Trajectory Generation for Robotic Systems with Contact Force Constraints

Jaemin Lee[1], Efstathios Bakolas[2], and Luis Sentis[3*]

Abstract— This paper presents a trajectory generation method for contact-constrained robotic systems such as manipulators and legged robots. Contact-constrained systems are affected by the interaction forces between the robot and the environment. In turn, these forces determine and constrain state reachability of the robot parts or end effectors. Our study subdivides the trajectory generation problem and the supporting reachability analysis into tractable subproblems consisting of a sampling problem, a convex optimization problem, and a nonlinear programming problem. Our method leads to significant reduction of computational cost. The proposed approach is validated using a realistic simulated contact-constrained robotic system.

I. INTRODUCTION

We aim to control contact-constrained robotic systems using optimal control in a computationally feasible way. In the case of high dimensional systems such as legged humanoid robots, a widely used control method is to create desired trajectories without prior knowledge of their feasibility, then rely on a feedback tracking controller to instantaneously realize them by projecting the controls into the null space of the constraints [1]–[3]. The drawback of this method is that the desired trajectories are often infeasible forcing the feedback controllers to fulfill them only in a least square error sense. Other methods used in humanoid robots rely on using simplified models to design feasible trajectories, e.g. using center of mass dynamics subject to contact constraints [4]. However, those methods cannot guarantee the feasibility of the trajectories because the mechanical degrees-of-freedom of the robot are ignored. In contrast, our approach generates feasible trajectories that fully comply with the robot's mechanics and dynamics as well as its contact state with its environment. In order to obtain feasible trajectories for the desired goals, we broadly employ reachability analysis. In particular contact constraints need different treatment than state constraints since they need to fulfill cone formulation requirements. This type of formulation has not been employed before for optimal control of robotic systems.

Reachability analysis is often used in optimal control for analyzing the performance and safety of various types of dynamical systems with bounded uncertainty, [5]–[9], for hybrid dynamical systems [10]–[14], and for stochastic systems [15], [16]. A common method to perform reachability analysis in the continuous domain is by solving the Hamilton-Jacobi-Bellman PDE [17]–[19]. But methods based on this process result in exponentially increasing computational cost as a function of the system's state and the discretization step. Therefore, for robotics it is not possible to use Hamilton-Jacobi-Bellan methods due to these limitations on scalability. Another approach is to employ the logarithmic norm of a Jacobian matrix producing over-approximated bounds of the reachable sets [20] and checking feasibility via simulations [21]. However, this kind of method does not incorporate system constraints as we do.

In robotics, configuration-based reachability analysis has been broadly used for motion planning of complex robotic systems. For instance, collision-free reachability maps in the configuration space are employed for the planning of humanoid motions [22] but without addressing contact forces or robot dynamics. The Monte-Carlo method has been used to obtain piecewise end-effector position samples by exploring configuration space samples and center of mass positions [23], however they ignore the robot's dynamics. Another idea is to connect nodes obtained via Rapidly-Exploring Random Trees using reachability analysis [24]. However, these two last works have not been extended to dynamical systems with contact force constraints.

Fundamentally, our method generates trajectories for contact force constrained robotic systems with differential constraints. We rely on randomly generated samples [25]–[28] and null-space projections associated with the differential constraints [29]. This approach is more efficient than employing Monte-Carlo methods to generate the set of samples fulfilling the system constraints. A key novelty in our sampling process is using a convex optimization stage to determine whether samples fulfill contact force constraints. We define the fraction of reachable samples (FRS) with respect to samples falling in the output regions that are feasible. The FRS, which has a dependency with the number of samples, is used as an indicator to guide the sampling process. We first solve a DP problem to obtain a candidate trajectory of the output guided by the properties of the samples. After this process we perform an additional optimal control procedure based on the reachability between the sampled points. We utilize a non-convex hull, which is the envelope of a set containing the output samples, to describe the set of reachable samples propagated from a given initial state over a finite time interval. By using small time intervals

[1]J. Lee is with the Department of Mechanical Engineering and Human-centered Robotics Laboratory, The University of Texas at Austin, Austin, TX, 78712, USA `jmlee87@utexas.edu`

[2]E. Bakolas is with Faculty of the Department of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, Austin, TX, 78712, USA `bakolas@austin.utexas.edu`

[3]L. Sentis is with Faculty of the Department of Aerospace Engineering and Engineering Mechanics, The University of Texas at Austin, Austin, TX, 78712, USA. *L. Sentis is the corresponding author. `lsentis@austin.utexas.edu`

for generating trajectories between adjacent regions we make use of model approximations that significantly reduce the computational burden required for state propagation. In addition, we increase computational efficiency by propagating only the dynamics of boundary states.

This paper is organized as follows. We introduce notations, the state space model of the constrained robotic system, and its time-discretization in Section II. In Section III, we explain how to obtain the samples that fulfill all system constraints and partition the sampled space based on the fraction of feasible samples. In Section IV, we characterize practical reachable sets and use them to design optimal control problems to generate trajectories over short time intervals. An example problem and simulation results are shown in Section V.

II. PRELIMINARIES

A. Notations

The sets of real n-dimensional vectors and $m \times n$ matrices are denoted by $\mathbb{R}^n$ and $\mathbb{R}^{m \times n}$, respectively. $\mathbb{R}^+$ and $\mathbb{R}^{++}$ indicate the sets of non-negative and strictly positive real numbers. $\mathbb{Z}^+$ and $\mathbb{Z}^{++}$ represent the sets of non-negative and strictly positive integers. When considering $z_1, z_2 \in \mathbb{Z}^+$ where $z_1 \leq z_2$, the interval of integers between z_1 and z_2 is represented as $[z_1, z_2]_d$ where d stands for discretization. The space of real symmetric $n \times n$ matrices is denoted by $\mathbb{S}_n$ and the spaces of positive semidefinite and positive definite matrices are denoted by $\mathbb{S}_n^+$ and $\mathbb{S}_n^{++}$, respectively. Given a matrix A, $A^\dagger$ and $\ker(A)$ denote the Moore-Penrose pseudo inverse and the kernel of A, respectively. Given multiple matrices $A_1, \ldots, A_k$ or a set $\mathcal{A} = \{A_1, \ldots, A_k\}$, $\text{Vertcat}(A_1, \ldots, A_k)$ or $\text{Vertcat}(\mathcal{A})$ denote the matrix formed by vertically concatenating the matrices A_1 to A_k. A diagonal matrix in $\mathbb{R}^{k \times k}$ with diagonal components $a_1, \cdots, a_k$ is denoted by $\text{diag}(a_1, \cdots, a_k)$. Considering a vector $a \in \mathbb{R}^n$, $\|a\|$ and $\|a\|_\infty$ denote the 2-norm and ∞-norm of the vector a, respectively. $\mathbb{E}[.]$ represents the probabilistic expectation operator. Given a set $\mathcal{A} \subseteq \mathbb{R}^n$, $\text{Int}(\mathcal{A})$ and $\text{Ext}(\mathcal{A})$ denote the interior and the exterior of $\mathcal{A}$. Furthermore, $\text{card}(\mathcal{A})$, $\text{bd}(\mathcal{A})$, and $\text{Nconv}(\mathcal{A})$ represent the cardinality, the boundary, and the non-convex hull of the set $\mathcal{A}$, respectively. Given two sets $\mathcal{A}_1$ and $\mathcal{A}_2$, the relative complement of $\mathcal{A}_1$ with respect to $\mathcal{A}_2$ is denoted by $\mathcal{A}_1 \backslash \mathcal{A}_2$, that is, $\mathcal{A}_1 \backslash \mathcal{A}_2 := \{x \in \mathcal{A}_1 : x \notin \mathcal{A}_2\}$. When $\mathcal{A} \subsetneq \mathbb{R}$, $\max(\mathcal{A})$ and $\min(\mathcal{A})$ denote the maximum and the minimum values among the elements of the set $\mathcal{A}$. Finally, if the k-th derivative of the function f exists and is continuous, the function is said to be of class $\mathcal{C}^k$.

