Quadratic Dynamic Matrix Control for Fast Cloth Manipulation

Edoardo Caldarelli, *Graduate Student Member, IEEE*, Adrià Colomé, *Member, IEEE*,
Carlos Ocampo-Martinez, *Senior Member, IEEE*, and Carme Torras, *Fellow, IEEE*

***Abstract*— Robotic cloth manipulation is an increasingly relevant area of research, challenging classic control algorithms due to the deformable nature of cloth. While it is possible to apply linear model predictive control to make the robot move the cloth according to a given reference, this approach suffers from a large dimensionality of the state-space representation of the cloth models. To address this issue, in this work we study the application of an input-output model predictive control strategy, based on quadratic dynamic matrix control, to robotic cloth manipulation. To account for uncertain disturbances on the cloth's motion, we further extend the algorithm with suitable chance constraints. In extensive simulated experiments, involving disturbances and obstacle avoidance, we show that quadratic dynamic matrix control can be successfully applied in different cloth manipulation scenarios, with significant gains in optimization speed compared to standard model predictive control strategies. The experiments further demonstrate that the closed-loop model used by quadratic dynamic matrix control can be beneficial to the tracking accuracy, leading to improvements over the standard predictive control strategy. Moreover, a preliminary experiment on a real robot shows that quadratic dynamic matrix control can indeed be employed in real settings.**

I. Introduction

Robotic cloth manipulation is a challenging and increasingly relevant area of research, and introduces novel problems, related to task learning [1]–[3], perception [4], dynamical system identification [5], [6], and benchmarking [7], due to the deformable nature of cloth. When considering cloth handling policies, *dynamic* strategies are particularly effective, as they leverage the dynamics and inertia of the cloth, being able to reach configurations outside the manipulator's workspace [1]. However, most of the approaches in the literature of dynamic cloth manipulation are tailored to a specific task (e.g., cloth folding), and are data and computationally intensive [8].

A more general framework, introduced by [9], considers moving the cloth based on the desired trajectory of some *points of interest*, such as the corners or the edges. A surrogate analytical model of the cloth is defined, and the cloth is controlled *in real-time* by means of *model predictive control* (MPC) [10]. Due to the highly nonlinear nature of the true cloth's dynamics, traditional control techniques, such as *static* feedback laws, might fail [11]. Conversely, MPC strategies are capable of retrieving *dynamic* feedback laws, that are endowed with optimality properties w.r.t. suitable figures of merit, and are subject to constraints that may stem from an evolving environment.

In particular, [9] proposes to combine a standard Cartesian controller of a bimanual robotic manipulator with the surrogate cloth model. This strategy yields a novel control pipeline where the robot controls the cloth movement in closed-loop by means of MPC, while the robot's end-effectors are moved with impedance control in the operational space [12]. In the case study proposed in [9], the robot is holding the cloth from its two upper corners, whose position is the input of the cloth dynamical system, while a reference signal is specified for the two lower corners, that serve as outputs. As we will show in our experiments, decoupling the dynamics of the cloth from that of the manipulator allows to *counteract disturbances*, such as wind, that do not affect the manipulator at all, but may alter the behavior of the garment significantly. While these disturbances are likely to be absent in controlled environments, they are the norm in domestic setups, in which robotic cloth manipulation may be applied successfully.

The surrogate cloth model defined in [9], referred to as *control oriented model* (COM), must give a proper approximation of the cloth's behavior, while being simple enough to be used in an online optimization, as it is done by MPC. Hence, [9] proposes to model the cloth with a mesh of particles connected with *linearized* springs and dampers, to be used in a linear MPC (LMPC) algorithm. While effective in practice, this modeling choice allows for arbitrarily stretching of the cloth. This fact underlines the need for constraints on the displacement of the upper corners, besides the ones on the mesh of the cloth, and both types of constraints can be easily handled by an MPC strategy.

All the aforementioned advantages of LMPC come with the burden of a computational cost related to solving an optimization problem online. This downside is particularly true for systems with a *high dimensionality* of the state-space representation, such as the mesh-based cloth model previously described. For instance, given a 4×4 mesh, the state of the model, which includes the position and velocity of each particle in the mesh, belongs to $\mathbb{R}^{96}$. One popular way to deal with high-dimensional state-space representations in the context of MPC is to use the so-called *quadratic dynamic matrix control* (QDMC) [13]. In contrast to standard output-feedback models, which require state estimation [14], QDMC bypasses the state-space representation, by directly modeling the system of interest with a discrete-time input-output

This work was supported by the project CLOTHILDE ("CLOTH manIpulation Learning from DEmonstrations"), funded by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Advanced Grant agreement No 741930).

All authors are with the Institut de Robòtica i Informàtica Industrial (IRI), CSIC-UPC, Barcelona, Spain. {ecaldarelli, acolome, cocampo, torras}@iri.upc.edu.

C. Ocampo-Martinez is also with the Automatic Control Department, Universitat Politècnica de Catalunya - BarcelonaTECH, Barcelona, Spain. carlos.ocampo@upc.edu.

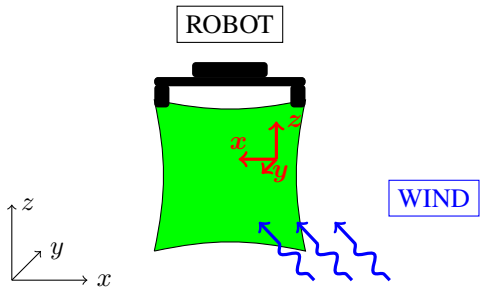

Fig. 1. Schematic representation of the setup considered in our work. A robot is grasping a piece of cloth (the green square in the picture) from its upper corners, to follow a reference with the lower corners. The motion of the cloth in space is perturbed by a possibly stochastic disturbance caused by wind. The model for the cloth is represented w.r.t. a local cloth base, shown in red. The black axes represent the world's frame.

relationship. This model boosts the speed of the control algorithm, making it viable for real-time requirements.

In this work, we investigate the use of QDMC in cloth manipulation. In addition to the standard deterministic QDMC implementation, we propose a stochastic formulation of QDMC, useful when the constraints stemming from the environment are combined with uncertain disturbances, as shown in Fig. 1. We consider *time-variant* uncertainty, in contrast to prior work in stochastic QDMC that was focusing on *time-invariant* uncertainty [15], [16]. While one way to deal with uncertainty is by means of *robust* predictive controllers [17], it is known that these approaches are often too conservative, and degrade the overall performance by considering unlikely scenarios with equal importance to the most likely ones [15]. In our approach, we mimic the derivations in *chance-constrained MPC* [18], where a probabilistic model of the uncertainty (e.g., the probability density function) is assumed to be known and used to derive constraints holding *with high probability*.

