This full text paper was peer-reviewed at the direction of IEEE Instrumentation and Measurement Society prior to the acceptance and publication.

In-Hand Object Material Characterization with Fast Level Set in Log-Polar Domain and Dynamic Time Warping

Fei Hui, Pierre Payeur
School of Electrical Engineering and Computer Science
University of Ottawa
Ottawa, Canada
fhui065, ppayeur@uottawa.ca

Ana-Maria Cretu
Department of Computer Science and Engineering
Université du Québec en Outaouais
Gatineau, Canada
ana-maria.cretu@uqo.ca

***Abstract*—Safe manipulation of soft deformable objects is a challenging task generally requiring a complete knowledge on the object, including its shape, material properties, surface texture, as well as its position in the environment. Due to the complexity of the problem, relatively few researchers dedicated their attention to the characterization of 3D object properties without imposing assumptions on the object's material. This paper proposes an original solution for in-hand characterization of deformable objects' material. A Kinect sensor observes the object during interaction with a Barrett robot hand. The proposed solution capitalizes on an advantageous combination of user-guided object selection, a RANSAC-based automated background removal procedure, an original fast level set method in the log-polar domain to identify the contour of the manipulated object, and dynamic time warping to characterize the object as rigid, elastic, plastic, or elasto-plastic, based on the tracked contour. When integrated in the control scheme for the robot hand, the proposed approach can contribute to ensuring a stable grasp and precise manipulation capability, enabling robots to handle efficiently unknown objects.**

Keywords—soft object characterization; deformation tracking; robot hand; Kinect; contours; material properties; log-polar transform; dynamic time warping.

I. INTRODUCTION

Safe and efficient manipulation of soft objects without human intervention, while being a fundamental capability of autonomous robot systems, is still a challenging area of research. Such dexterous manipulation generally requires a complete knowledge of the manipulated object, including its properties (i.e. shape, texture, material, stiffness) and its location in the environment. Relatively few publications address the issue of manipulation and grasping of 3D objects [1, 2], and even fewer researchers so far developed solutions for automated characterization of the properties of 3D deformable objects under interaction. In most cases presented in the literature, this characterization involves the approximate identification of the material properties (i.e. elasticity parameters) for the object model by comparing the real object subject to interactions and its simulated counterpart, and aiming to minimize the differences between them. The model is generally either a mass-spring model or a finite element (FEM) representation [3, 4]. A few other representations have been proposed as well, such as surfels [2, 5]. However, these approaches work by making assumptions on the object material, such as linearity, or isotropy, which are inadequate for several real-world materials such as foam or rubber for which the elastic parameters are not well defined.

In this paper we propose a solution for the automated characterization of 3D soft or deformable objects without assumptions on their material. The object in interaction with a robotic hand is observed by a Kinect sensor, placed above a robotic hand. The RGB-D data is analyzed to initially segment the object of interest from the background. Starting from a point selected by the user over the surface of the object in the color image, a background removal procedure is performed over the depth point cloud based on the commonly used random sample consensus (RANSAC) iterative method to fit a model with observed data. It is followed by a *k*-nearest search to identify the most probable 3D points representing the object of interest. The obtained point cloud containing the object is projected back in 2D and the corresponding area in the color image is further analyzed in the YUV color space. A color selection scheme is applied that uses either the U or V component and a novel fast level set method in the log-polar domain is proposed to enable real-time contour identification and subsequently contour tracking in the data stream coming from the Kinect. A solution based on dynamic time warping is employed over the tracked contour to characterize the object material as rigid, elastic, plastic or elasto-plastic. This characterization enables better manipulation capabilities for a robot hand. When included in the control scheme for the robotic hand, it contributes to ensuring stable grasp and precise manipulation capabilities without a priori knowledge on object's material, therefore enabling robots to handle efficiently unknown objects.

II. LITERATURE REVIEW

This section summarizes the most relevant related work on object material characterization and on robotic manipulation of 3D objects. The solution for in-hand modeling of 3D rigid objects from RGB-D data proposed by Krainin *et al.* [2]

employs a Kalman filter that produces at each frame estimates of the robot manipulator, the object and the Kinect sensor. These estimates enable the segmentation of the object, and its model is built as a series of surfels. Stuckler and Behnke's registration method [5] based on multi-resolution surfel maps provides a dense displacement field between object shapes monitored in RGB-D images. In [4], a linear isotropic 3D deformable object in interaction with a three-finger robot hand is modeled as a mass-spring system based on a tetrahedral mesh. The object deformations and the contact points estimation are based on node position tracking and solving the dynamic equations of Newton's second law. Mateo *et al.* [6] measure the stiffness of a 3D planar elastic object by the curvature of surface points extracted from the object's geometry. A level set curve describes the local deformation of the object. In the same line of research, the authors of [7] inscribe markers on the surface of a piece of paper to track its folding in visual data. The paper in interaction with a robot hand is represented as a 2D grid of nodes connected by links that specify the bending constraints. In [8], sparse sets of oriented 3D points along the contour of an object manipulated by a robotic manipulator are monitored using a stereo camera and then predicted based on the motion induced by the robot. Schulman *et al.* [9] track deformable objects from a sequence of point clouds by identifying the correspondence between the collected data and a model of the object represented by a collection of linked rigid particles, governed by dynamical equations. The most probable positions of the particles are found using an expectation-minimization algorithm. The experimentation is performed against a green background, limiting the applicability of the solution in normal conditions.