B. State Space Model of Robotic System

The equation of motion of a multi-body dynamical system enduring contact forces with the environments can be described as follows:

$$M(q)\ddot{q} + b(q, \dot{q}) + p(q) = u + J_c^\top F_c \qquad (1)$$

where $q \in \mathbb{R}^{n_q}$, $u \in \mathbb{R}^{n_u}$, $M(q) \in \mathbb{S}_{n_q}^{++}$, $b(q, \dot{q}) \in \mathbb{R}^{n_q}$, $p(q) \in \mathbb{R}^{n_q}$, $F_c \in \mathbb{R}^{n_c}$, and $J_c \in \mathbb{R}^{n_c \times n_q}$ are the joint variables, input commands, mass/inertia matrix, Coriolis/centrifugal force, gravitational force, contact force, and the Jacobian matrix corresponding to the contact force constraint, respectively. We can transform the above equation into state space form as

$$\dot{x} = f_x(x) + f_u(x)u + f_c(x)F_c$$
$$f_x(x) = \begin{bmatrix} x_2 \\ M^{-1}(x_1)\left(-b(x_1, x_2) - p(x_1)\right) \end{bmatrix} \qquad (2)$$
$$f_u(x) = \begin{bmatrix} 0 \\ M^{-1}(x_1) \end{bmatrix}, \quad f_c(x) = \begin{bmatrix} 0 \\ J_c^\top(x_1) \end{bmatrix}$$

where $x = \begin{bmatrix} x_1^\top & x_2^\top \end{bmatrix}^\top \in \mathbb{R}^{n_x}$, $x_1 = q$, and $x_2 = \dot{q}$. $f_x : \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_x}$, $f_u : \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_x \times n_u}$, and $f_c : \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_x \times n_c}$. The joint position and velocity limits of the robot are described as the state constraints, $C_x(x) \leq 0$, and torque limits are described as input constraints, $C_u(u) \leq 0$. In additional, more complicated interactions such as contact wrench cones constraints, are described as mixed state-input constraints $C_{x,u}(x,u) \leq 0$. Here, the constraint functions are $C_x : \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_{C_x}}$, $C_u : \mathbb{R}^{n_u} \mapsto \mathbb{R}^{n_{C_u}}$, and $C_{x,u} : \mathbb{R}^{n_x + n_u} \mapsto \mathbb{R}^{n_{C_{xu}}}$, which are $\mathcal{C}^1$. We discretize the state space dynamics in (2) as:

$$x_{k+1} = x_k + B_1(x_k, u_k, F_{c,k})\Delta t$$
$$+ \frac{B_2(x_k, u_k, F_{c,k})(\Delta t)^2}{2} + \mathcal{O}\left((\Delta t)^2\right) \qquad (3)$$
$$= \mathcal{F}(x_k, u_k, F_{c,k})$$

where $k \in [0, N-1]_d$. Δt and $\mathcal{O}\left((\Delta t)^2\right)$ denote the time discretization increment and terms higher than 2nd order in Taylor series expansion, respectively. $B_1(x_k, u_k, F_{c,k}) := f_x(x_k) + f_u(x_k)u_k + f_c(x_k)F_{c,k}$ and $B_2(x_k, u_k, F_{c,k}) := \frac{\partial B_1(x_k, u_k, F_{c,k})}{\partial x} B_1(x_k, u_k, F_{c,k})$.

The output state of the robotic system is defined as $y = g(x)$ where $y \in \mathbb{R}^{n_y}$ and $g : \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_y}$. g is continuous and differentiable. Our problem concerns the generation of feasible state and input trajectories given desired output goals and initial system states maintaining the solid contact with respect to the nonlinear system model in (3).

III. SAMPLING-BASED APPROACH

As we mentioned earlier, sampling based methods enable to solve complex computational problems like ours. In this section, we introduce the way to obtain the samples fulfilling the given constraints.

A. Mathematical Definitions for Sampling

We draw random samples of the system's states from a Gaussian distribution $x \sim \mathcal{N}(\mu_x, \Sigma_x)$ where $x \in \mathbb{R}^{n_x}$, $\mu_x \in \mathbb{R}^{n_x}$, and $\Sigma_x \in \mathbb{S}_{n_x}^{++}$ denote the sampled state vector, its mean, and its covariance matrix, respectively, where $\mu_x := \mathbb{E}[x]$ and $\Sigma_x := \mathbb{E}\left[(x - \mu_x)(x - \mu_x)^\top\right]$. In robotics, we can define μ_x and Σ_x based on joint position and velocity limits.

Given n_e equality constraints, we describe them using the function $C_{e,[h_e]}(x) = 0$ where $h_e \in [1, n_e]_d$ is an index. This index is used to address multiple equality constraints

separately. Likewise, given n_i inequality constraints, we describe them using the function $C_{i,[h_i]}(x) \leq 0$ where $h_i \in [1, n_i]_d$ is also an index. $C_{e,[h_e]} : \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_{e,h_e}}$ and $C_{i,[h_i]} : \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_{i,h_i}}$ denote functions for the h_e-th equality and h_i-th inequality constraints, respectively, and both functions are $\mathcal{C}^1$ functions. Then, we define vectors of values for the equality and inequality constraint functions in terms of the state sample x.

$$\begin{aligned} \mathcal{V}_E(x) &:= \text{Vertcat}\left(C_{e,[1]}(x), \cdots, C_{e,[n_e]}(x)\right) \\ \mathcal{V}_I(x) &:= \text{Vertcat}\left(C_{i,[1]}(x), \cdots, C_{e,[n_i]}(x)\right) \end{aligned} \quad (4)$$

where $\mathcal{V}_E(x) \in \mathbb{R}^{\sum_{h_e=1}^{n_e} n_{h_e}}$ and $\mathcal{V}_I(x) \in \mathbb{R}^{\sum_{h_i=1}^{n_i} n_{h_i}}$. For dividing the indices of the constraint functions, we define two sets as follows:

$$\begin{aligned} H_e(x) &:= \left\{h_e \in \mathbb{Z}^{++} : \left\|C_{e,[h_e]}(x)\right\| \leq \varepsilon_h, h_e \in [1, n_e]_d\right\} \\ H_i(x) &:= \left\{h_i \in \mathbb{Z}^{++} : C_{i,[h_i]}(x) \leq 0, h_i \in [1, n_i]_d\right\}. \end{aligned} \quad (5)$$

In addition, $H_{\backslash e}(x) := [1, n_e]_d \backslash H_e(x)$ and $H_{\backslash i}(x) := [1, n_i]_d \backslash H_i(x)$, respectively. To split all constraints into feasible and infeasible constraints in terms of the random sample x, we define the vectors whose elements are function values with respect to the index sets defined in (5) as follows:

$$\begin{aligned} \mathcal{V}_e(x) &:= \text{Vertcat}\left(C_{e,[h]}(x) : \forall h \in H_e(x)\right) \\ \mathcal{V}_{\backslash e}(x) &:= \text{Vertcat}\left(C_{e,[h]}(x) : \forall h \in H_{\backslash e}(x)\right) \\ \mathcal{V}_i(x) &:= \text{Vertcat}\left(C_{i,[h]}(x) : \forall h \in H_i(x)\right) \\ \mathcal{V}_{\backslash i}(x) &:= \text{Vertcat}\left(C_{i,[h]}(x) : \forall h \in H_{\backslash i}(x)\right). \end{aligned} \quad (6)$$

