Nonlinear Model Predictive Control for Robotic Pushing of Planar Objects With Generic Shape

Sara Federico, Marco Costanzo, *Member, IEEE*, Marco De Simone, and Ciro Natale, *Member, IEEE*

***Abstract*—Robotic manipulation of objects in cluttered dynamic scenes is challenging for a twofold reason. Object detection and localization are complex due to partial occlusions and high variability in the object classes and manipulation in tight spaces is difficult due to potential collisions. The present letter focuses on the low-level control of the non-prehensile pushing action aimed at moving planar objects of generic shape along a given path with an assigned time law. Based on the continuous and nonlinear dynamics of the system, we propose a nonlinear model predictive controller (NMPC), which avoids the need for linearization and, thus, the hybrid dynamics arising from it. An extensive comparison with a state-of-the-art linear MPC demonstrates that the NMPC can successfully react to more general disturbances, outperforming the linear one. Experimental results confirm the effectiveness of the method in a task where a robot is required to grasp fruits in a container with other obstructing objects (shown in the attached video).**

***Index Terms*—Contact modeling, dexterous manipulation, optimization and optimal control.**

I. Introduction

LOCALIZATION and manipulation of objects in cluttered environments are hard for robots mainly due to occlusions and obstacles. Because of this, a non-prehensile robotic action is very helpful for solving both problems for an occluded object (like the apple in Fig. 1), thus improving the autonomy and manipulation skill of a robotic system. In our letter, we focus on the *pushing*, a non-prehensile manipulation action to move a target object along a given path by letting it slide on its support surface without ever lifting it. A considerable bunch of literature, e.g., [1], [2], [3], has been produced since the first seminal letters by Mason on the robotic pushing such as [4], who first analyzed the stable pushing problem by using an analytic approach under the quasi-static assumption. In this letter, we propose a model-based approach, therefore we will explain the novelty of our contribution within this framework. On the other hand, we cannot ignore all the learning methods applied to the pushing task in the last years. Yang et al. [5] exploited reinforcement learning techniques and only tactile feedback to push a variety of unseen objects, while training the agent on a single one, but with poor dynamic performances in terms of task execution. Chi et al. [6] employed diffusion models for visuomotor policy learning, proving the effectiveness of their method on several manipulation tasks, including the so-called Push-T, aimed to push a T-shaped object to reach a target pose. In fact, learning-based approaches to robotic pushing generally pursue the objective of moving the object to a given fixed target position, instead here we intend to let the robot push the object along a given reference trajectory, that is a path and a time law. This feature is essential in those tasks where the robot has to act in a dynamic environment and the execution of the pushing manoeuvre has to be completed in a time compatible with possibly moving obstacles or targets.

(a) (b)

Fig. 1. Scenario illustrating the need for a pushing action: (a) The robot can push the obstacles away from the apple to grasp. (b) Close-up of the scene and desired paths for the obstacles to enable the grasp.

By following Mason's early results and those obtained in [7] as well as the ellipsoidal approximation of the limit surface in [8], Lynch et al. in [1] model the motion of a pushed object on a planar surface, mapping the involved friction forces into the corresponding velocities, through the motion cone concept. The resulting quasi-static model is nonlinear and is used in [9] to develop a linear control system based on a model predictive approach. They designed a sub-optimal solution, called Family Of Modes (FOM), to reduce the computational cost of the arising mixed-integer optimal control problem, otherwise unsolvable in real-time. By following this strategy, [10] proposes a model predictive controller with learned mode scheduling to reduce the computational complexity of the integer programming approach by employing learning techniques aimed at reconstructing an optimal sequence of contacts. Moreover, [11] uses a nonlinear solver based on Complementary Constraints (CC),

Received 23 October 2024; accepted 28 January 2025. Date of publication 5 February 2025; date of current version 18 February 2025. This article was recommended for publication by Associate Editor E. Mingo Hoffman and Editor L. Pallottino upon evaluation of the reviewers' comments. This work was supported by the European Commission under the Horizon Europe Research through INTELLIMAN project under Grant 101070136. *(Corresponding author: Marco Costanzo.)*

The authors are with the Dipartimento di Ingegneria, Università degli Studi della Campania Luigi Vanvitelli, 81031 Aversa, Italy (e-mail: marco.costanzo@unicampania.it).

This article has supplementary downloadable material available at https://doi.org/10.1109/LRA.2025.3539092, provided by the authors.

Digital Object Identifier 10.1109/LRA.2025.3539092

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

which, unlike [9], does not require a linearization along a given nominal path. However, it still considers the three contact modes. Instead, [12] avoids the mixed-integer programming by using State-Triggered Constraints (STCs) reformulating the problem as direct nonlinear programming, proven to be computationally more efficient.

Other works, such as [13] and [14], try to reduce the mismatch between the model and the reality by reinforcing the model-based approach with a data-driven one. While, in [15] human demonstrations are used to initialize the optimal control problem without adopting a predictive approach, thus limiting the autonomy of the solution that has to rely on human demonstrations sufficiently close to the desired task to be performed by the robot. Moreover, the powerful contact models based on the limit surface have been used for simulation and planning [16] as well as for computationally demanding control strategies [9], [17]. To overcome this problem, the approach in [18] assumes a specific dynamic regime (sticking) using a feedback linearization controller, whereas [19] uses model-free heuristic feedback loops to achieve a given fixed target pose.

Inspired by the model-based approach developed in [9], we propose two main novelties that extend our recent letter [17]. To reduce the computational cost of mixed-integer solutions, we developed a NMPC. However, unlike [9], our approach exploits the intrinsically continuous nature, even though nonlinear, of the pusher-slider dynamics avoiding the need for modeling discontinuous jumps between contact modes. Since no linearization is required, no constraints on the contact modes should be considered. Again, it also differs from the approach proposed in [12], where an automatic contact activation mechanism is developed relying on the STCs to reduce the number of contact pairs and consequently the computational complexity.

