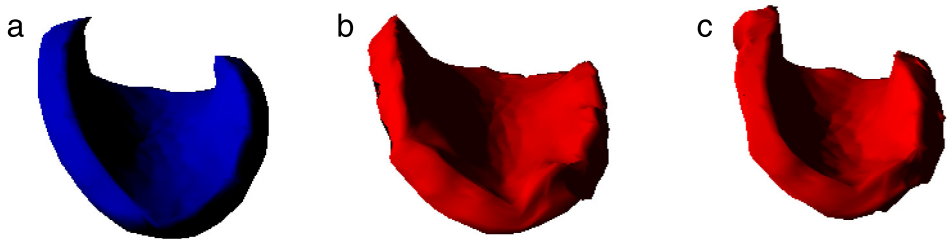


Fig. 9. Deformations on the generated sequence (a), on the tracked mesh with the standard FEM (b), and with the co-rotational FEM (c).

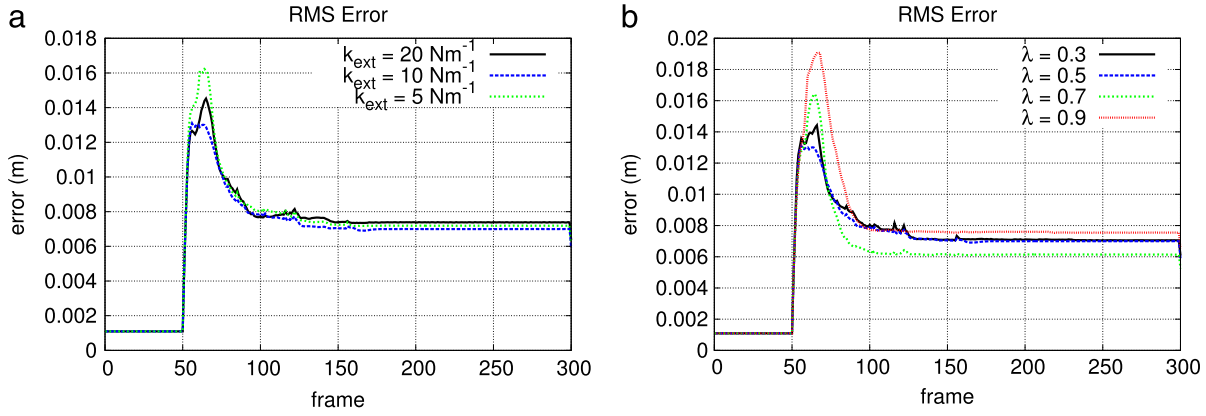


Fig. 10. Errors with respect to k_{ext} (a) and λ (b), tuning the balance between mesh to point cloud and point cloud to mesh forces.

k_{ext}, and for different values of λ. For the external stiffness, the value of k_{ext} acts as a gain and tunes the balance between tracking large stretching deformations, especially on the silhouette borders, and regularization effects, its influence being especially observable on the large initial deformations. With λ a good compromise is found with $\lambda = 0.7$, showing the relevance of point cloud to mesh correspondences in this case for which stretching actions prevail.

6.2. Results on real data

In order to carry out experiments on real data, the point cloud of the investigated scene is acquired from a calibrated RGB-D camera Asus Xtion, 320 × 240 RGB and depth images being processed. A standard laptop with an NVIDIA GeForce 720M graphic card has been used, along with a 2.4 GHz Intel Core i7 CPU. Here the segmentation process is involved in the loop, and since fast real-time performance is required, it relies on a CUDA implementation. The results presented here deal with a textureless and smooth elastic object made with silicon.

For the first mentioned "pizza-like" object, the idea has been to test motions and deformations similar to the ones involved in the pizza making process, in the scope of the RoDyMan project. The involved mesh is generated in a similar manner to Section 6.1, with radius/height dimensions of 0.12 m×0.01 m, and consisting in 574 vertices and 1675 elements. Let us note that the resolution of the point cloud is chosen, through sampling, to approximately coincide with the resolution of the visible mesh.

On the first featured sequence the object undergoes large rigid motions and various isometric and elastic deformations. The Young modulus and Poisson ratio of the material are unknown and good compromises have been found with $E = 300$ Pa and $\nu = 0.3$, by setting $k_{ext} = 1\ \mathrm{N\ m^{-1}}$.

Qualitative results are presented in Fig. 11, comparing our method using the co-rotational FEM approach with other models. On the first row are shown input RGB images, the second row features the corresponding segmented frame, the third row shows the 3D mesh tracking the object with the mass spring model, the fourth with the standard FEM model, the fifth with the proposed approach (with $\sigma = 30$ pixels).

We can visually notice the ability of the proposed method to correctly segment the visible part of the object, to track rigid motions, in contrast to the mass spring to accurately register stretching deformations, and in contrast to the standard linear FEM to be robust to folding/bending deformations thanks to the co-rotational approach. Robustness to occlusions due to the hands manipulating the object (third column in Fig. 11) or segmentation errors can also be observed. The second sequence has been worked out to stress out the robustness of the proposed method under large elastic deformations, due to stretching actions. In Fig. 12 are shown the results for the proposed method and in Fig. 13 are also compared the different models during stretching. Since the object is essentially stretched along the image plane, involving few bending deformations along the optical axis, we propose as an evaluation criteria to compute the mean 3D distance between the contour of the projected mesh in the image, and the contour the segmented silhouette of the object. This is achieved by searching from the 3D vertices corresponding to the extracted 2D contour of the mesh for the nearest 3D points corresponding to the 2D contour of the segmented silhouette. This way, the benefit of using the co-rotational approach along with the contour weighting function can also be stressed out (Fig. 14).