Since all constraint functions are differentiable, we can compute the Jacobian matrices of the constraint functions such that $J_{C_e,[h]}(x) := \frac{\partial C_{e,[h]}}{\partial x}(x) \in \mathbb{R}^{n_{e,h} \times n_x}$ and $J_{C_i,[h]}(x) := \frac{\partial C_{i,[h]}}{\partial x}(x) \in \mathbb{R}^{n_{i,h} \times n_x}$. In addition, we can define matrices by vertically concatenating the Jacobian matrices of the constraint functions with respect to the categorized index in (5):

$$\begin{aligned} \mathcal{J}_e(x) &:= \text{Vertcat}\left(J_{C_e,[h]}(x) : \forall h \in H_e(x)\right) \\ \mathcal{J}_{\backslash e}(x) &:= \text{Vertcat}\left(J_{C_e,[h]}(x) : \forall h \in H_{\backslash e}(x)\right) \\ \mathcal{J}_i(x) &:= \text{Vertcat}\left(J_{C_i,[h]}(x) : \forall h \in H_i(x)\right) \\ \mathcal{J}_{\backslash i}(x) &:= \text{Vertcat}\left(J_{C_i,[h]}(x) : \forall h \in H_{\backslash i}(x)\right) \end{aligned} \quad (7)$$

where all Jacobian matrices $\mathcal{J}_e(x)$, $\mathcal{J}_{\backslash e}(x)$, $\mathcal{J}_i(x)$, and $\mathcal{J}_{\backslash i}(x)$ are assumed as full row rank matrices. In order to have all sample states satisfying the state constraints, we will solve a least square error problem using the Moore-Penrose pseudo inverse.

B. Update of Samples for the State Constraints

We update the state sample for fulfilling all state constraints in the least square error sense. In addition, the orthogonal projection onto the kernel space of the constraint function is utilized to prevent modifications of the function values after fulfilling the constraints. Given a state sample $x^l \in \mathcal{X}$ in the l-th update iteration, we can compute the vertically concatenated Jacobian matrices $\mathcal{J}_e(x^l)$ and $\mathcal{J}_{\backslash e}(x^l)$.

The orthogonal projection onto $\ker\left(\mathcal{J}_e(x)\right)$ is defined as

$$P_e(x^l) := I_{n_x} - \mathcal{J}_e^\dagger(x^l)\mathcal{J}_e(x^l) \quad (8)$$

where I_{n_x} is the $n_x \times n_x$ identity matrix and $P_e : \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_x \times n_x}$. Based on the Jacobian matrix $\mathcal{J}_{\backslash e}(x^l)$ and $P_e(x^l)$, the sampled state is updated as follows:

$$x^{l+1} = x^l - \alpha P_e(x^l)\left(\mathcal{J}_{\backslash e}\left(x^l\right) P_e\left(x^l\right)\right)^\dagger \mathcal{V}_{\backslash e}(x^l) \quad (9)$$

where $l \in [0, N_{\text{iter}}]_d$ and the initial state is the originally sampled state, that is, $x^0 = x$. In addition, $\alpha \in \mathbb{R}^{++}$ is the gain for regulating the convergence speed to exponentially reduce $\left\|\mathcal{V}_{\backslash e}\left(x^l\right)\right\|$ to 0. When $\left\|\mathcal{V}_{\backslash e}\left(x^l\right)\right\| \leq \varepsilon_e$, the iteration process ends and the state sample x is altered to the result x^l in the set $\mathcal{X}$. If $\left\|\mathcal{V}_{\backslash e}\left(x^{N_{\text{iter}}}\right)\right\| > \varepsilon_e$, the state sample is discarded from the set $\mathcal{X}$ for the computational efficiency of the method.

After updating all samples in $\mathcal{X}$, we do not want to update the function values for which the constraints are already fulfilled. Therefore, an augmented Jacobian is defined with respect to the state x^l as

$$\mathcal{J}_{aug}(x^l) := \text{Vertcat}\left(\mathcal{J}_e(x^l), \mathcal{J}_i(x^l)\right) \quad (10)$$

where $P_{aug}(x^l)$ is computed in the same manner in (8). Using the Jacobian $\mathcal{J}_{\backslash i}(x^l)$ projected onto $\ker\left(\mathcal{J}_{aug}(x^l)\right)$, we can update the state sample without any modification of function values for already fulfilled constraints. The update of state sample is

$$\begin{aligned} x^{l+1} &= x^l + \alpha P_{aug}(x^l)\left(\mathcal{J}_{\backslash i}(x^l)P_{aug}(x^l)\right)^\dagger \mathcal{E}_i(x^l) \\ \mathcal{E}_i(x^l) &:= \mathcal{V}_{\backslash i}^{int}(x^l) - \mathcal{V}_{\backslash i}(x^l) \end{aligned} \quad (11)$$

where $\mathcal{V}_{\backslash i}^{int}(x^l)$ denotes the vector vertically concatenating the interior points fulfilling the constraints $C_{i,[h]}(x^l) \leq 0$ for all $h \in H_{\backslash i}(x^l)$. This update is terminated when the state x^l fulfills the inequality constraints and the existing component x is replaced by x^l. If the state update cannot be satisfied within N_{iter} iterations, the state sample is discarded from $\mathcal{X}$. Therefore, all components in the state set $\mathcal{X}$ fulfill the state constraints.

C. Sample Evaluation Given Contact Force Constraints

In this section, we check both input and contact force constraints in terms of the samples in $\mathcal{X}$. Optimization techniques are broadly utilized to find the contact force for the mechanical systems, thus, we also formulate the optimization problem with quadratic cost function to obtain the contact force in terms of the samples as follows:

$$\min_{F_c} \quad F_c^\top W_c F_c \quad (12a)$$

$$\text{s.t.} \quad x_{k+1} = \mathcal{F}\left(x_k, u, F_c\right), \quad x_k, x_{k+1} \in \mathcal{X}, \quad (12b)$$

$$\mathcal{D}_c(x_k)F_c \leq 0, C_u(u) \leq 0, C_{x,u}(x_k, u) \leq 0 \quad (12c)$$

where $\mathcal{D}_c : \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_{c'} \times n_c}$ denotes the unilateral constraints using a polyhedral approximation of the friction cone [30]. Solving this optimization problem for all states in $\mathcal{X}$, the set of feasible states in discrete state space $\mathcal{X}_R$ can be defined as the collection of the samples which result in the optimal contact force and input command given (12).

D. Fraction of Reachable Samples

Using the set $\mathcal{X}_R$, we will formulate a DP trajectory optimization problem. To that end, we define a measure called the fraction of reachable samples (FRS) regarding output regions. The output space is partitioned into n_o regions as $Y_m := \{y \in \mathbb{R}^{n_y} : \|y - \widetilde{y}_m\|_\infty \leq \delta_m\} \subsetneq \mathbb{R}^{n_y}$, where $m \in [1, n_o]_d$ and $\widetilde{y}_m \in \mathbb{R}^{n_y}$ denotes the center of the regions Y_m. Then, we define the subset of $\mathcal{X}_R$ as follows:

$$\mathcal{X}_{Y_m} := \{x \in \mathbb{R}^{n_x} : g(x) \in \text{Int}(Y_m), \forall x \in \mathcal{X}_R\}. \quad (13)$$

Using the set $\mathcal{X}_{Y_m}$, the FRS quantifies how many random samples are allocated in the region Y_m.