Moreover, to improve the autonomy of the robotic manipulator and make the pushing manoeuvre independent of the target object shape, as in [10] and [11] we take into account the shape of the pushed object. Both letters use the same geometric parameterization based on an angular variable suitable only for shapes that can be parameterized radially. In contrast, we can push planar objects of any shape by resorting to a B-spline function, thus with a known curvature in each contact point. The devised control algorithm allows the manipulator to move all around the object by always keeping one contact point between the pusher and the slider. In this way, the pushing manoeuvre is optimal without needing heuristic solutions to decide the pushing direction. To the best of our knowledge, this is the first NMPC approach for the pusher-slider system that can handle a generic planar slider shape by exploiting the B-spline formulation and that does not require a hybrid description of the pusher-slider dynamics owing to a direct nonlinear control strategy. Finally, all pushing experimental tests have been carried out with planar objects characterized by different shapes and curvatures (including both convex and concave objects) to validate our approach, highlighting the capability of the robot to autonomously move the pusher around the object without losing contact. The reported experiments are conducted not only in simulation but also in a real-world setup and show that the NMPC can be solved in real-time provided that a good initialization of the solver is available, as explained in Section IV. They also demonstrate how the possibility of turning the pusher around the object can greatly simplify the pushing task or make it feasible depending on the desired trajectory when this would be impossible if the object had to be pushed only on one of its faces. Specific experiments also compare our method with the approach proposed in [9]. The results highlight the limits of the linearization mainly in terms of robustness to disturbances.

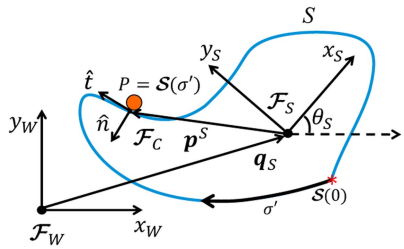

Fig. 2. Pusher-Slider kinematic model: the slider is represented by the blue contour and the pusher by the orange circle. $\mathcal{F}_W$, $\mathcal{F}_C$ and $\mathcal{F}_S$ are the world, contact and fixed to slider frames, respectively. $\boldsymbol{q}_S$ is the slider position in the world frame, θ_S is its orientation. $\hat{\boldsymbol{t}}$ and $\hat{\boldsymbol{n}}$ are the tangential and normal vector to the B-spline that define $\mathcal{F}_C$. The pusher position P at the path coordinate σ' is represented by the spline function $\boldsymbol{\mathcal{S}}(\sigma')$; $\boldsymbol{\mathcal{S}}(0)$ is the spline starting point and the black arrow indicates the increasing direction of σ'.

II. Pusher-Slider Nonlinear Model

To allow the pusher to freely move along the slider perimeter without losing contact, we generalized the pusher-slider model presented in [9] by extending it to planar sliders of generic shape through an analytical model of their 2D contour. Preserving the hypotheses used in [9], we assumed that the friction forces are governed by the Coulomb's law and adopted the quasi-static assumption. With reference to Fig. 2, given a planar slider S with a generic shape, its 2D contour has been analytically approximated by a spline function. Given n_c control points distributed along its contour of length L and the associated clamped knot vector, let $\boldsymbol{\mathcal{S}}(\sigma')$ be the spline two-dimensional vector function arising from the linear combination of a set of consecutive B-splines of fixed degree d, where σ' is the arc-length parameter of the resulting path, such that $\sigma' \in [0, L]$. To model multiple turns of the pusher along the object contour in both clockwise and counterclockwise directions, we introduce a path coordinate $\sigma \in (-\infty, \infty)$ linked to the arc-length by $\sigma' = \text{mod}(\sigma, L)$, where the function $\text{mod}(\cdot, \cdot)$ is the standard remainder after division function. Details about the B-spline formulation and its properties can be found in [20]. For $d \geq 1$, by using the analytical differentiation formula of $\boldsymbol{\mathcal{S}}(\sigma)$, we can evaluate the normal and the tangential unit vectors, $\hat{\boldsymbol{n}}(\sigma)$ and $\hat{\boldsymbol{t}}(\sigma)$, in each point of the path that define the contact frame $\mathcal{F}_C$ (see Fig. 2) located at the contact point $P(\sigma)$ between the pusher and the slider. Moreover, for reasons that will be clear in Section III, we also derived the following path curvature in each point, positive for σ increasing clockwise along the contour,

$$\mathcal{C}(\sigma) = -\frac{d}{d\sigma}(\text{atan2}(t_y(\sigma), t_x(\sigma))), \tag{1}$$

where $t_y(\sigma)$ and $t_x(\sigma)$ are the y and x components of the tangential unit vector $\hat{\boldsymbol{t}}$ evaluated in σ with respect to $\mathcal{F}_S$, respectively.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

The analytical formulation of $\mathcal{C}(\sigma)$ has been evaluated through the symbolic tool offered by CasADi [21].

This formulation generalizes the model in [9] to a generic slider shape by representing the pusher position with the path coordinate σ. With the symbols defined in the caption of Fig. 2, the four-dimensional state of the pusher-slider model is $\boldsymbol{x} = [\boldsymbol{q}_S^\top \quad \theta_S \quad \sigma]^\top$, where $\boldsymbol{q}_S = [x_S \quad y_S]^\top$ is expressed in $\mathcal{F}_W$. For any σ, the position $\boldsymbol{p}^S(\sigma) = [p_x \quad p_y]^\top$ of the pusher P, expressed in $\mathcal{F}_S$, is obtained by evaluating $\boldsymbol{\mathcal{S}}(\sigma)$. Let $\boldsymbol{p}^C = [p_n \quad p_t]^\top$ be the pusher position expressed in $\mathcal{F}_C$, that is obtained as $\boldsymbol{p}^C(\sigma) = \boldsymbol{R}_S^C(\sigma)\boldsymbol{p}^S(\sigma)$, being $\boldsymbol{R}_S^C$ the 2D rotation matrix representing the orientation of $\mathcal{F}_S$ with respect to $\mathcal{F}_C$; finally, let $\boldsymbol{u} = [u_n \quad u_t]^\top$ be the control input of the system, i.e., the pusher velocity expressed in $\mathcal{F}_C$. Using the motion cone concept and the ellipsoidal approximation of the limit surface as in [1], the nonlinear dynamic model follows

$$\begin{bmatrix} \dot{\boldsymbol{q}}_S \\ \dot{\theta}_S \end{bmatrix} = \frac{1}{c^2 + {p_n}^2 + {p_t}^2} \begin{bmatrix} \boldsymbol{R}_S^W(\theta_S)\boldsymbol{R}_C^S(\sigma)\boldsymbol{Q}(\sigma) \\ -p_t \quad p_n \end{bmatrix} \boldsymbol{v}_c, \tag{2}$$

$$\text{with } \boldsymbol{Q}(\sigma) = \begin{bmatrix} c^2 + {p_n}^2 & p_n p_t \\ p_n p_t & c^2 + {p_t}^2 \end{bmatrix}, \tag{3}$$