Sensitivity to material and tuning parameters.

Different stiffnesses of the material have been tested, $E = 300$ Pa appearing, as stated before, to be a good compromise between the ability to register large deformations, while providing sufficient regularization and robustness to noise, which can result in spurious registrations, and self-collisions, especially around the borders of the mesh, as seen in Fig. 16(a) and (b), and in Fig. 15(a) with the contour fitting errors. Different resolutions of the mesh

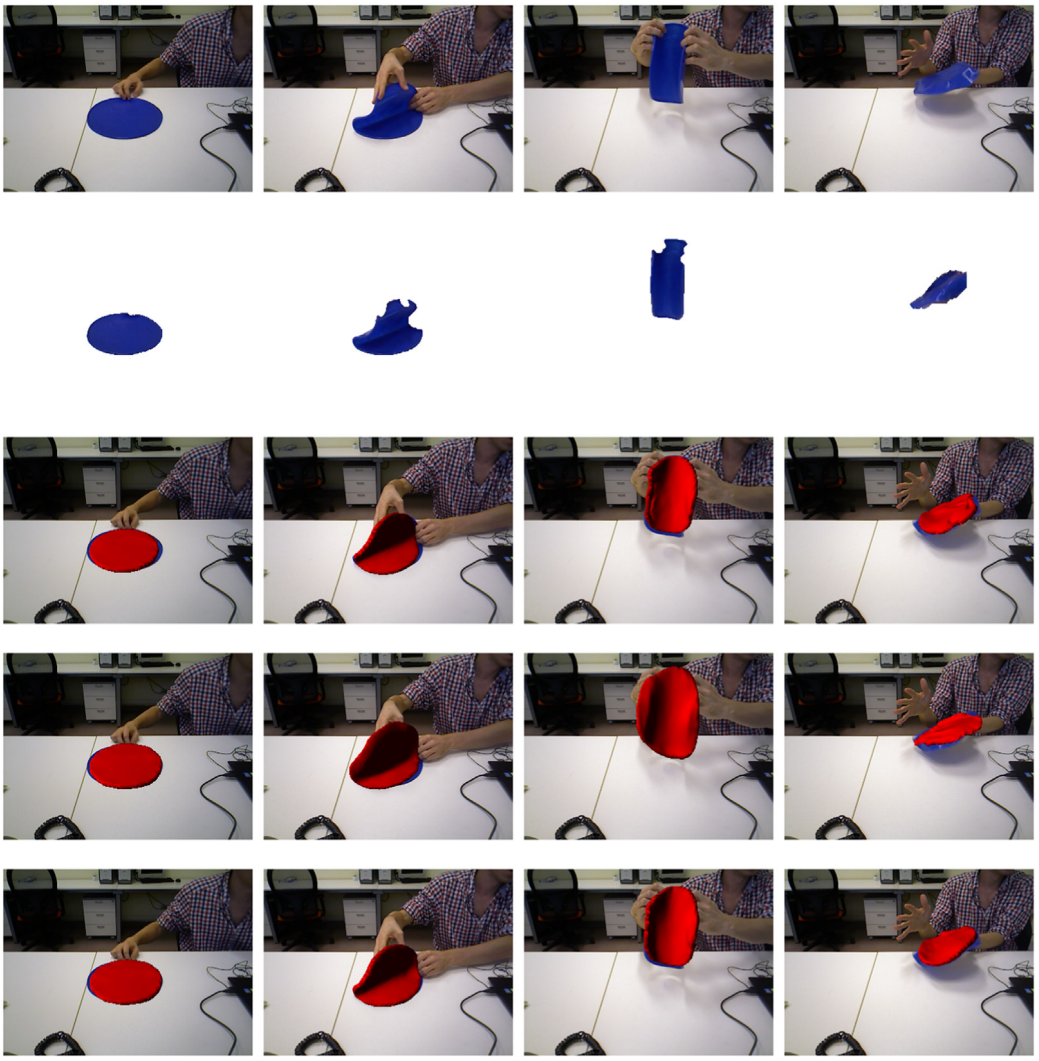
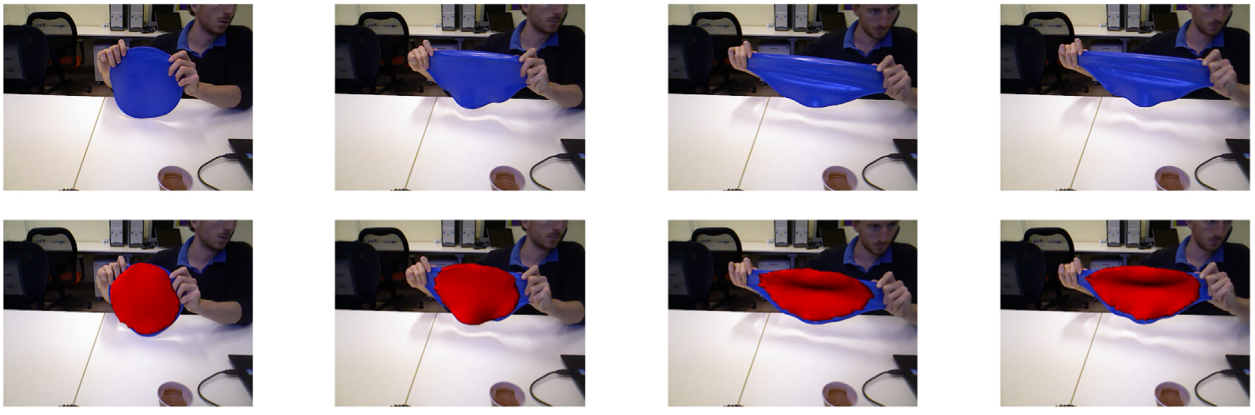


Fig. 11. Results of the tracking process for the pizza-like object, with the input images (first row), the segmented frames (second row), and the registered mesh reprojected in the input image, for the mass spring model (third row), the standard FEM model (fourth row) and finally with the co-rotational model, and with the contour weighting technique (CW).



