Efficient Reachability Analysis for Safe Motion Planning in High-Dimensional Robotic Systems

Manveer Singh[1], Jonathan Michaux[1], and Ram Vasudevan[1]

***Abstract*— High-dimensional robotic systems are being increasingly deployed in dynamic, human-centric environments, necessitating robust safety guarantees in their motion planning. These systems must generate collision-free trajectories that avoid harm to humans, damage to nearby objects, and self-collisions, while maintaining real-time performance to adapt to sudden environmental changes. However, safe motion planning for such systems presents a fundamental challenge: computing reachable sets quickly enough for real-time operation while maintaining safety guarantees. Traditional approaches to reachability analysis, while theoretically sound, often become computationally intractable as system dimensionality increases, forcing practitioners to compromise between safety and real-time performance. This paper extends the work on Autonomous Robust Manipulation via Optimization with Uncertainty-aware Reachability (ARMOUR) and Conformalized Reachable Sets for Obstacle Avoidance With Spheres (CROWS), for two challenging scenarios: safe bipedal locomotion in confined spaces and bimanual manipulation for pick-and-place tasks in constrained environments. We demonstrate that the method significantly reduces computational overhead while maintaining rigorous safety guarantees, enabling real-time performance in scenarios where traditional methods become prohibitively expensive.**

Fig. 1: This paper presents a comprehensive framework integrating neural networks for safe motion planning in high-dimensional robotic systems. The approach employs multiple specialized networks: one for locomotion safety guarantees, a second for predicting spherical approximations in bimanual tasks, a third for computing signed distance functions between arms, and a fourth for joint dynamics prediction. During runtime, these learned representations are combined with conformal prediction bounds to generate probabilistically-safe trajectories and control inputs. The framework operates in a receding horizon manner, simultaneously optimizing for locomotion and bimanual manipulation while maintaining safety constraints through learned inverse dynamics and inter-arm collision avoidance

I. Introduction

Numerous aspects of human life and work could be altered by high-dimensional robotic systems, such as multi-arm manipulators and humanoid robots. This includes using bipedal robots for complex locomotion tasks in human environments, deploying bimanual manipulation systems for manufacturing and assembly, and utilizing high-degree-of-freedom platforms for challenging scenarios such as disaster response or space exploration. In each of these settings, it is essential that the robot maintain safety at all times to prevent colliding with obstacles, damaging high-value objects, or harming nearby humans. It is also critical that the robot generates motion plans in real-time to ensure efficient task completion and rapid adaptation to environmental changes, particularly in dynamic human-centric spaces.

A modern approach to motion planning typically involves combining a high-level planner, a mid-level trajectory planner, and a low-level tracking controller into a hierarchical framework [1], [2]. The high-level planner generates a path between the robot's start and goal configurations consisting of a sequence of discrete waypoints. The mid-level trajectory planner computes time-dependent positions, velocities, and accelerations that move the robot from one waypoint to the next. The low-level tracking controller generates control inputs that attempt to minimize deviations between the robot's actual motion and the desired trajectory. For example, one may use a sampling-based planner such as an RRT* to generate a set of discrete waypoints for a trajectory optimization algorithm such as TrajOpt [3], and track the resulting trajectories with an inverse dynamics controller [4].

While variations of this planning framework have been demonstrated to work on various robotic platforms [1], [2], [5], several limitations prevent this approach from being widely deployed in real-world scenarios, particularly for high-dimensional systems. The computational complexity of these methods typically scales exponentially with the system's dimensionality, making them impractical for applications requiring both real-time performance and safety guarantees. For instance, computing reachable sets—crucial for ensuring safety—becomes prohibitively expensive as the number of degrees of freedom increases. Many algorithms introduce heuristics to improve computation speed, such as reducing the number of collision checks or simplifying the reachability analysis, to achieve real-time performance; however, this often comes at the expense of robot safety. Fur-

[1]Robotics Institute, University of Michigan, Ann Arbor, MI ⟨`manveer, jmichaux, ramv`⟩`@umich.edu`.