Navarro-Alarcon *et al.* [10] propose a solution for robotic manipulation of elastic objects that controls simultaneously the object's final position (i.e. interest points over the object and its centroid) as well as its deformations (i.e. the compression distance between points of interest, the folding angle and the normalized curvature of the object). The solution is however limited to elastic deformations only. Similarly, the work in [3] is dedicated to the real-time tracking of 3D elastic objects in RGB-D sensor data. Assuming that a prior segmentation of the object of interest exists, the object is tracked using a graph-cut approach, and an iterative closest point (ICP) approach is applied on the resulting point cloud to estimate a rigid transformation from the point cloud to a linear tetrahedral finite-element model representing the object. Linear elastic forces exerted on vertices are computed from the point cloud to the mesh based on closest point correspondence and the mechanical equations are solved numerically to simulate the deformed mesh. Hur *et al.* [11] propose a 3D deformable spatial pyramid model to find the dense 3D motion flow of deformable objects in RGB-D data without assuming a prior model or template for the object. A depth hole-filling algorithm is applied to correct the point cloud, followed by Gaussian filtering prior to the computation of a series of perspectively normalized descriptors. The 3D deformable spatial pyramid finds dense correspondences between instances of a deformed object by optimizing an objective function, in form of an energy corresponding to a Markov random field, taking into consideration factors such as: translation, rotation, warping costs and descriptors matching costs.

III. PROPOSED APPROACH FOR SOFT OBJECT TRACKING AND MATERIAL CHARACTERIZATION IN RGB-D DATA

The proposed approach for in-hand object material characterization from RGB-D data is illustrated in Fig. 1.

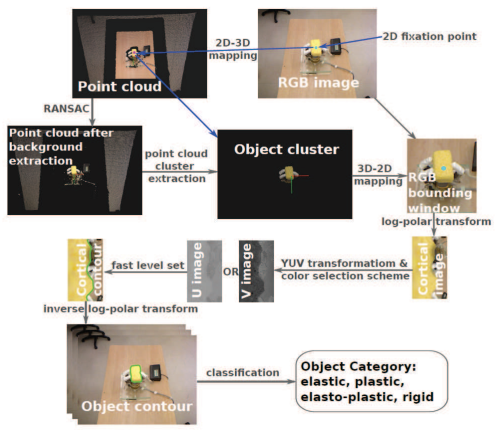


Fig. 1. System for object detection, tracking and material characterization.

The system takes as inputs the RGB-D data and a user selected point in the color image, so-called 2D fixation point. The latter guides the system towards the location of the object of interest. In an initial step, a 3D fixation point is calculated from the 2D fixation point provided by the user, using 2D-3D mapping in the RGB-D data. The depth point cloud recuperated from the Kinect is then subject to a background removal operation using the RANSAC algorithm [12], with the purpose to eliminate undesirable elements from the scene (i.e. measurement table, the robot hand mounting plate and other measurement equipment visible in the RGB-D data stream). This helps better identify the object shape and thus reduce the processing time. A cluster representing the object is identified based on the computed 3D fixation point in the point cloud from which the background is removed. This cluster is projected back to 2D to compute a bounding window around the identified object that is analyzed in the YUV color space and that serves for the initialization of the level set contour tracking method. A novel fast level set method in the log-polar domain is applied to extract and track the contour of the object within this bounding window. The object contour is then converted back to the RGB image coordinates and the object material is classified based on the contour information as belonging to a rigid, elastic, plastic or elasto-plastic object, or to the elastic, plastic, or elasto-plastic stages of deformation for more complex objects, using a dynamic time warping approach.

A. User-Selected Fixation Point and 2D-3D Mapping

As previously mentioned, the system is initially guided towards the location of the object of interest using a user-selected 2D fixation point. The user's intervention is only required for the first frame of the data stream. Such guidance is common in the current literature, but in many cases is more extreme than in the proposed solution. For example a prior segmentation of the object of interest is assumed to be available in [3], or the user is asked to fully crop the object in the initial frame. In this work, the fixation point can be selected randomly on the object of interest, but experiments demonstrated that a 2D fixation point chosen roughly at the centre of the object generally leads to better segmentation results and to smoother contours. To obtain the corresponding point in 3D, the 2D-3D mapping uses the intrinsic depth camera parameters of the Kinect sensor [13]. This 3D fixation point serves as a seed point to best separate the object of interest from the background in the RGB-D data stream.

B. Background Removal and Object Cluster Identification

During experimentation the object is placed in a robot hand situated on a table (Fig. 2a). Therefore an efficient way to remove most of the background is to locate the planar surface representing the table and to extract that surface from the RGB-D data. The planar surface identification is viewed as a plane-fitting problem which can be resolved with the random sample consensus (RANSAC) algorithm [12]. Once the best plane model is identified, the planar surface and the background information are represented by the inliers of the model. They can then be removed from the point cloud (Fig. 2c). In order to extract the object of interest from the remaining part of the point cloud, the nearest elements to the 3D fixation point, in a Euclidean sense, are extracted using a *k*-nearest neighbors search [14], with an empirically chosen value of *k*=100. The point cloud is mapped in a *k*-d tree data structure to accelerate the neighbor search.