(a) Frame 1. (b) Frame 20. (c) Frame 50. (d) Frame 150.

Fig. 12. Results of the tracking process for the silicon pizza, with the input images (first row), and the tracking results on the second row.

have been tried as well, adjusting through the manually set sample step the resolution of the point cloud accordingly. It can be observed in Fig. 15(b) that higher resolutions manage to fit better with the contour, and also, as observed in Fig. 16(c) and (d) to

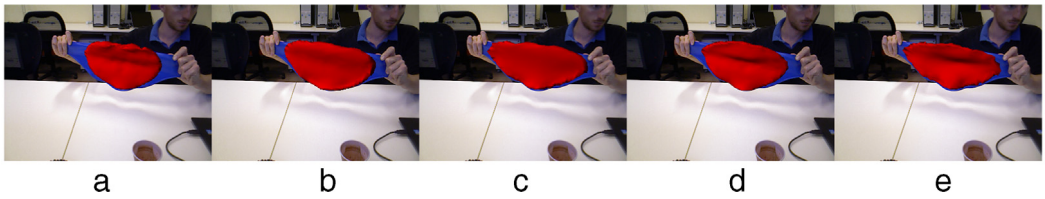


Fig. 13. Results of the tracking process for the silicon pizza during stretching for the mass spring model (a), the standard FEM model, with (b) and without (c) contour weighting, with the co-rotational model, with (d) and without the contour weighting technique (e), on frame 150.

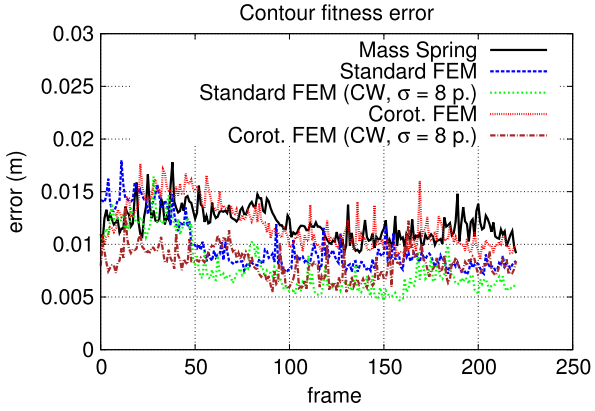


Fig. 14. Contour fitting error for the different tested approaches (mass spring, standard and co-rotational FEM).

capture higher frequency details, up to certain point for which the benefit becomes minor.

6.2.1. Computational costs

Regarding computational aspects, in Table 1 are shown the computation times of the various phases of the algorithm, for the different methods compared in this paper. *Visibility* corresponds to the process of determining the visible vertices of the rendered mesh, and in the case of using the contour weighting mode, extracting the vertices the lying on the contour. *Ext. forces* is the step involving the determination of the closest points between the mesh and the point cloud, and the computation the subsequent external forces exerted on the mesh. *Resolution* consists in the resolution of the Lagrangian mechanical equations, to compute the deformations. The presented figures are the averages of the execution times per frame (in milliseconds) for the sequence presented in Fig. 11. As noticed, the suggested method (co-rotational model with the contour weighting mode) runs on the sequence at around

Table 1
Execution times, in milliseconds, for the different phases of the approach, and the various models and methods employed in this paper.

	Mass spring	Stand. FEM	Corot.	Corot. - CW
Segmentation	10.7	10.5	10.7	10.7
Rigid ICP	3.0	2.5	2.7	2.6
Visibility	8.1	8.2	7.6	7.4
Ext. forces	3.4	3.5	3.5	4.0
Resolution	2.8	3.3	4.0	4.1
Total	28.0	27.9	28.6	28.8

35 fps. We can also observe that, the computational costs for the resolution phase being relatively small within the whole process, overall execution times are relatively independent of the selected model.

6.3. Preliminary robotic manipulation experiments

In the scope of the RoDyMan project, we present in this section some preliminary experiments integrating the tracking method described and validated above into a robotic manipulation task involving a deformable object, that is the silicon pizza dough. Based on the proposed registration method, the scenario consists in a kinematic control in the operational space in order to follow a trajectory aiming at manipulating the pizza from one hand to the other on the RoDyMan robotic platform.

