

Planning Framework for Robotic Pizza Dough Stretching with a Rolling Pin



Jung-Tae Kim, Fabio Ruggiero, Vincenzo Lippiello, and Bruno Siciliano

Abstract Stretching a pizza dough with a rolling pin is a nonprehensile manipulation. Since the object is deformable, force closure cannot be established, and the manipulation is carried out in a nonprehensile way. The framework of this pizza dough stretching application that is explained in this chapter consists of four sub-procedures: (i) recognition of the pizza dough on a plate, (ii) planning the necessary steps to shape the pizza dough to the desired form, (iii) path generation for a rolling pin to execute the output of the pizza dough planner, and (iv) inverse kinematics for the bi-manual robot to grasp and control the rolling pin properly. Using the deformable object model described in Chap. 3, each sub-procedure of the proposed framework is explained sequentially.

1 Brief Introduction

Making a pizza is a wonderful art, and it requires delicate skills like preparing a pizza dough mixed with wheat powder, water, salt, and other ingredients. An accurate ratio is put in the preparation, stretching it dynamically and quickly into a disk-shaped, saucing or dressing it with proper ingredients and quantity, and finally burning it evenly and sufficiently in a wood-burning oven.

J.-T. Kim (✉)

MIKNMEK Inc., CheongAm-ro 87, Nam-gu, Pohang-si, 37673 Gyeongsangbuk-do, Republic of Korea

F. Ruggiero · V. Lippiello · B. Siciliano

Department of Electrical Engineering and Information Technology, PRISMA Lab, CREATE Consortium & University of Naples Federico II, Via Claudio 21, 80125 Naples, Italy
e-mail: fabio.ruggiero@unina.it

V. Lippiello

e-mail: vincenzo.lippiello@unina.it

B. Siciliano

e-mail: bruno.siciliano@unina.it

© Springer Nature Switzerland AG 2022

B. Siciliano and F. Ruggiero (eds.), *Robot Dynamic Manipulation*,
Springer Tracts in Advanced Robotics 144,
https://doi.org/10.1007/978-3-030-93290-9_9

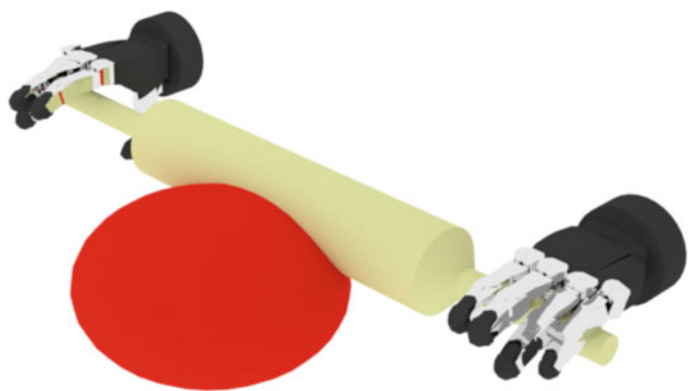


Fig. 1 Application: stretching a pizza dough with a rolling pin

Table 1 Main symbols used in this chapter

Definition	Symbol
Center of the pizza dough	$\mathbf{x}_0 = \begin{bmatrix} c_x & c_y \end{bmatrix}^T \in \mathbb{R}^2$
Generic point of the dough	$\mathbf{x} \in \mathbb{R}^2$
Boundary of the pizza dough	∂S
Thickness of the dough	$h > 0$
Frame associated to the plate	$\mathcal{D}$
Angle between the longest-axis of the dough shape and the x -axis of $\mathcal{D}$	$\theta \in \mathbb{R}$

Among these technical processes, this chapter focuses on stretching a pizza dough with a rolling pin (Fig. 1). This process includes two critical techniques: (i) the manipulation of a deformable object, and (ii) the control of the rolling pin to execute the proper actions. Regarding the former, difficulties arise during the manipulation planning of a deformable object because of the absence of a precise model. Indeed, typical deformable objects can not be described with just one property but have multiple properties like viscosity, elasticity, plasticity, or others, and mixed of them. Even though several proposed models represent these properties of a deformable object, it is not easy to find a suitable one for the pizza dough. In this chapter, the SPH formulation explained in Chap. 3 is used to model a highly viscous deformable object like the pizza dough, which is eligible to describe highly deformable objects, even for liquids. Regarding the latter, researches about controlling a tool with a robotic system are well established. However, nonprehensile manipulation is still a relevant and challenging topic. A general inverse-kinematics manipulation planning for the RoDyMan robot equipped with a rolling pin has been used in this pizza dough stretching application (Table 1).

Both techniques are explained in this chapter. The outline of the chapter consists of this introduction and a survey about the related state of the art (Sect. 2). The sketch of the proposed framework for the pizza dough stretching is depicted in Sect. 3, while the explanation of each sub-procedure is given from Sect. 4 to Sect. 7. Simulation results are explained in Sect. 8. Finally, Sect. 9 concludes the chapter.

2 Related Research

Manipulation of a deformable or a rheological object (i.e., the pizza dough) requires an understanding of the object's properties like viscosity, elasticity, and plasticity. For example, a bread dough consists of gluten proteins and various minor ingredients, including minerals. The gluten proteins play a crucial role in determining the unique baking quality of wheat by conferring water absorption capacity, cohesivity, viscosity, and elasticity. Gluten proteins can be divided into two main fractions according to their solubility in aqueous alcohols: the soluble gliadins and the insoluble glutenins [1]. It is widely accepted that gliadin accounts for the viscous properties and glutenin imparts the strength and elasticity that are necessary to hold the gases that are produced during fermentation and baking [2].

Many researchers have tried to characterize the bread dough's fundamental properties and analyze the influences of the substances. The densities of various doughs were measured in [3], while the viscoelasticity of bread dough was examined in [4–6]. The main substances for the bread dough that are H_2O (water), D_2O (heavy water), esterifying agents for glutamine residues, urea, salts, agents affecting disulfide bonding, and the protein subunits, were listed up in [7], also characterizing the influence of the substances for the rheology of the bread dough. Seventeen commercially available European wheat cultivars were sampled in [2]. Through these samples, the authors had the creep-recovery experiments and analyzed for a set of chemical and rheological parameters and baking quality using the PCA method. The dynamic rheological properties of gluteins fractions with two English-grown wheat cultivars, Hereward and Riband, were studied in [8]. The authors confirmed that the viscoelasticity of the glutenin sub-fraction of gluten and differences in the ratio of gliadin to glutenin are the main factors governing inter-cultivar differences in the viscoelasticity of wheat gluten. Similarly, in [9], the authors experimented with the uniaxial elongational and shear rheology properties of doughs affected by the protein contents or glutenin-to-gliadin ratio. The conclusion was that increasing protein content lowered the maximum shear viscosity while increasing the glutenin-to-gliadin ratio increased the maximum shear viscosity. Stress and strain of the dough related to the ratio of the feed sheet thickness of the roller gap and the roller's speed in the sheeting system were studied in [10]. The authors applied the lubrication approximation for the equation of motion and used an inelastic power-law model for the dough rheology. The relationship between the rheological properties for static dough and dynamic rheological properties for dough crumbs was also investigated in [11]. The former was evaluated by texture profile analysis, like uni-axis (stretching) and

bi-axial extension (inflation), and gluten index in static compression. The latter was evaluated through dynamic mechanical analysis and thermal mechanical analysis in dynamic compression. In [12], the authors described the influence of various substances in a dough. They introduced widely used modeling methods for dough rheology like power-law, Maxwell model, Lethersich model, Peleg model, and listing the various measurement methods for dough properties like farinograph, mixograph, rheomixer, extensigraph, alveograph, amylograph, maturograph, and so on. Based on these known properties, several visco-elastoplastic models were proposed, as the Herschel-Bulkley model and the K-BKZ model [13]. Bingham model is also one of the well-known models for representing plastic properties [14].

The baking industry uses rolling (or sheeting) process using counter-rotating rolls as a dough forming process for various products, such as cookies, crackers, pizza, bread, and pastry. The rolling process is akin to calendaring, which is used in many industries, such as the paper, plastics, rubber, and steel industries [15, 16]. An overall process for stretching of bread doughs was designed and implemented in [17].

There are more general researches for acquiring the properties of deformable objects. A four-element model for characterizing the viscoelasticity was proposed in [18]. In [19], a neural network model was employed to estimate an object's elastic properties. A FEM model was instead employed in [20] to estimate such properties. Interactive approaches to get object's elastic properties, even without the use of any specific model, were used in [21–25].

3 Framework for a Pizza Dough Stretching Behaviour

The application handled by this chapter roughly consists of three main components: (i) a robotic system grasping a tool, (ii) a deformable object, and (iii) the tool itself. In this application, the employed robotic system is the RoDyMan robot, the deformable object is a pizza dough, and the tool is a rolling pin (see Fig. 1).