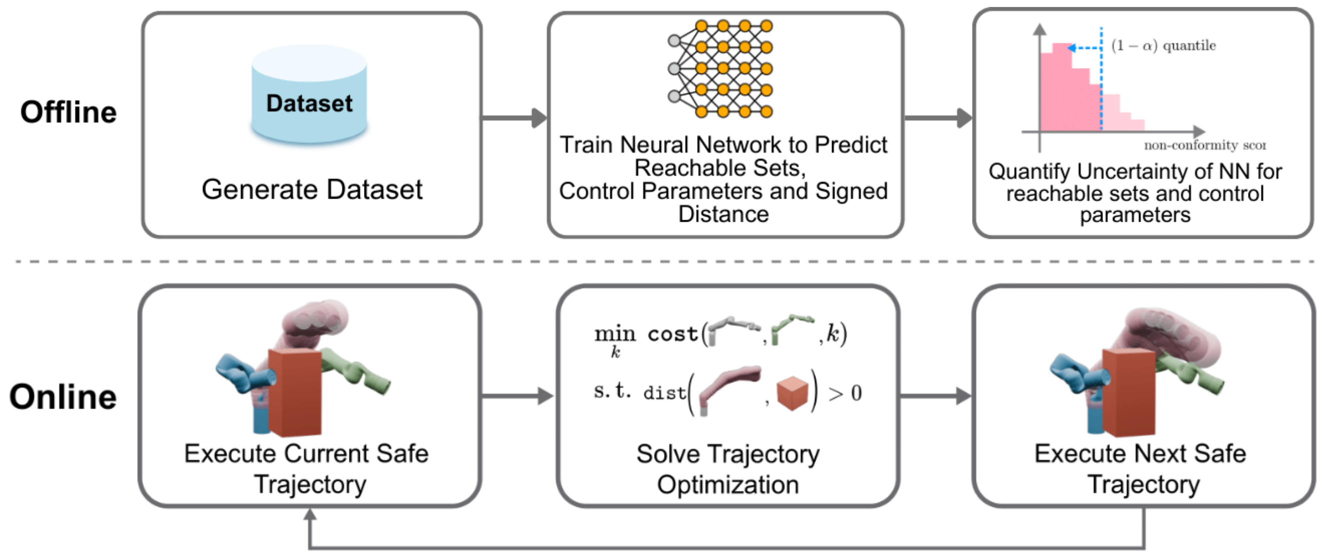
Offline
Dataset
Generate Dataset
Train Neural Network to Predict Reachable Sets, Control Parameters and Signed Distance
$(1-\alpha)$ quantile
non-conformity score
Quantify Uncertainty of NN for reachable sets and control parameters
Online
Execute Current Safe Trajectory
min cost(, , k)
s.t. dist(,) > 0
Solve Trajectory Optimization
Execute Next Safe Trajectory

Fig. 2: Overview of our multi-network framework for safe locomotion and bi-manual manipulation. Offline, the system trains multiple neural networks: one for locomotion safety representations, another for spherical approximations of dual-arm configurations, a third for signed distance function computation between arms, and a fourth for joint dynamics prediction. During runtime, these learned representations are combined with conformal prediction bounds to generate probabilistically-safe trajectories and control inputs. The framework simultaneously optimizes for both locomotion and bimanual manipulation while maintaining safety constraints through receding horizon planning.

thermore, many algorithms assume that the robot's dynamics are fully known, while in reality there can be considerable model uncertainty, particularly in high-dimensional systems where identifying accurate dynamic parameters becomes increasingly challenging. Unfortunately, both of these factors may increase the potential for the robot to collide with obstacles or exhibit unsafe behavior.

