Chapter 8
A coordinate-free framework for robotic pizza tossing and catching

Aykut C. Satici, Fabio Ruggiero, Vincenzo Lippiello, Bruno Siciliano

Abstract In this chapter, autonomous pizza tossing and catching is achieved. Under the assumption that some fingers grasp the pizza dough with soft contact, the grasp constraints are formulated and used to derive the individual and combined Euler-Lagrange dynamic equations of motion of the robotic manipulator and the dough. In particular, the dynamics of the dough is a modified version of the rigid-body dynamics, taking into account the change of inertia due to its deformation. Through these mathematical models, the two control problems of tossing and catching are formulated. For the tossing phase, an exponentially convergent controller that stabilizes a desired velocity of the dough as it leaves the fingers, is derived. On the other hand, to catch the dough, an optimal trajectory for the end-effector of the robotic manipulator is generated. Finally, the control laws to make the optimal trajectory exponentially attractive are derived. The developed theory is demonstrated with an elaborate simulation of the tossing and catching phases. This chapter is based on the works presented in [269].

Aykut C. Satici
Boise State University, 1910 University Drive, Boise, ID 83709, USA, e-mail: aykutsatici@boisestate.edu

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

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

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

Table 8.1: Main symbols used in this chapter.

Definition	Symbol
Frame at the i-th contact point moving with the dough	$\mathcal{C}_i$
Frame at the i-th contact point moving with the i-th finger	$\mathcal{F}_i$
Frame at the tool's center	$\mathcal{H}$
Frame at the dough's center of mass	$\mathcal{O}$
Standard unit vector	$\mathbf{e}_3 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$
Velocity (twist) of $\mathcal{C}_i$ with respect to $\mathcal{F}_i$, expressed in body coordinates	$\mathbf{v}^b_{f_i c_i} \in \mathbb{R}^6$
Homogeneous transformation between a generic frame $\mathcal{A}$ and a frame $\mathcal{B}$	$\mathbf{T}_{ab} \in SE(3)$
Velocity (twist) of $\mathcal{H}$ with respect to $\mathcal{W}$	$\mathbf{v}_{sh} \in \mathbb{R}^6$
Velocity (twist) of $\mathcal{O}$ with respect to $\mathcal{W}$	$\mathbf{v}_{so} \in \mathbb{R}^6$
Body manipulator Jacobian	$\mathbf{J}^b_{sh} = \mathbf{J} \in \mathbb{R}^{6\times n}$
Vector of joint angles	$\boldsymbol{\theta} = \begin{bmatrix} \theta_1 & \cdots & \theta_n \end{bmatrix}^T \in \mathbb{R}^n$
Radius of the pizza dough	$r > 0$
Height of the pizza dough's geometric center	$h > 0$
Orientation of $\mathcal{O}$ with respect to $\mathcal{W}$	$\mathbf{R}_{so} \in SO(3)$
Angular velocity of $\mathcal{O}$ with respect to $\mathcal{W}$ expressed in the body frame $\mathcal{O}$	$\boldsymbol{\omega}_o \in \mathbb{R}^3$
Inertia tensor of the pizza dough	$\mathbf{I}_o \in \mathbb{R}^{3\times 3}$
External torque applied to the pizza dough expressed in $\mathcal{O}$	$\boldsymbol{\upsilon} \in \mathbb{R}^3$.
Gravity acceleration	$\gamma \simeq 9.81$ m/s^2
Mass of the pizza dough	$m_o > 0$
Linear velocity of $\mathcal{O}$ with respect to $\mathcal{W}$ expressed in the body frame $\mathcal{O}$	$\mathbf{v} \in \mathbb{R}^3$
External force applied to the pizza dough expressed in $\mathcal{O}$	$\mathbf{f} \in \mathbb{R}^3$.
Mass matrix of the manipulator	$\mathbf{M}_r(\boldsymbol{\theta}) \in \mathbb{R}^{n\times n}$
Coriolis matrix of the manipulator	$\mathbf{C}_r(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) \in \mathbb{R}^{n\times n}$
Gravitational terms of the manipulator	$\mathbf{n}(\boldsymbol{\theta}) \in \mathbb{R}^n$
Manipulator joint torques	$\boldsymbol{\tau} \in \mathbb{R}^n$
Mass matrix of the pizza dough	$\mathbf{M}_o \in \mathbb{R}^{6\times 6}$
Wrench acting on the pizza dough	$\mathbf{f}_o \in \mathbb{R}^6$
Coriolis and gravity term of the pizza dough	$\boldsymbol{\phi}_o \in \mathbb{R}^6$
Contact force at the i-th finger	$\mathbf{f}_{c_i} \in \mathbb{R}^4$

8.1 Introduction

The problem of tossing and catching a pizza dough is a procedure that is frequently dexterously performed by human pizza makers. There are at least three important reasons why tossing the dough during the preparation of the pizza is attractive: (*i*) the dough is stretched to the desired size, (*ii*) the dough naturally assumes a configuration that is thicker at the ends and thinner in the middle, and (*iii*) as the spinning dough freely falls, the outside

of the dough dries, making it crunchy in the outside but light in the middle. The pizza maker is trained to perform a streamlined hand motion to toss and catch the dough, and we are aiming to achieve a similar feat with a humanoid robot.

There are several challenges to achieve this kind of manipulation. The fact that the object that we are trying to manipulate is deformable complicates the several previously well-studied problems, such as catching or grasping [272], motion estimation [172, 58] and manipulation of rigid objects [216]. Furthermore, such nonprehensile manipulation tasks typically require high-speed sensing and control action [182, 265]. While the deformation is beneficial for the grasping phase as it introduces more surface area where normal force may be applied, motion estimation is more problematic because the dynamic model of the rigid body is no longer valid. In this chapter, we estimate the equations of motion of the dough by those of a rigid body whose mass is constant but whose inertia varies due to its varying shape. We assume the ability to receive the shape information via a vision sensor, which enables us to estimate the motion of the dough while it is in the air. The specific literature for robotic tossing and catching of pizza dough is limited. In [175] the authors have studied how different forms of tossing might yield different desired behaviors.

With perfect knowledge of the motion of the dough, we can generate optimal trajectories to intercept it. The generation of optimal trajectories in $SE(3)$ has been studied in [348, 70]. Since we only estimate this motion, we repeat the optimal trajectory generation as we are fed with new sensor information. The optimal trajectories are generated to match the initial position, velocity, acceleration, and final velocity and accelerations. Therefore, it is at least thrice continuously differentiable. An optimal trajectory whose initial and final accelerations are desired to be prescribed has to satisfy a sixth-order BVP. We generate such a BVP using the necessary conditions for a path to minimize a convex combination of the jerk and acceleration functionals. While minimizing the jerk functional reduces the vibrations in the structure of the robotic manipulator, minimizing the acceleration functional reduces the total amount of energy expended during the catching motion [32]. We can determine which aspect to emphasize by choosing the convex coefficients suitably. As soon as the first optimal trajectory is generated, the robot is instructed to move along this trajectory and update its path as new measurements come in.

We derive control laws for both the tossing and catching phases. The control law for tossing the object makes sure the object has the desired velocity as it leaves the fingers, while the control law for catching moves the manipulator's end-effector along the previously generated optimal trajectory. The robotic manipulator is chosen with a redundant, S-R-S topology to mimic the human arm with 7 DoFs. The extra DoF is exploited to flow to the configuration with a maximum manipulability index along the nullspace of the manipulator Jacobian.

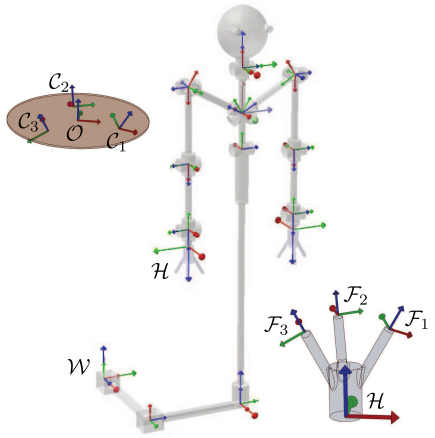


Fig. 8.1: RoDyMan Prototype: frames and joint axes.

Finally, we demonstrate the developed theory with an elaborate tossing and catching simulation.

8.2 Grasp constraints

For the formulation of the contact constraint, we follow [216]. We assume that the fingers contact the dough softly and that the contact points are fixed. The contact frame $\mathcal{C}_i$ moves with the object, while the finger frame $\mathcal{F}_i$ moves with the hand. The system and the frames are schematically displayed in Fig. 8.1.