The proposed framework is described in this section. Input data from a sensor device are acquired. This generates a proper action sequence for the robot. The employed 9 sensor device is a RGB-D Kinect camera mounted on the head of the RoDyMan robot.

The devised framework can be split into four components: (i) (deformable) object recognition, (ii) planning actions on the deformable object; (iii) planning actions for the tool, and (iv) robot manipulation planning. As evident from Fig. 2, these components are concatenated each other.

The first component, which is the object recognition module, gets sensor data as input. Its output is the status of the recognized object. In this application, an RGB-D Kinect camera takes pictures of the pizza dough on a plate. Afterwards, this component separates the area of the pizza dough and the background. It reconstructs the 3D shape of the pizza dough based on the 2D image data and some additional information about the pizza dough. Finally, this object recognition module describes

the shape of the pizza dough through numerical data. This description indicates the current status of the pizza dough.

The second component, which is the one in charge of planning the actions on the deformable object, gets the output of the previous component, the desired final shape of the dough, and some additional information (i.e., the transform look-up-table, which will be illustrated in the following sections). Its output is the planned sequence of actions on the deformable object. In this application, the current shape of the pizza dough, the desired shape of the pizza dough, and information about the dough deformation (i.e., how a particular action of the rolling pin deforms the object into another shape) are given. Then, this component finds out the best sequence of actions of the rolling pin to get the desired shape.

Similarly, the third component gets the output of the previous one and generates a continuous motion of the tool to realize the desired actions on the deformable object. Each action from the previous component is disconnected from the other. Therefore, it is necessary to generate a smooth continuous motion, which is the output of this module.

Finally, the last component is in charge of the robot planning to realize the output of the previous component. Considering the kinematic information of the employed robot and all the constraints, this component generates a smooth motion sequence for the robot to suitably moving the tool as specified by the previous component.

Each component is explained in detail within the sequel sections.

4 Pizza Dough Recognition

The pizza dough recognition component consists of two procedures: (i) digital image processing for raw data from the sensor device, and (ii) how to describe the status of the flatted deformable object.

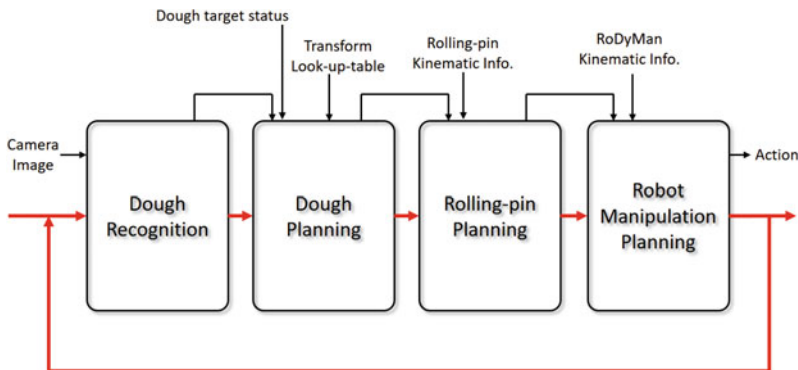
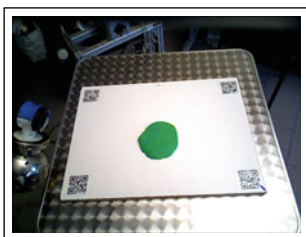


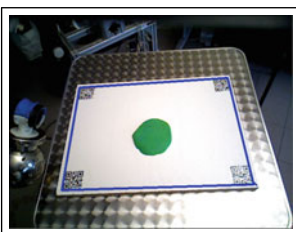
Fig. 2 Sketch of the overall process that consists of four concatenated components



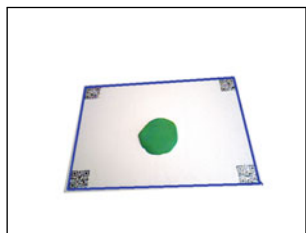
(a)



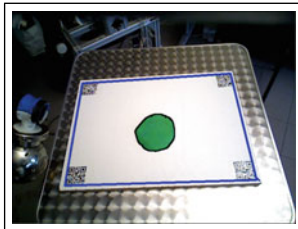
(b)



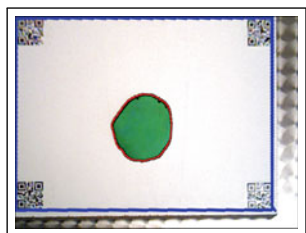
(c)



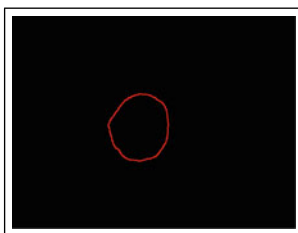
(d)



(e)



(f)



(g)

Fig. 3 Detection of the pizza dough on a white rectangular plate

4.1 Image Processing for Sensor Data

The perception procedure for a deformable object roughly consists of two parts in this application; one is the image processing for the camera sensor data, and the other is the representation of the status of the pizza dough. The perception method depends highly on the application; therefore, in this case, we analyse the stretching action of a pizza dough using the RoDyMan robot (Fig. 3a), and the recognition procedure naturally depends on a specific detector for the pizza dough. To reduce the difficulties of the perception based on image sensor data, some artificial restrictions are applied: restriction to the workspace by defining the boundary area (i.e., a plain rectangular plate) and the usage of a particular coloured deformable object (i.e., the blue colour) contrasted to the background colour (i.e., white colour). However, these restrictions are not critical but can be addressed with additional image processing.

With the help of the absolute position information of at least three corner markers in a rectangular plate and the kinematic information provided by the RoDyMan robot, it is not difficult to induce the frame transformation between a point in a given 2D camera image and the corresponding absolute position in the world frame [26]. In the carried out experiments, QR codes were placed at each corner of the rectangular plate, and then using an image matching with SIFT [27] the corner points were detected. Other feature detectors like the SURF [28] or the FAST [29] are also available. In order to easier detection of the corners, AR codes and the corresponding code detectors (i.e., ARToolkit [30]) can be used. The primary purpose of corner detection is to remove the dough plate's outlier and induce a transformation $\mathcal{T}$ from a 2D view image to a top-down viewed 2D space.

For the initial pizza dough, assuming that the ball- or bell-shaped pizza dough is symmetric to the vertical rotation axis, the reconstruction of the 3D shape is obtainable with partial camera views or depth sensors [31, 32]. Suppose the pizza dough is shallow after being pushed by a rolling pin so that the height difference is ignorable. In that case, the transformation $\mathcal{T}$ can be directly applied to the detected pizza dough as well as the plate. However, the initial pizza dough is more like a ball- or bell-shaped than a shallow disk. We do not commit much error in seeing the pizza dough as a 2D figure on a plate with uniform thickness: indeed, the more shallow the thickness of the dough, the minor error it occurs regarding a cylinder shape.

An example of the image processing steps to identify the pizza dough on the plate is given in Fig. 3b–g. With reference to the labels in the figure, the steps now briefly described. *b*) The raw image data from the camera mounted on the head of the RoDyMan robot is shown. With default information, we assume that the camera's view covers the whole area of the dough plate. *c*) There are four corner marks on the white dough plate. After detecting the marks, the area of the dough plate is outlined with blue lines. *d*) The outer area of the dough plate is removed from the image. *e*) Within the dough plate, the dough clay (the green object) is detected, and the outline is extracted through an edge detector [33], like the Canny edge detector [34]. *f*) The raw 2D camera image is deformed into the orthogonal top-down view with

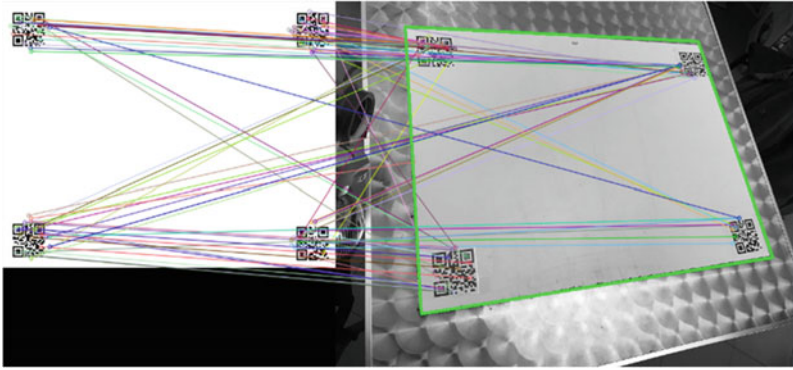


Fig. 4 Feature matching on the corners of the plate through the SURF method. On the left, the template image. On the right, the camera image. The green lines indicate the boundary of the detected plate

predefined width and height. *g*) The deformed outline of the detected pizza dough is finally obtained.