To address these challenges, this paper presents a unified framework that extends ARMOUR and CROWS methodologies to enable safe motion planning for high-dimensional robotic systems. Our approach introduces a neural network-based safety representation that efficiently integrates into a mid-level trajectory optimization algorithm for both bipedal locomotion and bimanual manipulation tasks (Fig. 2). The framework learns an overapproximation of the swept volume (*i.e.* reachable set) of high-dimensional robotic systems using spherical representations, which proves particularly effective for complex humanoid configurations. Prior to planning, our system trains neural networks to approximate sphere-based reachable sets for both locomotion and manipulation tasks, while simultaneously learning inverse dynamics for precise control input prediction (Sec. III-A). We then apply conformal prediction to compute confidence bounds that provide probabilistic safety guarantees for each motion component (Sec. III-B). During runtime, our framework leverages these conformalized reachable sets, their learned gradients, and predicted joint dynamics based on Newton Recursive Euler Algorithm (RNEA) to solve optimization problems that generate probabilistically-safe trajectories online (Sec. III-C). The effectiveness of this approach is demonstrated in two challenging scenarios: safe bipedal locomotion in confined spaces and coordinated bi-manual manipulation for pick-and-place tasks in constrained environments (Sec. IV-D). Our results demonstrate that this novel formulation enables real-time planning in extremely cluttered environments while maintaining rigorous safety guarantees, even for highly complex tasks like coordinated dual-arm manipulation (Fig. 3) where traditional methods become computationally intractable.

A. Related Work

Safe motion planning algorithms ensure that robots can move from one configuration to another while avoiding collisions for all time, whether in manipulation tasks or bipedal locomotion scenarios. Ideally, one would compute the robot's swept volume [6]–[11] and ensure that it does not intersect with any obstacles nor enter any unwanted regions of its workspace. This approach has been applied to both manipulator arms and humanoid robots [12]. Alternatively, many algorithms approximate the swept volume using occupancy grids, convex polyhedra, or CAD models [13]–[15]. However, both approaches often become intractable or suffer from high computational costs as system dimensionality increases, particularly for high-DOF systems like humanoid robots or dual-arm manipulators. Approximate swept volume algorithms may also be overly conservative [15], [16], limiting their utility to offline motion planning [12]. An alternative approach to robot collision avoidance involves modeling the robot or the environment with simple collision primitives such as spheres [17], [18], ellipsoids [19], capsules [20], [21], and then performing collision-checking along a given trajectory at discrete time instances. This is common for state-of-the-art trajectory optimization-based approaches in both manipulation [3], [22]–[24] and locomotion. While these methods have demonstrated impressive results, the resulting trajectories cannot be considered safe as collision avoidance is not enforced for all time, which is particularly concerning in human-centric environments. Reachability-based Trajectory Design (RTD) [5], a recent approach to

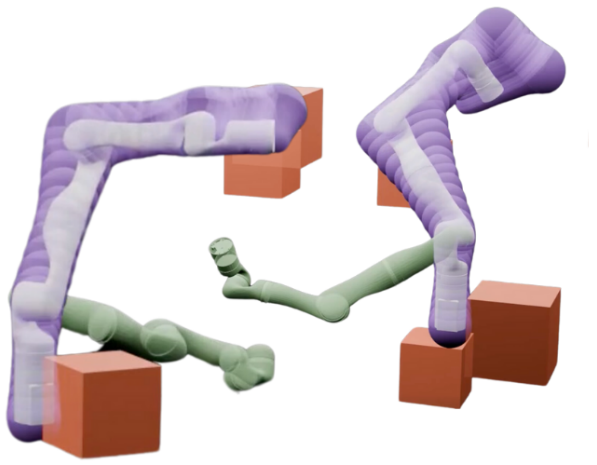
Fig. 3: A visualization of the bi-manual arm (white), the obstacles (red), Spherical Forward Occupancy (purple) and goal position(green) in an environment. The volume of each joint is overapproximated by a collection of spheres (Thm. 3).