where c is the ratio between the maximum frictional moment resulting from a pure rotation about the center of mass of the slider, which depends on the friction coefficient μ_{SS} between the slider and the support, and the maximum frictional force in pure translation [1]. Moreover, $\boldsymbol{R}_S^W(\theta_S)$ and $\boldsymbol{R}_C^S(\sigma)$ are rotation matrices used to switch from $\mathcal{F}_C$ to $\mathcal{F}_W$ and $\boldsymbol{v}_c$ is the velocity at the contact point transmitted from the pusher to the slider. Given the friction coefficient μ_{SP} between the pusher and the slider and the edge vectors of the motion cone

$$\begin{aligned} \boldsymbol{v}_l &= [1 \quad \gamma_l]^\top, \text{with } \gamma_l = \frac{\mu_{SP}c^2 - p_n p_t + \mu_{SP}{p_n}^2}{c^2 + {p_t}^2 - \mu_{SP}p_n p_t} \\ \boldsymbol{v}_r &= [1 \quad \gamma_r]^\top, \text{with } \gamma_r = \frac{-\mu_{SP}c^2 - p_n p_t - \mu_{SP}{p_n}^2}{c^2 + {p_t}^2 + \mu_{SP}p_n p_t}, \end{aligned} \tag{4}$$

the velocity at the contact $\boldsymbol{v}_c$ results

$$\boldsymbol{v}_c = \begin{cases} \boldsymbol{u}, & \text{if } \gamma_r u_n \le u_t \le \gamma_l u_n \\ u_n \boldsymbol{v}_l, & \text{if } u_t > \gamma_l u_n \\ u_n \boldsymbol{v}_r, & \text{if } u_t < \gamma_r u_n, \end{cases} \tag{5}$$

where the first condition corresponds to the so-called *sticking* contact, for which the pusher velocity is fully transmitted to the slider; the second and the third conditions correspond to a sliding contact, i.e., the pusher slides clockwise (*sliding left*) or counterclockwise (*sliding right*) along the object contour with respect to $\mathcal{F}_S$, for which the pusher velocity is only partially transmitted to the slider. The system dynamics is completed by computing $\dot{\sigma}$ as the tangential component of the pusher velocity relative to the slider

$$\dot{\sigma} = (\boldsymbol{u} - \boldsymbol{v}_c)^\top \hat{\boldsymbol{t}}, \tag{6}$$

where $\boldsymbol{v}_c$ is defined in (5). By combining (2) and (6), the resulting pusher-slider model can be formulated as a standard nonlinear state space dynamic system

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{u}(t)), \tag{7}$$

where the expression of the four-dimensional vector function $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})$ depends on the conditions involving the motion cone boundaries in (5). However, conditions (5) clearly show that $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})$ is continuous even though not differentiable where the contact modes change between sticking and sliding. This implies that there is no need to introduce any discrete variable describing the contact mode, unless a linearization is required along a nominal trajectory as in [9]. Indeed, they used a discrete variable to discriminate the state space model corresponding to the different contact modes in which the vector function $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})$ being linearized is differentiable.

III. Pushing Control Via Nonlinear MPC

In order to push the slider object along a desired trajectory $\hat{\boldsymbol{x}}(t)$ we adopted an NMPC strategy. The differential equation in (7) is first numerically integrated with an explicit 4th order Runge-Kutta (RK4) method with sampling time T_s, leading to the difference equation $\boldsymbol{x}_{h+1} = \boldsymbol{f}_d(\boldsymbol{x}_h, \boldsymbol{u}_h)$, being h the time step corresponding to the time instant hT_s, such that $\boldsymbol{x}_h = \boldsymbol{x}(hT_s), \boldsymbol{u}_h = \boldsymbol{u}(hT_s)$. Note that the analytical expression of the function $\boldsymbol{f}_d(\cdot, \cdot)$ is fully determined given the explicit numerical integrator used to solve the differential (7). This is not the case when using the FOM method, which requires a linearized model for each contact mode, hence the need for a mixed integer solver. Then, a quadratic cost function is defined over a prediction horizon of length H as

$$J = \sum_{h=0}^{H-1} \left(\tilde{\boldsymbol{x}}_h^\top \boldsymbol{W}_x \tilde{\boldsymbol{x}}_h + \boldsymbol{u}_h^\top \boldsymbol{W}_u \boldsymbol{u}_h \right) + \tilde{\boldsymbol{x}}_H^\top \boldsymbol{W}_H \tilde{\boldsymbol{x}}_H, \tag{8}$$

where $\tilde{\boldsymbol{x}}_h = \boldsymbol{x}_h - \hat{\boldsymbol{x}}_h$, $h = 0, \dots, H$ is the tracking error, and $\boldsymbol{W}_x$, $\boldsymbol{W}_u$ and $\boldsymbol{W}_H$ are weight matrices of proper dimensions. According to the receding horizon approach, the control input $\boldsymbol{u}_h$ is the first sample of the solution $\boldsymbol{u}_0, \dots, \boldsymbol{u}_{H-1}$ of the following constrained nonlinear optimization problem

$$\min_{\boldsymbol{u}_0, \dots, \boldsymbol{u}_{H-1}} \quad J \quad \text{subject to} \tag{9a}$$

$$\boldsymbol{x}_0 = \bar{\boldsymbol{x}}_0 \tag{9b}$$

$$\boldsymbol{x}_{h+1} = \boldsymbol{f}_d(\boldsymbol{x}_h, \boldsymbol{u}_h) \tag{9c}$$

$$\begin{bmatrix} 0 \\ -\bar{u}_t(\sigma_h) \end{bmatrix} \le \boldsymbol{u}_h \le \begin{bmatrix} \bar{u}_n \\ \bar{u}_t(\sigma_h) \end{bmatrix} \tag{9d}$$

where $\bar{\boldsymbol{x}}_0$ is the initial state measurement and $\bar{u}_n$ and $\bar{u}_t$ are the bounds on the normal and tangential pusher velocities, respectively.

The model described in (7) explicitly considers the variable shape of the pushed object but it neglects inertial forces, therefore, to avoid losing the contact between pusher and slider we had to impose a variable constraint on the maximum tangential velocity to let the robot to decelerate as the curvature of the slider increases. The term $\bar{u}_t(\sigma)$ is computed using the curvature expression in (1) as

$$\bar{u}_t(\sigma) = \min \left(\bar{u}_{t,\max}, \frac{\alpha}{||\mathcal{C}(\sigma)| - c_0|} \right), \tag{10}$$

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

where $\bar{u}_{t,\max}$ is the maximum allowed tangential velocity; $\alpha > 0$, and c_0 are two scalar values to be tuned depending on the performance of the robot and of the perception system. If the time delay arising from the communication with the robot or from the perception system is too high, or the robot tracks the commands with some delay, it is opportune to reduce the maximum velocity for a fixed curvature reducing both α and c_0. In this way, we can reduce the possibility that the robot loses contact with the slider.