4.2 Description for a Status of a Pizza Dough

Pizza dough has various shapes. However, according to general experiences and our experiments in making a pizza, there is no problem assuming that the pizza dough's shape is convex. Hence, we can define the state of the pizza dough as the set of distances between the center of the dough and its boundary and the related tickness

$$\mathcal{C}_X : \left\{ \left[\begin{array}{c} \|\mathbf{x} - \mathbf{x}_0\| \\ h \end{array} \right] \in \mathbb{R}^2 : \mathbf{x} \in \partial S \right\}. \quad (1)$$

The dough center is intended as the geometric center of its shape. During the implementation, we discretized the configuration as a set of angle-equally sampled distances (see Fig. 5a) and vectorized them clock-wisely. The histogram of the vectorized configuration is shown in Fig. 5b.

When the dough shape is deformed through the action of the rolling pin, the centre position and the rotated angle of the dough are less important. Indeed, the relative angle between the rolling pin and the pizza dough would affect the deformation. To get a rotation-invariant configuration, the order of the histogram elements is rearranged so that the longest distance vector is the first one in the set (see Fig. 5c). This method has been frequently used in object recognition algorithms of image processing, *e.g.*, SIFT [35], SURF [28], MSER [36], FAST [29], and BRISK [37].

An extended configuration space $\mathcal{C}_Y$ is introduced to include the position of the centre and the angle between the longest distance vector (the first one in the rearranged

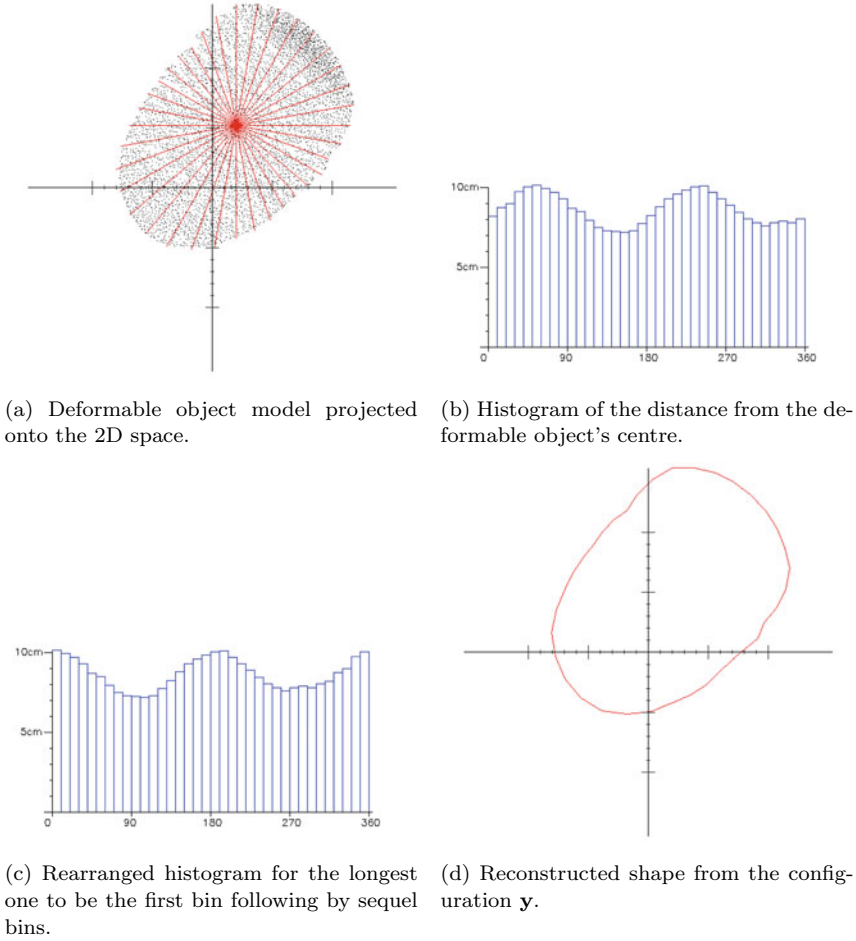


Fig. 5 Configuration for a dough state. The red radial lines in **a** indicate equal-angle sampled distance from the centre of the pizza dough shape to the boundary. Blue line is the longest distance from the centre. The sampled distances are ordered clock-wisely

histogram) and the x -axis of the frame $\mathcal{D}$

$$\mathcal{C}_Y : \left\{ \begin{bmatrix} \mathbf{x} \\ c_x \\ c_y \\ \theta \end{bmatrix} \in \mathbb{R}^5 : \mathbf{x} \in \mathcal{C}_x \right\}. \quad (2)$$

It can be seen in Fig. 5d how the reconstructed shape by the extended configuration $\mathcal{C}_Y$ is similar to the original shape. The original configuration space $\mathcal{C}_x$ of the pizza

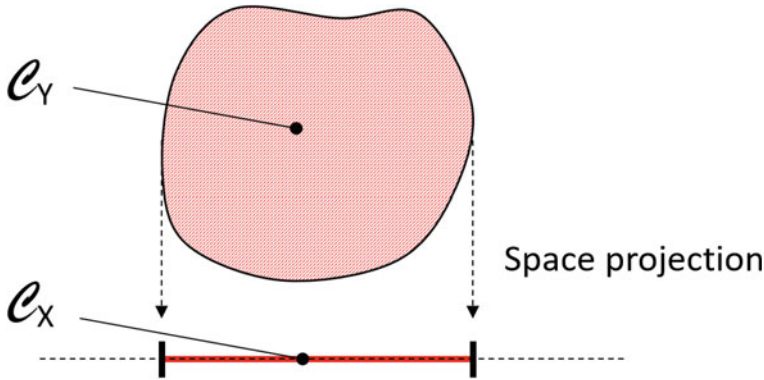


Fig. 6 The configuration space C_X and its extended configuration space C_Y

dough is a subspace of the extended configuration space C_Y : their relation is drawn in Fig. 6. Through space projection, the extended configuration space C_Y can be submerged into the original configuration space C_X , that is, $C_Y/\mathcal{I} = C_X$ where $\mathcal{I} = \{c_x = 0, c_y = 0, \theta = 0\}$.

5 Construction of a Planner for Pizza Dough Stretching

Given a generic configuration space $\mathcal{C}$, a initial status $q_I \in \mathcal{C}$, and a final status $q_G \in \mathcal{C}$, the planner finds the sequence of intermediate status

$$q_I \rightarrow q_1 \rightarrow \cdots \rightarrow q_{n-1} \rightarrow q_G, \quad (3)$$

to reach q_G from q_I , and where $q_1, \dots, q_{n-1} \in \mathcal{C}$. Each status is referred to as a configuration q in a given configuration space $\mathcal{C}$.

A way to find such a sequence of intermediate configurations is to associate to each of them a cost value relative to q_G . Let $V(q_i) > 0$ be a cost value related to the distance from $q_i \in \mathcal{C}$ to q_G , then a gradient descent method might be used to find the sequence of interemdiat configurations such that

$$V(q_I) \geq V(q_1) \geq \cdots \geq V(q_{n-1}) \geq V(q_G) = 0. \quad (4)$$

A movement from one configuration to another is called a transition $\mathcal{T}$. The transition occurs directly or indirectly through an action $\alpha \in \mathcal{A}$, where $\mathcal{A}$ is the set of admissible actions. Therefore, we can move from a configuration $q \in \mathcal{C}$ to a configuration $q' \in \mathcal{C}$ as

$$q \rightarrow q' = \mathcal{T}(q, \alpha). \quad (5)$$

The robotic system must execute the action α through a proper control design, taking into account possible errors during the execution.

Concerning the pizza stretching application, the configuration of the pizza dough is given by the sets $\mathcal{C}_X$ and $\mathcal{C}_Y$. The cost function $V(\cdot)$, the actions $\alpha \in \mathcal{A}$ and the transitions $\mathcal{T}$ to achieve the sough task are explained in the following.

5.1 Cost Value Function

In this application, the cost value function $V(\cdot)$ is defined as how the shape of the pizza in the configuration q is different from the shape of the pizza in the desired configuration q_G . Hence, the more similar the shape of q is to q_G , the smaller the cost value V is.

A comparison between two 3D objects is typically made by checking the respective volumes. However, the comparison can be simplified in this application by projecting the two 3D volumes into a 2D plane. In fact, the target object has a shallow disk, and the error between the approximated cylinder shape and the original dough shape decreases as deformations of rolling pin actions are made.