6.3.1. The RoDyMan humanoid robotic platform

As featured in Fig. 17(a), the robotic platform of the RoDyMan project is a humanoid robot made of an omnidirectional mobile platform, a torso with 2 DOF, two anthropomorphic arms with 7 Degrees of Freedom (DOF), two anthropomorphic hands and a sensorized-head (Fig. 17(b)) moved by a neck with two DOF about pan and tilt axis. Its current state can be seen in Fig. 17(c). The mobile platform consists in 4 omnidirectional wheels (meccanum-wheel) moved by Maxon motors. The chassis is able to extend itself, increasing the stability of the platform, in order to take advantage

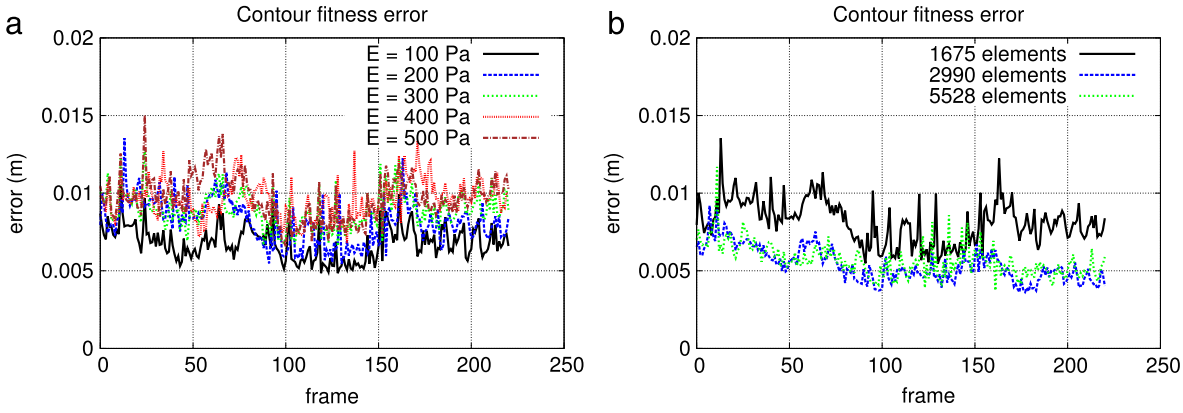


Fig. 15. Contour fitting error, for different stiffnesses (a) and resolutions (b).

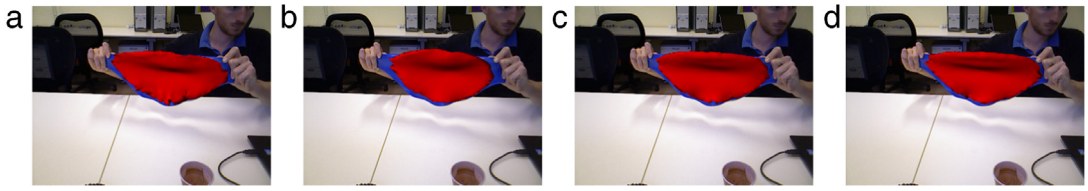


Fig. 16. Results of the tracking process with the proposed approach, for $E = 100$ Pa (a) and $E = 500$ Pa (b) with the standard resolution, and for resolutions of 2990 (c) and 5528 (d) elements, with $E = 100$ Pa.

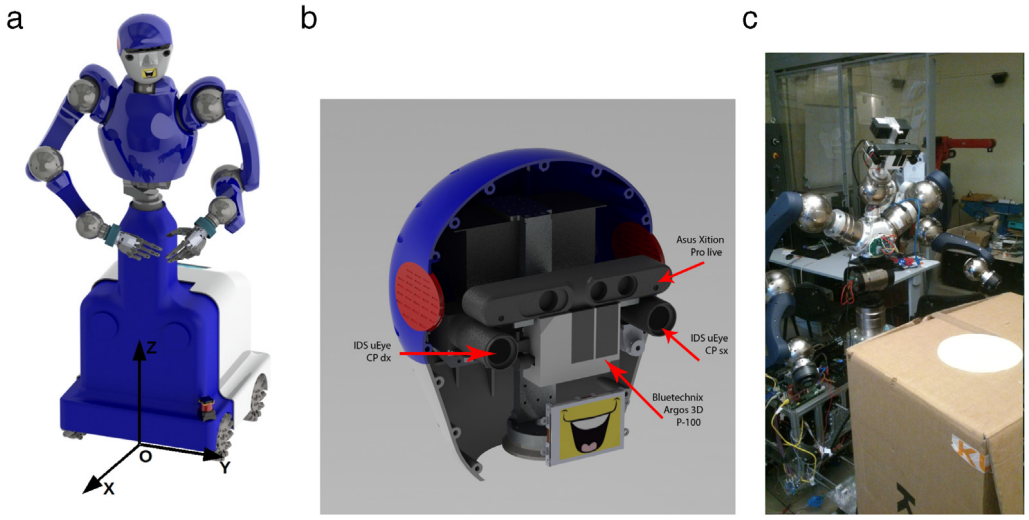


Fig. 17. Artistic view of the RoDyMan robotic platform (a) and of its head equipped with the vision sensors (b). On (c) can be seen the platform in its current state.

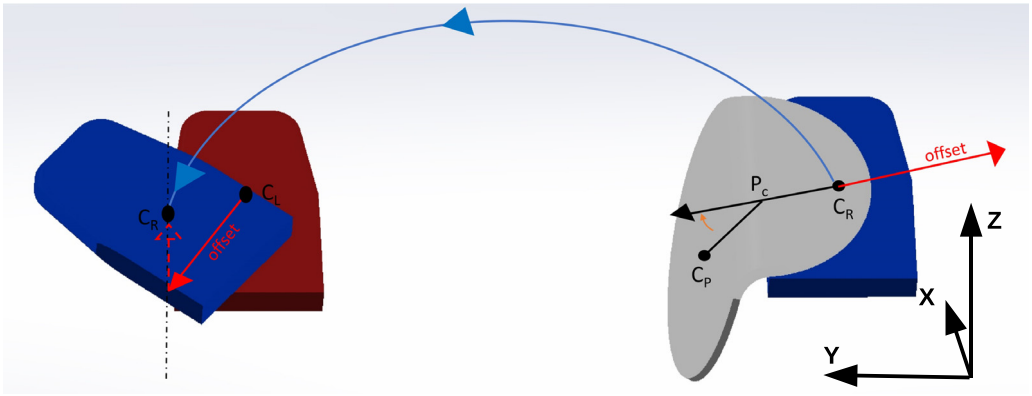


Fig. 18. Trajectory planning for the second circular path.