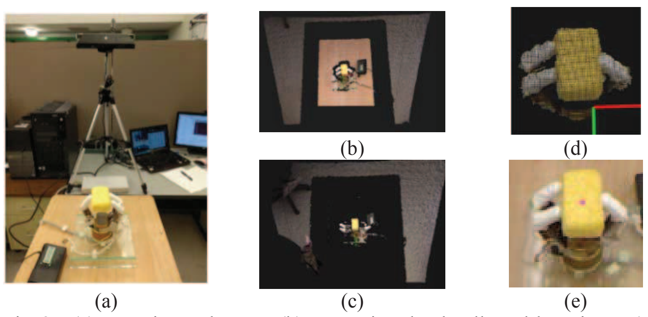


Fig. 2. (a) Experimental setup, (b) raw point cloud collected by Kinect, (c) background extraction result, (d) extracted point-cloud representing the object, and (e) color image with 2D fixation point shown in red.

The group of 2D pixels corresponding to the 3D points in the identified cluster (Fig. 2e) is then employed as initialization for the level set method. In particular, the bounding window with which the level set is initialized represents the bounding box of the 2D points (left, right, up, down extrema) enlarged by a border of 50 pixels all around to ensure that the contour remains within the search region in case the object shifts or rotates during manipulation.

C. Fast Level Set in Log-polar for Object Contour Tracking

In order to segment and track the object contour, a novel fast level set method in the log-polar domain is proposed. The advantage of using this transformation (Fig. 3a) is that the object of interest gets to fill a relatively large area of the log-polar image compared to the remaining background area, as can be observed in Fig. 3b. The percentage of space occupied by the object of interest (left side of the green separation line) is larger than in the original image as the log-polar map is centered on the fixation point. Moreover the mapping can reduce the size of the representation with respect to the original image (i.e. from 187×200 to 93×167 pixels in our case) based on the set resolution of the log-polar mapping. These contribute to accelerate object segmentation. Instead of directly using RGB color values in the log-polar image (Fig. 3b), a transformation to the YUV color space is performed, where Y is the luminance of a color, and U and V represent the two chromatic components.

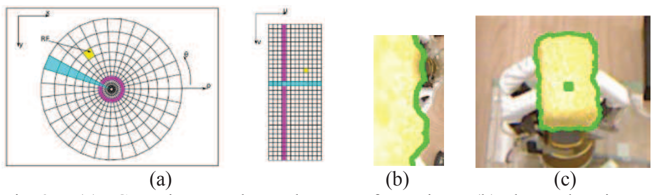


Fig. 3. (a) Cartesian to log-polar transformation, (b) log-polar image corresponding to the RGB image in Fig. 2d, and (c) RGB image with contour obtained by inverse mapping from log-polar domain.

The simplified UV map offers a more compact and therefore faster solution to process color information. In this work, either the U or the V component is selected for contour detection, depending on which one has the largest standard deviation. This choice is justified by the desire to work with higher contrast to enable more accurate segmentation.

The proposed method to segment and track the object contour builds on the fast implementation of level sets [15], while also drawing inspiration from [16, 17] that find contours in the log-polar domain based on the YUV color coding. In the log-polar domain, the curve, represented as the zero level set of the level set method is not an enclosing contour of the object, but rather an open boundary line in the vertical direction, as shown in green in Fig. 3b. As in [15], the proposed fast level set method in log-polar domain employs two neighboring lists, denoted by $\tilde{L}_{in}$ and $\tilde{L}_{out}$, respectively representing the inside and outside neighboring pixels of the curve, and defined as follows:

$$\begin{aligned}\tilde{L}_{out} &= \{\boldsymbol{x} \mid \tilde{\phi}(\boldsymbol{x}) > 0 \text{ and } \exists \boldsymbol{y} \in \tilde{N}_4(\boldsymbol{x}) \text{ such that } \tilde{\phi}(\boldsymbol{y}) < 0\} \\ \tilde{L}_{in} &= \{\boldsymbol{x} \mid \tilde{\phi}(\boldsymbol{x}) < 0 \text{ and } \exists \boldsymbol{y} \in \tilde{N}_4(\boldsymbol{x}) \text{ such that } \tilde{\phi}(\boldsymbol{y}) > 0\}\end{aligned} \tag{1}$$