The comparison is made by calculating the ratio of the occupied 2D area of the current dough shape over the 2D area of the target one while ignoring the space out of the target 2D area (see Fig. 7). The areas can be measured by counting the occupied grid cells after discretizing the 2D plate.

The designed cost value function for this application is

$$V(q) = 1 - \frac{\text{area}(q)}{\text{area}(q_G)}, \quad (6)$$

where $q, q_G \in \mathcal{C}_Y$ are the current and the target configurations of the pizza dough shape, respectively. The operator to compute the area is defined as

$$\text{area}(q) = \sum_{ij} \text{occ}(q, i, j),$$

where i and j are the indexes of discretized 2D space, and the occupancy function $\text{occ}(q, i, j)$ is defined as

$$\text{occ}(q, i, j) = \begin{cases} 1 & \text{if } (i, j) \in A(q) \cap A(q_G) \\ -\kappa & \text{if } (i, j) \in A(q) \cap \neg A(q_G) \\ 0 & \text{otherwise} \end{cases}, \quad (7)$$

where $A(q)$ is the part of the plate occupied by the dough in the configuration q , $\neg A(q)$ is the complement of $A(q)$, and $\kappa > 0$ is a penalty weight. In particular, $\kappa = 0$ means that we do not care if the current shape of the pizza dough is outside the target

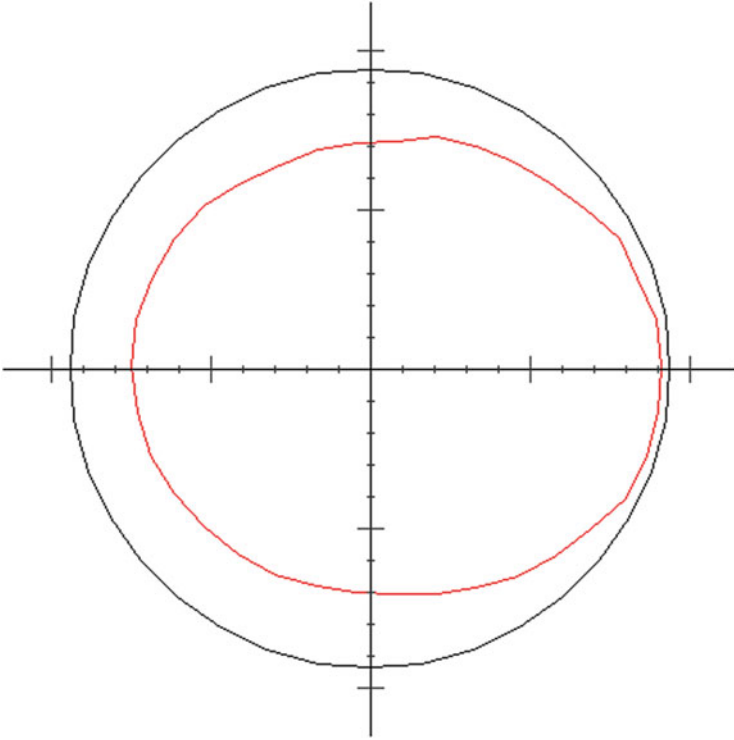


Fig. 7 An comparison between the current shape (red line) and the target shape (black line)

one, while $\kappa \gg 0$ indicates that we strongly penalize the case in which the current dough is outside the target one even partially. As it is built, the cost function assumes values in the range $[0, 1]$. When the cost function is zero, the current dough shape covers all the target one. In practice, we verified that it is difficult and inefficient to cover the target shape area completely: we thus recommend saturating to zero the cost function when its value is under 0.1, corresponding to the fact that about the 90% of the target shape area is covered.

5.2 *Actions for a Deformable Object*

There are a lot of possible actions to modify a pizza dough employing a rolling pin. For simplicity, the available actions in this application are limited. First, slanted or downing movements are prohibited. Besides, an action cannot change its angle during the movement. Then, the distance between the rolling pin and the plate is constant during all the movement. Finally, the rolling pin's movement is always done in contact with the dough.

Following the above constraints, the action set is defined as follows

$$\mathcal{A} : \left\{ \begin{bmatrix} \delta \\ \phi \end{bmatrix} \in \mathbb{R}^2 \right\}, \quad (8)$$

where $\delta > 0$ is the height between the rolling pin and the table and $\phi \in [0^\circ, \dots, 180^\circ]$ is the angle between the longest axis of the pizza dough and the x -axis in $\mathcal{D}$. Because the rolling pin's forward and backward movements are not discriminated in this application, the provided interval for ϕ is enough. During simulations, the action set $\mathcal{A}$ consisting of a height δ and an angle ϕ was discretized. The number of heights and angles used in our simulations are provided in Sect. 8.

Notably, the deformations caused by the actions defined above are relative to the local configuration space $\mathcal{C}_X$ and not the global configuration space $\mathcal{C}_Y$. This means that the deformation of the pizza after an action $\alpha \in \mathcal{A}$ is not affected by the global status $\mathbf{y} \in \mathcal{C}_Y$. However, it depends on $\chi \in \mathcal{C}_X$ and the relative angle between the pizza dough and the rolling pin.

5.3 Transition Originated from an Action

A transition changes the status of the dough to another thanks to an action. The transition function $\mathbf{T} : \mathcal{C}_Y \times \mathcal{A} \rightarrow \mathcal{C}_Y$ can be defined as

$$\mathbf{y}' = T(\mathbf{y}, \alpha), \quad (9)$$

where $\mathbf{y} \in \mathcal{C}_Y$ is the current state of the pizza dough and $\mathbf{y}' \in \mathcal{C}_Y$ is the one obtained after the execution of the action $\alpha \in \mathcal{A}$.

As previously mentioned, the deformation of the pizza dough through an action is more related to the relative angle between the rolling pin and the longest axis rather than the absolute pose of the pizza dough. Therefore, the Eq. (9) can be rewritten as

$$\begin{aligned} \mathbf{y}' &= T(\mathbf{y}, \alpha) \\ &= T([\mathbf{x}^T \ \mathbf{0}_3^T]^T, [\delta \ \phi - \theta]^T) + [\mathbf{0}_2^T \ \mathbf{x}_0^T \ \theta]^T. \end{aligned} \quad (10)$$

The new action $\alpha' = [\delta \ \phi - \theta]^T$ is already included within $\mathcal{A}$; hence, there is no change in the size of the action set $\mathcal{A}$. On the other hand, the state $[\mathbf{x}^T \ \mathbf{0}_3^T]^T$ might be interpreted as adimension reduction by projection (Fig. 6). In this way, the domain of the transition function T can be drastically reduced.

In the next section, how to generate such transitions will be explained.

5.4 LUT Method

Even though the deformation of an object due to an action is very well simulated through the SPH models (see Chap. 3.1), the problem is the computation time for each transition from one state to another state. The time depends on the resolution of the deformable object model and other conditions. In the chapter, the carried-out simulations took several minutes to hours for one transition. Considering that a planner requires at least several dozens, thousands, or much more transitions, this object model is not suitable for real-time or short-time planning.

LUT method is well-known for overcoming this computing-time problem. Possible transitions are calculated previously off-line, and then only the results are recorded into a transition database. Working online, the planner looks for a suitable transition in the database and uses it.

As explained in the previous Sect. 5.3, a transition T is independent from the position and rotation angle of the current state $\mathbf{y} \in \mathcal{C}_Y$. Therefore, LUT contains only transitions from a state $\chi \in \mathcal{C}_X$ at centre $\mathbf{x}_0 = \mathbf{0}_2$ and zero rotation angle, $\theta = 0$, with actions $\alpha \in \mathcal{A}$.

A transition space is continuous, but it is infeasible to generate and store all possible transitions. Therefore, discretizing the transition space is necessary, which requires storing selected transitions into a database and a method to find a proper transition within it. The conventional method to find a proper transition is to look for the most similar one and use it [38], or to use an interpolating method to estimate unknown transitions from a given state $\chi \in \mathcal{C}_X$ with the neighbour transitions. In the following subsections, we investigate how to find similar transitions into the database and interpolating them.