The soft contact condition yields the equation $\mathbf{B}_{c_i}^T \mathbf{v}_{f_i c_i}^b = \mathbf{0}_4$, for each finger enumerated by the index $i \in \mathfrak{I} := \{1, \ldots, k\}$, where $\mathbf{B}_{c_i} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{O}_3 & \mathbf{e}_3 \end{bmatrix} \in \mathbb{R}^{6\times 4}$. We adopt the convention that when the superscript over a quantity is omitted, it is expressed in the body coordinate frame, "b". After some algebra, we find that

$$\mathbf{b}_{f_i c_i} = -\mathrm{Ad}_{\mathbf{T}_{hc_i}}^{-1} \mathbf{v}_{sh} + \mathrm{Ad}_{\mathbf{T}_{oc_i}}^{-1} \mathbf{v}_{so}.$$

where $\mathrm{Ad} : SE(3) \to \mathrm{Aut}(\mathfrak{se}(3))$ is the adjoint operator on $SE(3)$ whose range is the set of isomorphisms of $\mathfrak{se}(3)$.Therefore, we have the contact constraint

$$\mathbf{B}_{c_i}^T \left(-\mathrm{Ad}_{\mathbf{T}_{hc_i}}^{-1} \mathbf{v}_{sh} + \mathrm{Ad}_{\mathbf{T}_{oc_i}}^{-1} \mathbf{v}_{so} \right) = \mathbf{0}_4,$$

where the $\vee$ and $\wedge$ operators denote the isomorphism from $\mathfrak{se}(3)$ to $\mathbb{R}^6$ and its inverse, respectively. Because $\mathbf{v}_{sh} = \mathbf{J}\dot{\boldsymbol{\theta}}$, we have

$$\mathbf{B}_{c_i}^T \mathbf{v}_{f_i c_i} = \mathbf{B}_{c_i}^T \left(-\mathrm{Ad}_{\mathbf{T}_{hc_i}}^{-1} J\dot{\boldsymbol{\theta}} + \mathrm{Ad}_{\mathbf{T}_{oc_i}}^{-1} \mathbf{v}_{so} \right) = \mathbf{0}_4. \tag{8.1}$$

Defining the following constraint matrices

$$\mathbf{B}_c = \begin{bmatrix} \mathbf{B}_{c_1} & \mathbf{O}_{6\times 4} & \cdots & \mathbf{O}_{6\times 4} \\ \mathbf{O}_{6\times 4} & \mathbf{B}_{c_2} & \cdots & \mathbf{O}_{6\times 4} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{O}_{6\times 4} & \mathbf{O}_{6\times 4} & \cdots & \mathbf{B}_{c_k} \end{bmatrix} \in \mathbb{R}^{6k\times 4k},$$

$$\mathbf{A} = \mathbf{B}_c^T \begin{bmatrix} \mathrm{Ad}_{\mathbf{T}_{oc_1}^{-1}} & -\mathrm{Ad}_{\mathbf{T}_{hc_1}^{-1}} \mathbf{J} \\ \mathrm{Ad}_{\mathbf{T}_{oc_2}^{-1}} & -\mathrm{Ad}_{\mathbf{T}_{hc_2}^{-1}} \mathbf{J} \\ \vdots & \vdots \\ \mathrm{Ad}_{\mathbf{T}_{oc_k}^{-1}} & -\mathrm{Ad}_{\mathbf{T}_{hc_k}^{-1}} \mathbf{J} \end{bmatrix} \in \mathbb{R}^{4k\times(6+n)},$$

allows us to rewrite the contact constraints (8.1) in the Pfaffian form $\mathbf{A}\mathbf{v} = \mathbf{0}_{4k}$, where $\mathbf{v} = \begin{bmatrix} \mathbf{v}_{so}^T & \dot{\boldsymbol{\theta}}^T \end{bmatrix}^T \in \mathbb{R}^{6+n}$. We call this equation the DCC

$$\mathbf{B}_{c_i}^T \left(\left[\mathbf{v}_{hc_i},\ \mathrm{Ad}_{\mathbf{T}_{hc_i}}^{-1} \mathbf{v}_{sh} \right] - \left[\mathbf{v}_{oc_i},\ \mathrm{Ad}_{\mathbf{T}_{oc_i}}^{-1} \mathbf{v}_{so} \right] - \mathrm{Ad}_{\mathbf{T}_{hc_i}}^{-1} \dot{\mathbf{v}}_{sh} + \mathrm{Ad}_{\mathbf{T}_{oc_i}}^{-1} \dot{\mathbf{v}}_{so} \right) = \mathbf{0}_4.$$

This can be simplified by considering the following

$$\mathbf{v}_{hc_i} = -\mathrm{Ad}_{\mathbf{T}_{hc_i}^{-1}} \mathbf{v}_{sh} + \mathrm{Ad}_{\mathbf{T}_{oc_i}^{-1}} \mathbf{v}_{so} + \mathbf{v}_{oc_i},$$

so that

$$\left[\mathbf{v}_{hc_i},\ \mathrm{Ad}_{\mathbf{T}_{hc_i}^{-1}} \mathbf{v}_{sh} \right] = \left[\mathrm{Ad}_{\mathbf{T}_{oc_i}^{-1}} \mathbf{v}_{so},\ \mathrm{Ad}_{\mathbf{T}_{hc_i}^{-1}} \mathbf{v}_{st} \right] + \left[\mathbf{v}_{oc_i},\ \mathrm{Ad}_{\mathbf{T}_{hc_i}^{-1}} \mathbf{v}_{st} \right].$$

Plugging this back to the DCC, we get

$$\mathbf{B}_{c_i}^T \left(\left[\mathrm{Ad}_{\mathbf{T}_{oc_i}}^{-1} \mathbf{v}_{so},\ \mathrm{Ad}_{\mathbf{T}_{hc_i}}^{-1} \mathbf{v}_{sh} \right] + \left[\mathbf{v}_{oc_i},\ \mathrm{Ad}_{\mathbf{T}_{hc_i}}^{-1} \mathbf{v}_{sh} \right] - \left[\mathbf{v}_{oc_i},\ \mathrm{Ad}_{\mathbf{T}_{oc_i}}^{-1} \mathbf{v}_{so} \right] - \mathrm{Ad}_{\mathbf{T}_{hc_i}}^{-1} \dot{\mathbf{v}}_{st} + \mathrm{Ad}_{\mathbf{T}_{oc_i}}^{-1} \dot{\mathbf{v}}_{so} \right) = \mathbf{0}_4.$$

Now, since the contact point is fixed in the object frame of reference, we have $\mathbf{v}_{oc_i}^b \equiv \mathbf{0}_6$ for all $i \in \mathfrak{I}$. Therefore, the second and the third brackets in the expression above drop, yielding simplified differential contact constraint

$$\mathbf{B}_{c_i}^T \left(\left[\mathrm{Ad}_{\mathbf{T}_{oc_i}^{-1}} \mathbf{v}_{so},\ \mathrm{Ad}_{\mathbf{T}_{hc_i}^{-1}} \mathbf{v}_{sh} \right] - \mathrm{Ad}_{\mathbf{T}_{hc_i}}^{-1} \dot{\mathbf{v}}_{sh} + \mathrm{Ad}_{\mathbf{T}_{oc_i}}^{-1} \dot{\mathbf{v}}_{so} \right) = \mathbf{0}_4.$$

Invoking the relationship between the joint velocities and the end-effector velocity, we get

$$\mathbf{B}_{c_i}^T \left(\left[\mathrm{Ad}_{\mathbf{T}_{oc_i}^{-1}} \mathbf{v}_{so},\ \mathrm{Ad}_{\mathbf{T}_{hc_i}^{-1}} \mathbf{v}_{sh} \right] - \mathrm{Ad}_{\mathbf{T}_{hc_i}}^{-1} \dot{\mathbf{J}}\dot{\boldsymbol{\theta}} - \mathrm{Ad}_{\mathbf{T}_{hc_i}}^{-1} \mathbf{J}\ddot{\boldsymbol{\theta}} + \mathrm{Ad}_{\mathbf{T}_{oc_i}}^{-1} \dot{\mathbf{v}}_{so} \right) = \mathbf{0}_4. \tag{8.2}$$

8.3 Kinematics

In order to map the tool velocities and forces to the joint velocities and torques, we need to derive the kinematics of the manipulator and the object. Object kinematics are well-known rigid body kinematics. To manipulate the pizza dough, we use a 7 DoF serial robotic manipulator. We want this manipulator to have similar characteristics as the human arm. To that end, we choose the topology of the manipulator to be of an S-R-S robot. We are developing a humanoid robotic torso mounted on an omnidirectional mobile platform, called RoDyMan, in our laboratory and the S-R-S manipulator corresponds to one of the arms of RoDyMan robot, whose frontal view is shown in Figure 8.1. We have chosen to perform the tossing and catching operations with the right arm. This viewpoint lets us identify the first joint with the shoulder, the second joint with the elbow, and the last joint with the wrist.