Definition 1. *Suppose that $m \in [0, n_0]_d$ is given and that $\mathcal{X}_R$ is not an empty set. The Fraction of Reachable Samples (FRS) is defined as the ratio:*

$$\mathfrak{F}_{m|N_s} := \frac{\text{card}(\mathcal{X}_{Y_m})}{\text{card}(\mathcal{X}_R)}. \quad (14)$$

where N_s is $\text{card}(\mathcal{X}_R)$ and $\mathcal{X}_{Y_m}$ is defined in (13).

We need sufficient number of samples so that the FRS is a reliable property. In order to check the rate of change of the FRS with respect to the number of samples, we define its gradient as follows:

$$\mathcal{G}\left(\mathfrak{F}_{m|N_s}, \Delta N_s\right) := \frac{\mathfrak{F}_{m|(N_s + \Delta N_s)} - \mathfrak{F}_{m|N_s}}{\Delta N_s} \quad (15)$$

where ΔN_s denotes the number of new samples generated in $\mathcal{X}_R$. Using the gradient of FRS, we address the convergence of the FRS with respect to N_s in the following theorem:

Theorem 1. *Let $\Delta N_s \in \mathbb{Z}^{++}$ and $m \in [0, n_o]_d$ be given and ΔN_s be less than N_s. Then, $\mathcal{G}\left(\mathfrak{F}_{m|N_s}, \Delta N_s\right)$ will converge to zero as $N_s \to \infty$, that is,*

$$\lim_{N_s \to \infty} \mathcal{G}\left(\mathfrak{F}_{m|N_s}, \Delta N_s\right) = 0 \quad (16)$$

In other words, $\lim_{N_s \to \infty} \mathfrak{F}_{m|N_s} = \overline{\mathfrak{F}}_m$ where $\overline{\mathfrak{F}}_m$ represents a convergence value for the FRS.

Proof. Suppose $\text{card}(\mathcal{X}_R) = N_s$ and $\text{card}(\mathcal{X}_{Y_m}) = N_m$. We consider a case in which all new samples are allocated within Y_m, which means the rate of change of the FRS is maximum. Hence, $|\mathcal{G}(\mathfrak{F}_{m|N_s}, \Delta N_s)|$ is bounded such that

$$|\mathcal{G}\left(\mathfrak{F}_{m|N_s}, \Delta N_s\right)| \leq \left| \frac{N_s \mathfrak{F}_{m|N_s} + \Delta N_s}{\Delta N_s (N_s + \Delta N_s)} - \frac{\mathfrak{F}_{m|N_s}}{\Delta N_s} \right|$$
$$= \left| \frac{N_s \Delta N_s - N_m \Delta N_s}{N_s \Delta N_s (N_s + \Delta N_s)} \right| = \left| \frac{N_s - N_m}{N_s (N_s + \Delta N_s)} \right|$$

Since $\Delta N_s, N_m \ll N_s$,

$$0 \leq \lim_{N_s \to \infty} |\mathcal{G}\left(\mathfrak{F}_{m|N_s}, \Delta N_s\right)| \leq \lim_{N_s \to \infty} \left| \frac{N_s - N_m}{N_s (N_s + \Delta N_s)} \right|$$
$$= \lim_{N_s \to \infty} \left| \frac{1 - N_m/N_s}{N_s + \Delta N_s} \right| = \lim_{N_s \to \infty} \left| \frac{1}{N_s} \right| = 0$$

It is therefore implied that $\lim_{N_s \to \infty} \mathcal{G}\left(\mathfrak{F}_{m|N_s}, \Delta N_s\right) = 0$. $\square$

For numerical implementation N_s is determined by using the inequality $\max(G_{N_s}) \leq \varepsilon_G$, where $G(N_s) := \{n_\mathcal{G} \in \mathbb{R}^+ : n_\mathcal{G} = |\mathcal{G}\left(\mathfrak{F}_{m|N_s}, \Delta N_s\right)|, m \in [1, n_o]_d\}$ and ε_G is a pre-defined small positive threshold.

Let us consider the set of output samples:

$$\mathcal{Y}_m := \{y \in \mathbb{R}^{n_y} : y = g(x), \forall x \in \mathcal{X}_{Y_m}\} \quad (17)$$

The mean vector and the covariance matrix of samples $y \in \mathcal{Y}_m$ are represented by $\mu_{y_m} := \mathbb{E}[y]$ and $\Sigma_{\mathcal{Y}_m} := \mathbb{E}\left[(y - \mu_{y_m})(y - \mu_{y_m})^\top\right]$, respectively. To analyze the patterns of the output samples, the covariance matrix $\Sigma_{\mathcal{Y}_m}$ is decomposed using the singular vale decomposition such that

$$\Sigma_{\mathcal{Y}_m} = U_m \Omega_m V_m^\top \quad (18)$$

where U_m and V_m are unitary matrices in $\mathbb{R}^{n_y \times n_y}$ and Ω_m represents a diagonal singular value matrix in $\mathbb{R}^{n_y \times n_y}$. We next define a vector that is associated with the largest singular value, to find the principle direction of the distribution of samples in $\mathcal{Y}_m$.

Definition 2. *Let $V_m = [\text{col}_1(V_m), \ldots, \text{col}_{n_y}(V_m)]$ and $\Omega_m = \text{diag}\left(\sigma_{m,1}, \ldots, \sigma_{m,n_y}\right)$ where $\sigma_{m,i}$ denotes the singular values of $\Sigma_{\mathcal{Y}_m}$. The principal singular vector (PSV) of $\mathcal{Y}_m$ is defined as the singular vector corresponding to the largest singular value such that*

$$\mathfrak{P}_m := \text{col}_j(V_m), \quad \sigma_{max} = \sigma_{m,j} \quad (19)$$

where $\mathfrak{P}_m \in \mathbb{R}^{n_y}$ and σ_{max} denotes the maximum singular value of $\Sigma_{\mathcal{Y}_m}$.

The PSV $\mathfrak{P}_m$ is a critical property for constructing the transition dynamics of the DP process.

E. Dynamic Programming based on Sample Properties

After computing the FRS, $\overline{\mathfrak{F}}_m$, and the PSV, $\mathfrak{P}_m$, with sufficient samples, we formulate the DP problem using a Markov Decision Process (MDP) to create an end-to-end trajectory. To start with, we define a discrete node associated with the output region Y_m as follows:

$$s_m := \text{node}(Y_m), m \in [1, n_o]_d. \quad (20)$$

For DP, we represent the FRS and the PSV corresponding to the node index s_m as $\mathfrak{F}_{s_m} = \overline{\mathfrak{F}}_m$ and $\mathfrak{P}_{s_m} = \mathfrak{P}_m$. In addition, we employ value iteration to solve the DP using the Bellman equation as follows:

$$\mathfrak{D}^\star(s^l) = \max_a \left(\mathfrak{R}(s^l) + \gamma \sum_{s^{l+1} \in S} \mathfrak{T}_a(s^l, s^{l+1}) \mathfrak{D}^\star(s^{l+1}) \right)$$

where $s^l \in S$, a, $\mathfrak{R}$, $\mathfrak{T}_a$, γ, and S are the node in the l-th iteration of the algorithm, the action, the reward, the transition dynamics, the discount factor, and the set of nodes, respectively. The reward function is defined to include as many samples as possible for the result of the DP:

$$\mathfrak{R}(s^l) = \begin{cases} -\eta_1 & \text{if } \mathfrak{F}_{s^l} = 0 \\ \eta_2 + K_\mathfrak{F} \mathfrak{F}_{s^l} & \text{if } s^l = s_\phi \\ -\eta_3 + K_\mathfrak{F} \mathfrak{F}_{s^l} & \text{else} \end{cases} \quad (21)$$

where s_ϕ denotes the node associated with the region containing the goal output, ϕ. $K_{\mathfrak{F}} \in \mathbb{R}^{++}$ and $\eta_{1,2,3} \in \mathbb{R}^{++}$ are the gain for $\mathfrak{F}_{s^l}$ and the offset values for the reward functions, respectively.