5.4.1 Similarity

In order to find similar transitions within the database, similarity measure functions between two states $\chi_1, \chi_2 \in \mathcal{C}_X$ and between two actions $\alpha_1, \alpha_2 \in \mathcal{A}$ are needed. We use the diagonal Mahalanobis distance $\text{sim}_X : \mathcal{C}_X \times \mathcal{C}_X \rightarrow \mathbb{R}^{\geq 0}$ as a function for the pizza dough configurations, whose definition is given below

$$\text{sim}_X(\chi_1, \chi_2) = \sqrt{\|\chi_1 - \chi_2\|_{\mathbf{S}_X^{-1}}}, \quad (11)$$

where $\mathbf{S}_X = \text{diag} \left(\begin{bmatrix} 1 & 1 \\ 1 & \beta \end{bmatrix} \right) \in \mathbb{R}^{2 \times 2}$, with $\beta > 1$. We use the function $\text{sim}_A : \mathcal{A} \times \mathcal{A} \rightarrow \mathbb{R}^{\geq 0}$ for the actions as

$$\text{sim}_A(\alpha_1, \alpha_2) = \sqrt{\|\alpha_1 - \alpha_2\|_{\mathbf{S}_A^{-1}}}, \quad (12)$$

where $\mathbf{S}_A = \text{diag} \left(\begin{bmatrix} \frac{1}{\gamma} & 1 \end{bmatrix} \right) \in \mathbb{R}^{2 \times 2}$, with $\gamma > 1$.

5.4.2 Interpolation

Like the basic idea of SPH formation in Sect. 3.2.2, we find the neighbour transitions within a specific distance, and then we interpolate them by substituting the transition function T in (9) with a new transition function $\bar{T} : \mathcal{C}_Y \times \mathcal{A} \rightarrow \mathcal{C}_Y$ as

$$\mathbf{y}' = \bar{T}(\mathbf{y}, \boldsymbol{\alpha}) = \frac{\sum_i \bar{T}_a(\mathbf{y}_i, \boldsymbol{\alpha}) W(\text{sim}_X(\bar{\mathbf{x}}, \bar{\mathbf{x}}_i), h_y)}{\sum_i W(\text{sim}_X(\bar{\mathbf{x}}, \bar{\mathbf{x}}_i), h_y)} \quad (13)$$

and

$$\bar{T}_a(\mathbf{y}, \boldsymbol{\alpha}) = \frac{\sum_j T(\mathbf{y}, \boldsymbol{\alpha}_j) W(\text{sim}_A(\boldsymbol{\alpha}, \boldsymbol{\alpha}_j), h_a)}{\sum_j W(\text{sim}_A(\boldsymbol{\alpha}, \boldsymbol{\alpha}_j), h_a)}, \quad (14)$$

where the kernel function $W : \mathbb{R}^{\geq 0} \times \mathbb{R}^+ \rightarrow \mathbb{R}^{\geq 0}$ and the kernel ranges $h_y, h_a \in \mathbb{R}$ have been used. The denominator of the previous expressions is used for normalization purposes, similarly to the SPH formation in Sect. 3.2.2. Notice that $\mathbf{y} = [\bar{\mathbf{x}}^T \mathbf{0}_3^T]^T$.

After execution of the action by the robot, if the deformed status is the same or similar to the expected one, then the following action will be executed sequentially. Otherwise, a new step by the planner is required.

6 Path Generation for a Rolling Pin

One of the particular features in the proposed planning framework is the independent planning for the tool itself. In contrast, most of the other planning frameworks integrate the planning of the tool and the planning of the robot manipulator. There are benefits and drawbacks to this planning separation. The separation makes the robot manipulation planning manageable. At the same time, there is a need to treat the infeasible action sequences that are a problem when the robot manipulator generates its action sequences to follow the trajectory of the tool. In this application, the preference of simplicity of the manipulation planning makes us separate the planning for a tool from the planning for the robot manipulator.

From the previous sections, the actions to stretch the dough towards the desired shape are generated. A possible sequence of actions is something like: stretch forward first, then stretch 30° , and so on, all with specified heights from the plate. There are two issues for generating a proper action sequence for the rolling pin: the generation of the action itself (called primitive action) and the connection to the following action (called connecting action).

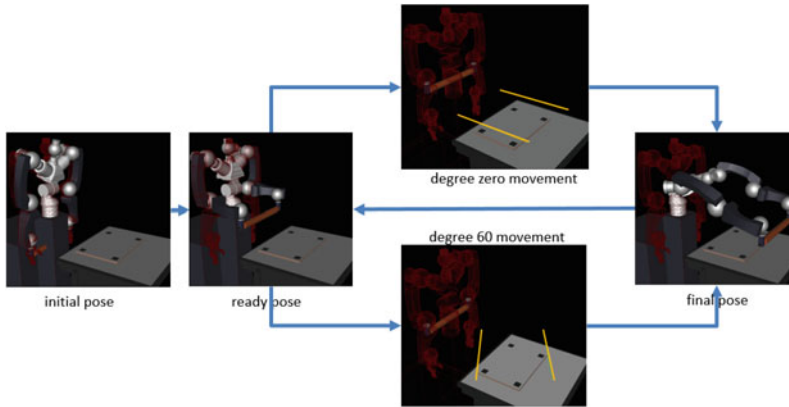


Fig. 8 Example of rolling-pin planning for two given actions on the dough

The primitive action is direct and intuitive. We must keep the desired height of the rolling pin from the plate and try to act at the specified angle by passing through the dough's center. The connecting action is instead artificial and specific to this application. First, two proper poses for the rolling pin are defined, namely the ready pose and the final pose. Then, a routine is repeated, starting with a connecting action from the initial pose to the ready pose until the generated action is completed. The routine steps are as follows: (a) doing a connecting action from the ready pose to the initial pose of a primitive action; (b) doing the primitive action; (c) doing a connecting action from the final pose of the primitive action to the defined final pose; (d) doing the connecting action from the defined final pose to the ready pose. This routine is capable of covering all generated actions from the previous dough planning.

There is an example of this process in Fig. 8 with two given actions from the previous dough planning, that is, a zero degree movement and a 60° movement. The generated action sequence consists of (1) a connecting action from the initial pose to the ready pose; (2) a connecting action from the ready pose to the initial pose of the zero-degree movement; (3) the primitive action related to the zero-degree movement; (4) a connecting action from the final pose of the zero-degree action to the final pose; (5) a connecting action from the final pose to the ready-pose; (6) a connecting action from the ready pose to the initial pose of the 60° movement; (7) the primitive action related to the 60° movement; (8) a connecting action from the final pose of the 60° movement to the final pose; (9) a connecting action from the final pose to the ready pose. The yellow lines indicate the trajectories of two holding points of the rolling pin while doing a primitive action.

The left figure in Fig. 9 shows the trajectory of the steps above from 1 to 5. The right figure in Fig. 9 shows the trajectory of the steps above from 6 to 9, in which the yellow lines and the red lines are those of the primitive and connecting actions, respectively.

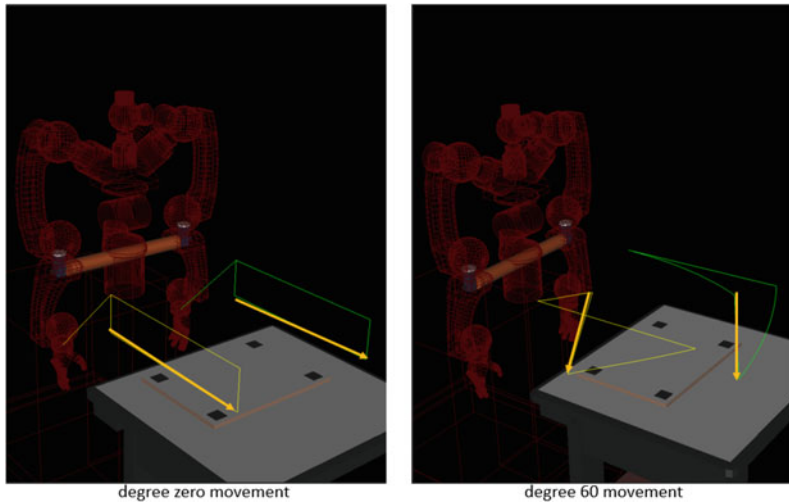


Fig. 9 Example trajectories of two holding points of a rolling pin for a given action on the dough. On the left, a zero-degree action. On the right, a 60° action. The yellow and red lines indicate the trajectories of the primitive and the connecting actions, respectively

7 Inverse Kinematics for the RoDyMan Robot

RoDyMan is the employed robot platform that consists of a mobile base, a 2-DoFs torso, a 2-DoFs neck, and two 7-DoFs arms. This process step aims to make the arms grasp the rolling pin properly and solve the inverse kinematics problem to plan the joint movements.

Many inverse kinematics algorithms are well established in the literature, as those using the LM algorithm or the damping least-squares method [39]. The employed method follows the closed-loop inverse kinematics algorithm with redundancy management explained in [39]. Redundancy is exploited to avoid unnatural postures of the robot.

Figure 10 shows the ready pose of RoDyMan before and after an action. Figure 10b shows the unnatural behaviour that can be avoided through a proper redundancy management, as shows in Fig. 11. Details are omitted here for brevity since this part is well established in the literature and does not bring any new insight into the problem faced by this chapter.