The structure of this paper is as follows: In Section II, we introduce the case study, and formalize the background of our work. In Section III, we describe the proposed chance-constrained QDMC formulation, while in Section IV we detail the experimental setup and the results for the algorithms used. Finally, the conclusions are reported in Section V

II. Problem Statement and Background

In this section, we introduce our case study, and report the main background of our work.

A. Linear Model of the Cloth

In this work, we consider the following setup. A bimanual robot is holding a square piece of cloth from the two upper corners, and the position of the two lower corners is measured. The goal is to move the cloth so that the lower corners follow a reference trajectory. This setup results in a multi-input-multi-output (MIMO) system modeling the cloth behavior, where the inputs are $\mathbf{u} \in \mathbb{R}^6$ and correspond to the 3-dimensional position of the upper corners of the piece of cloth. Likewise, the outputs are $\mathbf{y} \in \mathbb{R}^6$ and correspond to the 3-dimensional position of the lower corners of the piece of cloth. The cloth is modeled as a mesh of $P \times P$ particles, and the state of the system is given by the position and velocity of each particle in the mesh, that is, $\boldsymbol{\chi} \in \mathbb{R}^{N_x}$, where $N_x = 6P^2$. To retrieve a suitable state-space representation of the COM, the complex, nonlinear dynamics of the cloth must be linearized. This can be done as follows. Each particle in the mesh is connected to the neighboring ones by structural springs and a damper. This means that, at each time-step in the execution of a task, each particle is subject to three types of forces: the elastic force, the damping, and gravity. The elastic force is a nonlinear function of the difference between the nodes' positions, i.e., a nonlinear function of the states, as it depends on the Euclidean distance $\|\cdot\|_2$. The elastic term is linearized by replacing $\|\cdot\|_2$ with $\|\cdot\|_1$. Besides this linearization, the elastic and damping terms are modified so as to allow for different stiffness and damping constants along each direction of the space. In this way, a linear state-space representation is obtained and validated in [9].

Model Parameters: As described in [9], the spring-mass-damper system can potentially stretch under its own weight. To avoid this, we can set the initial length of the springs to an offset in the vertical axis, so that the weight of the particles is balanced by the actual elastic force. This initial offset, together with the stiffness and damping values along each Cartesian axis, are parameters to be tuned to obtain a stable system given the chosen sampling time of 0.02 s.

B. Quadratic Dynamic Matrix Control

As it is clear from the previous subsection, the major drawback of the linear cloth model is the high dimensionality of its state, which is *quadratic* in the number of nodes in the mesh. If a predictive controller with constraints is used, as we shall see in Section IV, this complexity slows down the optimization significantly, as the number of state constraints grows quadratically in the mesh size. Thus, we propose to design a QDMC algorithm for controlling the cloth, as we detail in the sequel.

1) Finite-Step Response Model: As a start, we can introduce a suitable finite-step response (FSR) model for our linear system. It is known that the response of a discrete-time, single-input-single-output (SISO) linear system, at time-step k, can be obtained from the superposition of responses to steps of varying amplitude, applied at subsequent time-steps [13]. In the MIMO case, thanks to the superposition principle, each output is obtained by summing up the responses due to every input. More formally, let N_u be the number of inputs, N_{ss} be the number of steps until the outputs are in steady-state (also known as *model length* [15]), N_y be the number of outputs. When considering cloth, $N_u = 6$, $N_y = 6$. Moreover, as we will show in Section IV-B, $N_{ss} = 500$. Furthermore, let $S^l_{i,j}$ be the l-th step response coefficient from input i to output j. This coefficient is the value, at time-step l, of the output j, when we observe a unitary step in input i at time-step 0. Lastly, let y^j_{ss} be the steady-state value of output j, and u^i_{ss} be the corresponding input value on channel i. It can be shown that the outputs

are given by the expression [13]

$$y_k^j = y_{ss}^j + \sum_{i=1}^{N_u} \left[\sum_{l=1}^{N_{ss}-1} S_{i,j}^l \Delta u_{k-l}^i + S_{i,j}^{N_{ss}} (u_{k-N_{ss}}^i - u_{ss}^i) \right], \tag{1}$$

for $j \in \{1, \dots, N_y\}$. Equation (1) indicates that the output value at time-step k is the convolution of suitable response coefficients and the input variations backwards in time. All the input variations Δu before time-step $k - N_{ss}$ affect the current output by the same steady-state coefficient $S_{i,j}^{N_{ss}}$. Note that the FSR model assumes a *stable* system [15].

2) Calculation of the FSR Coefficients: The first key challenge within the design of a QDMC strategy is the calculation of the aforementioned step response coefficients, when dealing with nonlinear systems. One option would be to directly approximate the nonlinear input-output relationship with a linear function, identified by means of a least-squares estimate [19]. However, as we have outlined in Section II-A, a linear model for the cloth is available, and we can use it to compute the coefficients. In order to compute the coefficients, we firstly put the cloth at the equilibrium so that no oscillations happen. Then, we apply a step variation in the x, y and z coordinates of both the upper corners, and measure all the outputs. The step coefficients are readily retrieved from the ratio between the output and the input amplitudes.

3) Predictive Model: Once we have reached time-step k of the simulation, the FSR model can be used to make predictions of the output values. Linearity of the output response w.r.t. the inputs allows us to decouple between the effect of *past* inputs (i.e., applied until time-step $k-1$), and *future* inputs, which need to be determined by optimizing a suitable risk function. The former output values are called *free response*, and are the value of output j at time-step $k+1$ if no input variation happens at time-step k. We denote the free response as $f_{k+1|k}^j$. We can therefore define the one-step-ahead predicted output j due to an input variation on channel i as

$$\hat{y}_{k+1|k}^j = S_{i,j}^1 \Delta u_k^i + f_{k+1|k}^j. \tag{2}$$

Stacking the predicted outputs over the prediction horizon H_p in the vector $\hat{\mathbf{y}}_k^j$, the control actions over the control horizon H_c in $\Delta \mathbf{u}_k^i$, and the free response in $\mathbf{f}_k^j$, the prediction model generalizes to

$$\hat{\mathbf{y}}_k^j = \underline{\mathbf{S}}_{i,j} \Delta \mathbf{u}_k^i + \mathbf{\Psi} \mathbf{f}_k^j. \tag{3}$$

Usually, $H_c \ll H_p$ for stability [13]. A detailed explanation on how to construct matrices $\underline{\mathbf{S}}_{i,j}$ and $\mathbf{\Psi}$ can be found in [15].