If the PSV of $\Sigma_{\mathcal{Y}_m}$ associated with the node s_m is not well defined, e.g., when the distribution of samples is isometric or uniform, the transition dynamics of the DP process is considered deterministic. Otherwise, the transition dynamics is computed by the direction cosine between the PSV $\mathfrak{P}_{s^l}$ and an action vector $\pi(a) \in \mathbb{R}^{n_y}$ such that

$$\Pi\left(s^l, s^{l+1}, a\right) := \max\left(\left\{0, \frac{\mathfrak{P}_{s^l}^\top \pi(a)}{|\mathfrak{P}_{s^l}|\, |\pi(a)|}\right\}\right). \quad (22)$$

Then, the transition dynamics is obtained by normalizing the direction cosine as follows:

$$\mathfrak{T}_a\left(s^l, s^{l+1}\right) = \frac{\Pi\left(s^l, s^{l+1}, a\right)}{\sum_{\hat{s} \in \hat{S}} \Pi\left(s^l, \hat{s}, a\right)} \quad (23)$$

where $\hat{s}$ is an individual neighboring node of s^l, and $\hat{S}$ is the collection of all neighboring nodes, respectively. The reward and transition dynamics are designed to exclude infeasible output regions ($\mathfrak{F}_{s_m} = 0$).

Using DP, we obtain a sequential set of nodes $\mathcal{S}^\star := \left\{s^1, s^2, \cdots, s^{n_{dp}}\right\}$. The set is converted to $\mathcal{Q}^\star := \left\{\text{node}^{-1}(s^1), \text{node}^{-1}(s^2), \cdots, \text{node}^{-1}(s^{n_{dp}})\right\}$ where $\text{node}^{-1}(s_m)$ denotes the mapping of the node s_m to its corresponding output region Y_m. This implies that the end-to-end trajectory generation problem can be formulated as a trajectory generation problems between the output regions in $\mathcal{Q}^\star$.

IV. TRAJECTORY GENERATION VIA REACHABILITY ANALYSIS

Given the resulting sequence of output regions $\mathcal{Q}^\star$ from the DP, we now generate feasible trajectories between output regions connecting $\text{node}^{-1}(s^l)$ to $\text{node}^{-1}(s^{l+1})$ for all $l \in [1, n_{dp}-1]_d$. After generating trajectories for all $l \in [1, n_{dp}-1]_d$, an entire end-to-end trajectory can be generated with feasibility guarantees.

A. Reachability Analysis

As discussed above, we seek to solve a nonlinear optimization problem for the sequence of output regions. The nonlinear optimization strategy requires a feasible initial condition so that the solution can converge to the local optimal point. For this reason, we compute a reachable set fulfilling the constraints given an initial state. The reachable set is defined for a continuous system with the contact force constraint as:

Definition 3. *Given an initial state $x_0 \in \mathbb{R}^{n_x}$ and a time instance t, the reachable state set of the robotic system given a contact force constraint is defined as:*

$$\begin{aligned} \mathcal{R}_t^x(x_0) := \{z \in \mathbb{R}^{n_x} : z = x(t), \exists u([t_0, t]), \\ \exists F_c([t_0, t]), C_x(x(t)) \leq 0, C_u(u(t)) \leq 0, \\ C_{x,u}(x(t), u(t)) \leq 0, \mathcal{D}(x(t))F_c(t) \leq 0, x(0) = x_0, \\ t \in [t_0, t], \dot{x} = f_x(x) + f_u(x)u + f_c(x)F_c\}. \end{aligned} \quad (24)$$

This reachable state set can be extended over a time interval $[t_i, t_{i+1}]$ as follows:

$$\mathcal{R}_{[t_i, t_{i+1}]}^x(x_0) := \bigcup_{t \in [t_i, t_{i+1}]} \mathcal{R}_t^x(x_0). \quad (25)$$

Using the reachable set for the states, the reachable set for the outputs is defined as:

$$\mathcal{R}_t^y(x_0) := \{\nu \in \mathbb{R}^{n_y} : \nu = g(\overline{x}), \forall \overline{x} \in \mathcal{R}_t^x(x_0)\} \quad (26)$$

which can be extended to the time interval $[t_i, t_{i+1}]$ as:

$$\mathcal{R}_{[t_i, t_{i+1}]}^y(x_0) := \bigcup_{t \in [t_i, t_{i+1}]} \mathcal{R}_t^y(x_0) \quad (27)$$

Since this paper considers the discrete state space model coupled with a sampling-based approach, we approximate the reachable set, e.g. Eq. (24), with a discrete state space model.

Before computing the reachable sets, we check the state bounds using the discrete state space model (3):

$$\begin{aligned} \|x_{k+1} - x_k\| = T\left\|B_1 + \frac{T}{2}B_2 + \frac{\mathcal{O}\left(T^2\right)}{T}\right\| \\ \leq T\left\|(I + \mathcal{Z}_1)\right\| \|B_1\| + K|T| = \mathcal{Z}_2(T, x_k) \end{aligned} \quad (28)$$

where $\mathcal{Z}_1 := J_x(x_k) + J_u(x_k)u_k + J_{\bar{c}}(x_k)F_{c,k}$, $J_x(x_k) = \frac{\partial f_x}{\partial x}(x_k)$, $J_u(x_k) = \frac{\partial f_u}{\partial x}(x_k)$, and $J_{\bar{c}}(x_k) = \frac{\partial f_c}{\partial x}(x_k)$. T is the time increment and it should be small satisfying $\mathcal{O}\left(T^2\right) < K|T|$. Also, the norm of the output update is bounded by:

$$\begin{aligned} \|y_{k+1} - y_k\| = \|J_y(x_k)(x_{k+1} - x_k)\| \\ \leq \|J_y(x_k)\| \|x_{k+1} - x_k\| = \mathcal{Z}_3(T, x_k) \end{aligned} \quad (29)$$

where $J_y(x_k) = \frac{\partial g}{\partial x}(x_k)$. Based on (29), we define the closed ball in the output space as follows:

$$\mathcal{B}^y(T, x_0) := \{y \in \mathbb{R}^{n_y} : \|y - g(x_0)\| \leq \mathcal{Z}_3(T, x_0)\}. \quad (30)$$

Since the reachable output set is a subset of $\mathcal{B}^y(T, x_0)$, it is necessary to consider a time interval wider than $[0, T_{\backslash min}]$ in the reachability analysis, where $T_{\backslash min} := \min(\{t : t = k\Delta t, \phi \in \mathcal{B}^y(t, x_0), k \in \mathbb{Z}^+\})$.

The reachable set is numerically constructed using the discrete state space model. To start, we formulate the optimization problem with the state, x_k, and input, u:

$$\min_{F_c, x_{k+1}} \quad F_c^\top W_c F_c \quad (31a)$$

$$\text{s.t.} \quad x_{k+1} = \mathcal{F}(x_k, u, F_c), \mathcal{D}_c(x_k)F_c \leq 0, \quad (31b)$$

$$C_{x,u}(x_k, u) \leq 0, C_x(x_{k+1}) \leq 0. \quad (31c)$$

Let us consider an initial state, $x_k = x_0$. We draw random input samples from a Gaussian distribution $u \sim \mathcal{N}(\mu_u, \Sigma_u)$ where $\mu_u := \mathbb{E}(u)$ and $\Sigma_u := \mathbb{E}[(u - \mu_u)(u - \mu_u)^\top]$ denote the mean vector and the covariance matrix of the input samples, respectively. For all of the generated input samples, we only select samples that fulfill the input constraints.