In Fig. 4a, the green line represents the curve $\tilde{C}$, which splits the log-polar image into two parts: the object on its left and the background to its right. The list $\tilde{L}_{in}$ contains the pixels located on the left side of the curve, shown in dark gray, while the list $\tilde{L}_{out}$ contains the pixels located on the right side of the curve, shown in light gray. $\tilde{N}_4(\boldsymbol{x})$ represents a 4-connected discrete neighborhood of a pixel, $\boldsymbol{x} = (x, y)$, in a two-dimensional cortical image. $\tilde{N}_4(\boldsymbol{x})$ contains the four neighboring pixels of $\boldsymbol{x}$, $\boldsymbol{x}_{up}$, $\boldsymbol{x}_{down}$, $\boldsymbol{x}_{left}$, $\boldsymbol{x}_{right}$, as illustrated in Fig. 4c. $\tilde{\phi}$ is the level set function. It is negative for the object and positive for the background and is defined as follows [15]:

$$\tilde{\phi}(\boldsymbol{x}) = \begin{cases} 3, & \text{if } \boldsymbol{x} \text{ is exterior pixel;} \\ 1, & \text{if } \boldsymbol{x} \text{ is in } \tilde{L}_{out}; \\ -1, & \text{if } \boldsymbol{x} \text{ is in } \tilde{L}_{in}; \\ -3, & \text{if } \boldsymbol{x} \text{ is interior pixel.} \end{cases} \tag{2}$$

To illustrate the movement of switching pixels from $\tilde{L}_{in}$ to $\tilde{L}_{out}$ and vice versa during the tracking of the object, Fig. 4b shows an example in which the curve $\tilde{C}$ moves outwards relative to the object at a pixel B and inwards at a pixel D. This behavior is represented as a switch of pixel B from $\tilde{L}_{out}$ to $\tilde{L}_{in}$ and a switch of pixel D from $\tilde{L}_{in}$ to $\tilde{L}_{out}$.

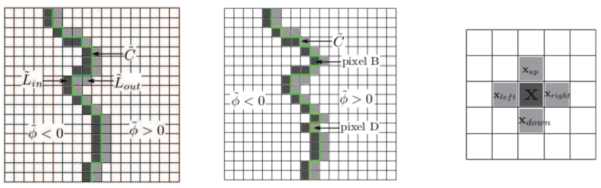


Fig. 4. (a) Representation of the curve $\tilde{C}$, and the two lists of neighboring pixels $\tilde{L}_{in}$ and $\tilde{L}_{out}$ in the log-polar domain, (b) the motion of switching pixels of $\tilde{L}_{in}$ and $\tilde{L}_{out}$, (c) representation of the 4-connected discrete neighborhood, $\tilde{N}_4(x)$, of a pixel $\boldsymbol{x}$. Each element of the grid denotes a pixel of the image.

The functions of switching the pixels from one list to the other are described in pseudocode as follows [15]:

```
switch_in($\boldsymbol{x}$): Remove $\boldsymbol{x}$ from $\tilde{L}_{out}$ and add it to $\tilde{L}_{in}$. Set $\tilde{\phi}(\boldsymbol{x}) = -1$.
For $\forall\, \boldsymbol{y} \in \tilde{N}_4(\boldsymbol{x})$ with $\tilde{\phi}(\boldsymbol{y}) = 3$, add $\boldsymbol{y}$ to $\tilde{L}_{out}$. Set $\tilde{\phi}(\boldsymbol{y}) = 1$.
```

and

```
switch_out($\boldsymbol{x}$): Remove $\boldsymbol{x}$ from $\tilde{L}_{in}$ and add it to $\tilde{L}_{out}$. Set $\tilde{\phi}(\boldsymbol{x}) = 1$.
For $\forall\, \boldsymbol{y} \in \tilde{N}_4(\boldsymbol{x})$ with $\tilde{\phi}(\boldsymbol{y}) = -3$, add $\boldsymbol{y}$ to $\tilde{L}_{in}$. Set $\tilde{\phi}(\boldsymbol{y}) = -1$.
```

The function switch_in(**x**) is employed for a curve moving outwards at a pixel $\boldsymbol{x} \in \tilde{L}_{out}$, and switch_out(**x**) is employed for a curve moving inwards at a pixel $\boldsymbol{x} \in \tilde{L}_{in}$. The curve is evolved according to the speed function, which is separated into two parts: the data-dependent speed function, $\tilde{F}_{ext}$ and the curve smoothness regularization speed function, $\tilde{F}_{in}$, where the data-dependent speed function, $\tilde{F}_{ext}$, is computed as in [18]:

$$\tilde{F}_{ext} = \begin{cases} -\lambda_1 (I(x,y) - c_1)^2 + k\lambda_1 (I(x,y) - c_2)^2, \text{if } \left|\tilde{F}_{ext}\right| < th \\ -(I(x,y) - c_1)^2 + (I(x,y) - c_2)^2, \quad \text{otherwise} \end{cases} \tag{3}$$

where $I(x, y)$ is the color information at pixel (x, y), coming from either the U or V component. The parameters c_1 and c_2 are the mean intensities of the images on the left and right side of the curve $\tilde{C}$, respectively, given by:

$$c_1(\tilde{\phi}) = \frac{\int I(x,y) H(\tilde{\phi}) dxdy}{\int H(\tilde{\phi}) dxdy} \quad c_2(\tilde{\phi}) = \frac{\int I(x,y)(1 - H(\tilde{\phi})) dxdy}{\int (1 - H(\tilde{\phi})) dxdy} \tag{4}$$

where $H\left(\tilde{\phi}\right)$ is the Heaviside function such that $H\left(\tilde{\phi}\right) = \begin{cases} 1, & \tilde{\phi} < 0 \\ 0, & \tilde{\phi} > 0 \end{cases}$.