8 Simulations

For simulation purposes, the employed system consists of an Intel®Core™ i7-6500U CPU@2.50 GHz, Memory 8.0 GB with Windows®10 x64 operating system. We used

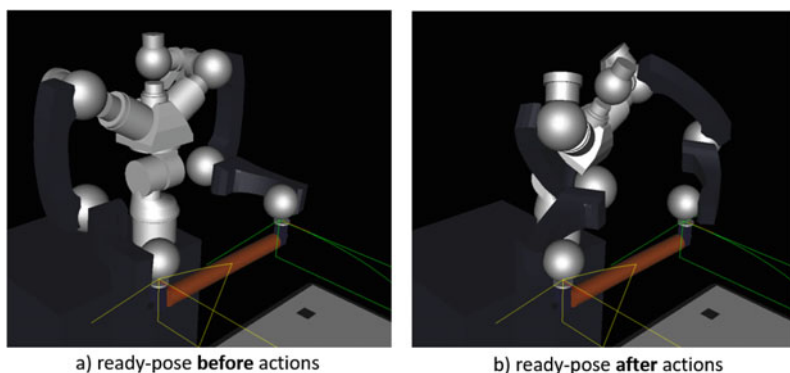


Fig. 10 Ready pose of RoDyMan before and after an action

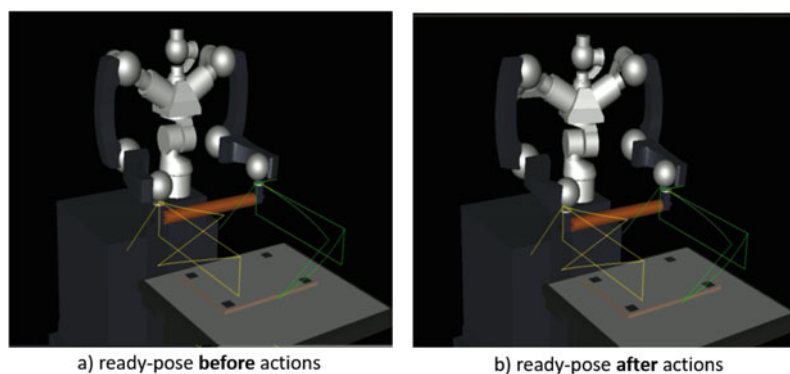


Fig. 11 Improved ready pose of RoDyMan before and after an action thanks to the redundancy management

the Microsoft Foundation Class (MFC) library of Microsoft®, the Qt library,¹ Boost,² Eigen,³ and OpenCV⁴ for 2D graphic or OpenSceneGraph⁵ for 3D graphic libraries based on C++11 programming language. Additionally, Houdini™ of SideFX® and Blender™ were used to reconstruct the mesh from particles and for graphical rendering, respectively. 3D reconstruction is done with VisualSFM⁶ and MeshLab.⁷

¹ <https://www.qt.io>.

² <https://www.boost.org>.

³ <http://eigen.tuxfamily.org>.

⁴ <https://opencv.org>.

⁵ <http://www.openscenegraph.org>.

⁶ <http://ccwu.me/vsfm/>.

⁷ <http://www.meshlab.net/>.

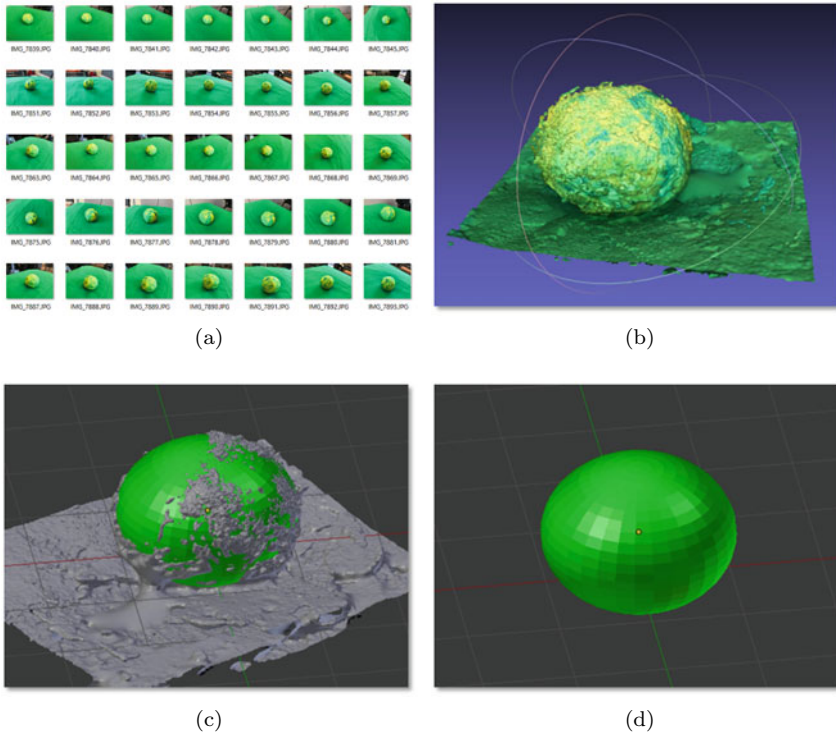


Fig. 12 Measuring of a real dough shape at initial status. **a** pictures with various views for real dough; **b** 3D reconstruction; **c** matching of the reconstructed shapes with a sphere; **(d,h)** the final matched shapes

8.1 Modelling of a Deformable Object

Even though a model for a deformable object like a pizza dough is designed in Chap. 3.1, the deformable object's properties like viscosity and elasticity vary case by case. Therefore, there is the need to tuning the model's parameters to fit the target object. Figure 12 shows how to measure a real deformable object. During preliminary experiments, a toy clay was used instead of pizza dough for convenience.

The process during the preliminary experiments has been carried out as follows. At first, the measure of the deformable object's surface and the reconstruction of its shape in 3D virtual space, at the initial and deformed statuses, have been carried out. Afterwards, an SPH-based object model is made following the reconstructed shape of the initial and deformed object. Finally, by applying the various actions on the SPH-based object model for the initial object, the best matching parameters for the model are found, which generates the SPH-based object model that is close to the reconstructed shape of the deformed object.

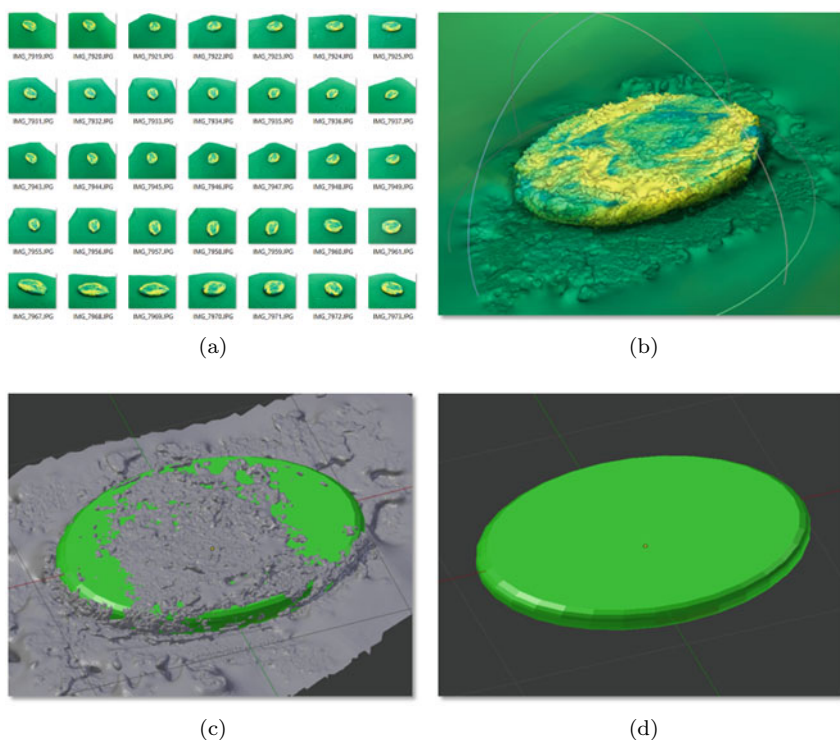


Fig. 13 Measuring of a real dough shape at initial status. **a** pictures with various views for real dough; **b** 3D reconstruction; **c** matching of the reconstructed shapes with a disk; **d,h** the final matched shapes

The reconstruction of a deformable object in the 3D virtual space at the initial status and the deformed status, respectively, are shown in Figs. 12 and 13. The pictures in Fig. 12 deals with a ball shape, representing the pizza dough before any deformation. The pictures in Fig. 13 show a disk shape after the deformation actuated by the rolling pin. To measure the shape of the object, a structure from motion method is used, which gathers some pictures (Figs. 12a and 13a) with various views and generates a 3D model (Figs. 12b and 13b). Usually, the generated 3D model is very rough and too complex. Hence, it needs post-processing to smooth the surface and remove the outliers. A simpler ball mesh-model (Fig. 12c) and cylinder mesh-model (Fig. 13c) are used to match the generated 3D model as close as possible, and the final reconstructed 3D model (Figs. 12d and 13d) are fixed, respectively.