real-time motion planning, uses (polynomial) zonotopes [25] to construct reachable sets that overapproximate all possible robot positions corresponding to a pre-specified family of parameterized trajectories. RTD's reachable sets are constructed to ensure that its obstacle-avoidance constraints are satisfied in continuous-time, making it particularly suitable for safety-critical applications. Recent work has extended these concepts to both manipulation and locomotion domains. For manipulation, ARMOUR [2] and WAITR [26] have demonstrated safe planning for robotic arms, while locomotion approaches like [1] have applied similar principles to bipedal systems. CROWS [27], a recent advancement, utilizes sphere-based reachable sets to generate certifiably-safe motion plans that are considerably less conservative than previous approaches across both domains. Several methods have been developed to quantify the uncertainty of neural network models such as Monte Carlo dropout [28]–[30], Laplace approximation [31]–[33], and deep ensembles [34], [35]. More recently, conformal prediction [36], [37] has gained popularity due to its ability to provide probabilistic guarantees on coverage. Conformal prediction has been employed in various robotics applications including: ensuring the safety of learning-based object detection systems [38]; quantifying the uncertainty of human inputs during teleoperation [39]; ensuring safety in environments with dynamic obstacles [40], [41]; and applying conformal prediction to align uncertainty in large language models. Notably, [42] is closely related to our work, as it enhances the efficiency of reachable set computations for high-dimensional systems with neural networks while verifying them with conformal prediction. Our work extends these concepts to both manipulation and locomotion domains, demonstrating their efficacy in reducing computational overhead while maintaining safety guarantees.

II. Background

This section summarizes the background necessary for the development of neural networks in Sec. III.

A. Mathematical Preliminaries

Sets, subspaces, and matrices are typeset using capital letters. Let $\mathbb{R}$ and $\mathbb{N}$ denote the spaces of real numbers and natural numbers, respectively. Subscripts are primarily used as an index or as a label for relevant countable sets. For example, if $n_\alpha \in \mathbb{N}$, then we denote the set $N_\alpha = \{1, \cdots, n_\alpha\}$. Given a set $\mathcal{A}$, denote its power set as $P(\mathcal{A})$. Given a set $\Omega \subset \mathbb{R}^{n_d}$, $co(\Omega)$ denote its convex hull.

B. Robot Occupancy

This subsection describes how to overapproximate the forward occupancy, or swept volume, of a moving robot arm. Consider a compact time interval $T \subset \mathbb{R}$. We define a trajectory for the robot's configuration as $q : T \to Q \subset \mathbb{R}^{n_q}$ and a trajectory for the velocity as $\dot{q} : T \to \mathbb{R}^{n_q}$. To facilitate the later discussion of the robot's occupancy, we begin by restating an assumption [43, Ass. 4] about the robot model:

Assumption 1. *The robot operates in an three-dimensional workspace, denoted $W_s \subset \mathbb{R}^3$. There exists a reference frame called the base frame, denoted the 0^{th} frame, that indicates the origin of the robot's kinematic chain within the workspace. The robot is fully actuated and composed of only revolute joints, where the j^{th} joint actuates the robot's j^{th} link. The robot's j^{th} joint has position and velocity limits given by $q_j(t) \in [q^-_{j,\text{lim}}, q^+_{j,\text{lim}}]$ and $\dot{q}_j(t) \in [\dot{q}^-_{j,\text{lim}}, \dot{q}^+_{j,\text{lim}}]$ for all $t \in T$, respectively.*

Let $L_j \subset W_s \subset \mathbb{R}^3$ denote the volume occupied by the robot's j^{th} link with respect to the j^{th} reference frame. Then the *forward occupancy* of the j^{th} link is the map $\text{FO}_j : Q \to \mathcal{P}(W_s)$ defined as

$$\text{FO}_j(q(t)) = p_j(q(t)) \oplus R_j(q(t))L_j, \tag{1}$$

where $p_j(q(t))$ and $R_j(q(t))$ are computed by the forward kinematics [44] and specify the pose of the j^{th} joint, and $R_j(q(t))L_j$ is the rotated volume of link j. The volume occupied by the entire arm in the workspace is the union of the link volumes defined by the map $\text{FO} : Q \to W_s$:

$$\text{FO}(q(t)) = \bigcup_{j=1}^{n_q} \text{FO}_j(q(t)) \subset W_s. \tag{2}$$

Due to the potentially complex geometry of a robot's links, we reiterate an assumption [43, Ass. 5] that facilitates the construction of an overapproximation of the forward occupancy:

Assumption 2. *Given a robot configuration $q(t)$ and any $j \in \{1, \dots, n_q\}$, there exists a ball with center $p_j(q(t))$ and radius r_j that overapproximates the volume occupied by the j^{th} joint in W_s. We further assume that link volume L_j is a subset of the tapered capsule formed by the convex hull of the balls overapproximating the j^{th} and $j + 1^{th}$ joints.*