When the slider presents sharp corners, the pusher would ideally reach a zero tangential velocity to avoid losing contact. In practice, using the spline formulation, real sharp corners are rounded up automatically owing to the differentiability of $\boldsymbol{\mathcal{S}}(\sigma)$. This causes a mismatch between the actual slider shape and the shape model used by the controller which ends with the pusher's inability to pass the corner. A sharp corner represents a singularity that must be suitably handled. We propose a singularity handling strategy such that, when the pusher is near a sharp corner, i.e., $|\mathcal{C}(\sigma)| > \mathcal{C}^*$, being $\mathcal{C}^*$ a certain threshold, the controller output u_n is set to a negative value u_n^* that allows the pusher to lose the contact when the singularity is detected and only for the necessary time to cross it. Selection guidelines for the threshold and the normal negative velocity will be given in the next section.

To counteract the communication delays of the manipulator and perception interfaces, we adopted the delay compensation strategy proposed in [17].

IV. Experimental Results

The experiments are carried out on a 7-axis robot Yaskawa Motoman SIA5F equipped with a WSG32 gripper by Weiss Robotics. The pusher extension, suitably designed and realized in ABS plastic, is linked to the SUNTouch force/tactile right finger [22] mounted on the robot. The contact point to perform the pushing manoeuvre is at its extremity. The nonlinear optimization problem described in Section III is solved by using the open-source framework acados [23] on Matlab running on an Intel i9 processor at 3.7 GHz. In particular, we employed CasADi to implement the symbolic expression of the continuous function in (7), and exploited the explicit RK4 method implemented by acados to obtain (9c). The 2D pose of the slider required by the feedback control law is measured through a 3D model-based tracking system based on the ViSP platform [24]. Relying on the availability of 3D CAD models, the tracker considers both moving edges and Kanade–Lucas–Tomasi (KLT) keypoints to track first the features of the objects and then to recover the pose of the camera w.r.t. the tracked object. The pusher position $\boldsymbol{p}^S$ is available through the robot forward kinematics, while, the measure of the state variable σ is obtained in real-time by numerically minimizing the function $\|\boldsymbol{p}^S(\sigma) - \boldsymbol{\mathcal{S}}(\sigma)\|$. In this way, the measured pusher position is projected onto the slider-shape curve despite the measurement noise. The overall communication with the robot takes place through the Robot Operating System (ROS).

To highlight the performance of the proposed nonlinear controller and its capability to address the shape variability of the slider, we selected the four objects with different shapes shown in Fig. 3. Fig. 4 shows $\boldsymbol{\mathcal{S}}(\sigma)$, $\bar{u}_t(\sigma)$ and $\mathcal{C}(\sigma)$ for each object. Each spline $\boldsymbol{\mathcal{S}}(\sigma)$, of order $d = 3$, has been evaluated by using a planar

Fig. 3. Selected objects for the experiments. From left to right and from top to bottom: "santal", "montana", "balea", and "madel".

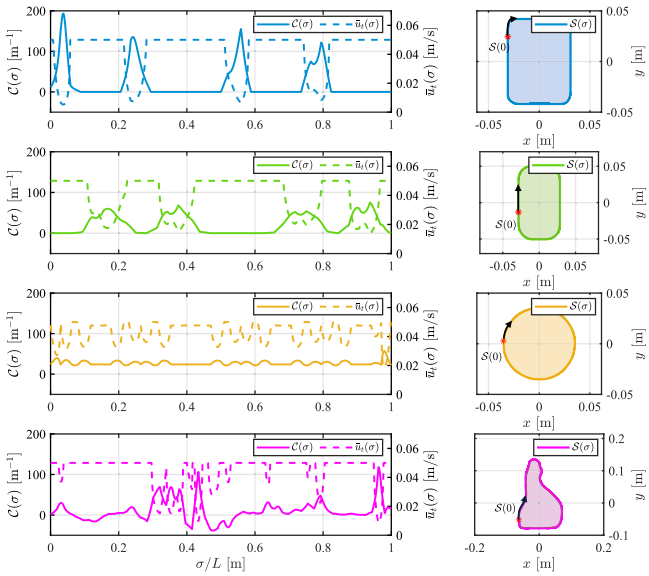

Fig. 4. From top to bottom: $\mathcal{C}(\sigma)$, $\bar{u}_t(\sigma)$, and $\boldsymbol{\mathcal{S}}(\sigma)$ for santal, montana, balea, and madel objects. The plots on the left are displayed with σ normalized to the length L of each contour. In the $\boldsymbol{\mathcal{S}}(\sigma)$ plots on the right, the starting point of the spline is highlighted in red and the black arrow indicates the increasing direction of σ.

TABLE I
Physical and Geometrical Parameters of the Four Objects

	m [kg]	μ_{SP}	μ_{SS}	n_c	L [m]
santal	0.29	0.19	0.32	37	0.29
montana	0.25	0.10	0.20	35	0.30
balea	0.17	0.20	0.35	37	0.22
madel	0.50	0.10	0.22	56	0.60

surface arising from a simplified CAD model of each object. Note that the santal juice brick is a rectangular object, but due to the differentiability property, the corresponding spline cannot exactly model the four sharp corners of the object. Indeed, being the maximum value of the santal curvature greater than $\mathcal{C}^*$ (see Table II), the singularity handling strategy described in Section III becomes here relevant. Moreover, being the balea object a cylinder, its curvature is almost constant, while the madel object is non-convex, hence its curvature is negative in correspondence with concave sections. The spline model of the madel contour has been simplified by neglecting the spray head

TABLE II
CONTROLLER PARAMETERS

T_s	0.07 s	c_0	3
H	10	$\bar{u}_n$	0.03 m/s
$\boldsymbol{W}_x$	$\text{diag}(1, 1, 10^{-2}, 0)$	$\bar{u}_{t,max}$	0.05 m/s
$\boldsymbol{W}_u$	$\text{diag}(10^{-3}, 10^{-3})$	$\mathcal{C}^*$	$120\,\text{m}^{-1}$
$\boldsymbol{W}_H$	$20\text{diag}(10^4, 10^4, 1, 0)$	u_n^*	-0.001 m/s
α	0.29	d_{delay}	5

to reduce the number n_c of control points from 155 to 56 (see Table I) thus reducing the computational complexity of the spline evaluation. During the experiments, we constrained the value of σ to prevent the pusher from reaching the spray head.