of the large workspace and the dynamics of the upper body. The base contains all the electronics for the power supply and the control facilities of the platform. The torso consists in two SCHUNK PRL-120 revolute joints and the arms are formed by a SCHUNK LWA 6DOF arm along with a further PRL-100 revolute joint in the shoulder, reaching a total of seven DOF for each arm. In this configuration, the kinematics of the robot arms are described by two kinematic chains with 12 DOF (3 DOF from the mobile base, 2 from the torso and 7 from the arm). The two anthropomorphic hands would provide enhanced dexterous skills. SCHUNK has provided us with 2 Right/Left 5-finger gripping hands SVH which are to be mounted on the last SCHUNK ERB115 revolute joints of the Powerball robotic arms. These hands consist in 21 joints, 9 of which are actuated and the remaining ones are passive. Between the end-effector and the hands is installed a force–torque sensor able to measure the generalized forces applied to the end-effector of the robot. The neck is actuated by a SCHUNK ERB-115 Powerball which allows to move on both pan and tilt axis a sensorized-head (see Fig. 17(b)). The head is indeed equipped with the vision sensors of the robots, which consist in an Asus Xtion structured light RGB-D sensor, a BlueTechnix Argos P-100 time-of-flight L-D sensor and two IDS uEye CP cameras mounted in a stereo configuration.

6.4. Trajectory planning and control

For each hand the idea is to follow a path which can be decomposed into three paths: the first one is a circular path with constant orientation to make the hand supporting the pizza dough approaching the other one. The second path is also circular, while this time performing a rotation of 180°around X (in the base frame, as seen on Fig. 17(a)) and a rotation of 50°around Z until the pizza

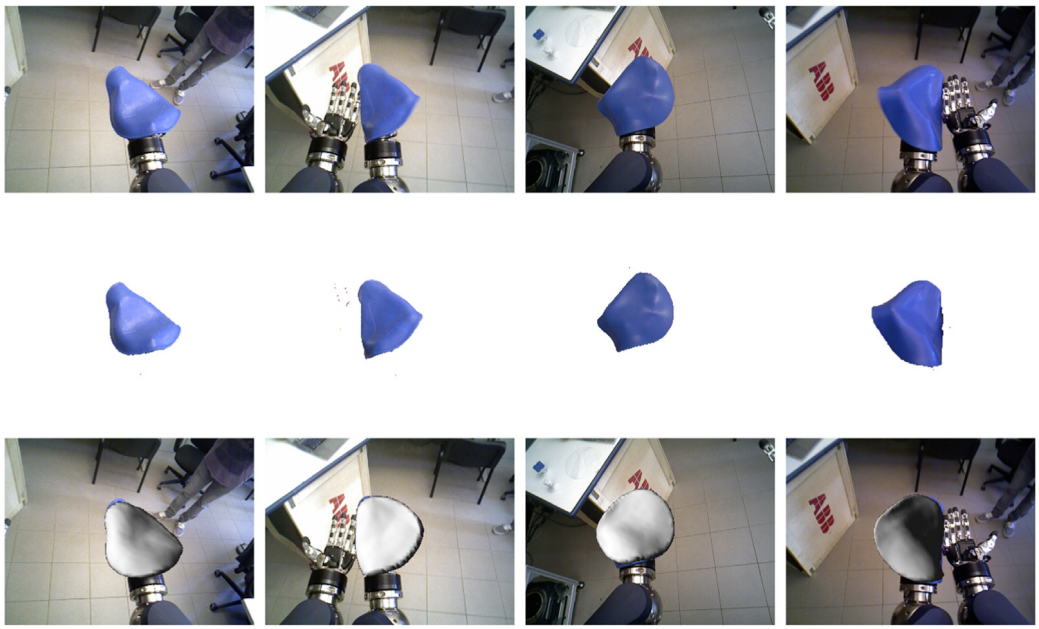


Fig. 19. Results of the tracking process for the silicon pizza, with the input images (first and fourth rows), the segmented frames (second row), and the registered mesh reprojected in the input image with the proposed method.

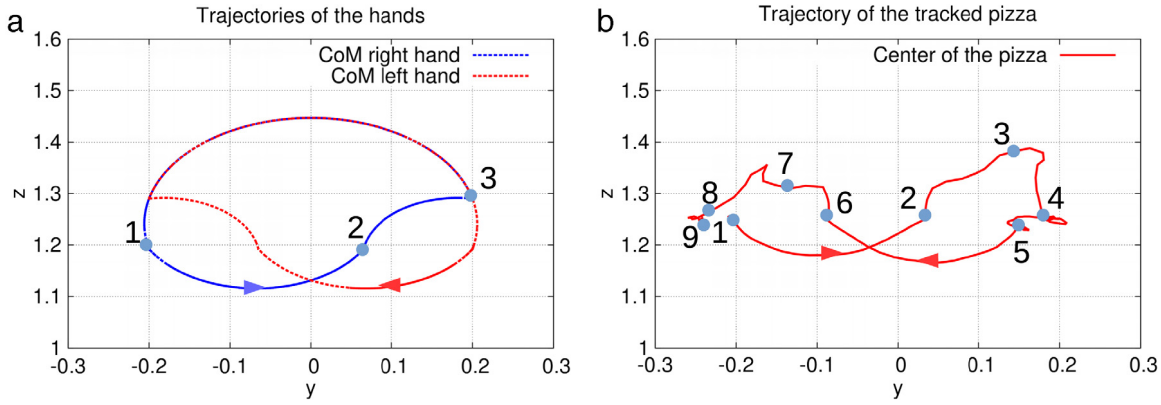


Fig. 20. Trajectories in $(Y–Z)$ plane of the centers of mass of both hands (a) and trajectory of the vertex corresponding to the center of mass of the object at rest (b).