Via our numerical strategy, the reachable state set at Δt is approximated as $\overline{\mathcal{R}}_{\Delta t}^x(x_0)$ in the form of the collection

x_1 from the optimization (31) for all input samples. The reachable set of the output samples is computed as:

$$\overline{\mathcal{R}}^y_{\Delta t} := \{\nu \in \mathbb{R}^{n_y} : \nu = g(\overline{x}), \overline{x} \in \overline{\mathcal{R}}^x_{\Delta t}(x_0)\}. \tag{32}$$

We can extend this method to obtain the reachable set over multiple time steps. Suppose a time step, $T_k = k\Delta t$, and a reachable set for the previous time step, $\overline{\mathcal{R}}^x_{0,T_{k-1}}(x_0)$. Our strategy is to solve the optimization problem in (31) with respect to the samples in the set $\{\overline{x} : g(\overline{x}) \in \mathrm{bd}(\overline{\mathcal{R}}^y_{[0,T_{k-1}]}(x_0))\}$ instead of with respect to x_k. Then the computation of the reachable set at T_k, $\overline{\mathcal{R}}^x_{T_k}(x_0)$ results in a collection of x_{k+1} for all $\overline{x}$ and u. Based on $\overline{\mathcal{R}}^x_{[0,T_{k-1}]}(x_0)$ and $\overline{\mathcal{R}}^x_{T_k}(x_0)$, we can obtain the reachable state set over the time horizon $[0, T_k]$ and the corresponding output set as:

$$\overline{\mathcal{R}}^x_{[0,T_k]}(x_0) = \overline{\mathcal{R}}^x_{[0,T_{k-1}]}(x_0) \bigcup \overline{\mathcal{R}}^x_{T_k}(x_0) \tag{33a}$$

$$\overline{\mathcal{R}}^y_{[0,T_k]}(x_0) = \{\nu \in \mathbb{R}^{n_y} : \nu = g(\overline{x}), \overline{x} \in \overline{\mathcal{R}}^x_{[0,T_k]}(x_0)\} \tag{33b}$$

where $k \geq 2$, $\overline{\mathcal{R}}^x_{[0,T_1]}(x_0) = \overline{\mathcal{R}}^x_{\Delta t}(x_0)$, and $\overline{\mathcal{R}}^y_{[0,T_1]}(x_0) = \overline{\mathcal{R}}^y_{\Delta t}(x_0)$. In terms of the computational efficiency, the proposed strategy can reduce the computational cost to $\mathcal{O}(N_b^{N_u})$, where N_b and N_u denote the number of samples at the boundary of the reachable set and the number of input samples, respectively.

As mentioned above, we must check whether the desired goal belongs to the reachable set of output samples before solving the optimal control problem given an initial state. We first approximate $\overline{\mathcal{R}}^x_{[0,T]}(x_0)$ by a non-convex hull. We then check whether the desired goal output y_ϕ is reachable or not: $y_\phi \in \mathrm{Nconv}\left(\overline{\mathcal{R}}^y_{[0,T]}(x_0)\right)$, where $\overline{\mathcal{R}}^y_{[0,T]}(x_0)$ is addressed in (33b). If we find a T satisfying $y_\phi \in \mathrm{Nconv}\left(\overline{\mathcal{R}}^y_{[0,T]}(x_0)\right)$, then we can conclude that it implies that the desired goal output position y_ϕ is achievable over the time interval $[0, t]$ where $t \geq T \geq T_{\backslash min}$.

B. Optimal Control

To generate the whole trajectory, we recursively solve the optimal control problem between $\mathrm{node}^{-1}(s^l)$ and $\mathrm{node}^{-1}(s^{l+1})$ for all $l \in [1, n_{dp} - 1]_d$ in $\mathcal{Q}^\star$, as defined in Section III. To begin the optimal control process, the time interval T^l and the desired output $\widetilde{y}^{l+1}$ for the l-th optimal control problem are determined through the previous reachability analysis and initial state, x_0^l. We also define a performance measure of the optimal control problem in the discrete time domain as follows:

$$\mathcal{L}^l(u(.), N^l) := \sum_{k=0}^{N^l-1} \left(u_k^\top W_1 u_k + F_{c,k}^\top W_2 F_{c,k}\right) + L^l(\xi_N)$$

$$L^l(\xi_N) := \left(\widetilde{y}^{l+1} - \xi_N\right)^\top W_3 \left(\widetilde{y}^{l+1} - \xi_N\right)$$

where ξ denotes the trajectory of the output and $N^l = T^l/\Delta t$. $W_1 \in \mathbb{S}^{++}_{n_u}$, $W_2 \in \mathbb{S}^{++}_{n_c}$, and $W_3 \in \mathbb{S}^{++}_{n_y}$ are the weighting matrices for the components of the performance measure.

The optimal control problem is formulated using the reachable sets, the constraints, and the discrete state space model as follows:

$$\min_{\zeta(.), u(.)} \quad \mathcal{L}^l(u(.), N^l) \tag{34a}$$

$$\text{s.t.} \quad \xi_k \in \overline{\mathcal{R}}^y_{[0,T^l]}(x_0^l) \tag{34b}$$

$$\zeta_k \in \overline{\mathcal{R}}^x_{[0,T^l]}(x_0^l) \tag{34c}$$

$$\zeta_{k+1} = \mathcal{F}\left(\zeta_k, u_k, F_{c,k}\right) \tag{34d}$$

$$C_x(\zeta_k) \leq 0, C_u(u_k) \leq 0, \tag{34e}$$

$$C_{x,u}(\zeta_k, u_k) \leq 0, \mathcal{D}(\zeta_k)F_{c,k} \leq 0, \tag{34f}$$

$$\zeta_0 = x_0^l, \xi_0 = g(x_0^l) \tag{34g}$$

where ζ denotes the state trajectory. As a result of the optimal control problem, we obtain the state trajectory to control the system output from region $\mathrm{node}^{-1}(s^l)$ to region $\mathrm{node}^{-1}(s^{l+1})$ in $\mathcal{Q}^\star$:

$$\Psi_l := \mathrm{Vertcat}\left(\zeta_k^\top : \forall k \in [0, N^l]_d\right). \tag{35}$$

This process is sequentially implemented for all pairs $(\mathrm{node}^{-1}(s^l), \mathrm{node}^{-1}(s^{l+1}))$ in $\mathcal{Q}^\star$ by replacing the initial state x_0^l with the last component of Ψ_{l-1} when $l \neq 1$. In particular, we set $x_0^1 = x_0$ and $\widetilde{y}^{n_{dp}} = \phi$. Note that all initial states in the optimal control problem are feasible due to the constraints (34b) and (34c). After solving for all trajectories $\Psi_1, \ldots, \Psi_{n_{dp}-1}$, we obtain the start-to-end state trajectory by connecting the individual trajectories such that:

$$\psi = \mathrm{Vertcat}\left(\Psi_1, \ldots, \Psi_{n_{dp}-1}\right). \tag{36}$$

The full state trajectory and the corresponding input are able to control the robotic system, fulfilling not only the required differential constraints but also the contact force constraints.

V. SIMULATIONS

In this simulation, we consider a 2-dimensional output space and a 4-DOF planar robot with a contact point at the end-effector as shown in Fig. 1(a). For the simulation, we use the optimization toolbox of MATLAB and the tool DYNOPT [31] on a laptop with an i7-8650U CPU and 16.0 GB RAM.