Table I reports the geometrical and physical parameters of the four objects; friction coefficients have been estimated by manually pushing objects over the given surface with a force sensor. Table II shows the controller parameters, tuned with the help of the simulator set up in Matlab through acados.[1] The matrices $\boldsymbol{W}_x$, $\boldsymbol{W}_H$, and $\boldsymbol{W}_u$ have been tuned with the help of the simulator to obtain a satisfactory performance in the nominal conditions; note how the variables with the same physical meaning have been weighed with equal values, and the weight $\boldsymbol{W}_H$ has been chosen larger than $\boldsymbol{W}_x$ to obtain a smaller final error. The prediction horizon H has been chosen as a trade-off between computational time and the length of the predicted slider path. Moreover, we selected the Sequential Quadratic Programming (SQP) solver provided by acados, employing the QP solver High-Performance Interior Point Method (HPIPM) to solve the local quadratic approximations of the problem. As usual in the solution of optimization problems, the solver is warm-started with the previous solution, while for the first iteration, we initialize the control with zero pusher velocity, that is a reasonable choice for generic initial conditions.

Looking at the curvature of the santal and the madel in Fig. 4, the only objects with sharp corners, we selected $\mathcal{C}^*$ as a value a bit lower than the lowest peak corresponding to a sharp corner. Accordingly, the negative value u_n^* was chosen starting from a very small fraction of $\bar{u}_n$ and gradually increasing it until the singularity handling strategy described in Section III allowed the pusher to cross a corner.

A. Comparison With a Linear Approach

We compare our controller to the linear FOM method proposed in [9] and implemented using Gurobi[2] in the available repository,[3] first through Matlab simulations, and then on the real setup to validate our approach and demonstrate its performance improvement. Both the controllers run at 50 ms, which is the minimum sampling time the nonlinear solver can converge even though with sporadic outliers. Note that in the experiments presented in the next sections, we set the sampling time at 70 ms, i.e., a trade-off between having a small sampling time and allowing the nonlinear solver to achieve at least a suboptimal solution with no outliers. Both methods minimize the same index in (8), even though the FOM minimizes the control input difference with respect to the nominal one, i.e., $u_t = 0$ and u_n equal to the nominal trajectory speed.

[1]Further implementation details in the repo: https://github.com/Vanvitelli-Robotics/uclv_qs_pushing_matlab.git

[2]Gurobi Optimization, LLC, *Gurobi Optimizer Reference Manual*, 2024.

[3]https://github.com/Vanvitelli-Robotics/uclv_fompush.git

TABLE III
MEAN AND STANDARD DEVIATION (M(STD)) OF PERFORMANCE INDEXES IN THE SIMULATION COMPARISON BETWEEN FOM AND NMPC

	e_x [mm]	e_y [mm]	e_θ [rad]	$I_{\bar{J}}$
FOM	7.2(2.3)	8.0(3.0)	0.29(0.13)	3.6e-6(4.0e-6)
NMPC	4.4(1.7)	6.0(2.4)	0.21(0.092)	7.4e-7(2.1e-6)

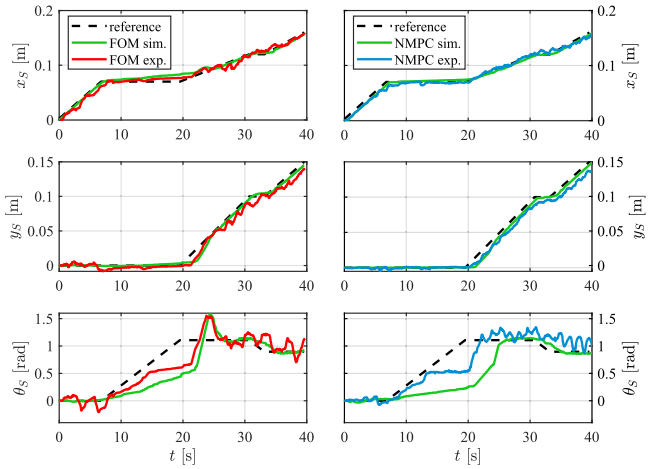

Fig. 5. Comparison with a linear approach: simulation and experiment for the santal S-shaped trajectory tracking (FOM–left, NMPC–right).

Table III shows the statistical results of an extensive comparative study performed in simulation by considering the santal object that has to track two different trajectories. FOM requires linearizing the system dynamics along the given reference trajectory. For a generic one, this step is computationally expensive if done online, or memory greedy if done offline. Thus, for a fair comparison, we implemented the idea in [17], where the following simplifications have been adopted. First, an offline linearization of the system around a straight line with a constant velocity has been done. Then, to always use the same resulting linear system, given a set of waypoints, any arbitrary trajectory is generated as a sequence of segments, separating the rotations from the translations. The overall trajectory cannot have an arbitrary velocity profile, but it has to be constant at the same value used for the preliminary linearization. By using this approach, we generated a S-shaped path (see Fig. 6) consisting of three segments at a constant velocity of $0.01\,\text{m/s}$ and two turns at a velocity of 0.09 rad/s, not requiring the pusher to move around the object, and a straight line path consisting of one segment at a constant velocity of $0.01\,\text{m/s}$. Note that the FOM approach can handle pushing from a single side of the object, hence, for a fair comparison, in the NMPC we constrained the pusher to stay on a single side of the object. We compared the controllers in a variety of conditions by applying five random initial conditions uniformly distributed in the following intervals:

$$x_S(0), y_S(0) \sim \mathcal{U}\{-0.03, 0.03\}\,[\text{m}],$$
$$\theta_S(0) \sim \mathcal{U}\{-10, 10\}\,[\text{deg}],\ \sigma(0) \sim \mathcal{U}\{-0.045, 0.003\}.$$

Moreover, five random disturbances uniformly distributed as $y_S^S(t) \sim \mathcal{U}\{-0.03, 0.03\}$ [m] have been applied on y_S^S at time

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

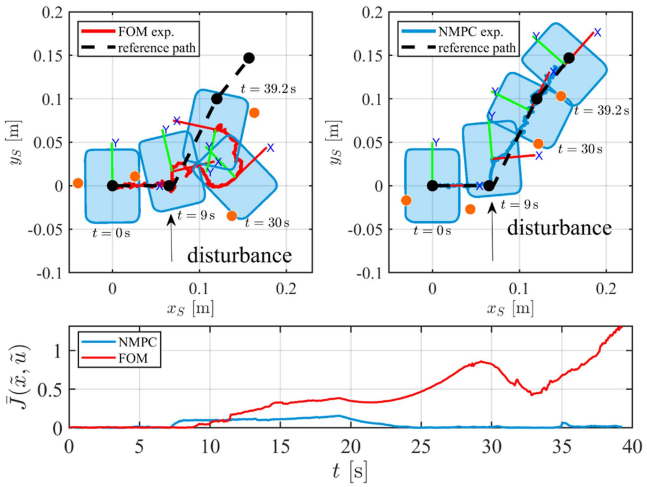

Fig. 6. Disturbance rejection experiment: from top to bottom, snapshots along the S-shaped path and performance index (11). The orange dot and the dashed line indicate the position of the pusher and the reference path, respectively. The black dots are the path waypoints.