Again, we follow [216] for the derivation of the kinematics of the S-R-S manipulator. This procedure involves using the exponential coordinates for the end-effector position and therefore maintains the inherent geometric features of the manipulator. The joint axes, expressed in $\mathcal{W}$, are given by $\boldsymbol{\xi}_i = \begin{bmatrix} -\boldsymbol{\omega}_i \times \mathbf{p}_i \\ \boldsymbol{\omega}_i \end{bmatrix}$, for each joint $i \in \{1, \ldots, 7\}$, where, $\boldsymbol{\omega}_i \in \mathbb{R}^3$ is the axis of rotation of each of the i^{th} joint expressed in $\mathcal{W}$ and $\mathbf{p}_i \in \mathbb{R}^3$ is any point on this axis of rotation, written in $\mathcal{W}$.

The position forward kinematics is then given by

$$\mathbf{T}_{sh}(\theta) = e^{\hat{\boldsymbol{\xi}}_1 \theta_1} e^{\hat{\boldsymbol{\xi}}_2 \theta_2} e^{\hat{\boldsymbol{\xi}}_3 \theta_3} e^{\hat{\boldsymbol{\xi}}_4 \theta_4} e^{\hat{\boldsymbol{\xi}}_5 \theta_5} e^{\hat{\boldsymbol{\xi}}_6 \theta_6} e^{\hat{\boldsymbol{\xi}}_7 \theta_7} \mathbf{T}_{sh}(0), \tag{8.3}$$

where $\mathbf{T}_{sh}(0)$ is the initial pose of the end-effector with respect to the base frame and the exponential mapping $exp : \mathfrak{se}(3) \to SE(3)$ is as defined in [216].

The relationship between the joint velocities and the end-effector velocities is given by the body manipulator Jacobian, $\mathbf{J}^b \in \mathbb{R}^{6 \times n}$. This is found by the following computation

$$\mathbf{J}^b = \mathbf{J}_{sh}^b = \left[\boldsymbol{\xi}_1^\dagger \ \ \boldsymbol{\xi}_2^\dagger \ \ \boldsymbol{\xi}_3^\dagger \ \ \boldsymbol{\xi}_4^\dagger \ \ \boldsymbol{\xi}_5^\dagger \ \ \boldsymbol{\xi}_6^\dagger \ \ \boldsymbol{\xi}_7^\dagger \right],$$

where

$$\boldsymbol{\xi}_i^\dagger = \mathrm{Ad}^{-1}_{\left(e^{\hat{\boldsymbol{\xi}}_i \theta_i} \cdots e^{\hat{\boldsymbol{\xi}}_7 \theta_7} \mathbf{T}_{sh}(0)\right)} \boldsymbol{\xi}_i,$$

with $i = 1, \ldots, 7$. Given the joint velocities, we can then find the end-effector velocities by the relation $\mathbf{v}_{sh}^b = \mathbf{J}^b \dot{\boldsymbol{\theta}}$. Note that away from the singularities, $\mathbf{J}^b$ has rank 6. Given an end-effector velocity, there are infinitely many joint velocities that supply this end-effector velocity. To achieve the desired end-effector velocity, we need a redundancy resolution technique to decide which joint velocity we want to supply.

8.4 Dynamics

We have to model the natural motion of the pizza dough, which is a deformable object. We assume the dough has a circular shape with a radius that changes dues to the various S^1-symmetric forces. The height of the geometric center of the pizza may change due to fluid dynamical forces on the dough.

8.4.1 Variable inertia rigid body orientation dynamics

Although the mass of the dough remains constant, its moment of inertia changes due to its shape's change. In turn, this variation affects the conventional Euler-Lagrange equations of motion of the rigid body. In this section, we derive the appropriate equations of motion for the orientation dynamics of the dough in the body frame.

The Lagrangian of the pizza dough is given by

$$L(\mathbf{R}_{so}, \dot{\mathbf{R}}_{so}) = -\frac{\det(\mathbf{I}_o)}{2} \operatorname{tr}\left[\left(\mathbf{I}_o^{-1} \hat{\boldsymbol{\omega}}_o\right)^2\right] = \frac{1}{2} \boldsymbol{\omega}_o^T \mathbf{I}_o \boldsymbol{\omega}_o, \tag{8.4}$$

where $\det()$ and $\operatorname{tr}()$ indicates the determinant and the trace of a square matrix, respectively. The dependence of the Lagrangian (8.4) on $\mathbf{R}_{so}$ and $\dot{\mathbf{R}}_{so}$ stems from the orientation kinematics, namely, from the equation $\boldsymbol{\omega}_{so} = \mathbf{R}_{so}^T \dot{\mathbf{R}}_{so}$.

We would like to calculate the conventional Euler-Lagrange equations for the rigid body with a variable moment of inertia. It is well-known that the rotational dynamics is symmetric with respect to the action of the group $SO(3)$ on itself. Therefore, by the theory of reduction of dynamics [191], the equations of motion drop to the quotient $TSO(3)/SO(3) \cong \mathfrak{so}(3)$. We have

$$\frac{\partial L}{\partial \mathbf{R}_{so}} = -\det(\mathbf{I}_o)\frac{\partial}{\partial \mathbf{R}_{so}}\operatorname{tr}\left[\left(\mathbf{I}_o^{-1}\mathbf{R}_{so}^T\dot{\mathbf{R}_{so}}\right)^2\right] = -\det(\mathbf{I}_o)\mathbf{R}_{so}\left[\hat{\boldsymbol{\omega}}_o, \mathbf{I}_o^{-1}\hat{\boldsymbol{\omega}}_o\mathbf{I}_o^{-1}\right]$$
$$\frac{\partial L}{\partial \dot{\mathbf{R}}_{so}} = -\frac{\det(\mathbf{I}_o)}{2}\frac{\partial}{\partial \dot{\mathbf{R}}_{so}}\operatorname{tr}\left[\left(\mathbf{I}_o^{-1}\mathbf{R}_{so}^T\dot{\mathbf{R}}_{so}\right)^2\right] = \det(\mathbf{I}_o)\mathbf{R}_{so}\mathbf{I}_o^{-1}\hat{\dot{\boldsymbol{\omega}}}_o\mathbf{I}_o^{-1}.$$

Here, the differentiation with respect to the matrices $\mathbf{R}_{so}$ and $\dot{\mathbf{R}}_{so}$ is performed using the familiar matrix differentiation rules, followed by projecting the result onto the tangent space $T_R SO(3)$.

Pulling the derivatives above back to the Lie algebra via the left translation map yields

$$\frac{\partial L}{\partial \mathbf{R}_{so}} = -\det(\mathbf{I}_o)\left[\hat{\boldsymbol{\omega}}_o, \mathbf{I}_o^{-1}\hat{\boldsymbol{\omega}}_o\mathbf{I}_o^{-1}\right], \quad \frac{\partial L}{\partial \dot{\mathbf{R}}_{so}} = \det(\mathbf{I}_o)\mathbf{I}_o^{-1}\hat{\dot{\boldsymbol{\omega}}}_o\mathbf{I}_o^{-1}.$$

Finally, the Euler-Poincaré equations motion are found to be

$$\mathbf{I}_o\dot{\boldsymbol{\omega}}_o - \mathbf{I}_o\left(\hat{\boldsymbol{\omega}}_o\mathbf{I}_o^{-1}\frac{\mathrm{d}\mathbf{I}_o}{\mathrm{d}t} + \frac{\mathrm{d}\mathbf{I}_o}{\mathrm{d}t}\mathbf{I}_o^{-1}\hat{\boldsymbol{\omega}}_o\right)^{\vee} + \operatorname{tr}\left(\mathbf{I}_o^{-1}\frac{\mathrm{d}\mathbf{I}_o}{\mathrm{d}t}\right)\mathbf{I}_o\boldsymbol{\omega}_o + \boldsymbol{\omega}_o \times \mathbf{I}_o\boldsymbol{\omega}_o = \boldsymbol{\upsilon}. \tag{8.5}$$

8.4.2 Rigid body translational dynamics

The translational dynamics of the object are the usual Newton's equations written in the frame $\mathcal{O}$

$$m_o\dot{\mathbf{v}}_o + \boldsymbol{\omega}_o \times m_o\mathbf{v}_o + m_o\gamma\mathbf{R}_{so}^T\mathbf{e}_3 = \mathbf{f}. \tag{8.6}$$

8.4.3 RoDyMan arm manipulator dynamics

We follow [216] once again to derive the manipulator dynamics, whose details have been omitted for the sake of simplicity. This procedure yields

$$\mathbf{M}_r(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \mathbf{C}_r(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} + \mathbf{n}(\boldsymbol{\theta}) = \boldsymbol{\tau}. \tag{8.7}$$