A. A Planar Robot with Contact

A 4-DOF planar robot is set with parameters $M = [2.0, 1.5, 1.0, 1.0]$ kg and $L = [1.0, 1.5, 2.5, 1.0]$ m where M and L denote the masses and link lengths, respectively. It is assumed that each link's center of mass is located at the geometric center of the link. We consider joint position, velocity, and torque limits such that:

$$-(2/3)\pi \leq q_i \leq (2/3)\pi \quad \forall i \in [1, 4]_d$$

$$-(3/2)\pi \leq \dot{q}_i \leq (3/2)\pi \quad \forall i \in [1, 4]_d$$

$$-1000 \leq u_i \leq 1000 \quad \forall i \in [1, 4]_d$$

where q_i, $\dot{q}_i$, and u_i denote the i-th joint position, velocity, and torque input, respectively. We also consider contact geometry constraints for the end-effector. Specifically, the contact position and the orientation of the end-effector should
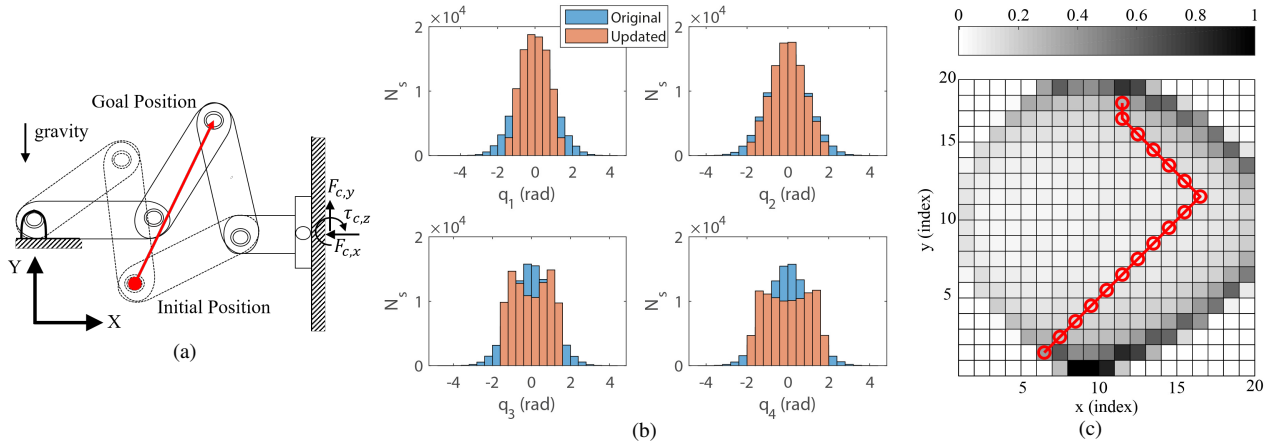
Fig. 1. (a) Conceptual robotic system, (b) histograms of the original samples and the samples updated to fulfill the constraints, including contact force constraints. N_s is the number of samples in desired specific range, (c) is the color-map of the FRS $\mathfrak{F}_m$ and the solution of the first DP.
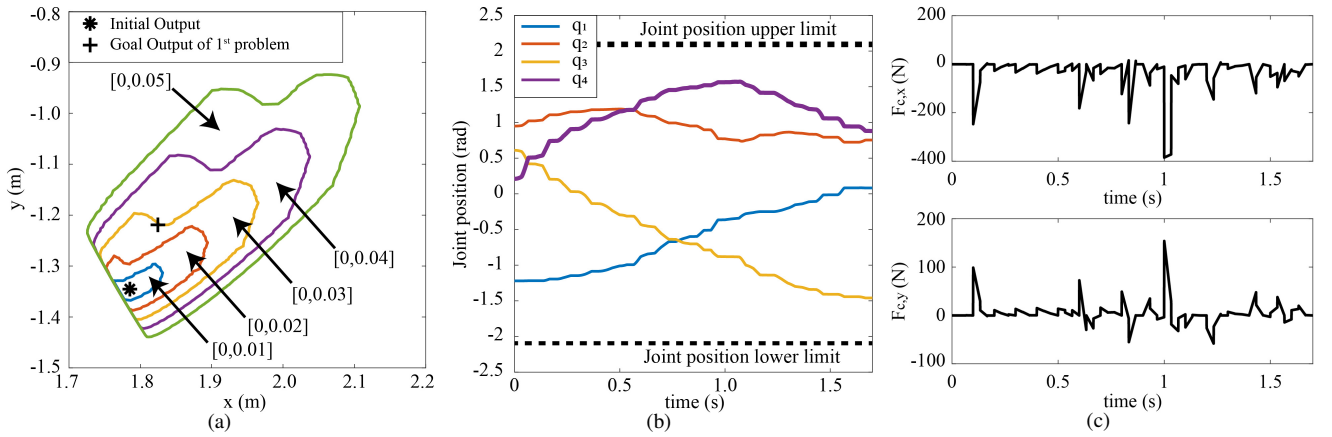

Fig. 2. (a) Non-convex hull of the reachable output set within the time interval: $\text{Nconv}\left(\overline{\mathcal{R}}^y_{[0,T]}(x_0)\right)$, (b) joint position trajectory, (c) contact force

be consistent, and its velocity should be zero. The contact force constraint is defined as follows:

$$-\mu|F_{c,x}| < F_{c,y} < \mu|F_{c,x}|, \quad F_{c,x} < 0$$
$$-L_c|F_{c,x}| \leq \tau_{c,z} \leq L_c|F_{c,x}|$$

where $F_c = [F_{c,x}, F_{c,y}, \tau_{c,z}]^\top$ and L_c denotes the moment arm of the contact link. The focus of the performance for this simulation is on the 2nd link which is required to move while a solid contact should be maintained on the end-effector, as shown in Fig. 1(a). The initial configuration, the joint velocity, and the goal output are $q_0 = [-1.22, 0.949, 0.610, 0.210]$ rad, $\dot{q}_0 = [0.0, 0.0, 0.0]$ rad/s, and $\phi = [2.0, 1.2]$ m, respectively.

We then generate random samples. The threshold ε_G used to determine the appropriate number of samples is set to be 6.0×10^{-5}. The Jacobian matrices of all the constraint functions are full row rank. As a result, the required number of samples is 2.0×10^5 to obtain the reliable FRS. The raw random samples and the modified samples, which fulfill the constraints, are represented by histograms as shown in Fig. 1(b). Specifically, the updated samples satisfy not only joint position limits but also the contact geometric constraints. We next consider a 20×20 grid box to create trajectories using

the DP process described in Section III-E. As a result of the DP process, the trajectory generation problem over long-term interval is broken into 16 short-term problems as shown Fig. 1(c).

The reachable sets are numerically computed to check whether or not the goal outputs resulting from solving the DP problem are reachable via the proposed approach. Fig. 2(a) describes the non-convex hull of the reachable output sets with respect to small time intervals $[0, 0.01]$, $[0, 0.02]$, $[0, 0.03]$, $[0, 0.04]$, and $[0, 0.05]$. The non-convex hull of the reachable output set, $\text{Nconv}\left(\overline{\mathcal{R}}^y_{[0,0.03]}(x_0)\right)$, contains both the initial and goal output positions from the first trajectory generation problem. Therefore, we set the final time interval of the first nonlinear optimization as 0.1 s > 0.03 s based on the reachable set of Fig. 2(a). We repeat the process for the remaining problems. Fig. 2(b) shows the generated joint position trajectory, and Fig. 2(c) shows the corresponding contact force. The final configuration of the robot is $q_f = [0.084, 0.755, -1.460, 0.880]$ rad and the final output is $g(q_f) = [2.0, 1.2]$ m, which is identical to the desired goal output. Furthermore, all considered constraints, related to the joint position, joint velocity, and input limits and the contact geometry/force constraints, are fulfilled while

achieving the desired goal.