The curve smoothness regularization term of the speed function $\tilde{F}_{in}$ is approximated by:

$$\tilde{F}_{in} = \mu \nabla \cdot (\nabla \tilde{\phi} / |\nabla \tilde{\phi}|) = \mu \kappa \tag{5}$$

where κ is the curvature of the evolving curve, μ is the regularization parameter, and ∇ represents the derivative function. The curvature is computationally expensive and therefore the Laplacian of $\tilde{\phi}$ is used as a simplified term of the curvature [15]. Furthermore, the evolution of the Laplacian of a function is equivalent to Gaussian filtering this function. As a result, a Gaussian filter, *G*, is employed as a smoothness regularization term to accelerate the fast level set implementation. The Gaussian filter is applied only on the pixels of $\tilde{L}_{out}$ and $\tilde{L}_{in}$ in order to smooth the zero level set. With the defined speed function, $\tilde{F}$, the evolution of curve $\tilde{C}$ follows the following algorithm:

```
Initialize the array $\tilde{\phi}$, $\tilde{F}_{ext}$, and the two lists $\tilde{L}_{in}$ and $\tilde{L}_{out}$.
for $i = 1 : N_a$ do                                          // Cycle one
  Compute $\tilde{F}_{ext}$, for pixels in $\tilde{L}_{in}$ and $\tilde{L}_{out}$;
  For each pixel $\boldsymbol{x} \in \tilde{L}_{out}$, apply switch_in($\boldsymbol{x}$) if $\tilde{F}_{ext}(\boldsymbol{x}) > 0$;
  For each pixel $\boldsymbol{x} \in \tilde{L}_{in}$, remove $\boldsymbol{x}$ from $\tilde{L}_{in}$ and set $\tilde{\phi}(\boldsymbol{x}) = -3$ if
        $\tilde{\phi}(\boldsymbol{y}) < 0, \forall \boldsymbol{y} \in \tilde{N}_4(\boldsymbol{x})$;
  For each pixel $\boldsymbol{x} \in \tilde{L}_{in}$apply switch_out($\boldsymbol{x}$) if $\tilde{F}_{ext}(\boldsymbol{x}) \le 0$;
  For each pixel $\boldsymbol{x} \in \tilde{L}_{out}$,remove $\boldsymbol{x}$ from $\tilde{L}_{out}$ and set $\tilde{\phi}(\boldsymbol{x}) = 3$ if
        $\tilde{\phi}(\boldsymbol{y}) > 0, \forall \boldsymbol{y} \in \tilde{N}_4(\boldsymbol{x})$;
  Check the stop condition and if satisfied go to Cycle Two
  Else continue this cycle.
end for
for $i = 1 : N_g$ do                                          // Cycle two
  For each pixel $\boldsymbol{x} \in \tilde{L}_{out}$,compute $G \otimes \tilde{\phi}(\boldsymbol{x})$ and apply switch_in($\boldsymbol{x}$)
        if $G \otimes \tilde{\phi}(\boldsymbol{x}) < 0$;
  For each pixel $\in \tilde{L}_{in}$, remove $\boldsymbol{x}$ from $\tilde{L}_{in}$ and set $\tilde{\phi}(\boldsymbol{x}) = -3$
        if $\tilde{\phi}(\boldsymbol{y}) < 0, \forall \boldsymbol{y} \in \tilde{N}_4(\boldsymbol{x})$;
  For each pixel $\boldsymbol{x} \in \tilde{L}_{in}$, compute $G \otimes \tilde{\phi}(\boldsymbol{x})$. Apply switch_out($\boldsymbol{x}$)
        if $G \otimes \tilde{\phi}(\boldsymbol{x}) > 0$;
  For each pixel $\boldsymbol{x} \in \tilde{L}_{out}$, remove $\boldsymbol{x}$ from $\tilde{L}_{out}$ and set $\tilde{\phi}(\boldsymbol{x}) = 3$
        if $\tilde{\phi}(\boldsymbol{y}) > 0, \forall \boldsymbol{y} \in \tilde{N}_4(\boldsymbol{x})$;
end for
If the stop condition is satisfied in Cycle One, terminate the algorithm; else
go back to Cycle One.
```

The two stop conditions for this algorithm are as follows: (1) the speed at every neighboring pixel satisfies:

$$\begin{aligned} \tilde{F}_{ext}(\boldsymbol{x}) \le 0, \forall \boldsymbol{x} \in \tilde{L}_{out} \\ \tilde{F}_{ext}(\boldsymbol{x}) \ge 0, \forall \boldsymbol{x} \in \tilde{L}_{in} \end{aligned} \tag{6}$$

(2) a pre-specified maximum number of iterations is reached.

The variable N_g represents the size of the Gaussian filter, *G*, applied over the contour in order to smooth it. In this work, the two parameters are set empirically such that N_a=80 and N_g= 3. The resulting contours are remapped into the RGB image domain (Fig. 3c) for object material characterization.