8.4.4 Object and manipulator combined dynamics

When the fingers and the object are in contact, there are forces arising from this interaction. In the combined system, the forces on the object may only

be imparted via the contact, therefore we have the following object dynamics

$$\mathbf{M}_o \dot{\mathbf{v}}_{so}^b + \boldsymbol{\phi}_o \left(\mathbf{T}_{so}, \mathbf{v}_{so}^b \right) = \mathbf{f}_o, \tag{8.8}$$

where

$$\mathbf{M}_o = \begin{bmatrix} m_o \mathbf{I}_3 & \mathbf{O}_3 \\ \mathbf{O}_3 & \mathbf{I}_o \end{bmatrix} \in \mathbb{R}^{6\times 6},$$

$$\mathbf{f}_o = \sum_{i=1}^{k} \mathrm{Ad}^T_{\mathbf{T}^{-1}_{oc_i}} \mathbf{B}_{c_i} \mathbf{f}_{c_i} \in \mathbb{R}^6,$$

$$\boldsymbol{\phi}_o = \begin{bmatrix} \boldsymbol{\omega}_o \times m_o \mathbf{v}_o + m_o \gamma \mathbf{R}_{so}^T \mathbf{e}_3 \\ -\mathbf{I}_o \left(\hat{\boldsymbol{\omega}}_o \mathbf{I}_o^{-1} \dfrac{\mathrm{d}\mathbf{I}_o}{\mathrm{d}t} + \dfrac{\mathrm{d}\mathbf{I}_o}{\mathrm{d}t} \mathbf{I}_o^{-1} \hat{\boldsymbol{\omega}}_o \right)^{\vee} + \mathrm{tr}\left(\mathbf{I}_o^{-1} \dfrac{\mathrm{d}\mathbf{I}_o}{\mathrm{d}t} \right) \mathbf{I}_o \boldsymbol{\omega}_o + \hat{\boldsymbol{\omega}}_o \mathbf{I}_o \boldsymbol{\omega}_o \end{bmatrix} \in \mathbb{R}^6,$$

where the dependencies of $\boldsymbol{\phi}_o$ have been dropped.

Similarly, each finger will experience a reaction force due to Newton's third law. These reaction forces occurring at the contact locations are mapped to the hand frame by the corresponding adjoint transformation. Subsequently, the wrench at the tool frame are mapped to the joint torques by the body manipulator Jacobian. Therefore, the reaction force on the hand frame and on the joints are given by

$$\mathbf{f}_h = -\sum_{i=1}^{k} \mathrm{Ad}^T_{\mathbf{T}^{-1}_{hc_i}} \mathbf{B}_{c_i} \mathbf{f}_{c_i} \in \mathbb{R}^6, \qquad \boldsymbol{\tau}_{\mathrm{reac}} = \mathbf{J}^{b^T} \mathbf{f}_h \in \mathbb{R}^n.$$

Consequently, the manipulator dynamics become

$$\mathbf{M}_r \ddot{\boldsymbol{\theta}} + \underbrace{\mathbf{C}_r(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) \dot{\boldsymbol{\theta}} + \mathbf{n}(\theta)}_{\triangleq \boldsymbol{\phi}_r\left(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}\right)} = \boldsymbol{\tau} - \sum_{i=1}^{k} \mathbf{J}^{b^T} \mathrm{Ad}^T_{\mathbf{T}^{-1}_{hc_i}} \mathbf{B}_{c_i} \mathbf{f}_{c_i}.$$

In order to express these equations together, we are going to introduce some more definitions, let

$$\mathbf{M} = \begin{bmatrix} \mathbf{M}_o & \mathbf{O}_{6\times n} \\ \mathbf{O}_{n\times 6} & \mathbf{M}_r \end{bmatrix} \in \mathbb{R}^{(6+n)\times(6+n)},$$

$$\boldsymbol{\phi} = \begin{bmatrix} \boldsymbol{\phi}_o \\ \boldsymbol{\phi}_r \end{bmatrix} \in \mathbb{R}^{6+n},$$

$$\bar{\boldsymbol{\tau}} = \begin{bmatrix} \mathbf{0}_6 \\ \boldsymbol{\tau} \end{bmatrix} \in \mathbb{R}^{6+n}, \quad \mathbf{f}_c = \begin{bmatrix} \mathbf{f}_{c_1} \\ \vdots \\ \mathbf{f}_{c_k} \end{bmatrix} \in \mathbb{R}^{4k}.$$

Then, we have the following system of differential-algebraic equations

$$\mathbf{M}\dot{\bar{\mathbf{v}}} + \boldsymbol{\phi}(\mathbf{T}_{so}, \mathbf{v}_{so}, \boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) = \bar{\boldsymbol{\tau}} + \mathbf{A}^T \mathbf{f}_c \tag{8.9}$$

$$\mathbf{f}_c = \left(\mathbf{A}\mathbf{M}^{-1}\mathbf{A}^T\right)^{-1} \left(\mathbf{A}\mathbf{M}^{-1}(\boldsymbol{\phi} - \bar{\boldsymbol{\tau}}) - \boldsymbol{\chi}\right), \tag{8.10}$$

or, equivalently, we have

$$\begin{bmatrix} \mathbf{M} & -\mathbf{A}^T \\ -\mathbf{A} & \mathbf{O}_{4k} \end{bmatrix} \begin{bmatrix} \dot{\bar{\mathbf{v}}} \\ \boldsymbol{\lambda} \end{bmatrix} + \begin{bmatrix} \boldsymbol{\phi} \\ \mathbf{0}_{4k} \end{bmatrix} = \begin{bmatrix} \bar{\boldsymbol{\tau}} \\ \dot{\mathbf{A}}\bar{\mathbf{v}} \end{bmatrix}, \tag{8.11}$$

with

$$\bar{\mathbf{v}} = \begin{bmatrix} \mathbf{v}_{so}^{b^T} & \boldsymbol{\theta}^T \end{bmatrix}^T \in \mathbb{R}^{6+n},$$

$$\mathbf{A}^T = \begin{bmatrix} \mathrm{Ad}^T_{\mathbf{T}^{-1}_{oc_1}} \mathbf{B}_{c_1} & \cdots & \mathrm{Ad}^T_{\mathbf{T}^{-1}_{oc_k}} \mathbf{B}_{c_k} \\ -\mathbf{J}^{b^T} \mathrm{Ad}^T_{\mathbf{T}^{-1}_{hc_1}} \mathbf{B}_{c_1} & \cdots & -\mathbf{J}^{b^T} \mathrm{Ad}^T_{\mathbf{T}^{-1}_{hc_k}} \mathbf{B}_{c_k} \end{bmatrix} \in \mathbb{R}^{(6+n)\times 4k}.$$

8.5 Trajectory generation

In order to render the catching problem as easy as possible, it may be desirable to toss the dough in the air in such a way that it neither has any lateral linear velocity nor any angular velocity along the roll and pitch axes. As desirable as it might be, unmodeled dynamics due to aerodynamic forces and the compliance of the dough inevitably imparts such velocities while the dough is in flight. We assume the presence of an estimator that predicts the dough's motion while it is in the air and updates its prediction at regular intervals. To begin with, we intend to consider the visual tracking problem of a dough of a particular rigid shape, concentrating on obtaining the motion of its center of mass using our prior experience in the estimation of such objects [172, 58]. Afterwards, we intend to develop visual tracking algorithms to track the freely falling object's deformation and orientation. Preliminary results on tracking deformation and orientation can be found in the previous chapters. In this section, we tackle the problem of generating an optimal trajectory for the end-effector of the robotic hand, given the prediction of the motion of the dough.

8.5.1 Theory

We are interested in trajectories for which we can specify the initial position, velocity, acceleration, and final velocity and acceleration of the motion. The final position is going to be dictated by the final position of the dough. This final position of the dough is a function of the final time, which will be

a parameter to be determined by the optimization process. Note that the motion may be specified in either the joint space, a torus, or the task space, the special Euclidean group of three dimensions over the reals. At this stage, we are going to evade this distinction by assuming that the path we would like to generate is on an arbitrary Riemannian manifold [80] Q. Let us call this path : $(\mathbf{a}, \mathbf{b}) \to \mathbf{Q}$, and the metric on Q is denoted by $\langle \cdot, \cdot \rangle$. We let $\mathbf{f} : (-\epsilon, \epsilon) \times (a, b) \to Q$ be a variation of c, with $\epsilon > 0$,

$$\begin{aligned} \mathbf{f}(0,t) &= \mathbf{c}(t), \quad \forall t \\ &\in [a,b] \quad \text{and} \quad f(s,a) \\ &= c(a). \end{aligned}$$

We have two vector fields of importance along the path $\mathbf{c}$. The first one is called the *variation field*, defined by

$$\mathbf{s}_{c(s)} = \frac{\partial \mathbf{f}(s,t)}{\partial s} = \frac{\mathrm{d}\mathbf{f}_t(s)}{\mathrm{d}s},$$

and second one is the velocity vector field of $\mathbf{c}$, given by

$$\mathbf{v}_{c(s)} = \frac{\mathrm{d}\mathbf{c}(t)}{\mathrm{d}t} = \frac{\partial \mathbf{f}(s,t)}{\partial t} = \frac{\mathrm{d}\mathbf{f}_s(t)}{\mathrm{d}t},$$

where this last represents the velocity along the particular path.