gets overturned, the third circular path is intended to move the hand back to its initial position. In a feed-forward manner, the final position of the second path is computed based on data provided by the registration algorithm initially, which enables to retrieve the position of the vertex in the mesh corresponding to the center of mass of the object in its rest shape, C_P in Fig. 18. It is given by the center of mass of the hand (C_R) on which the pizza shall be laid on, plus a translation along Z, and an offset, as seen on Fig. 18. This offset is approximated as the length of the path between (C_R) and (C_P), and passing by P_C, which is the intersection between the edge of the hand and the plane spanned by $(\overrightarrow{C_R C_P}, \overrightarrow{Z})$. The closed-loop inverse kinematic (CLIK) based controller also controls the joints of the torso and the head so that the optical axis of the RGB-D sensor intersects throughout the manipulation task.

6.4.1. Experimental set-up

The images and point clouds of the investigated scene are in these experiments provided by the Asus Xtion RGB-D sensor mounted on the head, 320 × 240 RGB and depth images being processed. The computer hardware and software is the same as the one used for the experiments shown in Section 6.2. A view of the overall set-up can be seen in Fig. 17(c).

6.4.2. Results

Two passes, way and back, are performed, based on the described methodology, to bring the silicon pizza dough from the right hand to the left and conversely. The resulting trajectories in the $Y–Z$ plane of the centers of mass (CoM) of both hands can be observed in Fig. 20(a), starting on step 1 (dot 1), handling then 3 phases, first the circular path to bring the right hand close to the left one, until step 2, then the corrected circular path to flip the pizza (dot 3), and finally a third circular path to bring the hand back to its initial position. The reverse trajectory is then achieved by the left hand. For the registration phase of the pizza dough during the manipulation phase, results are presented in Fig. 19, with on the first row the input RGB images, on the second row the corresponding segmented frame, the third row shows the 3D mesh tracking the object. Fig. 20(b) shows the trajectory of C_P. Let us note that the position of C_P is smoothed, using a constant velocity Kalman filter.

When the pizza gets flipped, the occlusions by the hands prevailing too much, the deformation registration process is stopped to avoid spurious deformations on the mesh that may not be recovered afterwards when the pizza gets reasonably visible again.

Thus only the rigid ICP process is maintained during these phases, from step 3 to step 4 and from step 7 to step 8.

7. Conclusion

Real-time perception is a crucial issue for robotic dynamic manipulation tasks, especially when considering deformable objects such as food, which has to be manipulated in a very challenging way, as in the case of a pizza being stretched and tossed by a humanoid pizza chef robot. The recent development of physics-based modeling methods for deformable elastic objects for registration purposes and the availability of real-time implementations have lead us to choosing such an approach to track a textureless and smooth object subjected to various large deformations, with an RGB-D sensor. The use of a pertinent linear FEM model, of an efficient segmentation method, and of classical point cloud registration techniques have made our system a promising real-time tracking method able to handle various deformations and motions. A first step towards robotic manipulation of such objects has been achieved by performing a simple manipulation task on the RoDyMan platform. Regarding future works, efforts could focus on different aspects of the registration process, such as segmentation, which could benefit from the depth data, the point cloud matching procedure, and the physical model, by extending it to other deformations such as plastic ones. For manipulation, more complex trajectory planning, based on accurate predictive physical simulation, would be considered.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at http://dx.doi.org/10.1016/j.robot.2016.08.023.

References

[1] A. Nealen, M. Müller, R. Keiser, E. Boxerman, M. Carlson, Physically based deformable models in computer graphics, in: Computer Graphics Forum, vol. 25, Wiley Online Library, 2006, pp. 809–836.

[2] J. Bender, M. Müller, M.A. Otaduy, M. Teschner, M. Macklin, A survey on position-based simulation methods in computer graphics, in: Computer Graphics Forum, vol. 33, Wiley Online Library, 2014, pp. 228–251.

[3] C. Smith, Y. Karayiannidis, L. Nalpantidis, X. Gratal, P. Qi, D.V. Dimarogonas, D. Kragic, Dual arm manipulation: A survey, Robot. Auton. Syst. 60 (10) (2012) 1340–1353.

[4] M. Tenorth, M. Beetz, A unified representation for reasoning about robot actions, processes, and their effects on objects, in: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2012, pp. 1351–1358.

[5] M. Bollini, S. Tellex, T. Thompson, N. Roy, D. Rus, Interpreting and executing recipes with a cooking robot, in: Experimental Robotics, Springer, 2013, pp. 481–495.

[6] P. Kormushev, S. Calinon, D.G. Caldwell, Robot motor skill coordination with em-based reinforcement learning, in: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2010, pp. 3232–3237.

[7] M.C. Gemici, A. Saxena, Learning haptic representation for manipulating deformable food objects, in: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2014, pp. 638–645.

[8] M. Higashimori, K. Utsumi, Y. Omoto, M. Kaneko, Dynamic manipulation inspired by the handling of a pizza peel, IEEE Trans. Robot. 25 (4) (2009) 829–838.