Following Assum. 2, we now define the ball $S_j(q(t))$ overapproximating the volume occupied by the j^{th} joint as

$$S_j(q(t)) = (p_j(q(t)), r_j) \tag{3}$$

and the tapered capsule $TC_j(q(t))$ overapproximating the j^{th} link as

$$TC_j(q(t)) = co\Big(S_j(q(t)) \cup S_{j+1}(q(t))\Big). \tag{4}$$

Then, the volumes occupied by the j^{th} link (1) and the entire arm (2) can be overapproximated by

$$\text{FO}_j(q(t)) \subset TC_j(q(t)) \tag{5}$$

and

$$\text{FO}(q(t)) \subset \bigcup_{j=1}^{n_q} TC_j(q(t)) \subset W_s, \tag{6}$$

respectively.

Note that if the forward occupancy (or *reachable set*) $\text{FO}(q(T))$ does not intersect with the environment, then the arm is *collision-free* over the time interval T. To facilitate the exposition of our approach, we summarize the construction of the robot's safety representation by restating [43][Thm. 10]:

Theorem 3. *Given a serial link robot with $n_q \in \mathbb{N}$ revolute joints, a time partition T of a finite set of intervals, T_i (i.e., $T = \cup_{i=1}^{n_t} T_i$), the swept volume corresponding to the robot's motion over T is overapproximated by a collection of L_2 balls in $\mathbb{R}^3$, which we call the Spherical Forward Occupancy (SFO [43] defined as*

$$\mathcal{SFO} = \cup_{j=1}^{n_q} \cup_{i=1}^{n_t} \cup_{m=1}^{n_S} S_{j,i,m}(q(T_i;k)), \tag{7}$$

where each $S_{j,i,m}(q(T_i;k))$ is an L_2 ball in $\mathbb{R}^3$, $n_S \in \mathbb{N}$ is a parameter that specifies the number of closed balls overapproximating each of the robot's links, and k is a trajectory parameter that characterizes the motion of the robot over T.

Note that one can explicitly construct an $\mathcal{SFO}$ that satisfies this assumption using the approach described in [43].

C. Serial Link Dynamics

The robot is composed of n_q rigid links with inertial parameters:

$$[\Delta] = (m_1, \ldots, m_{n_q}, c_{x,1}, \ldots, c_{z,n_q}, I_{xx,1}, \ldots, I_{zz,n_q})^\intercal \tag{8}$$

where m_j, $c_j = (c_{x,j}, c_{y,j}, c_{z,j})$, and $(I_{xx,j}, \ldots, I_{zz,j})$ represent the mass, center of mass, and inertia tensor of the j^{th} link, respectively. The dynamics are represented by the standard manipulator equations for bimanual manipulation[44]:

$$M(q(t),\Delta)\ddot{q}(t) + C(q(t),\dot{q}(t),\Delta)\dot{q}(t) + G(q(t),\Delta) = u(t) \tag{9}$$

where $M(q(t),\Delta) \in \mathbb{R}^{n_q \times n_q}$ is the positive definite inertia matrix, $C(q(t),\dot{q}(t),\Delta)$ is the Coriolis matrix, $G(q(t),\Delta)$ is the gravity vector, and $u(t)$ is the input torque all at time t. For simplicity, we do not explicitly model friction, but it can be incorporated in $G(q(t),\Delta)$ We assume the following about the inertial parameters:

Assumption 4 (Inertial Parameter Bounds)**.** *The model structure (i.e., number of joints, sizes of links, etc.) of the robot is known, but its true inertial parameters are unknown. The uncertainty in each inertial parameter is given by*

$$\Delta = \big([m_1], \ldots, [m_{n_q}], [c_{x,1}], \ldots, [c_{z,n_q}], [I_{xx,1}], \ldots [I_{zz,n_q}]\big) \tag{10}$$

where $[m_j] \subset \mathbb{IR}$, $[c_j] \subset \mathbb{IR}^3$, and $[I_j] \subset \mathbb{IR}^{3\times3}$ represent the uncertainties in the mass, center of mass, and inertia tensor of the j^{th} link, respectively. The true parameters lie in this interval. Any inertia tensor drawn from $[I_j]$ must be positive semidefinite. If we let $[\Delta]_j$ denote the j^{th} component of $[\Delta]$, then $\texttt{inf}([\Delta]_j) > -\infty$ and $\texttt{sup}([\Delta]_j) < \infty$ for all j. There exists a nominal vector of inertial parameters $\in$ that serves as an estimate of the true parameters of the system.