The process of finding the best matching parameters for the deformable object model is shown in Fig. 14. Various deformed shapes are generated with various parameters: among them, we must find the best matching shape and its parameter. The matching process is done offline.

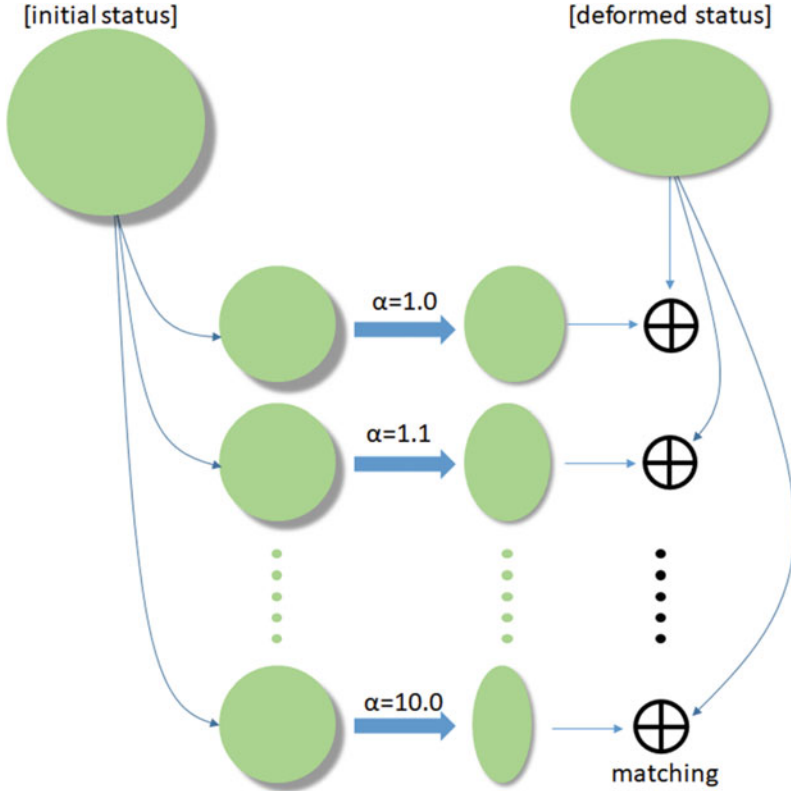


Fig. 14 Schematic diagram for finding the parameters of the deformable object model. α is just an example of a parameter for a SPH-based model: name and values are different depending on the model

8.2 Pizza Dough Transition Look-up-table

The transition LUT is used to speed up the planning algorithm, which is generated using various statuses of the pizza dough and various actions. However, it is not easy, and it needs much time to have experiments with real pizza dough. Therefore, using the deformable object model obtained before is more efficient than a real deformable object.

The previous section defines the model for a deformable object and its parameters obtained from real experiments. We had experiments with SPH particle radius of $1.25 \cdot 10^{-3}$ m, a pizza dough density of 1.276 kg/m^3 , a solid density of $200 \cdot 10^3 \text{ kg/m}^3$, an IISPH [40] for an incompressible fluid, and a viscosity coefficient of 250 kg/ms . Based on this model, the transition LUT is made by simulating the deformation from several configurations of the pizza dough and with various actions on the object. In particular, the action set $\mathcal{A}$ consists of seven heights, from $8 \cdot 10^{-3}$ m

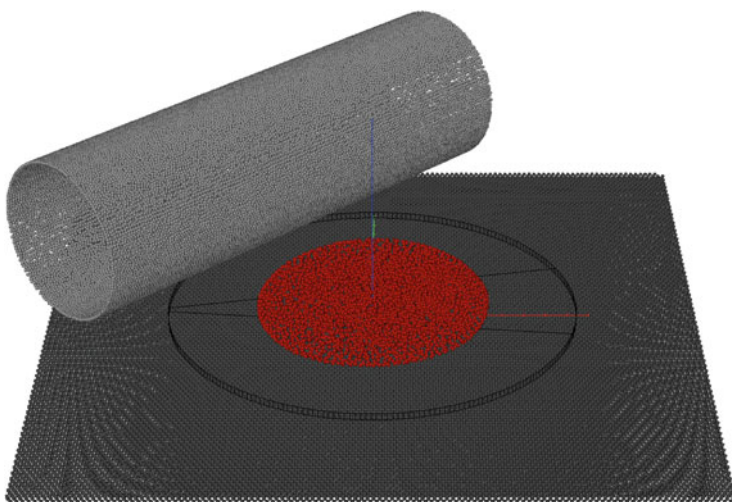


Fig. 15 SPH-based transition simulation for the pizza dough (red object) with a rolling pin (grey object)

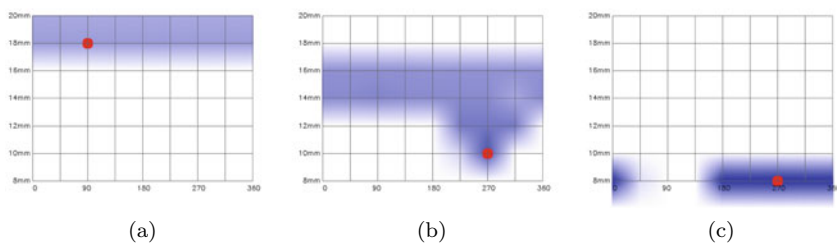
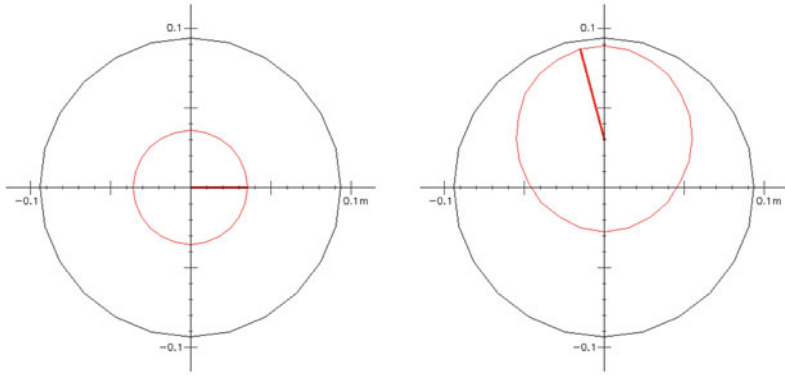


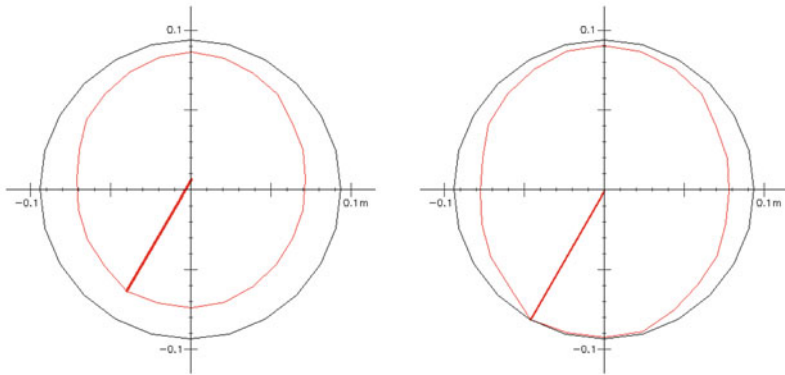
Fig. 16 LUT method. The x -axis and the y -axis indicate the angle and the height of a rolling pin motion, respectively. The white colour means lower score, while the darker blue colour means higher score. Red ball is the highest score action

to $20 \cdot 10^{-3}$ m spanned each $2 \cdot 10^{-3}$ m, and eight directions, from zero degrees to 180° , spanned each 45° and considering both forward and backward motions.

A simulation is shown in Fig. 15 where the pizza dough is represented by the red particles and the rolling pin by the green ones. An example where the pizza dough is stretched is depicted in Figs. 16 and 17. The former shows the transition LUT for sequential statuses of the pizza dough with height and angular table, while the latter shows the occupancy of the pizza dough for each status.