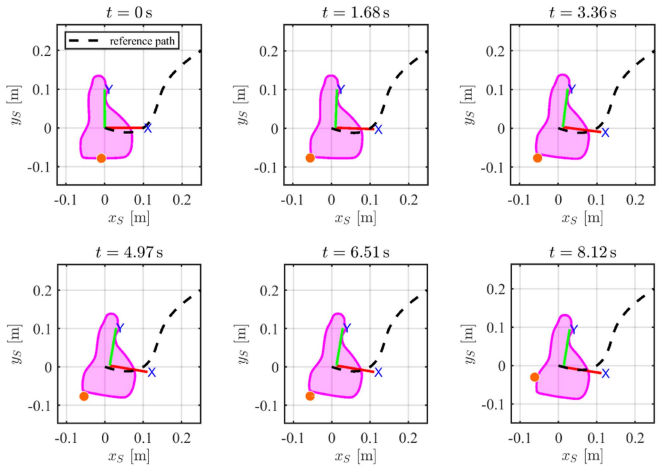

Fig. 7. Trajectory tracking experiment: snapshots during the initial phase for the madel object. The orange dot and the dashed line indicate the position of the pusher and the reference path, respectively.

instants $t = 4$ s and $t = 7$ s for the straight line, and $t = 6$ s and $t = 27$ s for the S-shaped trajectory. The maximum magnitude of the disturbances and the time instants have been selected so that the pusher never loses contact with the slider. By combining 2 trajectory types, 2 time instants, 5 amplitudes of the disturbances and 5 initial conditions, we performed 100 simulations for each controller computing the following metrics: e_x, e_y, e_θ, which are the mean (and standard deviation), reported in Table III with the notation M(STD), of the RMS value of the tracking error on x_S, y_S, θ_S, respectively. Moreover, we evaluated $I_{\bar{J}}$, i.e., the mean (and standard deviation) of the time integral of the performance index defined as

$$\bar{J}(\tilde{\boldsymbol{x}}, \boldsymbol{u}) = \tilde{\boldsymbol{x}}(t)^\top \boldsymbol{W}_x \tilde{\boldsymbol{x}}(t) + \boldsymbol{u}(t)^\top \boldsymbol{W}_u \boldsymbol{u}(t), \qquad (11)$$

where $\tilde{\boldsymbol{x}}(t) = \boldsymbol{x}(t) - \hat{\boldsymbol{x}}(t)$ is the tracking error, and $\boldsymbol{W}_x$, $\boldsymbol{W}_u$ are the same weight matrices in (8). Table III confirms the expected results: the nonlinear approach outperforms the linear one, especially when the applied disturbances deviate the system too far from the nominal trajectory around which it has been linearized, as discussed below.

Two case studies similar to the simulations above have also been tested on the real robot and the results are reported in Fig. 5 and Fig. 6. Note that, as for our NMPC, also for the FOM approach it was necessary to compensate a delay of 350 ms arising from the perception pipeline in the experimental setup, as described in Section III ([17] shows the effects of a non-compensated delay on the system).

Despite the unavoidable uncertainties affecting the simulation model, the simulation results are very close to the experimental ones for both the analysed methods. Moreover, they are both able to track the reference in the nominal condition with a satisfactory performance.

Fig. 6 shows the results obtained by applying a disturbance of about 0.03 m by hand on y_S while tracking the same trajectory of the previous experiments. Note that the capability of the FOM to reject disturbances highly depends on the linearization of the system. Indeed, as proved also in [17], the linear approach can reject small disturbances, which deviate the system not too far from the nominal trajectory, otherwise the system would be unable to recover from them. This conclusion is in line with the outcome of other works comparing with the same FOM method [11].

B. Trajectory Tracking

The following experiments aim at demonstrating the performance of the proposed control system and the effectiveness of the approach adopted to allow the pusher to move the object along a desired trajectory by sliding all around the contour while always keeping the contact. In each experiment, the object has to track the same reference trajectory and the initial position of the pusher is selected such that the optimal manoeuvre, to be computed by the NMPC, to pursue the goal is moving along the contour.

The reference trajectory is generated by defining the path waypoints and interpolating them such that the x-axis of $\mathcal{F}_S$ is always tangent to the S-shaped path shown in Fig. 7. The maximum reference slider speed is 11 mm/s.

Fig. 8 shows the experimental results for the trajectory tracking of the four proposed objects. The tracking performance is very satisfactory, despite the initial condition of the pusher being far from the optimal one. With reference to Fig. 7, the initial pusher position is clearly not the right one to push the madel object along the reference path. The snapshots show the pusher position resulting from the application of the optimal control input in the first time instants of the task. The pusher moves around the corner to reach the optimal position on the correct side to start pushing the object. A similar behaviour arises for all objects. Note that the tracking of the orientation angle θ_S in Fig. 8 for balea, montana, and madel is less accurate than the one for santal. For the balea, this is due to the low accuracy in the angle measured by the visual tracker for such a symmetric object; for the montana and the madel objects, this is due to the low friction between the pusher and the slider that forces the controller to prefer sliding over sticking, that means the pusher has to continuously slide left and right on the same face to push

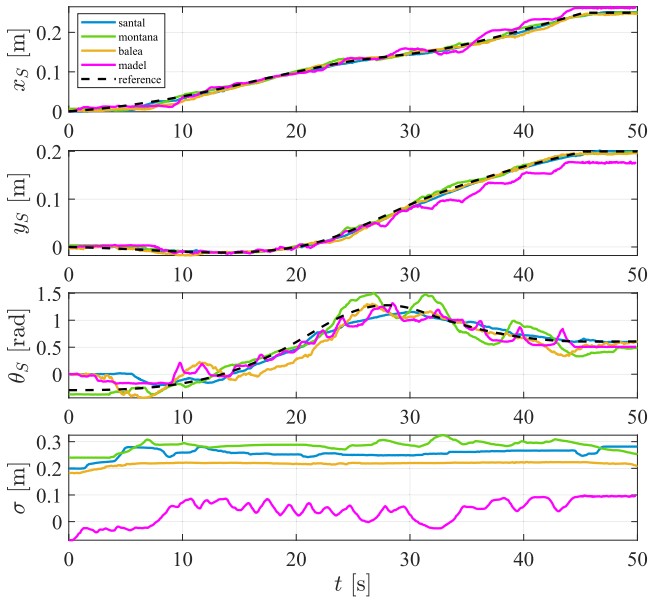

Fig. 8. Trajectory tracking experiment: reference (black), santal (blue), montana (green), balea (yellow), madel (magenta).