We do not assume that Δ_0 is equal to the true parameters.

D. Polynomial Zonotope RNEA

High-dimensional systems must also avoid saturating their available motor torques. We describe how to overapproximate the set of torques that may be required for the robust passivity-based control input to track any parameterized trajectory. We define a total feedback trajectory polynomial zonotope $(\mathbf{T_i}; \mathbf{K})$ by computing a polynomial zonotope representation of each of the components of . We then define modified desired velocity and acceleration trajectory polynomial zonotopes:

$$\begin{aligned}\dot{\mathbf{q}}_\mathbf{a}(\mathbf{T_i};\mathbf{K}) &= \mathbf{q_d}(\mathbf{T_i};\mathbf{K}) \oplus K_r,\\ \ddot{\mathbf{q}}_\mathbf{a}(\mathbf{T_i};\mathbf{K}) &= \dot{\mathbf{q}}_\mathbf{d}(\mathbf{T_i};\mathbf{K}) \oplus K_r.\end{aligned} \tag{11}$$

Using these definitions and plugging into nominal model of the robot's dynamics given by inertial parameters Δ, we obtain

$$\begin{aligned}\boldsymbol{\tau}(\mathbf{q_A}(\mathbf{T_i};\mathbf{K}),\Delta_0) = {} & (\mathbf{q}(\mathbf{T_i};\mathbf{K}),\Delta_0)\ddot{\mathbf{q}}_\mathbf{a}(\mathbf{T_i};\mathbf{K}) + \\ & + (\mathbf{q}(\mathbf{T_i};\mathbf{K}),\dot{\mathbf{q}}(\mathbf{T_i};\mathbf{K}),\Delta_0)\dot{\mathbf{q}}_\mathbf{a}(\mathbf{T_i};\mathbf{K}) + \\ & + (\mathbf{q}(\mathbf{T_i};\mathbf{K}),\Delta_0).\end{aligned} \tag{12}$$

To compute $\boldsymbol{\tau}(\mathbf{q_A}(\mathbf{T_i};\mathbf{K}),\Delta_0)$, we introduce the Polynomial Zonotope RNEA (PZRNEA). Note just as in the original RNEA and IRNEA Algorithms, we have suppressed the dependence on $\mathbf{q_j}(\mathbf{T_i};\mathbf{K})$ in $\mathbf{R^j_{j-1}}$. PZRNEA takes as inputs , and the nominal inertial parameters Δ_0 to find the nominal input:

$$\boldsymbol{\tau}(\mathbf{q_A}(\mathbf{T_i};\mathbf{K}),\Delta_0) = \texttt{PZRNEA}(\mathbf{q_A}(\mathbf{T_i};\mathbf{K}),\Delta_0,a_0^0), \tag{13}$$

where $a_0^0 = [0, 0, 9.81]^\top$.

Next, we describe a bound on the robust control input at each time step.

Theorem 5 (Robust Input Bound)**.** *Suppose $\alpha : \mathbb{R} \to \mathbb{R}$ in* **(??)** *is a linear function, i.e., $\alpha(x) = \alpha_c x$, where $\alpha_c > 0$ is a user-specified constant. Suppose the disturbance is over approximated using*

$$\begin{aligned}\mathbf{w}((\mathbf{T_i};\mathbf{K}),\Delta_0,[\Delta]) = {} & \texttt{PZRNEA}((\mathbf{T_i};\mathbf{K}),[\Delta],a_0^0) + \\ & -\boldsymbol{\tau}((\mathbf{T_i};\mathbf{K}),\Delta_0)\end{aligned} \tag{14}$$