(a) Start shape of a dough placed in center (b) The shape after applying the highest scored action in Fig. 9.15a



(c) The shape after applying the highest scored action in Fig. 9.15b (d) The shape after applying the highest scored action in Fig. 9.15c

Fig. 17 Shapes of the current dough (red) and its target (black). The red line indicates the angle of the first bin in the dough's state

9 Discussion and Conclusion

This chapter explained a planning method to stretch a pizza dough with a rolling pin actuated by a robotic system. Based on the perception chapters, the deformable object is modelled, and the deformation information is used to plan the stretching actions on the pizza dough to reach the desired shape. An object recognition algorithm, a method to model deformable objects with high viscosity, the definition of the status of the pizza dough, the definition of the actions through the rolling pin, an inverse kinematics algorithm for the robot have been integrated to achieve the sought goal. Experiments were carried out to identify the parameters of the pizza dough. The

stretching actions were planned offline thanks to a LUT database. Future experiments will definitely validate the proposed approach.

References

1. H. Wieser, Chemistry of gluten proteins. *Food Microbiol.* **24**(2), 115–119 (2007)
2. F. Van Bockstaele, I. De Leyn, M. Eeckhout, K. Dewettinck, Rheological properties of wheat flour dough and the relationship with bread volume. I. Creep-recovery measurements. *Cereal Chem. J.* **85**(6), 753–761 (2008)
3. G.M. Campbell, C.D. Rielly, P.J. Fryer, P.A. Sadd, Measurement and interpretation of dough densities. *Cereal Chem.* **70**(5), 517–521 (1993)
4. M. Charalambides, L. Wanigasooriya, S.M. Goh, J.G. Williams, Viscoelasticity of bread dough, in *2005 SEM Annual Conference and Exposition on Experimental and Applied Mechanics*, pp. 879–888 (2005)
5. J.M. Faubion, R.C. Hosney, The viscoelastic properties of wheat flour doughs, in *Dough Rheology and Baked Product Texture* (Springer, 1990), pp. 29–66
6. E. Magaña Barajas, B. Ramírez-Wong, P.I. Torres-Chávez, I. Morales-Rosas. Use of the stress-relaxation and dynamic tests to evaluate the viscoelastic properties of dough from soft wheat cultivars, in *Viscoelasticity*, ed. by J. de Vicente (IntechOpen, 2012), pp. 259–272
7. M.R. Amjid, A. Shehzad, S. Hussain, M.A. Shabbir, M.R. Khan, M. Shoaib, A comprehensive review on wheat flour dough rheology. *Pak. J. Food Sci.* **23**, 105–123 (2013)
8. B.S. Khatkar, A.E. Bell, J.D. Schofield, The dynamic rheological properties of glutes and gluten sub-fractions from wheats of good and poor bread making quality. *J. Cereal Sci.* **22**(1), 29–44 (1995)
9. S. Uthayakumaran, M. Newberry, M. Keentok, F.L. Stoddard, F. Bekes, Basic rheology of bread dough with modified protein content and glutenin-to-gliadin ratios. *Cereal Chem. J.* **77**(6), 744–749 (2000)
10. J. Engmann, M.C. Peck, D.I. Wilson, An experimental and theoretical investigation of bread dough sheeting. *Food Bioprod. Process.* **83**(3), 175–184 (2005)
11. C. Bollain, A. Angioloni, C. Collar, Relationships between dough and bread viscoelastic properties in enzyme supplemented wheat samples. *J. Food Eng.* **77**, 665–671 (2006)
12. H. Mirsaeedghazi, Z. Eman-Djomeh, S.M.A. Mousavi, Rheometric measurement of dough rheological characteristics and factors affecting it. *Int. J. Agric. Biol.* **10**(1), 112–119 (2008)
13. S. Sofou, E.B. Muliawan, S.G. Hatzikiriakos, E. Mitsoulis, Rheological characterization and constitutive modeling of bread dough. *Rheol. Acta* **47**(4), 369–381 (2008)
14. C.R. Beverly, R.I. Tanner, Numerical analysis of three-dimensional Bingham plastic flow. *J. Nonnewton. Fluid Mech.* **42**(1), 85–115 (1992)
15. E. Mitsoulis, S.G. Hatzikiriakos, Rolling of bread dough: experiments and simulations. *Food Bioprod. Process.* **87**(2), 124–138 (2009)
16. S. Sofou, E. Mitsoulis, Calendaring of pseudoplastic and viscoplastic sheets of finite thickness. *J. Plastic Film & Sheet.* **20**, 185–222 (2004)
17. S. Tokumoto, S. Hirai, Deformation control of rheological food dough using a forming process model, in *2002 IEEE International Conference on Robotics and Automation*, pp. 1457–1464 (2002)
18. M. Higashimori, K. Yoshimoto, M. Kaneko, Active shaping of an unknown rheological object based on deformation decomposition into elasticity and plasticity, in *2010 IEEE International Conference on Robotics and Automation*, pp. 5120–5126 (2010)
19. A.-M. Cretu, P. Payeur, E.M. Petriu, Neural network mapping and clustering of elastic behavior from tactile and range imaging for virtualized reality applications. *IEEE Trans. Instrum. Meas.* **57**(9), 1918–1928 (2008)

20. Z. Wang, S. Hirai, Modeling and estimation of rheological properties of food products for manufacturing simulations. *J. Food Eng.* **102**(2), 136–144 (2011)
21. D. Berenson, Manipulation of deformable objects without modeling and simulating deformation, in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4525–4532 (2013)
22. B. Frank, C. Stachniss, N. Abdo, W. Burgard, Efficient motion planning for manipulation robots in environments with deformable objects, in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2180–2185 (2011)
23. J. Lang, An acquisition method for interactive deformable models, in *2nd International Conference on Creating, Connecting and Collaborating through Computing*, pp. 165–170 (2004)
24. C. Phillips-Grafflin, D. Berenson, A representation of deformable objects for motion planning with no physical simulation, in *2014 IEEE International Conference on Robotics and Automation*, pp. 98–105 (2014)
25. J. Smolen, A. Patriciu, Deformation planning for robotic soft tissue manipulation, in *2009 Second International Conferences on Advances in Computer-Human Interactions*, pp. 199–204 (2009)
26. R. Hartley, A. Zisserman, *Multiple View Geometry in Computer Vision* (Cambridge University Press, 2004)
27. D.G. Lowe, Object recognition from local scale-invariant features, in *International Conference on Computer Vision*, pp. 1150–1157 (1999)
28. H. Bay, A. Ess, T. Tuytelaars, L. Van Gool, SURF: speeded up robust features, in *Computer Vision and Image Understanding*, pp. 346–359 (2008)
29. E. Rosten, T. Drummond, Machine learning for high-speed corner detection. *Eur. Conf. Comput. Vis.* **1**, 430–443 (2006)
30. H. Kato, M. Billinghurst, Marker tracking and HMD calibration for a video-based augmented reality conferencing system, in *2nd IEEE and ACM International Workshop on Augmented Reality*, pp. 85–94 (1999)
31. A. Bey, R. Chaine, R. Marc, G. Thibault, S. Akkouché, Reconstruction of consistent 3D CAD models from point cloud data using a priori CAD models. *ISPRS - Int. Arch. Photogram. Remote Sens. Spatial Inf. Sci.* **3812**, 289–294 (2011)
32. M. Berger, A. Tagliasacchi, L.M. Seversky, P. Alliez, J.A. Levine, A. Sharf, C.T. Silva, State of the art in surface reconstruction from point clouds, in *Eurographics 2014 - State of the Art Reports*, ed. by S. Lefebvre, M. Spagnuolo (The Eurographics Association, 2014)
33. R.C. Gonzalez, R.E. Woods, *Digital Image Processing* (Prentice-Hall, 2006)
34. J. Canny, A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-8**(6), 679–698 (1986)
35. D.G. Lowe, Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **60**(2), 91–110 (2004)
36. J. Matas, O. Chum, M. Urban, T. Pajdla, Robust wide-baseline stereo from maximally stable extremal regions. *Image Vis. Comput.* **22**(10), 761–767 (2004)
37. S. Leutenegger, M. Chli, R.Y. Siegwart, BRISK: Binary Robust Invariant Scalable Keypoints, in *2011 International Conference on Computer Vision*, pp. 2548–2555 (2011)
38. B. Frank, M. Becker, C. Stachniss, W. Burgard, M. Teschner, Efficient path planning for mobile robots in environments with deformable objects, in *2008 IEEE International Conference on Robotics and Automation*, pp. 3737–3742 (2008)
39. B. Siciliano, L. Sciacivco, L. Villani, G. Oriolo, *Robotics: Modelling, Planning and Control*. (Springer, London, UK, 2009)
40. M. Ihmsen, J. Cornelis, B. Solenthaler, C. Horvath, M. Teschner, Implicit incompressible SPH. *IEEE Trans. Vis. Comput. Gr.* **20**(3), 426–435 (2014)