4) Model Mismatch Handling: QDMC accounts for mismatches in the prediction model. In particular, at time-step k, the measured output j is denoted by $y_k^{j,meas}$. The current mismatch, $\tilde{d}_k^j$, is computed as the difference between $y_k^{j,meas}$ and the predicted output after the last input has been applied (at time-step $k-1$), that is, $f_{k|k}^j$. The disturbance is assumed to be constant along the whole prediction horizon, and summed to $\mathbf{\Psi} \mathbf{f}_k^j$. Note that, when a new input is applied, the free response must be updated with the FSR model [19]. The model mismatch is used by QDMC to close the loop with measurements from the real system, while the nominal dynamics associated to the COM are propagated recursively. Note that the fact that the model mismatch error is kept constant over the prediction window makes closed-loop QDMC different from closed-loop LMPC. In the latter, the prediction starts from the current measured state in an autoregressive way, and therefore the error between the nominal and the real state is propagated through the system's dynamics.

5) Final Prediction Model: The predictions on the different output channels, along with the control actions and the free responses (with the model mismatch), can be stacked in column vectors to obtain the following model, for a suitable coefficient matrix $\tilde{\mathbf{S}}$, as defined in [19]:

$$\tilde{\mathbf{y}}_k = \tilde{\mathbf{S}} \Delta \tilde{\mathbf{u}}_k + \tilde{\mathbf{f}}_k. \tag{4}$$

6) Optimization Problem: Let $\tilde{\mathbf{r}}_k$ be a vectorized reference signal, i.e., a vector where the references for the output channels have been stacked vertically. We can use Equation (4) to compute the tracking error $\mathbf{e}_k$ over a prediction horizon H_p, that is,

$$\mathbf{e}_k = \tilde{\mathbf{S}} \Delta \tilde{\mathbf{u}}_k + \tilde{\mathbf{f}}_k - \tilde{\mathbf{r}}_k \in \mathbb{R}^{N_y \cdot H_p}. \tag{5}$$

This output can be used to define a suitable risk to be optimized online throughout the simulation, depending on weighting factors $\tilde{\mathbf{Q}}$ and $\tilde{\mathbf{R}}$:

$$\mathcal{R}_k = \mathbf{e}_k^T \tilde{\mathbf{Q}} \mathbf{e}_k + \Delta \tilde{\mathbf{u}}_k^T \tilde{\mathbf{R}} \Delta \tilde{\mathbf{u}}_k. \tag{6}$$

Let us now define a region for admissible input variations, $\Delta \mathbb{U}$, and a region of admissible outputs, $\Delta \mathbb{Y}$. The final optimization problem that we solve is given by

$$\begin{aligned} &\min_{\Delta \tilde{\mathbf{u}}_k} \ \mathcal{R}_k \\ &\text{subject to } \mathbf{e}_k = \tilde{\mathbf{f}}_k + \tilde{\mathbf{S}} \Delta \tilde{\mathbf{u}}_k - \tilde{\mathbf{r}}_k, \\ &\qquad \Delta \mathbf{u}_k \in \Delta \mathbb{U}, \\ &\qquad \tilde{\mathbf{y}}_k \in \mathbb{Y}. \end{aligned} \tag{7}$$

III. Proposed Chance-Constrained QDMC

In this section, we propose a stochastic QDMC (SQDMC) formulation with chance constraints, by detailing how the QDMC prediction model in Section II-B.3 can be extended to include exogenous disturbances, and how the optimization problem (7) can be modified to account for time-variant uncertainty. As we will show in Section IV, this formulation allows us to *reject disturbances* such as wind acting on the cloth (thanks to the disturbance included in the prediction model), and *avoid obstacles* with safety margins (thanks to the proposed chance constraints). In particular, the chance constraints allow to safely move apart from the obstacle, with a confidence that is proportional to the aforementioned uncertainty, quantified in terms of standard deviation of the outputs of the cloth system.

A. QDMC with Disturbances

In order to embed the disturbance in the QDMC prediction model, we can start by observing that the linear state-space model described in Section II-A can be extended to include

the effect of the disturbances on the states. For a state vector $\boldsymbol{\chi}$ (position and velocity of the particles in the cloth mesh), constant offset on the states $\boldsymbol{\chi}_{ct}$, exogenous disturbance $\boldsymbol{\delta}_k \in \mathbb{R}^{N_d}$, and a matrix $\mathbf{B}_{\boldsymbol{\delta}} \in \mathbb{R}^{N_x \times N_d}$, we have that

$$\boldsymbol{\chi}_{k+1} = \mathbf{A}\boldsymbol{\chi}_k + \mathbf{B}\Delta\mathbf{u}_k + \boldsymbol{\chi}_{ct} + \mathbf{B}_{\boldsymbol{\delta}}\boldsymbol{\delta}_k. \tag{8}$$

The disturbance vector can be considered as a d-dimensional *non-controllable* input. Thus, we need to find a suitable FSR model for mapping every channel in $\boldsymbol{\delta}_k$ to every output of our MIMO system. Thanks to the linearity of the COM, its output vector is given by the superposition (i.e., sum) of the outputs due to $\Delta\mathbf{u}$ and $\Delta\boldsymbol{\delta}$. As before, let N be the model length, N_u the number of input channels, and N_d the number of disturbance channels, i.e., the number of components in the disturbance vector. Output channel j at time-step k, originally given by Equation (1), becomes

$$\begin{aligned} y_k^j = y_{ss}^j + \sum_{i=1}^{N_u}\left[\sum_{l=1}^{N-1} S_{i,j}^l \Delta u_{k-l}^i + S_{i,j}^N (u_{k-N}^i - u_{ss})\right] \\ + \sum_{i=1}^{N_d}\left[\sum_{l=1}^{N-1} D_{i,j}^l \Delta\delta_{k-l}^i + D_{i,j}^N (\delta_{k-N}^i - \delta_{ss})\right]. \end{aligned} \tag{9}$$

The new coefficients $D_{i,j}^l$ can be obtained as we described in Section II-B.2. In particular, we start at the equilibrium. Setting all the other inputs to 0, we apply a step to each of the disturbance components, measure the corresponding output, and obtain the coefficients as the ratio between output and input at every time-step l. We can group the disturbance FSR coefficients from channel i to output channel j, over H_p, as a square lower triangular matrix

$$\underline{\mathbf{D}}_{i,j} = \begin{bmatrix} D_{i,j}^1, & 0 & \dots & 0 \\ D_{i,j}^2, & D_{i,j}^1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ D_{i,j}^{H_p} & D_{i,j}^{H_p-1} & \dots & D_{i,j}^1 \end{bmatrix} \in \mathbb{R}^{H_p \times H_p}. \tag{10}$$