In order to perform calculus on the curves of this Riemannian manifold, we introduce the Levi-Civita connection, ∇. Given a curve $\mathbf{c}(t)$ and a connection, there exists a covariant derivative, which we denote by $\dfrac{\mathrm{D}}{\mathrm{d}t}$. The curvature R of a Riemannian manifold Q is a correspondence that associates to every pair $X, Y \in \mathfrak{X}(Q)$ a mapping $R(X,Y) : \mathfrak{X}(Q) \to \mathfrak{X}(Q)$ given by

$$\begin{aligned} R(X,Y)Z &= \nabla_Y \nabla_X Z - \nabla_X \nabla_Y Z + \nabla_{[X,Y]} Z, \quad Z \\ &\in \mathfrak{X}(M). \end{aligned}$$

Many paths satisfy the conditions on initial and final positions, velocities, and accelerations. Since we would like to specify the initial position, velocity, acceleration, and final velocity and acceleration, we need a sixth-order differential equation to plug these constraints as boundary conditions. Such a differential equation is what we end up with when we minimize the jerk functional, for example. On the other hand, the jerk functional is not necessarily a measure of how much effort is expended as it is a measure of vibrations within the system. If we would like to minimize the end-effector wrenches needed to catch the dough, we would have to minimize the acceleration functional. This approach, however, yields a fourth-order differential equation and so does not lend itself to imposing the desired boundary conditions.

In order to overcome this quandary, we propose to minimize not just the acceleration nor just the jerk, but a convex combination of the two. As long as

we keep the weight of the jerk functional away from zero, we shall still end up with a sixth-order differential equation and will be able to impose the desired boundary conditions. Furthermore, we can tune the weight of the acceleration functional so that it is arbitrarily close to unity, which would practically ignore the effect of the jerk functional and yield an almost optimal minimum acceleration path that successfully catches the dough. With this motivation, we define the cost functional to be minimized to be

$$L_{\mathbf{c}}(s) := \int_{t_0}^{t_f+s\delta t_f} \alpha\langle \frac{\mathrm{D}^2\mathbf{v}_{c(s)}}{\partial t^2},\ \frac{\mathrm{D}^2\mathbf{v}_{c(s)}}{\partial t^2}\rangle + \beta\langle \frac{\mathrm{D}\mathbf{v}_{c(s)}}{\partial t},\ \frac{\mathrm{D}\mathbf{v}_{c(s)}}{\partial t}\rangle \,\mathrm{d}t \tag{8.12}$$

where the weights α and β satisfy $\alpha+\beta=1$. Here, we consider only the case where the final position is left free and is part of the minimization problem. We calculate the first variation of this functional using analogous calculations as in [70] and, in addition, taking special care of the free endpoint conditions. This yields

$$\begin{aligned}\frac{1}{2}\frac{\mathrm{d}}{\mathrm{d}s}L_{\mathbf{c}}(s) = \int_{t_0}^{t_f} \Bigg\langle \alpha \Bigg(&-\frac{\mathrm{D}^5\mathbf{v}_{c(s)}}{\partial t^5} - R\left(\mathbf{v}_{c(s)}, \frac{\mathrm{D}^3\mathbf{v}_{c(s)}}{\partial t^3}\right)\mathbf{v}_{c(s)} \\ &+ R\left(\frac{\mathrm{D}\mathbf{v}_{c(s)}}{\partial t}, \frac{\mathrm{D}^2\mathbf{v}_{c(s)}}{\partial t^2}\right)\mathbf{v}_{c(s)}\Bigg) \\ &+ \beta\left(\frac{\mathrm{D}^3\mathbf{v}_{c(s)}}{\partial t^3} + R\left(\mathbf{v}_{c(s)}, \frac{\mathrm{D}\mathbf{v}_{c(s)}}{\partial t}\right)\mathbf{v}_{c(s)}\right),\ \mathbf{s}_{c(s)}\Bigg\rangle \mathrm{d}t \\ &+ \left(\frac{1}{2}\alpha\langle\frac{\mathrm{D}^2\mathbf{v}_{c(s)}}{\partial t^2}, \frac{\mathrm{D}^2\mathbf{v}_{c(s)}}{\partial t^2}\rangle + \langle\alpha\frac{\mathrm{D}^4\mathbf{v}_{c(s)}}{\partial t^4} - \beta\frac{\mathrm{D}^2\mathbf{v}_{c(s)}}{\partial t^2}, \frac{\mathrm{d}\zeta}{\mathrm{d}t}\rangle\right)\Bigg|_{t_f}\delta t_f.\end{aligned}$$

Consequently, the necessary conditions for the minimization are

$$\begin{aligned}&\alpha\left(-\frac{\mathrm{D}^5\mathbf{v}_{c(s)}}{\partial t^5} - R\left(\mathbf{v}_{c(s)}, \frac{\mathrm{D}^3V}{\partial t^3}\right)\mathbf{v}_{c(s)} + R\left(\frac{\mathrm{D}\mathbf{v}_{c(s)}}{\partial t}, \frac{\mathrm{D}^2\mathbf{v}_{c(s)}}{\partial t^2}\right)\mathbf{v}_{c(s)}\right) \\ &+\beta\left(\frac{\mathrm{D}^3\mathbf{v}_{c(s)}}{\partial t^3} + R\left(\mathbf{v}_{c(s)}, \frac{\mathrm{D}\mathbf{v}_{c(s)}}{\partial t}\right)\mathbf{v}_{c(s)}\right) = \mathbf{0}_3,\ \ \forall t_0 \le t \le t_f \qquad (8.13a)\\ &\left(\frac{1}{2}\alpha\langle\frac{\mathrm{D}^2\mathbf{v}_{c(s)}}{\partial t^2}, \frac{\mathrm{D}^2\mathbf{v}_{c(s)}}{\partial t^2}\rangle + \langle\alpha\frac{\mathrm{D}^4\mathbf{v}_{c(s)}}{\partial t^4} - \beta\frac{\mathrm{D}^2\mathbf{v}_{c(s)}}{\partial t^2}, \frac{\mathrm{d}\zeta}{\mathrm{d}t}\rangle\right)\Bigg|_{t_f} = 0 \qquad (8.13b)\\ &\mathbf{c}(t_0) = \boldsymbol{\gamma}_0,\quad \mathbf{v}_{c(s)}(t_0) = \mathbf{v}_0,\quad \frac{\mathrm{D}\mathbf{v}_{c(s)}}{\partial t}(t_0) = \mathbf{a}_0, \qquad (8.13c)\\ &\mathbf{c}(t_f) = \boldsymbol{\zeta}(t_f),\quad \mathbf{v}_{c(s)}(t_f) = \frac{\mathrm{D}\mathbf{v}_{c(s)}}{\partial t}(t_f) = \mathbf{0}_3, \qquad (8.13d)\end{aligned}$$

where suitable parametrized initial and final conditions terms have been introduced.

8.5.2 Generating tool frame trajectories

In this section, we discuss how we generate optimal trajectories for the motion of the end-effector, given the initial position $\boldsymbol{\gamma}_o = \begin{bmatrix} x_0 & y_0 & z_0 \end{bmatrix}^T \in \mathbb{R}^3$ and velocity $\mathbf{v}_{c(s)} = \begin{bmatrix} v_{d0,x} & v_{d0,y} & v_{d0,z} \end{bmatrix}^T \in \mathbb{R}^3$ of the dough at the moment it is tossed by the end-effector. With this data, we can integrate the dynamics of the center of mass of the dough to find the path it takes

$$\mathbf{p}_d(t) = \begin{bmatrix} x_0 + v_{d0,x}t \\ y_0 + v_{d0,y}t \\ z_0 + v_{d0,z}t - \dfrac{1}{2}\gamma t^2 \end{bmatrix} \in \mathbb{R}^3.$$

Moreover, we can also numerically integrate the rotational dynamics of the dough, derived in Section 8.4.1, to find the rotational path the dough takes. We then use this information to find the extremal of the cost functional defined in the previous Section 8.5.1. In particular, we break this problem into two. We compute the optimal translational path using the necessary conditions (8.13). The final time generated from this procedure is then used to determine the orientation the tool needs to have by evaluating the orientational trajectory of the dough at this instant. Without breaking the problem into two, one could also find the optimal trajectory directly in $SE(3)$. However, in this case, the rotational motion of the dough must be interpolated using one of the available techniques [70]. Moreover, the free endpoint condition presented in the necessary conditions (8.13) has to be solved iteratively, injecting a fair deal of complexity into the problem.