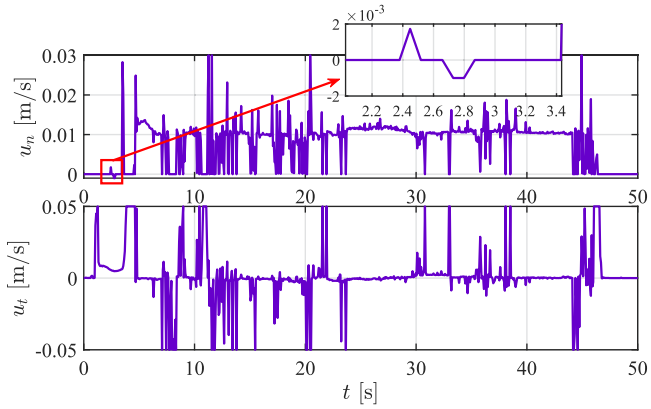

Fig. 9. Trajectory tracking experiment: control signals for the santal object.

the object forward, as $\sigma(t)$ in Fig. 8 shows. Due to the similarity of the obtained results, we show the control inputs only for the santal object, which is challenging due to the sharp corners. Fig. 9 highlights the manoeuvre performed by the pusher to cross the corner. It is realized in about six seconds by applying only a tangential velocity, thus the pusher slides along the object without pushing it. Note that the maximum tangential velocity applied at the beginning is reduced as the curvature increases due to the constraint (9d). Simultaneously, when $\mathcal{C}^*$ is reached, the singularity handling strategy described in Section III allows the pusher to cross the corner by applying a negative normal velocity, as the zoomed plot shows.

C. Disturbance Rejection

In this experiment, the santal object has to track a more complex trajectory by keeping a constant orientation along the whole U-shaped path with a maximum reference slider speed of 5 mm/s. To evaluate the robustness of the proposed NMPC,

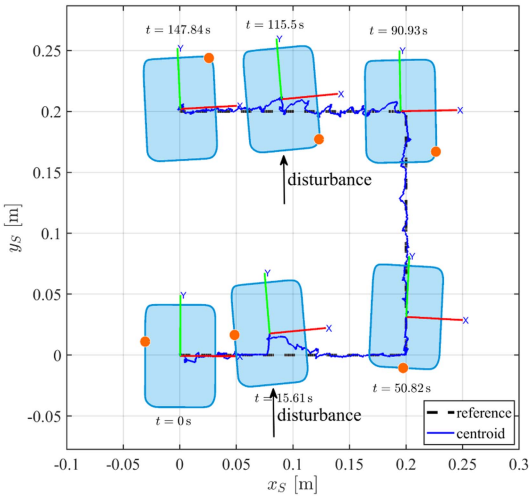

Fig. 10. Disturbance rejection experiment: snapshots during the trajectory tracking (U-shaped path) for the santal object.

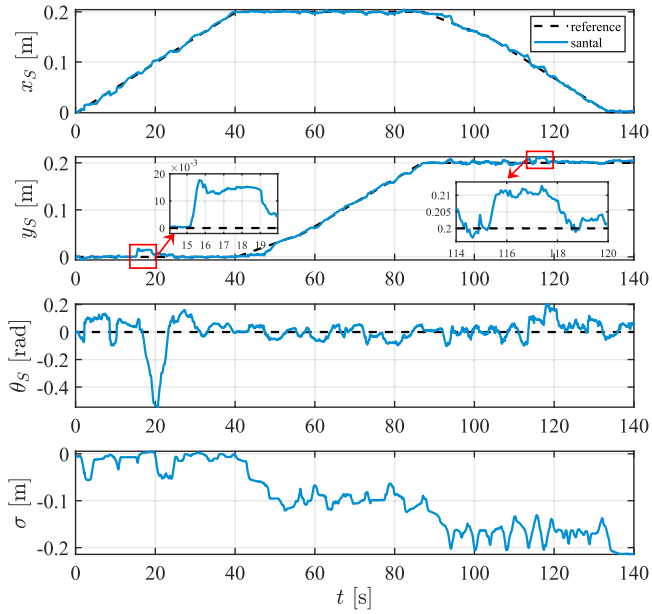

Fig. 11. Disturbance rejection experiment: trajectory tracking for the santal object. The effect of the disturbances is highlighted in the zoomed plots.

during the motion, the object is touched two times (see the accompanying video). Fig. 10 shows the desired trajectory in the $x - y$ plane and the actual trajectory followed by the origin of the slider frame.

When the disturbances are applied, the controller is able to reject them, allowing the object to return to its path. Quality of the tracking can be appreciated in Fig. 11. Note that tracking the angle reference would be unfeasible if the pusher could not move from one face to the other.

D. Pushing to Enable Pick-and-Place in a Cluttered Scene

This last experiment shows how the pushing manoeuvre is useful to make the pick-and-place task of a red apple feasible in a cluttered scene. Fig. 1(a) shows the initial setup of the

experiment. The grasp of the apple is attempted with the pipeline proposed in [25]. It consists of a perception module, devoted to the detection of the apple, a grasp planning module, devoted to the computation of a feasible grasp pose despite the obstacles in the scene, and a grasp controller, which uses tactile measures to compute the minimum force necessary to lift the apple. Obviously, the apple cannot be grasped in the case of Fig. 1(a) due to obstructing objects. The problem becomes feasible by pushing them away along the red paths reported in Fig. 1(b). This and all other experiments can be found in the accompanying video.

V. Conclusion

The letter presented a novel NMPC for robotic pushing. The nonlinear approach gives the system greater robustness and better performance than the linear one, even if it is computationally more expensive. The analytical strategy adopted to model the shape of the slider allows the pusher to follow complex trajectories by turning around the object when required. However, the success of the pushing execution strongly depends on the length of the prediction horizon which is limited by the computational power available. Indeed, as shown in the last part of the attached video, our NMPC could not solve the optimization problem when the initial position of the pusher is too far from the optimum value, due to the limited prediction horizon; hence the controller cannot predict the system behavior sufficiently forward. This limitation could be overcome by dynamically changing the control horizon of the NMPC formulation, depending on the pusher and slider velocities, or by adopting warm-start procedures of the solver based on collected data from human demonstrations or simulations executed with a proper prediction horizon depending on the task complexity. Future work will extend the result to the dynamic case when inertial forces are not negligible and explore learning-based methods, e.g., diffusion models and reinforcement learning, to see if they can overcome the limitations above.