[9] A. Bartoli, V. Gay-Bellile, U. Castellani, J. Peyras, S. Olsen, P. Sayd, Coarse-to-fine low-rank structure-from-motion, in: 2008 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2008, pp. 1–8.

[10] A. Weiss, D. Hirshberg, M.J. Black, Home 3d body scans from noisy image and range data, in: 2011 IEEE International Conference on Computer Vision, IEEE, 2011, pp. 1951–1958.

[11] M. Kass, A. Witkin, D. Terzopoulos, Snakes: Active contour models, Int. J. Comput. Vis. 1 (4) (1988) 321–331.

[12] A. Bartoli, A. Zisserman, et al., Direct estimation of non-rigid registrations, in: British Machine Vision Conference, 2004, pp. 899–908.

[13] J. Pilet, V. Lepetit, P. Fua, Fast non-rigid surface detection, registration and realistic augmentation, Int. J. Comput. Vis. 76 (2) (2007) 109–122.

[14] D. Terzopoulos, A. Witkin, M. Kass, Constraints on deformable models: Recovering 3d shape and nonrigid motion, Artificial Intelligence 36 (1) (1988) 91–123.

[15] M. Salzmann, J. Pilet, S. Ilic, P. Fua, Surface deformation models for nonrigid 3d shape recovery, IEEE Trans. Pattern Anal. Mach. Intell. 29 (8) (2007).

[16] A. Jordt, R. Koch, Direct model-based tracking of 3d object deformations in depth and color video, Int. J. Comput. Vis. (2013) 1–17.

[17] M. Zollhöfer, M. Nießner, S. Izadi, C. Rehmann, C. Zach, M. Fisher, C. Wu., A. Fitzgibbon, C. Loop, C. Theobalt, et al., Real-time non-rigid reconstruction using an rgb-d camera, in: ACM Trans. Graph., TOG, 2014.

[18] B. Allain, J.-S. Franco, E. Boyer, An efficient volumetric framework for shape tracking, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition, 2015.

[19] R.A. Newcombe, D. Fox, S.M. Seitz, Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 343–352.

[20] V. Lippiello, F. Ruggiero, B. Siciliano, Floating visual grasp of unknown objects using an elastic reconstruction surface, in: Robotics Research, Springer, 2011, pp. 329–344.

[21] C. Elbrechter, R. Haschke, H. Ritter, Bi-manual robotic paper manipulation based on real-time marker tracking and physical modelling, in: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2011, pp. 1427–1432.

[22] J. Schulman, A. Lee, J. Ho, P. Abbeel, Tracking deformable objects with point clouds, in: Robotics and Automation, ICRA, 2013 IEEE International Conference on, IEEE, 2013, pp. 1130–1137.

[23] L.D. Cohen, I. Cohen, Deformable models for 3-d medical images using finite elements and balloons, in: 1992 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 1992, pp. 592–598.

[24] T. McInerney, D. Terzopoulos, A finite element model for 3d shape reconstruction and nonrigid motion tracking, in: 1993 IEEE International Conference on Computer Vision, IEEE, 1993, pp. 518–523.

[25] A. Malti, R. Hartley, A. Bartoli, J.-H. Kim, Monocular template-based 3d reconstruction of extensible surfaces with local linear elasticity, in: Computer Vision and Pattern Recognition, CVPR, 2013 IEEE Conference on, IEEE, 2013, pp. 1522–1529.

[26] N. Haouchine, J. Dequidt, I. Peterlik, E. Kerrien, M.-O. Berger, S. Cotin, Image-guided simulation of heterogeneous tissue deformation for augmented reality during hepatic surgery, in: Mixed and Augmented Reality, ISMAR, 2013 IEEE International Symposium on, IEEE, 2013, pp. 199–208.

[27] N. Haouchine, J. Dequidt, M.-O. Berger, S. Cotin, Monocular 3d reconstruction and augmentation of elastic surfaces with self-occlusion handling, IEEE Trans. Vis. Comput. Graphics 21 (12) (2015) 1363–1376.

[28] C. Rother, V. Kolmogorov, A. Blake, Grabcut: Interactive foreground extraction using iterated graph cuts, in: ACM Transactions on Graphics, TOG, vol. 23, 2004, pp. 309–314.

[29] Y. Boykov, O. Veksler, R. Zabih, Fast approximate energy minimization via graph cuts, in: IEEE Trans. on Pattern Analysis and Machine Intelligence, 2001, pp. 1222–1239.

[30] R.D. Cook, Finite Element Modeling for Stress Analysis, Wiley, 1994.

[31] O. Etzmuß, M. Keckeisen, W. Straßer, A fast finite element solution for cloth modelling, in: In: 11th Pacific Conference on Computer Graphics and Applications, IEEE, 2003, pp. 244–251.

[32] M. Müller, M. Gross, Interactive virtual materials, in: Proceedings of Graphics Interface 2004, Canadian Human-Computer Communications Society, 2004, pp. 239–246.

[33] M. Nesme, Y. Payan, F. Faure, Efficient, physically plausible finite elements, in: Eurographics, 2005.

[34] Y. Chen, G. Medioni, Object modelling by registration of multiple range images, Image Vis. Comput. 10 (3) (1992) 145–155.

[35] F. Faure, C. Duriez, H. Delingette, J. Allard, B. Gilles, S. Marchesseau, H. Talbot, H. Courtecuisse, G. Bousquet, I. Peterlik, et al., Sofa: A multi-model framework for interactive physical simulation, in: Soft Tissue Biomechanical Modeling for Computer Assisted Surgery, Springer, 2012, pp. 283–321.