The necessary conditions presented in equations (8.13) yields the differential equation $-\alpha\mathbf{p}^{(6)} + \beta\mathbf{p}^{(4)} = \mathbf{0}_3$, with $\mathbf{p}(t) \in \mathbb{R}^3$ the position of the end effector at time t, in addition to the following concrete boundary condition equations for the translational motion of the end-effector

$$\begin{bmatrix} \mathbf{p}(t_0) - \mathbf{p}_0 \\ \dot{\mathbf{p}}(t_0) - \mathbf{v}_0 \\ \ddot{\mathbf{p}}(t_0) - \mathbf{a}_0 \\ \mathbf{p}(t_f) - \mathbf{p}_d(t_f) - l_h\mathbf{R}_d(t_f)\mathbf{e}_3 \\ \dot{\mathbf{p}}(t_f) \\ \ddot{\mathbf{p}}(t_f) \\ \mathbf{p}^{(5)}(t_f)^T\dot{\mathbf{p}}_d(t_f) + \dfrac{1}{2}\mathbf{p}^{(3)}(t_f)^T\mathbf{p}^{(3)}(t_f) \end{bmatrix} = \mathbf{0}_{19},$$

where the first three equations are self-explanatory. They are boundary conditions to match the tool frame's initial position, velocity, and acceleration with that of the dough. Assuming the body z-axis of the dough points orthogonal to the plane of the dough and $l_h > 0$ is a constant, the fourth equation states that the end-effector should be placed directly underneath the dough at the final time. The fifth and sixth conditions impose the design

choice that the final translational velocity of the tool frame vanishes. Upon experimentation, it was empirically observed that no slippage occurs if the hand is stopped at the time of contact (catching). Lastly, the final equation is the free endpoint condition from which we solve for the final time at which the catching should occur.

The general solution to the differential equation is given by

$$\begin{aligned}\mathbf{p}(t) &= \begin{bmatrix} x(t) \\ y(t) \\ z(t) \end{bmatrix} \\ &= \sum_{n=0}^{3} \begin{bmatrix} a_n t^n \\ b_n t^n \\ c_n t^n \end{bmatrix} + \left(\frac{\alpha}{\beta}\right)^2 \begin{bmatrix} a_4 \cosh\left(\sqrt{\frac{\beta}{\alpha}}t\right) + a_5 \sinh\left(\sqrt{\frac{\beta}{\alpha}}t\right) \\ b_4 \cosh\left(\sqrt{\frac{\beta}{\alpha}}t\right) + b_5 \sinh\left(\sqrt{\frac{\beta}{\alpha}}t\right) \\ c_4 \cosh\left(\sqrt{\frac{\beta}{\alpha}}t\right) + c_5 \sinh\left(\sqrt{\frac{\beta}{\alpha}}t\right) \end{bmatrix}.\end{aligned}$$

We use this analytic expression in the boundary conditions presented above along with a nonlinear root finder to solve for the 19 unknowns, namely, the coefficients a_0 through c_5 and the final time t_f.

The next task is to find the optimal rotational path the end-effector is desired to follow. The necessary conditions in equations (8.13) yields the differential equation

$$\begin{aligned}-\alpha \Big(&\boldsymbol{\omega}^{(5)} + 2\boldsymbol{\omega} \times \boldsymbol{\omega}^{(4)} + \frac{5}{4}\boldsymbol{\omega} \times \left(\boldsymbol{\omega} \times \boldsymbol{\omega}^{(3)}\right) + \frac{5}{2}\dot{\boldsymbol{\omega}} \times \boldsymbol{\omega}^{(3)} + \frac{1}{4}\boldsymbol{\omega} \\ &\times (\boldsymbol{\omega} \times (\boldsymbol{\omega} \times \ddot{\boldsymbol{\omega}})) + \frac{3}{2}\boldsymbol{\omega} \times (\dot{\boldsymbol{\omega}} \times \ddot{\boldsymbol{\omega}}) - (\boldsymbol{\omega} \times \ddot{\boldsymbol{\omega}}) \times \dot{\boldsymbol{\omega}} - \frac{1}{4}(\boldsymbol{\omega} \times \dot{\boldsymbol{\omega}}) \times \ddot{\boldsymbol{\omega}} - \frac{3}{8}\boldsymbol{\omega} \\ &\times ((\boldsymbol{\omega} \times \dot{\boldsymbol{\omega}}) \times \dot{\boldsymbol{\omega}}) - \frac{1}{8}(\boldsymbol{\omega} \times (\boldsymbol{\omega} \times \dot{\boldsymbol{\omega}})) \times \dot{\boldsymbol{\omega}} \Big) + \beta \left(\boldsymbol{\omega}^{(3)} + \boldsymbol{\omega} \times \ddot{\boldsymbol{\omega}}\right) = \mathbf{0}_3,\end{aligned}$$

along with the boundary conditions

$$\begin{bmatrix} \mathbf{I}_3 - \mathbf{R}_{sh}(t_0)^T \mathbf{R}_{so}(t_0) \\ \hat{\boldsymbol{\omega}}_{sh}(t_0) - \mathbf{R}_{oh}^{-1}(t_0)\hat{\boldsymbol{\omega}}_{so}(t_0)\mathbf{R}_{oh}(t_0) \\ \hat{\boldsymbol{\alpha}}_{sh}(t_0) - \mathbf{R}_{oh}^{-1}(t_0)\hat{\boldsymbol{\alpha}}_{so}(t_0)\mathbf{R}_{oh}(t_0) \\ \mathbf{I}_3 - \mathbf{R}_{sh}(t_f)^T \mathbf{R}_{so}(t_f) \\ \hat{\boldsymbol{\omega}}_{sh}(t_f) \\ \hat{\boldsymbol{\alpha}}_{sh}(t_f) \end{bmatrix} = \mathbf{O}_{18\times 3},$$

where $\boldsymbol{\alpha} \in \mathbb{R}^3$ is the angular acceleration. The first three equations match the initial orientation, angular velocity, and angular acceleration of the tool frame and the dough, taking into account the constant rotational offset between them due to the choice of initial conditions of the reference frames. The

Time of flight	t_f	$\frac{2\mathbf{v}_o^T\mathbf{e}_3}{\gamma}$
Max projectile height	h_d	$\frac{(\mathbf{v}_o^T\mathbf{e}_3)^2}{2\gamma}$
No. of revolutions	n_d	$\frac{\boldsymbol{\omega}_o^T\mathbf{e}_3 t_f}{2\pi}$

Table 8.2: Projectile motion

fourth equation is a condition that the final orientations of the end effector and the dough should match, where the yaw rotation of the dough has been eliminated. The final two equations dictate that the final angular velocity and acceleration of the tool vanish.

This boundary value problem has no analytical solution and needs to be found using a numerical boundary-value problem solver. We have used Matlab's "*bvp4c*" function to accomplish this task.

8.6 Control law

There are two distinct control objectives that we are pursuing in this work. The first one is to toss the pizza dough such that it reaches a certain height, $h_d > 0$, and by the time it comes back to the level of the fingers it has spun a certain number of times, $n_d > 0$. The second objective, which begins after the tossing phase, is to catch the pizza dough. We assume that we have generated an optimal trajectory for the S-R-S manipulator's end-effector to follow to intercept the dough. The second control objective is to have the end-effector follow this generated trajectory.

8.6.1 Tossing

The simplest plan to achieve the tossing goal may be excogitated by assuming that the center of mass of the dough behaves like a point mass during the free-fall phase. We can analytically determine how long the flight takes and the maximum height that the center of mass reaches with this assumption. These are the familiar formulas from elementary mechanics, cf. Table 8.2.