References

[1] K. Lynch, H. Maekawa, and K. Tanie, "Manipulation and active sensing by pushing using tactile feedback," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 1992, pp. 416–421.

[2] S. Akella and M. Mason, "Posing polygonal objects in the plane by pushing," in *Proc. 1992 IEEE Int. Conf. Robot. Automat.*, vol. 3, 1992, pp. 2255–2262.

[3] K.-T. Yu, J. Leonard, and A. Rodriguez, "Shape and pose recovery from planar pushing," in *Proc. 2015 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 1208–1215.

[4] M. T. Mason, "Mechanics and planning of manipulator pushing operations," *Int. J. Robot. Res.*, vol. 5, no. 3, pp. 53–71, 1986.

[5] M. Yang et al., "Sim-to-real model-based and model-free deep reinforcement learning for tactile pushing," *IEEE Robot. Automat. Lett.*, vol. 8, no. 9, pp. 5480–5487, Sep. 2023.

[6] C. Chi et al., "Diffusion policy: Visuomotor policy learning via action diffusion," *Int. J. Robot. Res.*, to be published, doi: 10.1177/02783649241273668.

[7] S. Goyal, A. Ruina, and J. Papadopoulos, "Planar sliding with dry friction part 1. Limit surface and moment function," *Wear*, vol. 143, no. 2, pp. 307–330, 1991. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0043164891901043

[8] S.-H. Lee and M. R. Cutkosky, "Fixture planning with friction," *J. Eng. Ind.*, vol. 113, pp. 320–327, 1991.

[9] F. R. Hogan and A. Rodriguez, "Feedback control of the Pusher-Slider system: A story of hybrid and underactuated contact dynamics," in *Algorithmic Foundations of Robotics XII*. Cham: Springer, 2020, pp. 800–815, doi: 10.1007/978-3-030-43089-4_51.

[10] F. R. Hogan and A. Rodriguez, "Reactive planar non-prehensile manipulation with hybrid model predictive control," *Int. J. Robot. Res.*, vol. 39, no. 7, pp. 755–773, 2020, doi: 10.1177/0278364920913938.

[11] J. Moura, T. Stouraitis, and S. Vijayakumar, "Non-prehensile planar manipulation via trajectory optimization with complementarity constraints," in *Proc. 2022 Int. Conf. Robot. Automat.*, 2022, pp. 970–976.

[12] M. Wang, A. Ã. Ãnol, P. Long, and T. Padir, "Contact-implicit planning and control for non-prehensile manipulation using state-triggered constraints," in *Proc. Int. Symp. Robot. Res.*, 2022, pp. 189–204.

[13] M. Bauza, F. R. Hogan, and A. Rodriguez, "A data-efficient approach to precise and controlled pushing," in *Proc. 2nd Conf. Robot Learn.*, Oct. 2018, vol. 87, pp. 336–345. [Online]. Available: https://proceedings.mlr.press/v87/bauza18a.html

[14] C. Chamzas, C. R. Garrett, B. Sundaralingam, L. E. Kavraki, and D. Fox, "Meta-policy learning over plan ensembles for robust articulated object manipulation," in *Proc. RSS 2023 Workshop Learn. Task Motion Plan.*, 2023. [Online]. Available: https://openreview.net/forum?id=68N8Dj6KVv

[15] T. Xue, H. Girgin, T. S. Lembono, and S. Calinon, "Demonstration-guided optimal control for long-term non-prehensile planar manipulation," in *Proc. 2023 IEEE Int. Conf. Robot. Automat.*, 2023, pp. 4999–5005.

[16] N. Chavan-Dafle and A. Rodriguez, "Sampling-based planning of in-hand manipulation with external pushes," in *Robotics Research*, N. M. Amato, G. Hager, S. Thomas, and M. Torres-Torriti, Eds., Cham, Switzerland: Springer, 2020, pp. 523–539.

[17] M. Costanzo, M. De Simone, S. Federico, and C. Natale, "Non-prehensile manipulation actions and visual 6D pose estimation for fruit grasping based on tactile sensing," *Robotics*, vol. 12, no. 4, 2023, Art. no. 92. [Online]. Available: https://www.mdpi.com/2218-6581/12/4/92

[18] J. Zhou, Y. Hou, and M. T. Mason, "Pushing revisited: Differential flatness, trajectory planning, and stabilization," *Int. J. Robot. Res.*, vol. 38, no. 12-13, pp. 1477–1489, 2019.

[19] J. Lloyd and N. F. Lepora, "Goal-driven robotic pushing using tactile and proprioceptive feedback," *IEEE Trans. Robot.*, vol. 38, no. 2, pp. 1201–1212, Apr. 2022.

[20] T. Lyche, C. Manni, and H. Speleers, "Foundations of spline theory: B-splines, spline approximation, and hierarchical refinement," in *Splines and PDEs: From Approximation Theory to Numerical Linear Algebra*. Cham: Springer, 2018, pp. 1–76, doi: 10.1007/978-3-319-94911-6_1.

[21] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi–A software framework for nonlinear optimization and optimal control," *Math. Program. Comput.*, vol. 11, no. 1, pp. 1–36, 2019.

[22] M. Costanzo, G. De Maria, C. Natale, and S. Pirozzi, "Design and calibration of a force/tactile sensor for dexterous manipulation," *Sensors*, vol. 19, no. 4, 2019, Art. no. 966. [Online]. Available: https://www.mdpi.com/1424-8220/19/4/966

[23] R. Verschueren et al., "acados–a modular open-source framework for fast embedded optimal control," *Math. Program. Comput.*, vol. 14, no. 1, pp. 147–183, Oct. 2022, doi: 10.1007/s12532-021-00208-8.

[24] E. Marchand, F. Spindler, and F. Chaumette, "ViSP for visual servoing: A generic software platform with a wide class of robot control skills," *IEEE Robot. Automat. Mag.*, vol. 12, no. 4, pp. 40–52, Dec. 2005.

[25] M. Costanzo, M. D. Simone, S. Federico, C. Natale, and S. Pirozzi, "Enhanced 6D pose estimation for robotic fruit picking," in *Proc. 9th Int. Conf. Control, Decis. Inf. Technol.*, 2023, pp. 901–906.