with PZRNEA *and* $a_0^0 = [0, 0, 9.81]^\top$*, let*

$$\rho(\mathbf{w}((\mathbf{T_i}; \mathbf{K}), \Delta_0, [\Delta])) = \max\Big(|\texttt{inf}(\mathbf{w}((\mathbf{T_i}; \mathbf{K}), \Delta_0, [\Delta]))|, \\ |\texttt{sup}(\mathbf{w}((\mathbf{T_i}; \mathbf{K}), \Delta_0, [\Delta]))| \Big), \tag{15}$$

and let

$$w_M(*) = \rho(\mathbf{w}((\mathbf{T_i}; \mathbf{K}), \Delta_0, [\Delta])) \tag{16}$$

where $w_M(*) := w_M((\mathbf{T_i}; \mathbf{K}), \Delta_0, [\Delta])$*. Then for all* $(t; k)$ *that satisfies the uniform bound, the following bound is satisfied for each* $j \in N_q$

$$|v(q_A(t; k), \Delta_0, [\Delta])_j| \leq \frac{\alpha_c \epsilon(\sigma_M - \sigma_m)}{2} + \\ + \frac{\|w_M(*)\| + w_M(*)_j}{2}, \tag{17}$$

where $|v(q_A(t; k), \Delta_0, [\Delta])_j|$ *is the* j^{th} *component of the robust input.*

This theorem requires that α is linear with positive slope rather than an arbitrary extended class $\mathcal{K}_\infty$ function. The proof could be extended to any $\mathcal{K}_\infty$ function, but would generate a bound that is more difficult to compute. As from previous work a linear α works well. Using (17), we bound the robust input by a constant in each dimension. That is, we let

$$\mathbf{v}_j(\mathbf{q_A}(\mathbf{T_i}; \mathbf{K}), \Delta_0, [\Delta]) = \Big(\frac{\alpha_c \epsilon(\sigma_M - \sigma_m)}{2} + \\ + \frac{\|(w_M(*)\| + w_M(*)_j}{2} \Big) x_{v_j} \tag{18}$$

with $w_M(*)$ as in Thm. 5 and x_{v_j} an indeterminate in $[-1, 1]$. Using this we let

$$\mathbf{v}(\mathbf{q_A}(\mathbf{T_i}; \mathbf{K}), \Delta_0, [\Delta]) = \bigtimes_{j=1}^{n_q} \mathbf{v}_j(\mathbf{q_A}(\mathbf{T_i}; \mathbf{K}), \Delta_0, [\Delta]). \tag{19}$$

We define the *input reachable set* for the ith timestep as

$$\mathbf{u}(\mathbf{q_A}(\mathbf{T_i}; \mathbf{K}), \Delta_0, [\Delta]) = \boldsymbol{\tau}(\mathbf{q_A}(\mathbf{T_i}; \mathbf{K}), \Delta_0) + \\ - \mathbf{v}(\mathbf{q_A}(\mathbf{T_i}; \mathbf{K}), \Delta_0, [\Delta]). \tag{20}$$

Because the robust input bound is treated as constant in each dimension in (18), when we slice the input reachable set by k, we only slice the polynomial zonotope representation of the nominal input by k, *i.e.*,

$$\mathbf{u}(\mathbf{q_A}(\mathbf{T_i}; \mathbf{K}), \Delta_0, [\Delta]) = \boldsymbol{\tau}(, \Delta_0) + \\ - \mathbf{v}(, \Delta_0, [\Delta]). \tag{21}$$

Using these definitions and the fact that all relevant operations involving polynomial zonotopes are either exact or over approximate, one can prove that the input reachable set contains the robust passivity-based control input:

Lemma 6 (Input PZ Conservativeness)**.** *The input reachable set is over approximative, i.e., for each* $k \in \mathbf{K}$

$$\mathbf{u}(q_A(t; k), \Delta_0, [\Delta]) \in \mathbf{u_j}(\mathbf{T_i}; k), \quad \forall t \in \mathbf{T_i}. \tag{22}$$

E. Polynomial Zonotope Signed Distance Function

This subsection derives an approximation to the reachability-based signed distance function. The core idea is to approximate the distance between an obstacle and the polynomial zonotope forward occupancy $\mathbf{FO}(\mathbf{q}(\mathbf{T_i}; \mathbf{K}))$ over of time for an entire trajectory. Note that slicing a polynomial zonotope of all of its *dependent* coefficients results in a zonotope. This allows CROWS to approximate both positive and negative (*i.e.* signed) distances by leveraging the zonotope arithmetic.