Then, for a known disturbance profile on channel i defined by

$$\Delta\boldsymbol{\delta}_k^i = [\Delta\delta_{k|k}^i, \dots, \Delta\delta_{k+H_p-1|k}^i]^T \in \mathbb{R}^{H_p}, \tag{11}$$

the prediction model is given by

$$\tilde{\mathbf{y}}_k = \tilde{\mathbf{S}}\Delta\tilde{\mathbf{u}}_k + \tilde{\mathbf{f}}_k + \underbrace{\begin{bmatrix} \underline{\mathbf{D}}_{1,1} & \dots & \underline{\mathbf{D}}_{N_d,1} \\ \vdots & \ddots & \vdots \\ \underline{\mathbf{D}}_{1,N_y} & \dots & \underline{\mathbf{D}}_{N_d,N_y} \end{bmatrix}}_{:=\tilde{\mathbf{D}}} \underbrace{\begin{bmatrix} \Delta\boldsymbol{\delta}_k^1 \\ \vdots \\ \Delta\boldsymbol{\delta}_k^{N_d} \end{bmatrix}}_{:=\Delta\tilde{\boldsymbol{\delta}}_k}. \tag{12}$$

The error to be used in the optimization problem (7) becomes

$$\mathbf{e}_k = \tilde{\mathbf{f}}_k + \tilde{\mathbf{S}}\Delta\tilde{\mathbf{u}}_k + \tilde{\mathbf{D}}\Delta\tilde{\boldsymbol{\delta}}_k - \tilde{\mathbf{r}}_k \in \mathbb{R}^{N_y \cdot H_p}. \tag{13}$$

B. Stochastic Formulation

The disturbance profile in Equation (11) is often affected by some degree of uncertainty. In particular, let us consider a random Gaussian noise, $\boldsymbol{\epsilon}_i$, corrupting the disturbance channel i independently at every time-step. That is, the joint distribution of the noise over the whole simulation horizon T is given by $\boldsymbol{\epsilon}^i \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{\boldsymbol{\epsilon}^i})$, $\boldsymbol{\Sigma}_{\boldsymbol{\epsilon}^i} = \operatorname{diag}\left({\sigma_1^i}^2, {\sigma_2^i}^2, \dots {\sigma_T^i}^2\right)$. When applying the predictive model in Equation (12), we consider a slice of such disturbance vector, that is,

$$\boldsymbol{\epsilon}_k^i = [\epsilon_{k|k}^i, \dots, \epsilon_{k+H_p-1|k}^i]^T \in \mathbb{R}^{H_p}, \tag{14}$$

The FSR model in Equation (9) depends on the disturbance variations ($\Delta\boldsymbol{\delta}$) at each time-step. When the disturbance is affected by noise, we can introduce the variation vector

$$\boldsymbol{\nu}_k^i \coloneqq \Delta\boldsymbol{\delta}_k^i + \Delta\boldsymbol{\epsilon}_k^i \in \mathbb{R}^{H_p}. \tag{15}$$

Note that, for suitable selection matrices $\mathbf{N}_k$ and $\boldsymbol{\Xi}_k$,

$$\Delta\boldsymbol{\epsilon}_k^i = \boldsymbol{\epsilon}_k^i - \boldsymbol{\epsilon}_{k-1}^i = \mathbf{N}_k\boldsymbol{\epsilon}^i - \boldsymbol{\Xi}_k\boldsymbol{\epsilon}^i. \tag{16}$$

Thus, $\boldsymbol{\nu}_k^i$ is distributed as

$$\boldsymbol{\nu}_k^i \sim \mathcal{N}\left\{\Delta\boldsymbol{\delta}_k^i, (\mathbf{N}_k - \boldsymbol{\Xi}_k)\,\boldsymbol{\Sigma}_{\boldsymbol{\epsilon}^i}\,(\mathbf{N}_k - \boldsymbol{\Xi}_k)^T\right\}. \tag{17}$$

Furthermore, let $\Delta\tilde{\boldsymbol{\delta}}_k$ be as of Equation (12), and $\tilde{\boldsymbol{\nu}}_k$ be the vector in which the $\boldsymbol{\nu}_k^i$ have been stacked vertically, for all channels. Its distribution is given by $\tilde{\boldsymbol{\nu}}_k \sim \mathcal{N}\left(\Delta\tilde{\boldsymbol{\delta}}_k, \boldsymbol{\Sigma}_{\tilde{\boldsymbol{\nu}}_k}\right)$, where the covariance matrix is block diagonal, and each block is the covariance of a single $\boldsymbol{\nu}_k^i$, since we assume that the noise on each disturbance channel is independent of the others. Since the predicted output in Equation (12) is a linear transformation of the noisy vector $\tilde{\boldsymbol{\nu}}_k$, it is in turn a Gaussian random variable with distribution

$$\tilde{\mathbf{y}}_k \sim \mathcal{N}(\tilde{\mathbf{f}}_k + \tilde{\mathbf{S}}\Delta\tilde{\mathbf{u}}_k + \tilde{\mathbf{D}}\Delta\tilde{\boldsymbol{\delta}}_k, \tilde{\mathbf{D}}\boldsymbol{\Sigma}_{\tilde{\boldsymbol{\nu}}_k}\tilde{\mathbf{D}}^T) \sim \mathcal{N}(\boldsymbol{\mu}_{\tilde{\mathbf{y}}_k}, \boldsymbol{\Sigma}_{\tilde{\mathbf{y}}_k}). \tag{18}$$

The distribution of the predicted output can be used to formulate a novel optimization problem accounting for stochasticity. In particular, let $\mathbb{E}[\cdot]$ denote the *expectation* of a random variable, and $\Pr[\cdot]$ the probability of an event. Then, Equation (7) becomes

$$\begin{aligned} \min_{\Delta\tilde{\mathbf{u}}_k} \ & \mathbb{E}\left[\mathcal{R}_k\right] \\ \text{subject to } & \mathbf{e}_k = \boldsymbol{\mu}_{\tilde{\mathbf{y}}_k} - \tilde{\mathbf{r}}_k, \\ & \Delta\mathbf{u}_k \in \Delta\mathbb{U}, \\ & \Pr\left[\tilde{\mathbf{y}}_k \in \mathbb{Y}\right] > 1 - \alpha, \\ & \alpha > 0. \end{aligned} \tag{19}$$

The risk $\mathcal{R}_k$ contains a quadratic form in the random variable $\tilde{\mathbf{y}}$. Let $\operatorname{Tr}[\cdot]$ be the *matrix trace* operator. Linearity of the expectation leads to the well-known result [20]

$$\mathbb{E}\left[\mathcal{R}_k\right] = \operatorname{Tr}\left[\tilde{\mathbf{Q}}\boldsymbol{\Sigma}_{\tilde{\mathbf{y}}_k}\right] + \mathbf{e}_k^T\tilde{\mathbf{Q}}\mathbf{e}_k + \Delta\tilde{\mathbf{u}}_k^T\tilde{\mathbf{R}}\Delta\tilde{\mathbf{u}}_k. \tag{20}$$

The first addendum can be dropped from the risk since it is independent of $\Delta\tilde{\mathbf{u}}_k$.