VI. CONCLUSION AND FUTURE WORK

This paper proposes an approach to generate feasible trajectories for robotic systems with contact force constraints. The proposed approach consists of a sampling-based method and two optimization processes to generate trajectories that maintain solid contacts in an effective way. Using properties from our sampling approach, the end-to-end trajectory generation problem over a long-term interval is replaced by multiple sub-problems over short-term intervals. This strategy also enables us to perform numerical reachability analysis for the finite time interval before implementing an optimal control process. The simulation results show that the proposed approach successfully generates a feasible trajectory with contact force constraints.

In the near future, we will conduct an extended analysis of our method. Real experiments on a robot will be pursued to verify the scalability of our method. We will also apply our method to more complex systems such as dual-arm and bipedal robots.

ACKNOWLEDGMENTS

REFERENCES

[1] L. Sentis and O. Khatib, "Synthesis of whole-body behaviors through hierarchical control of behavioral primitives," *International Journal of Humanoid Robotics*, vol. 2, no. 04, pp. 505–518, 2005.

[2] M. Mistry, "Operational space control of constrained and underactuated systems," *Robotics: Science and systems VII*, pp. 225–232, 2012.

[3] A. Escande, N. Mansard, and P.-B. Wieber, "Hierarchical quadratic programming: Fast online humanoid-robot motion generation," *The International Journal of Robotics Research*, vol. 33, no. 7, pp. 1006–1028, 2014.

[4] B. J. Stephens and C. G. Atkeson, "Dynamic balance force control for compliant humanoid robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010, pp. 1248–1255.

[5] D. Jia and B. Krogh, "Min-max feedback model predictive control for distributed control with communication," in *Proceedings of American Control Conference*, vol. 6, 2002, pp. 4507–4512.

[6] J. M. Bravo, T. Alamo, and E. F. Camacho, "Robust MPC of constrained discrete-time nonlinear systems based on approximated reachable sets," *Automatica*, vol. 42, no. 10, pp. 1745–1751, 2006.

[7] R. Gonzalez, M. Fiacchini, T. Alamo, J. L. Guzmán, and F. Rodríguez, "Online robust tube-based MPC for time-varying systems: a practical approach," *International Journal of Control*, vol. 84, no. 6, pp. 1157–1170, 2011.

[8] S. V. Rakovic, B. Kouvaritakis, M. Cannon, C. Panos, and R. Findeisen, "Parameterized tube model predictive control," *IEEE Transactions on Automatic Control*, vol. 57, no. 11, pp. 2746–2761, 2012.

[9] S. Subramanian, S. Lucia, and S. Engell, "A novel tube-based output feedback MPC for constrained linear systems," in *Proceedings of American Control Conference*, 2017, pp. 3060–3065.

[10] I. Mitchell, A. M. Bayen, and C. J. Tomlin, "Validating a Hamilton-Jacobi approximation to hybrid system reachable sets," in *International Workshop on Hybrid Systems: Computation and Control*. Springer, 2001, pp. 418–432.

[11] C. J. Tomlin, J. Lygeros, and S. S. Sastry, "A game theoretic approach to controller design for hybrid systems," *Proceedings of the IEEE*, vol. 88, no. 7, pp. 949–970, 2000.

[12] L. Habets, P. J. Collins, and J. H. van Schuppen, "Reachability and control synthesis for piecewise-affine hybrid systems on simplices," *IEEE Transactions on Automatic Control*, vol. 51, no. 6, pp. 938–948, 2006.

[13] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin, "A time-dependent Hamilton-Jacobi formulation of reachable sets for continuous dynamic games," *IEEE Transactions on automatic control*, vol. 50, no. 7, pp. 947–957, 2005.

[14] M. Maiga, N. Ramdani, L. Travé-Massuyès, and C. Combastel, "A comprehensive method for reachability analysis of uncertain nonlinear hybrid systems," *IEEE Transactions on Automatic Control*, vol. 61, no. 9, pp. 2341–2356, 2016.

[15] S. Summers, M. Kamgarpour, C. Tomlin, and J. Lygeros, "Stochastic system controller synthesis for reachability specifications encoded by random sets," *Automatica*, vol. 49, no. 9, pp. 2906–2910, 2013.

[16] K. Lesser and M. Oishi, "Reachability for partially observable discrete time stochastic hybrid systems," *Automatica*, vol. 50, no. 8, pp. 1989–1998, 2014.

[17] A. B. Kurzhanski and P. Varaiya, "Dynamic optimization for reachability problems," *Journal of Optimization Theory and Applications*, vol. 108, no. 2, pp. 227–251, 2001.

[18] E. Asarin, O. Bournez, T. Dang, and O. Maler, "Approximate reachability analysis of piecewise-linear dynamical systems," in *International Workshop on Hybrid Systems: Computation and Control*. Springer, 2000, pp. 20–31.

[19] N. Kariotoglou, S. Summers, T. Summers, M. Kamgarpour, and J. Lygeros, "Approximate dynamic programming for stochastic reachability," in *Proceedings of European Control Conference*, 2013, pp. 584–589.

[20] J. Maidens and M. Arcak, "Reachability analysis of nonlinear systems using matrix measures," *IEEE Transactions on Automatic Control*, vol. 60, no. 1, pp. 265–270, 2015.

[21] M. Arcak and J. Maidens, "Simulation-based reachability analysis for nonlinear systems using componentwise contraction properties," *arXiv preprint arXiv:1709.06661*, 2017.

[22] Y. Yang, W. Merkt, H. Ferrolho, V. Ivan, and S. Vijayakumar, "Efficient humanoid motion planning on uneven terrain using paired forward-inverse dynamic reachability maps," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 2279–2286, 2017.

[23] Y. Guan and K. Yokoi, "Reachable space generation of a humanoid robot using the monte carlo method," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 1984–1989.

[24] A. Shkolnik, M. Walter, and R. Tedrake, "Reachability-guided sampling for planning under differential constraints," in *IEEE International Conference on Robotics and Automation*, 2009, pp. 2859–2865.

[25] V. Boor, M. H. Overmars, and A. F. Van Der Stappen, "The Gaussian sampling strategy for probabilistic roadmap planners," in *IEEE International Conference on Robotics and Automation*, vol. 2, 1999, pp. 1018–1023.

[26] S. Patil, J. Van Den Berg, and R. Alterovitz, "Estimating probability of collision for safe motion planning under gaussian motion and sensing uncertainty," in *IEEE International Conference on Robotics and Automation*, 2012, pp. 3238–3244.

[27] J. Carpentier and N. Mansard, "Multi-contact locomotion of legged robots," *IEEE Transactions on Robotics*, 2018.

[28] K. Hauser, T. Bretl, J.-C. Latombe, K. Harada, and B. Wilcox, "Motion planning for legged robots on varied terrain," *The International Journal of Robotics Research*, vol. 27, no. 11-12, pp. 1325–1349, 2008.

[29] M. Stilman, "Task constrained motion planning in robot joint space," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007, pp. 3074–3081.

[30] S. Caron, Q.-C. Pham, and Y. Nakamura, "Stability of surface contacts for humanoid robots: Closed-form formulae of the contact wrench cone for rectangular support areas," in *IEEE International Conference on Robotics and Automation*, 2015, pp. 5107–5112.

[31] M. Cizniar, M. Fikar, M. Latifi *et al.*, "A matlab package for dynamic optimisation of processes," in *Proceedings of International Scientific-Technical Conference-PROCESS CONTROL*. Kouty nad Desnou, Czech Republic, 2006.