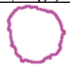
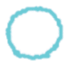
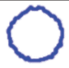
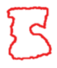
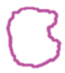
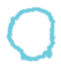
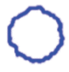
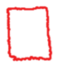
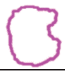
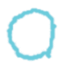
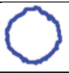
D. Object Material Characterization

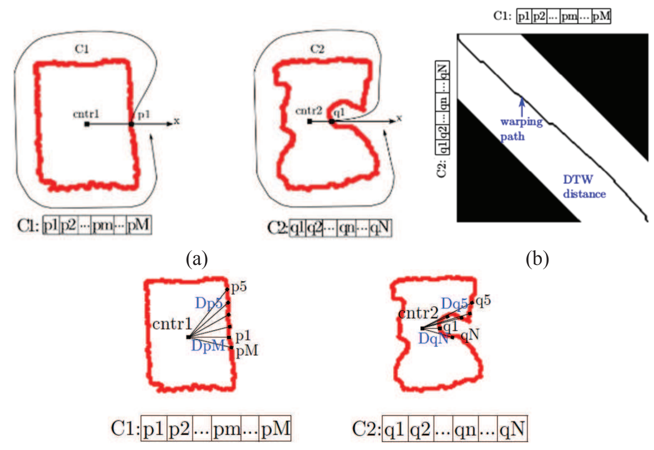
The comparison between the initial contour, the contour under the largest deformation, and the final object contour once the force is removed can be exploited to detect the object material as illustrated in Table 1.

An elastic object returns to its initial shape (contour) once the deformation force is removed. In this case the initial and the final object contour are identical within a certain tolerance (Table 1, column 1). A plastic object remains in the same state of deformation when the force is removed (Table 1, column 2); while in the case of an elasto-plastic object, the object partially retrieves its shape once the deforming force ceases (Table 1, column 3). Finally, for a rigid object, the three contours are roughly identical (Table 1, column 4) as no deformation occurs when forces are applied.

Exploiting these observations, we propose a solution based on dynamic time warping (DTW) to compare these three contours. Starting from a contour, its center (***cntr***) is first identified by applying principal component analysis. The first point of the contour is then identified as the pixel whose slope with respect to the center is 0 and is located on the right side of the center. Once this pixel is identified, the contour sequence is filled up with the remaining pixels in the counter-clockwise direction (along the arrow in Fig. 5a).

TABLE I. CONTOURS COMPARISON FOR MATERIAL CHARACTERIZATION

	Elastic material (stage)	*Plastic material (stage)*	*Elasto-plastic material (stage)*	*Rigid object*
Initial contour				
Contour under largest deformation				
Contour after the force is removed				



(c)

Fig. 5. Contour comparison for objects with various material properties: (a) contour alignment, (b) DTW matrix with warping path for two consecutive contours, (c) distance computation for score calculation.

Figure 5a shows an example of two consecutive contours of different lengths, $C_1 = \boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_m, \ldots, \boldsymbol{p}_M$ and $C_2 = \boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_n, \ldots, \boldsymbol{q}_N$, at time t and $t + 1$ respectively, extracted from the data stream. Dynamic time warping is employed to optimally match pixels from the two contours. In order to compare the differences between C_1 and C_2, the DTW matrix maps the similarity of each pair of pixels in the sequences of C_1 and C_2. In the DTW matrix, the value of each element is the sum of accumulated value of the previous element and the local cost that is the Euclidean distance between the pixel $\boldsymbol{p}_m$ from the first contour and the pixel $\boldsymbol{q}_n$ from the second contour. The value of local cost is low if $\boldsymbol{p}_m$ and $\boldsymbol{q}_n$ are similar to each other, or high otherwise [19]. The warping matrix is computed with an additional locality constraint [20]. This means that for the contour sequences C_1 and C_2, instead of calculating the DTW distance over all pairs of pixels, the DTW distance is only calculated for the part of warping matrix where the condition $|M - N| \leq v$ is satisfied. With this locality constraint (i.e. $v = 1/5 \times \max(M, N)$ and then set to $v = \max(v, |M - N|)$), the computed DTW distance is restricted to a band (shown in white in Fig. 5b), along the diagonal of the warping matrix. In Fig. 5b, the black corners are eliminated from the calculations to accelerate the computation.

To compute the differences between the initial contour, bw_0, the contour under largest deformation, bw_1, and the contour after the force is removed, bw_2, we compute a score inspired from [21] which monitors their similarity. But in our case the score also takes into account shifting and slight rotation movements that the object could exhibit during manipulation. The proposed score is calculated as:

$$score(C_1, C_2) = e^{-\frac{\sum_{l=1}^{L} DIST_l}{L}} \qquad (7)$$

where L is the length of the warping path (marked as a black line over the white diagonal in Fig. 5b), and $DIST_l = |Dp_l - Dq_l|$, where Dp_l is the Euclidean distance between the pixel $\boldsymbol{p}_m$ and its contour center $\boldsymbol{cntr}_1$, $Dp_l = ||\boldsymbol{p}_m - \boldsymbol{cntr}_1||$, and Dq_l is the Euclidean distance between the pixel $\boldsymbol{q}_n$ and its contour centre $\boldsymbol{cntr}_2$, $Dq_l = ||\boldsymbol{q}_n - \boldsymbol{cntr}_2||$. Fig. 5c illustrates the Euclidean distances between a contour pixel and the center of the contour. Because it is desired that the method is robust with respect to noise and fast at the same time, we impose two conditions in order to consider that a contour has been deformed. In particular, we verify if a displacement of the contour occurred and that the displacement affects more than three continuous pixels over the contour. Considering the sequence of $DIST_l$ values, if the value of an element in this vector is larger than a threshold (e.g. 4 pixels in our work), it is considered that a displacement occurred, otherwise, the difference is attributed to noise and it is considered that no displacement occurred. In case a displacement occurred, we also verify the number of continuous pixels affected by the displacement. In particular, if the number of continuous pixels with displacement is larger than 3 pixels, the contour is considered being deformed. Only the deformed contours are evaluated using eq. (7). The decision making process based on the computed score is summarized in Fig. 6, where the threshold, thr, on the score was empirically set to 0.75. Experiments revealed that for larger threshold values (e.g. $thr = 1.0$), the proposed classification approach is over sensitive and noise impacts the classification; on the other hand, for lower threshold values (e.g. $thr = 0.5$), the approach is insufficiently responsive to slightly different contours.

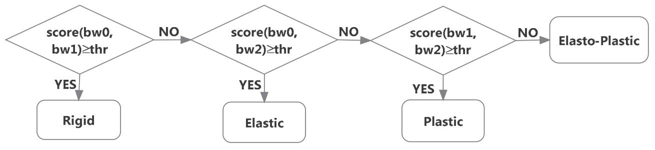


Fig. 6. Classification process for rigid, elastic, plastic, and elasto-plastic materials.

IV. EXPERIMENTAL RESULTS

The proposed approach was initially tested on four objects with different material properties, as illustrated in Fig. 7 and 8, to evaluate the performance of the framework. Fig. 8 also shows that the contour is correctly tracked regardless of the position of the object in the robot hand. Fig. 9 demonstrates that the solution is relatively robust to the location of the user-selected fixation point, marked by a green dot. However, one can notice that a more central position of the 2D fixation point leads to a smoother contour, while the contour is more jagged if the user selected point is closer to the sides of the object. This is a consequence of the use of a log-polar mapping which centers the contour extraction on the selected fixation point.

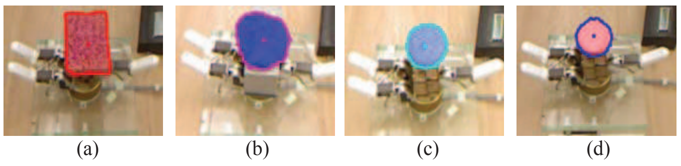


Fig. 7. (a) Elastic, (b) plastic, (c) elasto-plastic and (d) rigid material object.

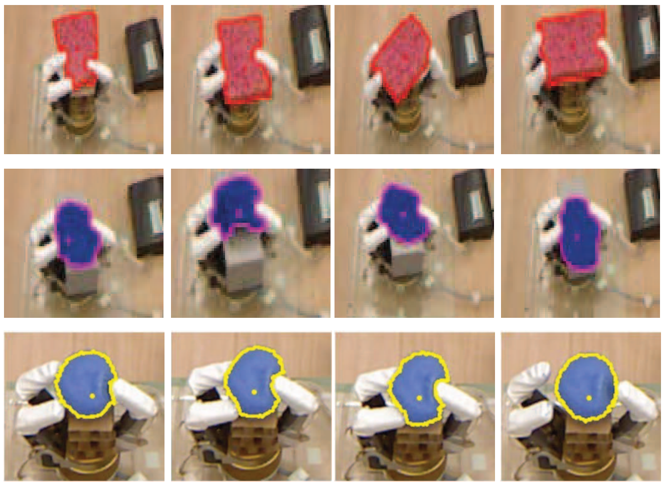


Fig. 8. Tracked contours at various stages of deformation for: elastic, plastic and elasto-plastic objects.

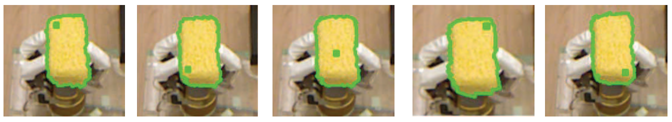


Fig. 9. Initial contour for various positions of the 2D fixation point.

To evaluate the object material characterization with the proposed dynamic time warping approach, tests were repeated 20 times for the 4 objects. During each repetition, the object is squeezed with a strong force and released three times and the result of material characterization is obtained by voting over the three trials. A recognition rate of 100% was achieved for rigid material, 100% for elastic material, 100% for elasto-plastic material, and 95% for plastic material. These results show that the approach works very well for most types of object material. The lower performance for plastic material is related to the use of plasticine packed in a transparent plastic bag, which is in fact not a perfect plastic material. The robot fingers tend to stick to the plasticine when a strong force is applied, impacting on the contour recognition.