[36] A.V. Gelder, Approximate simulation of elastic membranes by triangulated spring meshes, J. Graph. Tools 3 (2) (1998) 21–41.

[37] B.A. Lloyd, G. Székely, M. Harders, Identification of spring parameters for deformable object simulation, IEEE Trans. Vis. Comput. Graphics 13 (5) (2007) 1081–1094.

Antoine Petit was born in Dieppe, France, on May 12, 1986. He graduated from Supélec in 2010 as an Engineer, with majors in Control and Signal Processing, and from Centrale Paris for a Research Master in Aerospace Engineering. He received his Ph.D. from University de Rennes 1 in Computer Vision in December 2013, focusing on vision based tracking, with an application regarding Orbital Rendez-vous between spacecrafts, in partnership with Airbus. He is now working as a postdoc researcher in the PRISMA group in the Department of Electrical Engineering and Information Technology of the University of Naples Federico II. His research interests involve perception and computer vision for robotics.

Vincenzo Lippiello was born in Naples, Italy, on June 19, 1975. He received the Laurea degree in Electronic Engineering and the Research Doctorate degree in Information Engineering from the University of Naples Federico II in 2000 and 2004, respectively. From 2000 he was with the Department of Electrical Engineering and Information Technology of the University of Naples Federico II, where he is currently Researcher in Robotics and Automation. His research interests include visual tracking and servoing of robot manipulators, 3D object grasping, grasp synthesis, visual/force control, and aerial manipulation. He has published 60 journal and conference papers.

Giuseppe Andrea Fontanelli was born in Avellino, Italy, on June 09, 1991. He attended high school at "Liceo Scientifico Ettore Majorana" in Caltagirone specializing in scientific studies. He obtained his Laurea Triennale (equivalent to a Bachelor's Degree) in Automation Engineering on October 2012. His thesis concerned a development of an experimental setup for identification of flexinol based actuator. He obtained the Laurea Specialistica degree (equivalent to a Master's Degree) in Automation Engineering with highest honors in February 2015, with a dissertation entitled: "Development of a omnidirectional mobile platform with a 8 DOF and torque sensor integrated", supervised by Dr. V. Lippiello.

He is currently a collaborator in PRISMA lab (Naples), under the supervision of Dr. V. Lippiello.

Bruno Siciliano was born in Naples, Italy, on October 27, 1959. He received the Laurea degree and the Research Doctorate degree in Electronic Engineering from the University of Naples in 1982 and 1987, respectively. From 1983 to 2000 he was with the Department of Electrical Engineering and Information Technology of the University of Naples Federico II. From 2000 to 2003 he was Professor of Automatic Control in the Department of Information and Electrical Engineering of the University of Salerno. He is currently Professor of Control and Robotics, and Director of the PRISMA Lab at University of Naples Federico II. From September 1985 to June 1986 he was a visiting scholar at the School of Mechanical Engineering of the Georgia Institute of Technology. His research interests include model identification and adaptive control, impedance and force control, visual tracking and servoing, redundant and cooperative manipulators, lightweight flexible arms, aerial robots, human-centered and service robotics. He has published more than 300 journal and conference papers, he is co-author of the books: Robotics—Modelling, Planning and Control (Springer 2009), Modelling and Control of Robot Manipulators (McGraw-Hill 1996, 2nd Edition Springer 2000), Robot Force Control (Kluwer 1999), Theory of Robot Control (Springer 1996), and he is co-editor of the books: Control Problems in Robotics and Automation (Springer 1998), RAMSETE — Articulated and Mobile Robotics for Services and Technologies (Springer 2001), Experimental Robotics VIII (Springer 2003), Advances in Control of Articulated and Mobile Robots (Springer 2004), Advanced Bimanual Manipulation: Results from the DEXMART Project, Springer 2012). He has delivered more than 130 invited seminars abroad. Since 2002 Professor Siciliano is Co-Editor of the Springer Tracts in Advanced Robotics book series. He has served as an Associate Editor of the IEEE Transactions on Robotics and Automation from 1991 to 1994, and of the ASME Journal of Dynamic Systems, Measurement, and Control from 1994 to 1998. He has served on the Editorial Boards of Robotica since 1994, the Journal of Robotic Systems from 2001 to 2005, and the JSME International Journal from 1998 to 2001. He is also Co-Editor of the Springer Handbook of Robotics which received the AAP PROSE Award for Excellence in Physical Sciences & Mathematics and was also the winner in the category Engineering & Technology. He is an ASME Fellow, an IEEE Fellow, and an IFAC Fellow. He has held representative positions within the IEEE Robotics and Automation Society: Administrative Committee Member from 1996 to 1999 (elected for two consecutive terms) and again since 2005, Vice-President for Publications in 1999, Vice-President for Technical Activities from 2000 to 2003, Distinguished Lecturer from 2004 to 2007, and Fellow Evaluation Committee Member in 2004, 2006, 2007 and 2010. He has served as President-Elect for 2006–2007, President for 2008–2009, Junior Past-President for 2010–2011, Senior Past-President for 2012–2013. He has received several awards including the 2010 IEEE RAS Distinguished Service Award and the 2015 IEEE RAS George Saridis Leadership Award in Robotics and Automation. From 1996 to 1999 he has been Chair of the Technical Committee on Manufacturing and Automation Robotic Control of the IEEE Control Systems Society. He has served as chair or co-chair for numerous international conferences. His group has been granted fourteen European projects, including an Advanced Grant from the European Research Council.