C. Chance Constraints

The problem in (19) includes probabilistic constraints that can be converted into deterministic ones to be implemented within the controller. Since the predicted output is a Gaussian random vector, we can leverage a concentration bound to retrieve a deterministic constraint on $\tilde{\mathbf{y}}$. In particular, let us assume that the set $\mathbb{Y}$ is given by the outputs satisfying

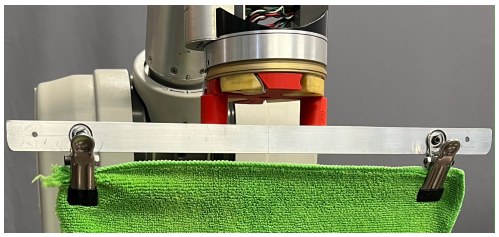

Fig. 2. The tool used in our real-world experiments. To resemble it, the distance between the upper corners in our simulation is fixed.

$\tilde{\mathbf{y}}_{min} \leq \tilde{\mathbf{y}}_k \leq \tilde{\mathbf{y}}_{max}$. For a suitable matrix $\mathbf{H}$ and vector $\mathbf{h}$, this constraint can be rewritten as

$$\mathbf{H}\tilde{\mathbf{y}}_k \leq \mathbf{h}. \tag{21}$$

This inequality results in a set of $2N_yH_p$ constraints that must hold simultaneously. We can now follow the derivations in [21]. In particular, let $\mathbf{H}_m$ be the m-th row of $\mathbf{H}$, and h_m the corresponding upper bound. By using the union bound, for $\alpha \in (0, 1]$, we can readily observe that

$$\Pr[\mathbf{H}\tilde{\mathbf{y}}_k \leq \mathbf{h}] = \Pr\left[\bigwedge_{m=1}^{2N_yH_p} \mathbf{H}_m\tilde{\mathbf{y}}_k \leq h_m\right] \tag{22}$$

$$\geq 1 - \sum_{m=1}^{2N_yH_p} \Pr\left[\mathbf{H}_m\tilde{\mathbf{y}}_k \geq h_m\right] \tag{23}$$

$$\geq 1 - \alpha, \tag{24}$$

provided that

$$\Pr\left[\mathbf{H}_m\tilde{\mathbf{y}}_k \geq h_m\right] \leq \alpha/\left(2N_yH_p\right), \ \forall m \in \{1, \ldots, 2N_yH_p\}. \tag{25}$$

This condition is the well-known uniform risk allocation policy, i.e., enforcing that the bounds on the probabilities of violating each constraint are equal [22]. Now, let $\Phi(\cdot)$ be the cumulative densitive function of the standard Gaussian distribution. Therefore, as shown in [21], after standardization, the individual constraints given by Equation 25 are fulfilled if and only if

$$\mathbf{H}_m\tilde{\mathbf{y}}_k \leq h_m - \Phi^{-1}\left[1 - \alpha/\left(2N_yH_p\right)\right]\tilde{\mathbf{\Sigma}}^{1/2}_{\mathbf{H}_m\tilde{\mathbf{y}}_k}. \tag{26}$$

IV. Experiments

In this section, we will show the experimental results of the application of QDMC to the task of cloth manipulation. After some preliminary definitions and tuning, we show two experiments that benchmark LMPC, QDMC, and their stochastic counterparts. After these simulations, we show a preliminary experiment on a real robot that indicates that QDMC is a feasible control algorithm in practice. All the simulated experiments were performed on a custom laptop (MacBook Pro 2019), and were implemented in `Matlab`, with the `CasADi` library for optimization and the `IPOPT` solver. The real-world experiment uses `ROS` and a 7-DOF Barrett WAM manipulator[1]. We refer the interested reader to [9] for an extensive and detailed empirical evaluation of the linear state-space representation of the cloth, the LMPC baseline, and its implementation, as used here.

[1]The code for simulations, the trajectories used, and the `ROS` nodes are available at `https://github.com/caedoard/qdmc-cloth-manipulation`.

A. Preliminary Definitions and Parameter Tuning

We start by introducing some definitions and parameter tuning related to our experiments.

1) Additional Constraints and Local Cloth Base: To avoid potential synchronization errors in a real-world bimanual setup, we instead used a single manipulator with an end-effector with two linked grasp points, as shown in Fig. 2, in all of our experiments. This end-effector introduced an additional constraint in the optimization problem (the distance between the two upper corners of the cloth must be constant), turning it into a *quadratically constrained quadratic programming problem* [23]. In practice, this modification did not slow down the optimization excessively, as already discussed in [9]. Moreover, in order to account for different cloth orientations w.r.t. the world's frame, a *local cloth base* for the COM was used, as introduced in [9].

2) Simulation-Oriented Model: Besides the COM, a *simulation-based* MPC algorithm requires the usage of a simulation-oriented model (SOM), which is as complex as possible, to mimic the behavior of the real-world system. In our case, we choose as SOM the mesh-based system presented in [5], which is treated as a black-box to get the measurements of the cloth *outputs* and close the output feedback loop. Note that no knowledge of the internal state-space representation of the SOM is needed in the design of QDMC. The COM is a 4×4 mesh, while the SOM is 7×7.

3) FSR coefficients: As outlined in Section II-B, QDMC requires to compute the step-response coefficients $S^l_{i,j}$ and $D^l_{i,j}$ of the linear cloth system, over a time window of $[0, N_{ss}]$. To guarantee that the system is in steady-state regime, we choose $N_{ss} = 500$. We report in Fig. 3 some illustrative examples for the FSR coefficients obtained with the methodology described in Section II-B.2.

4) Disturbance Model: When performing our experiments, we consider *wind* as a disturbance, which is often overlooked in standard robotic tasks, but is important in cloth manipulation. Wind is modeled as a 3-dimensional vector, representing the force exerted on each particle of the cloth's mesh in the Cartesian space. Let $\mathbf{v}_{wind} \in \mathbb{R}^3$ be the wind speed along each axis, A_{cloth} be the area of the cloth's surface, and ρ_a be the air density. The overall wind force exerted on the cloth's surface is

$$\mathbf{F}_{wind} = 1/2\rho_a A_{cloth}\mathbf{v}^2_{wind}. \tag{27}$$

Note that we assume that the wind force vector is the same for all the points in the mesh, except for the upper corners, which are grasped by the manipulator. All particles are subject to an additive acceleration that can be readily fit in the state-space model of Equation (8). As we will detail in the next subsection, the wind force is corrupted by i.i.d., zero-mean Gaussian noise.