V. Conclusion

The paper proposed a novel approach for real-time object deformation tracking in RGB-D data with the purpose of characterizing deformable objects under robot hand manipulation. The approach combines advantageously an automated background removal procedure based on the RANSAC algorithm over the depth point cloud, and a novel fast level set method in the log-polar domain accompanied by a color selection scheme to identify the contour of the object of interest in the RGB-D data stream. Dynamic time warping and an adapted scoring scheme are then employed to characterize the object properties based on the recuperated contour.

References

[1] A.M. Cretu, P. Payeur, E. M. Petriu, "Soft object deformation monitoring and learning for model-based robotic hand manipulation", *IEEE Trans. Systems, Man and Cybernetics – Part B*, vol. 42, no. 3, 2012, pp. 740-753.

[2] M. Krainin, P. Henry, X. Ren, D. Fox, "Manipulator and object tracking for in-hand 3D object modeling", *Int. Journal of Robotics Research*, 2011, pp. 1311-1327.

[3] A. Petit, V. Lippiello, B. Siciliano, "Real-time tracking of 3D elastic objects with an RGB-D sensor", *Proc. IEEE Int. Conf. Intelligent Robots and Systems,* Hamburg, Germany, Oct. 2015, pp. 3914-3921.

[4] L. Zaidi, B. Bouzgarrou, L. Sabourin, Y. Menzouar, "Interaction modeling in the grasping and manipulation of 3D deformable objects", *Proc. IEEE Int. Conf. on Advanced Robotics*, Istanbul, Turkey, 2015, pp. 504-509.

[5] J. Stückler, S. Behnke, "Perception of deformable objects and compliant manipulation for service robots", *Soft Robotics*, A. Verl et al. (Eds.), Springer, 2015, pp. 69-80.

[6] C.M. Mateo, P. Gil, D. Mira, F. Torres, "Analysis of shapes to measure surfaces", *Proc. IEEE Conf. Informatics in Control, Automation and Robotics*, Colmar, Jul. 2015, pp. 60-65.

[7] C. Elbrechter, R. Haschke, H. Ritter, "Folding paper with anthropomorphic robot hands using real-time physics-based modeling", Proc. *IEEE Int. Conf. Humanoid Robots*, Osaka, 2012, pp. 210-215.

[8] D. Kraft, N. Pugeault, E. Baseski, M. Popovic, D. Kragic, S. Kalkan, F. Worgetter, N. Kruger, "Birth of the object: Detection of objectness and extraction of object shape through object-action complexes", *Int. Journal Humanoid Robotics*, vol. 5, no. 2, 2008, pp. 247-265.

[9] J. Schulman, A. Lee, J. Ho, P. Abbeel, "Tracking deformable objects with point clouds", *Proc. IEEE Int. Conf. Robotics and Automation*, 2013, pp. 1130-1137.

[10] D. Navarro-Alarcon, H.M Yip, Z. Wang, Y.-H Liu, F. Zhong, T. Zhang, P. Li, "Automatic 3-D manipulation of soft object by robotic arms with an adaptive deformation model", *IEEE Trans. Robotics*, vol. 32, no. 2, 2016, pp. 429-441.

[11] J. Hur, H. Lim, S.C. Ahn, "3D deformable spatial pyramid for dense 3D motion flow of deformable object", *Proc. of ISVC 2014*, G. Bebis et al. (Eds.), LNCS 8887, 2014, pp. 118-127.

[12] M.A. Fischler, R.C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography", *Communications of the ACM*, vol. 24, 1981, pp. 381-395.

[13] K. Kramer, N. Burrus, F. Echtler, D.C. Herrera, M. Parker, *Hacking the Kinect*, Apress, Berkeley, USA, 2012.

[14] R.F. Sproull, "Refinements to nearest-neighbor searching in k-dimensional trees", *Algorithmica*, vol. 6, 1991, pp. 579-589.

[15] Y. Shi, W.C. Karl, "A real-time algorithm for the approximation of level-set-based curve evolution", *IEEE Trans. Image Processing*, vol. 17, no. 5, 2008, pp. 645-656.

[16] A.K. Mishra, Y. Aloimonos, "Active segmentation", *Int. Journal Humanoid Robotics*, vol. 6, no. 3, 2009, pp. 361-386.

[17] M. Chessa, F. Solari, "Local feature extraction in log-polar images", *Proc. of ICIAP 2015*, Springer, LNCS 9279, 2015, pp. 410-420.

[18] M. Thida, K.L. Chan, H.-L. Eng, "An improved real-time contour tracking", Pacific-Rim Symp. Image and Video Tech., 2006, pp. 702-711.

[19] M. Muller, *Information Retrieval for Music and Motion*, Springer, 2007.

[20] D. Lemire, "Faster Retrieval with a Two-Pass", *Pattern Recognition,* vol. 42, no. 9, 2009, pp. 2169-2180.

[21] E. Gonzalez-Sosa, R. Vera-Rodriquez, J. Fierrez, J. Ortega-Garcia, "Comparison of body shape descriptors for biometric recognition using MMW images", *EURASIP Journal on Image and Video Processing*, 2015, vol. 1, pp. 1-13.