From the maximum height formula, we can derive an expression for the desired velocity of the dough at the time of release: $\mathbf{v}_{o,d} = \begin{bmatrix} 0 & 0 & \sqrt{2\gamma h_d} \end{bmatrix}^T$. Substituting the formula for the time of flight into the formula for n_d, we find a formula for the desired angular velocity of the dough at the time of release

$$n_d = \frac{\boldsymbol{\omega}_o^T \mathbf{e}_3 t_f}{2\pi} = \frac{\boldsymbol{\omega}_o^T \mathbf{e}_3 \mathbf{v}_o^T \mathbf{e}_3}{\gamma\pi} = \frac{\boldsymbol{\omega}_o^T \mathbf{e}_3}{\pi}\sqrt{\frac{2h_d}{\gamma}} \implies \boldsymbol{\omega}_{o,d} = \left[0 \;\; 0 \;\; \pi n_d \sqrt{\frac{\gamma}{2h_d}}\right]^T.$$

Combining these last two expressions for the desired linear and angular velocities, we get the desired velocity for the dough as

$$\mathbf{v}_{so,d} = \left[0 \;\; 0 \;\; \sqrt{2\gamma h_d} \;\; 0 \;\; 0 \;\; \pi n_d \sqrt{\frac{\gamma}{2h_d}}\right]^T.$$

In order to achieve this velocity, we need to act at the acceleration level and control the velocity accordingly. We go back to equation (8.8) and compute the contact forces, $\mathbf{f}_c$, that will impose an exponentially stable velocity dynamics to the desired velocity. To that end, let us prescribe

$$\sum_{i=1}^{k} \mathrm{Ad}^T_{\mathbf{T}^{-1}_{oc_i}} \mathbf{B}_{c_i} \mathbf{f}_{c_i} = -\mathbf{M}_o \mathbf{K}_d \left(\mathbf{b}_{so} - \mathbf{v}_{so,d}\right) + \boldsymbol{\phi}_o, \tag{8.14}$$

where $\mathbf{K}_d \in \mathbb{R}^{6\times 6}$ is a symmetric, positive-definite matrix. If we can find forces to apply to our fingers such that the contact forces are as given above, then the object velocity will converge to the desired velocity exponentially fast with the rate of convergence given by the eigenvalues of $\mathbf{K}_d$. To find suitable forces that we can apply to the fingers in order to generate the desired contact forces (8.14), we go back to the defining equation of the contact forces given by the second equation in (8.10). Rearranging this equation gives

$$\mathbf{A}\mathbf{M}^{-1}\bar{\boldsymbol{\tau}} = \mathbf{A}\mathbf{M}^{-1}\boldsymbol{\phi} - \mathbf{A}\mathbf{M}^{-1}\mathbf{A}^T\mathbf{f}_c - \boldsymbol{\chi}.$$

However, we need to have the first six components of the vector $\bar{\boldsymbol{\tau}}$ to vanish because those are the forces that are imposed on the object directly and there is no such physical for at our disposal. Imposing this constraint yields the following equation

$$\underbrace{-\mathbf{B}_c^T \left[\mathrm{Ad}^T_{\mathbf{T}^{-1}_{tc_1}} \;\; \mathrm{Ad}^T_{\mathbf{T}^{-1}_{tc_2}} \;\; \cdots \;\; \mathrm{Ad}^T_{\mathbf{T}^{-1}_{tc_k}}\right]^T \mathbf{J}^b \mathbf{M}_r^{-1}}_{\triangleq \mathbf{G}} \boldsymbol{\tau} = \mathbf{A}\mathbf{M}^{-1}\boldsymbol{\phi} - \mathbf{A}\mathbf{M}^{-1}\mathbf{A}^T\mathbf{f}_c - \boldsymbol{\chi}.$$

Therefore, one possible set of joint torques that would yield the desired contact forces is given by

$$\boldsymbol{\tau} = \mathbf{G}^{\dagger}\left(\mathbf{A}\mathbf{M}^{-1}\boldsymbol{\phi} - \mathbf{A}\mathbf{M}^{-1}\mathbf{A}^T\mathbf{f}_c - \boldsymbol{\chi}\right), \tag{8.15}$$

where $\mathbf{G}^{\dagger}$ is the pseudo-inverse of $\mathbf{G}$ and $\mathbf{f}_c$ is any solution of the underdetermined equation (8.14).

8.6.2 Catching

The strategy we employ to catch the dough is to move the end-effector directly underneath the dough. This direction is determined by the body negative z-axis of the object frame. Moreover, we orient the end-effector such that the z-axes of the dough and the end-effector are parallel. The latter behavior is imposed so that all the fingers come in contact with the dough almost simultaneously, which reduces the likelihood that slippage occurs during the catching phase.

We refer to the optimal trajectory, generated in Section 8.5 by $\boldsymbol{\zeta}^*(t) = (\mathbf{R}^*(t), \mathbf{p}^*(t)) \in SE(3)$ and to the actual end-effector trajectory by $\boldsymbol{\zeta}(t) = (\mathbf{R}(t), \mathbf{p}(t)) \in SE(3)$. In order to follow the desired optimal trajectory, we find the desired velocity field $\mathbf{v}_{\text{des}} \in \mathfrak{X}(SE(3))$ whose integral curves converge to the desired optimal trajectory. In the next step, we control the joints of the manipulator so that the velocity of the end-effector exponentially converges to this desired velocity. We utilize the redundancy of the manipulator by adding an additional control term, which is the orthogonal projection onto the null space of the manipulator Jacobian of the gradient vector field of the dynamic manipulability index.

The desired vector field $\mathbf{v}_{\text{des}}$ is found as the gradient vector field of the potential function $\Psi : SE(3) \to \mathbb{R}$, given by

$$\Psi(\mathbf{R}, \mathbf{p}) = k_R \| \left(\mathbf{I}_3 - \mathbf{R}^{*T} \mathbf{R} \right) \mathbf{e}_3 \|^2 + \frac{k_p}{2} \| \mathbf{R}^T \left(\mathbf{p} - \mathbf{p}^* \right) \|^2, \qquad (8.16)$$

where $k_R, k_p > 0$ are rotational and translational proportional gains, respectively. The gradient of this potential function is then computed to be

$$\mathbf{v}_{\text{des}}(\mathbf{R}, \mathbf{p}) = \left(-k_R \mathbf{R} \left(\mathbf{R}^T \mathbf{R}^* \mathbf{e}_3 \mathbf{e}_3^T - \mathbf{e}_3 \mathbf{e}_3^T \mathbf{R}^{*T} \mathbf{R} \right), \ -k_p \mathbf{R}^T (\mathbf{p} - \mathbf{p}^*) \right).$$

Note that, the integral curves of $\mathbf{v}_{\text{des}} \in \mathfrak{X}(SE(3))$ are such that $(\mathbf{R}\mathbf{e}_3, \mathbf{p}) \xrightarrow[t \to \infty]{} (\mathbf{R}^* \mathbf{e}_3, \mathbf{p}^*)$, exponentially fast. The final step in the control design procedure is then to control the velocity $\mathbf{v}$ of the end-effector such that $\mathbf{v} \xrightarrow[t \to \infty]{} \mathbf{v}_{\text{des}}$, exponentially fast. If this is accomplished then by the vanishing perturbation theory [152], we can claim that $(\mathbf{R}\mathbf{e}_3, \mathbf{p}) \xrightarrow[t \to \infty]{} (\mathbf{R}^* \mathbf{e}_3, \mathbf{p}^*)$, exponentially fast.

We solve this problem by differentiating the relation $\mathbf{J}^b \dot{\boldsymbol{\theta}} = \mathbf{v}$ to get $\dot{\mathbf{J}}^b \dot{\boldsymbol{\theta}} + \mathbf{J}^b \ddot{\boldsymbol{\theta}} = \dot{\mathbf{v}}$. Substituting for $\ddot{\boldsymbol{\theta}}$ from equation (8.7), we arrive at the relation

$$\dot{\mathbf{v}} = \mathbf{J}^b \mathbf{M}_r^{-1} \left(\boldsymbol{\tau} - \mathbf{C}_r(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) \dot{\boldsymbol{\theta}} - \mathbf{n}(\boldsymbol{\theta}) \right) + \dot{\mathbf{J}}^b \dot{\boldsymbol{\theta}}.$$

If we set the right hand side equal to $-k_d \left(\mathbf{v} - \mathbf{v}_{des} \right)$, then from the linear control theory, we know that $\mathbf{v} \xrightarrow[t \to \infty]{} \mathbf{v}_{\text{des}}$. For that purpose, we first set

$\boldsymbol{\tau} = \mathbf{C}_r(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} + \mathbf{n}(\boldsymbol{\theta}) + \boldsymbol{\tau}'$ and we then set

$$\boldsymbol{\tau}' = \tilde{\mathbf{J}}^T \left(\tilde{\mathbf{J}}\tilde{\mathbf{J}}^T\right)^{-1} \left(-k_d \left(\mathbf{v} - \mathbf{v}_{\text{des}}\right) - \dot{\mathbf{J}}^b \dot{\boldsymbol{\theta}}\right) + \left(\mathbf{I}_n - \tilde{\mathbf{J}}^T \left(\tilde{\mathbf{J}}\tilde{\mathbf{J}}^T\right)^{-1} \tilde{\mathbf{J}}\right) \boldsymbol{\eta},$$

where $\tilde{\mathbf{J}} = \mathbf{J}^b \mathbf{M}_r^{-1} \in \mathbb{R}^{6\times n}$ is an auxiliary control term that is designed to maximize the dynamic manipulability index, $\boldsymbol{\eta} = \dfrac{\partial \sqrt{\det\left(\tilde{\mathbf{J}}\tilde{\mathbf{J}}^T\right)}}{\partial \boldsymbol{\theta}} \in \mathbb{R}^n$.