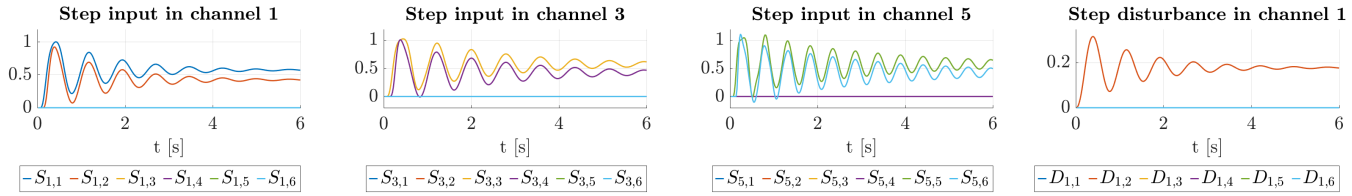

Fig. 3. Illustrative values of the FSR coefficients $S_{i,j}$ and $D_{i,j}$. Input and disturbance channels are numbered increasingly from the x coordinates of left (resp. right) upper corner. By *input channel* and *disturbance channel* we denote the entries of the input vector and of the disturbance vector. The outputs are numbered accordingly. In all these cases, the system is stable, making QDMC a viable control algorithm.

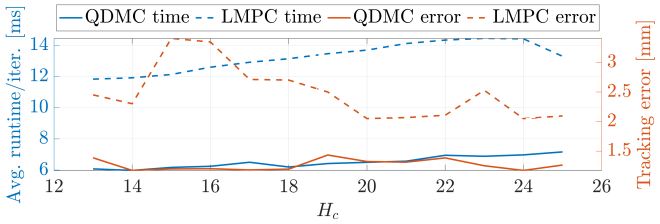

Fig. 4. Runtime vs. the tracking error of both LMPC and QDMC, on a sample trajectory, with different control horizons H_c and a fixed prediction horizon $H_p = 25$ observations. As expected, QDMC is much faster than LMPC.

5) Key Performance Indicators: In order to quantitatively evaluate the performance of the control algorithms, we can consider the following two key performance indicators (KPIs). Since we are interested in tracking the output of the cloth system, we can consider the *absolute tracking error* E, averaged over the whole time horizon T and the number of outputs. While the main goal of the control algorithm is to provide a proper tracking of the reference trajectory, it is also important to consider the time taken to run the MPC simulations. Hence, we also consider the *average runtime per iteration* across the simulation (T_{iter})[2].

6) Tuning the QDMC Control Horizon: Having defined suitable KPIs, we can now move on to consider two of the parameters of the QDMC algorithm, namely the control horizon, H_c, and the prediction horizon, H_p, introduced in Section II. As described in [13], QDMC benefits from a shorter control horizon than H_p, which is fixed to 25 observations. This is confirmed by the results in Fig. 4, where we observe that the control horizon can be reduced significantly, with no loss in the tracking accuracy, up to the minimum considered of 13 observations. One might argue that we could, in principle, shrink the control horizon of LMPC as well. While this is not a conventional operation for standard LMPC, it would be beneficial for the running time, as fewer optimization variables are used. However, if the control horizon of LMPC is reduced, there is no clear benefit in the tracking error, as shown in Fig. 4. Thus, the control horizon of LMPC is set equal to H_p, as it is usually done in the literature.

[2]Note that in our simulations the optimization step freezes the execution of the code, i.e., the program waits for the optimization problem to be solved before updating the state of the SOM. On the other hand, one could also consider as KPI the number of successfully completed optimization steps in the timespan $[0, T]$, if the implementation is asynchronous and the state of the SOM is updated independently of the optimization being completed or not (i.e., with a default control policy in the latter case). This KPI would be higher (better) for QDMC, as it is faster in solving the optimization problem. Conversely, in the real-robot experiment, the control algorithm does not freeze, to make the algorithms physically implementable.

TABLE I

MEAN TRACKING ERROR E, AND RUNTIME PER ITERATION T_{iter}, WITH STANDARD DEVIATION, EVALUATED ON 10 TRAJECTORIES WITHOUT DISTURBANCES, FOR LMPC, QDMC, AND A MODEL-FREE BASELINE.

	NoMDL	**LMPC**	**QDMC**
E [mm]	3.19 ± 1.19	2.96 ± 0.90	1.83 ± 0.85
T_{iter} [ms]	–	13.79 ± 0.98	5.97 ± 0.14

TABLE II

MEAN TRACKING ERROR E, AND RUNTIME PER ITERATION T_{iter}, WITH STANDARD DEVIATION, EVALUATED ON AN OBSTACLE AVOIDANCE TASK WITH A STOCHASTIC WIND DISTURBANCE. 10 DIFFERENT NOISY REALIZATIONS ARE CONSIDERED, AND $\alpha = 0.1$.

	NoMDL	**SLMPC**	**SQDMC**
E [mm]	15.57 ± 0.50	11.01 ± 0.42	10.50 ± 1.37
T_{iter}[ms]	–	33.14 ± 1.46	23.14 ± 1.12

B. Simulated Experiments

With the setup described before, we can now consider the KPIs of our QDMC implementation and LMPC on different simulated scenarios.

a) Simulated Trajectories without Disturbances: As a start, we can test both LMPC and QDMC (in their deterministic version) on a set of 10 different trajectories involving different speeds, durations, and orientations of the cloth w.r.t. the world's frame. Table I reports the mean KPIs for this first experiment, with standard deviation. The algorithms are also benchmarked with a simple baseline (NoMDL), in which the cloth model is ignored and the upper corners move following the reference of the lower corners. Our QDMC implementation exhibits a running-time that is roughly 43% of the LMPC's one. Furthermore, the tracking error is roughly 61% of the benchmark. Moreover, the model-free baseline exhibits a worse performance. An illustrative tracking result is shown in Fig. 5.

b) Chance Constraints and Obstacle Avoidance: As a second experiment, we can benchmark our chance-constrained SQDMC and the standard stochastic LMPC (SLMPC) [18] in presence of a noisy wind disturbance on

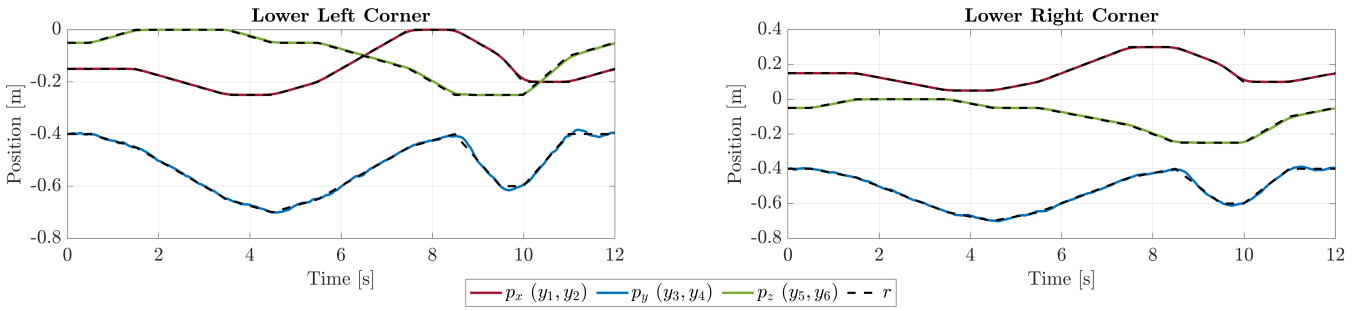

Fig. 5. The tracking obtained by QDMC on one of the considered simulated motions. QDMC is able to accomplish both slow (e.g., between 0 s and 4 s) and fast movements o the cloth (e.g., between 8 s and 10 s).

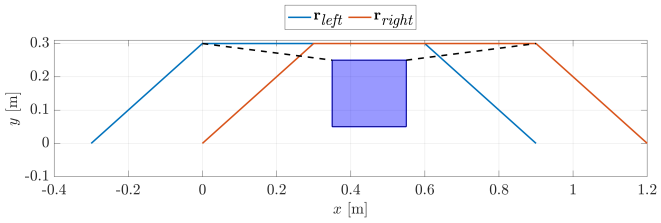

Fig. 6. View from above of our simulated obstacle avoidance setup. The bounding box of the obstacle is represented by the blue square. The plots also shows the reference signals for the lower corners, along with the two lines separating the different constraints used (black dashed).

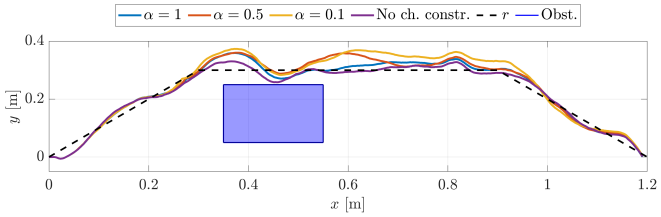

Fig. 7. View from above of the trajectory performed in the obstacle avoidance task by the right lower corner of the cloth, for various values of constraint violation probability α. As this parameter increases, the trajectory performed moves further from the obstacle. We further show the reference signal (black dashed) and the trajectory we would obtain without chance constraints (i.e., with QDMC instead of SQDMC). As expected, this gets very close to the obstacle.

the cloth. In order to test the proposed chance constraint formulation for SQDMC, we compare the algorithms on the task of obstacle avoidance. The robot is required to move the piece of cloth around an obstacle, without touching it. The obstacle is represented as a cube imposing a constraint in the x-y plane, as shown in Fig. 6. Note that the plane is split into three regions, each one corresponding to a different linear inequality. The nominal wind speed is a Gaussian curve peaking at -2.4 m/s on the y axis, and 0.8 m/s on the x axis, at 5 s, and is assumed to be known. Moreover, the noise standard deviation is 0.07 N. As we discussed in Section III-C, a chance constraint reshapes the original constraint by an amount that depends on the noise variance, based on the desired constraint violation probability α, defined in Section III-C. As expected, tuning this parameter has the effect of moving the cloth far from the obstacle, as we can appreciate from Fig. 7. Moreover, for a fixed value of α,

TABLE III

AVERAGE TRACKING ERROR E, AND RUNTIME PER ITERATION T_{iter}, EVALUATED IN A REAL-WORLD EXPERIMENTS.

	E [mm]	T_{iter} [ms]
LMPC	33.72	15.23
QDMC	32.30	10.89

Table II shows the mean and standard deviation of the KPIs across 10 noisy realizations of the disturbance profiles. The presence of the obstacle constraint significantly slows down the algorithms, nonetheless, SQDMC is much faster than SLMPC. The performance of NoMDL is poor due to the fact that the dynamics of the cloth (and therefore the effect of wind) are not taken into consideration.

C. Real-Robot Tracking

Besides testing our algorithm in simulation, we propose a preliminary implementation on a real manipulator that indicates that our QDMC controller can be ported to the real world. For the real-world implementation, we performed some slight modifications to the setup presented in the simulation part. In particular, to favor slower responses from the controller (needed for safety reasons) we reduced the prediction horizon to 20 observations, both for LMPC and QDMC. The feedback on the state of the cloth was measured with a Kinect camera and processed by a black-box vision node [3]. The experiment consists of running both LMPC and QDMC on the aforementioned setup, on a trajectory without wind. A visualization of the tracking for the right lower corner is shown in Fig. 8, while the KPIs for all outputs are reported in Table III. While the tracking errors for LMPC and QDMC are similar, QDMC allows to perform smoother trajectories than LMPC, as shown in Fig. 8. Moreover, this real-robot experiment confirms that QDMC is much faster than LMPC in solving the associated optimization problem.

V. CONCLUSION

In this work, we have studied the application of quadratic dynamic matrix control to dynamic cloth manipulation. Furthermore, we have introduced a suitable stochastic formulation of such a control algorithm, which is effective

[3]The code of this ROS node is available at `https://github.com/miguelard/cloth_point_cloud_segmentation`.