8.7 Simulation

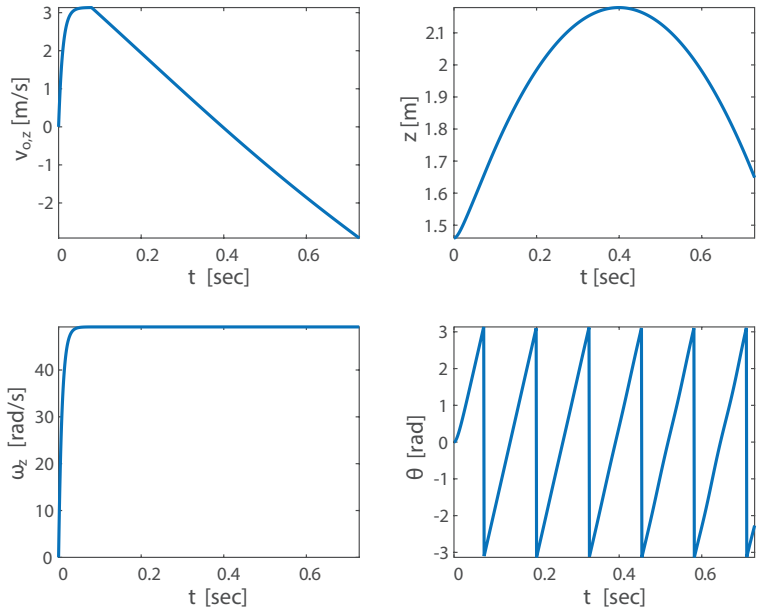
In the RoDyMan prototype that we have developed (see Fig. 8.1, the $n = 7$ axes of rotation $\boldsymbol{\omega}_i$ and the points $\mathbf{q}_i$ on these axes of rotation are given by

$$\begin{aligned}
\boldsymbol{\omega}_1 &= \begin{bmatrix} 0 & -\sin\left(\frac{\pi}{3}\right) & \cos\left(\frac{\pi}{3}\right) \end{bmatrix}^T, \quad \boldsymbol{\omega}_2 = \begin{bmatrix} -1 & 0 & 0 \end{bmatrix}^T, \\
\boldsymbol{\omega}_3 &= \boldsymbol{\omega}_4 = \boldsymbol{\omega}_6 = \begin{bmatrix} 0 & -1 & 0 \end{bmatrix}^T, \quad \boldsymbol{\omega}_5 = \boldsymbol{\omega}_7 = \begin{bmatrix} 0 & 0 & -1 \end{bmatrix}^T \\
\mathbf{q}_1 &= \mathbf{q}_2 = \mathbf{q}_3 = \begin{bmatrix} 0 & -l_{0y} & l_{0z} \end{bmatrix}^T, \ \mathbf{q}_4 = \begin{bmatrix} 0 & -l_{0y} & l_{0z} - l_1 \end{bmatrix}^T, \\
\mathbf{q}_5 &= \mathbf{q}_6 = \mathbf{q}_7 \begin{bmatrix} 0 & -l_{0y} & l_{0z} - l_1 - l_2 \end{bmatrix}^T,
\end{aligned}$$

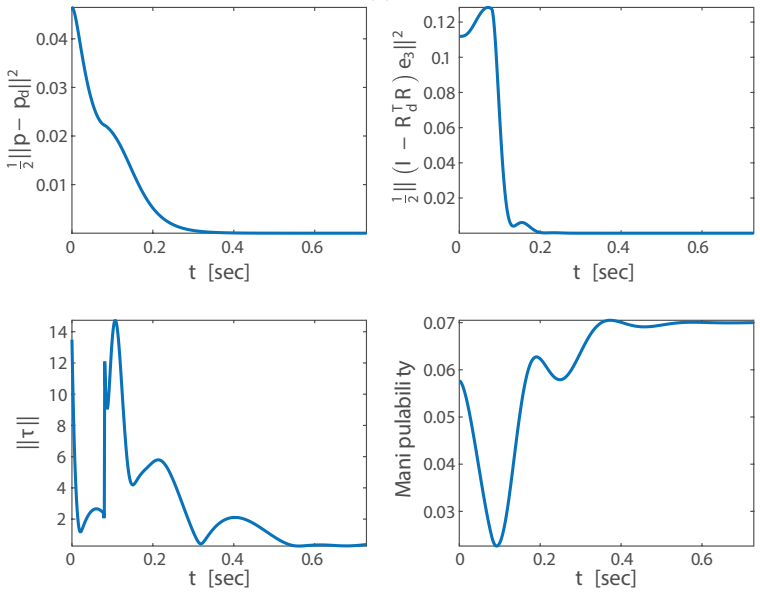
with $l_{0y}, l_{0z}, ml_1, l_2, l_3 > 0$ proper lengths. To complete the derivation of position forward kinematics, we have selected the initial pose of the end-effector with respect to the base frame to be the following homogeneous matrix

$$\mathbf{T}_{st}(0) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & -l_{0y} \\ 0 & 0 & -1 & l_{0z} - l_1 - l_2 - l_3 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Using the dynamics of the combined system introduced in Section 8.4, we have tested the controllers developed in Section 8.6. When integrating the equations of motion, we use the constraints due to the first finger and the z-components of the second and third fingers since these constraints form a maximally independent set of constraints. For the tossing objective, we have set $h_d = 0.5$ m and $n_d = 5$. With these numbers, the desired object velocity at the instant of release reads $\mathbf{v}_d = \begin{bmatrix} 0 & 0 & 3.1321 & 0 & 0 & 49.199 \end{bmatrix}^T$, in which the first three components have m/s as unit measure, and rad/s for the last three components. However, in order to account for unmodelled dynamics and disturbances, we intentionally impart nonzero lateral linear velocity and roll-pitch angular velocities to the dough by setting $\mathbf{v}_d = \begin{bmatrix} 0.1 & -0.1 & 3.1321 & 0.75 & -0.5 & 49.199 \end{bmatrix}^T$. We consider the situation where there are $k = 3$ fingers in contact with the dough.



(a)



(b)

Fig. 8.2: (a) Tossing: Object motion; (b) Catching: End-effector motion.

In this simulation the tossing phase lasts about 0.08 s. The plots in Fig. 8.2(a) show the yaw velocity and angle of the object along with its velocity and position along the inertial z-axis. We observe in the top-left subfigure that the vertical velocity of the object has reached 3.132 m/s and in the bottom-left subfigure its yaw velocity has reached 49.198 rad/s at the time of release. In the top-right figure we observe the parabolic profile of the z-axis position of the object, and in the bottom-right figure, we see that after the time of release (around 0.08 s), the object has rotated 5 times by the time it is caught.

After the release has taken place, the optimal trajectory generation is performed and the manipulator is commanded with the relevant control law. It turns out that the best time to intercept the pizza is $t_f = 0.729$ s at which instant the translation of the center of mass of the dough and the orientation of the dough read

$$\mathbf{p}_d(t_f) = \begin{bmatrix} 0.31331 & -0.26305 & 1.6793 \end{bmatrix} \text{ m},$$
$$\mathbf{R}_d(t_f) = \begin{bmatrix} -0.3925 & 0.84341 & -0.36686 \\ -0.91162 & -0.30383 & 0.27684 \\ 0.12203 & 0.4431 & 0.88813 \end{bmatrix},$$

and the interception pose of the tool is computed to be

$$\mathbf{p}(t_f) = \begin{bmatrix} 0.48 & -0.38 & 1.46 \end{bmatrix} \text{ m},$$
$$\mathbf{R}(t_f) = \begin{bmatrix} 0.77963 & -0.50743 & -0.36686 \\ 0.61235 & 0.74053 & 0.27684 \\ 0.13116 & -0.44048 & 0.88813 \end{bmatrix}.$$

As desired, while the z-axes of the end-effector and dough frames are matched, the x- and y-axes are not necessarily parallel. We observe in the top two plots of Fig. 8.2(b) that the end-effector is successfully controlled to its desired pose, as the positive semidefinite potential function Ψ presented in equation (8.16) is bounded below and above by a positive constant. In the usual Euclidean norm, the bottom left plot depicts the total amount of torque expended at each instant in time. Finally, the bottom right plot shows how the dynamic manipulability index evolves throughout the manipulation. The decrease at the beginning is due to the rapid tossing of the pizza. As the redundancy is exploited to favour the manipulability during the catching phase, we see that it climbs back up slightly above its starting value.

8.8 Discussion and conclusion

In this chapter, we have tackled tossing and catching a pizza dough with a humanoid robot. We have developed the models of grasping the dough with robotic fingers, the kinematics and dynamics of the robotic manipulator, the deformable pizza dough, and the combined system. We have developed a control law that achieves the desired tossing motion using these kinematic and dynamic models. In order to plan the catching of the dough as efficiently as possible, we have found an optimal path that minimizes a convex combination of the acceleration and jerk functionals. Once the trajectory to be taken has been decided, the control law we developed for the catching phase makes sure the robotic manipulator follows this desired path. We have then presented simulation results in plots of interesting quantities.