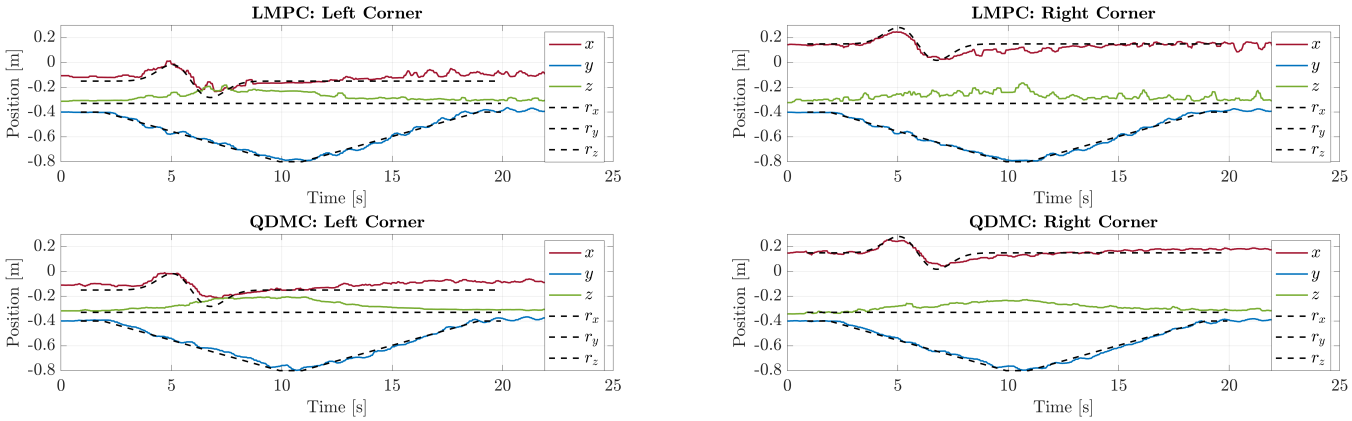

Fig. 8. The real-robot results of QDMC and LMPC on a reference trajectory for the right and left lower corners of the cloth. The reference is shown in black dashed in the picture.

in presence of disturbances corrupted by time-variant noise. Our approach has been shown to be successful in simulation, and a viable option for real-robot control. While the main focus of this work was on offering a showcase of the potentialities of dynamic matrix control applied to cloth manipulation, it constitutes the starting point of interesting future research. In particular, the joint effects of prediction horizon, control horizon and controller gains on disturbance rejection can be studied in depth in a sensitivity analysis, similar to [9]. Moreover, the vision feedback can be closed with a new vision node, specifically designed for tracking the lower corners of the cloth. This extension will allow to test the behavior of the proposed controller under disturbances in the real setup, potentially validating the solid performance shown in simulation. Lastly, the predictive controller can be enhanced with a *fast learning module*, such as the kernel-based approach discussed in [24], to infer and predict the disturbance profile directly from data.

REFERENCES

[1] J. Hietala, D. Blanco-Mulero, G. Alcan, and V. Kyrki, "Learning visual feedback control for dynamic cloth folding," *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1455–1462, 2022.

[2] Y. Avigal, L. Berscheid, T. Asfour, T. Kröger, and K. Goldberg, "Speedfolding: Learning efficient bimanual folding of garments," *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–8, 2022.

[3] T. Weng, S. M. Bajracharya, Y. Wang, K. Agrawal, and D. Held, "Fabricflownet: Bimanual cloth manipulation with a flow-based policy," *Conference on Robot Learning*, pp. 192–202, 2022.

[4] J. Maitin-Shepard, M. Cusumano-Towner, J. Lei, and P. Abbeel, "Cloth grasp point detection based on multiple-view geometric cues with application to robotic towel folding," *2010 IEEE International Conference on Robotics and Automation*, pp. 2308–2315, 2010.

[5] F. Coltraro, J. Amoros, M. Alberich-Carramiñana, and C. Torras, "An inextensible model for the robotic manipulation of textiles," *Applied Mathematical Modelling*, vol. 101, pp. 832–858, 2022.

[6] C. X. Zheng, A. Colomé, L. Sentis, and C. Torras, "Mixtures of controlled Gaussian processes for dynamical modeling of deformable objects," *Learning for Dynamics and Control Conference*, pp. 415–426, 2022.

[7] I. Garcia-Camacho, M. Lippi, M. C. Welle, H. Yin, R. Antonova, A. Varava, J. Borras, C. Torras, A. Marino, G. Alenyà, and D. Kragic, "Benchmarking bimanual cloth manipulation," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1111–1118, 2020.

[8] R. Jangir, G. Alenya, and C. Torras, "Dynamic cloth manipulation with deep reinforcement learning," *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4630–4636, 2020.

[9] A. Luque, D. Parent, A. Colomé, C. Ocampo-Martinez, and C. Torras, "Model predictive control for dynamic cloth manipulation: Parameter learning and experimental validation," *arXiv:2209.05798 [cs.RO]*.

[10] E. F. Camacho and C. Bordons Alba, *Model predictive control*. Springer science & business media, 2013.

[11] R. Mahony, I. Mareels, G. Bastin, and G. Campion, "Static-state feedback laws for output regulation of non-linear systems," *Control Engineering Practice*, vol. 4, no. 7, pp. 1009–1014, 1996.

[12] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: modelling, planning and control*. Springer, 2009.

[13] C. E. Garcia and A. Morshedi, "Quadratic programming solution of dynamic matrix control (QDMC)," *Chemical Engineering Communications*, vol. 46, no. 1-3, pp. 73–87, 1986.

[14] D. Q. Mayne, S. V. Raković, R. Findeisen, and F. Allgöwer, "Robust output feedback model predictive control of constrained linear systems," *Automatica*, vol. 42, no. 7, pp. 1217–1222, 2006.

[15] J. A. Paulson, A. Mesbah, S. Streif, R. Findeisen, and R. D. Braatz, "Fast stochastic model predictive control of high-dimensional systems," *IEEE Conference on decision and Control*, pp. 2802–2809, 2014.

[16] M. von Andrian and R. D. Braatz, "Offset-free input-output formulations of stochastic model predictive control based on polynomial chaos theory," *2019 American Control Conference (ACC)*, pp. 360–365, 2019.

[17] A. Bemporad and M. Morari, "Robust model predictive control: A survey," in *Robustness in identification and control*. Springer, 1999, pp. 207–226.

[18] A. Mesbah, "Stochastic model predictive control: An overview and perspectives for future research," *IEEE Control Systems Magazine*, vol. 36, no. 6, pp. 30–44, 2016.

[19] M. Torchio, C. Ocampo-Martinez, L. Magni, M. Serra, R. D. Braatz, and D. M. Raimondo, "Fast model predictive control for hydrogen outflow regulation in ethanol steam reformers," *IEEE American Control Conference (ACC)*, pp. 5044–5049, 2016.

[20] K. B. Petersen, M. S. Pedersen *et al.*, "The matrix cookbook," *Technical University of Denmark*, vol. 7, no. 15, p. 510, 2008.

[21] J. Grosso, C. Ocampo-Martinez, and V. Puig, "Chance-constrained model predictive control for drinking water networks," *Journal of Process Control*, vol. 24, pp. 504–516, 2014.

[22] A. Nemirovski and A. Shapiro, "Convex approximations of chance constrained programs," *SIAM Journal on Optimization*, vol. 17, no. 4, pp. 969–996, 2007.

[23] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[24] S. L. Brunton, M. Budišić, E. Kaiser, and J. N. Kutz, "Modern Koopman theory for dynamical systems," *SIAM Review*, vol. 64, no. 2, pp. 229–340, 2022.