Theorem 7. *Suppose a robot is following a parameterized trajectory* $q(t; k)$ *for all* $t \in T$*. Consider an obstacle* O *with center* c_o *and generators* G_o *and* $\mathbf{FO_j}(\mathbf{q}(\mathbf{T_i}; k))$ *with center* c_F *and generators* G_F*. Let* $\mathcal{P}_j := \bigcup_{i=1}^{n_t} \mathbf{FO_j^{buf}}(\mathbf{q}(\mathbf{T_i}; k))$ *where* $\mathbf{FO_j^{buf}}(\mathbf{q}(\mathbf{T_i}; k)) = (c_F, G_o \cup)$*. Define the function* $_j$ *as follows:*

$$_j(c_o, \mathcal{P}_j) = \begin{cases} \mathbf{d}(c_o, \partial\mathcal{P}_j) & \text{if } c_o \notin \mathcal{P} \\ -\mathbf{d}(c_o, \partial\mathcal{P}_j) & \text{otherwise}, \end{cases} \tag{23}$$

and define the function as follows:

$$(c_o, \cup_{j \in N_q} \mathcal{P}_j) = \min_{j \in N_q} {}_j(c_o, \mathcal{P}_j). \tag{24}$$

If $\mathbf{FO_j}(\mathbf{q}(\mathbf{T_i}; k)) \cap O \neq \emptyset$*, then* $(c_o, \mathcal{P}_j) \geq$ *RDF*$(O, \mathrm{FO}(q(T; k)))$*. If* $\mathbf{FO_j}(\mathbf{q}(\mathbf{T_i}; k)) \cap O = \emptyset$*, then* $(c_o, \mathcal{P}_j) \leq$ *RDF*$(O, \mathrm{FO}(q(T; k)))$*.*

This theorem is useful because it allows us to conservatively approximate RDF using which computes the distance between a point and a convex set. As we show next, this distance computation can be done by solving a convex program. Recall that a convex hull can be represented as the intersection of a finite number of half planes, *i.e.,*

$$\mathcal{P}_j = \bigcap_{h \in N_{h,j}} \mathcal{H}_j^{(h)} \tag{25}$$

where $N_{h,j} = \{1, \cdots, n_{h,j}\}$ and $n_{h,j}$ is the number of half-spaces $\mathcal{H}_j^{(h)}$. As a result, one can determine whether a point $p \in \mathbb{R}^{n_d}$, is in side of $\mathcal{P}_j$ using the following property:

$$p \in \mathcal{P}_j \iff A_j^{(h)} p - b_j^{(h)} \leq 0 \quad \forall i = N_{h,j}, \tag{26}$$

where $A_j^{(h)} \in \mathbb{R}^{n_d}$, $b_j^{(h)} \in \mathbb{R}$ represents each half-space, $\mathcal{H}_j^{(h)}$. As a result, one can compute the distance in (24) as:

$$\mathbf{d}(c_o, \partial\mathcal{P}_j) = \min_p \quad ||p - c_o|| \tag{27}$$

$$A_j^{(h)} p - b_j^{(h)} \leq 0 \qquad \forall h \in N_{h,j}, \tag{28}$$

where depending upon the norm chosen in the cost function one can solve the optimization problem using a linear or a convex quadratic program. In the instance that there is non-trivial intersection between an obstacle and $\mathcal{P}_j$, one can apply the Euclidean projection [45, p.398] to directly calculate the distance between c_o and every half-space $\mathcal{H}_j^{(h)}$ supporting $\mathcal{P}_j$:

$$\mathbf{d}(c_o, \partial\mathcal{P}_j) = \min_{h\in N_{h,j}} \mathbf{d}(c_o, \mathcal{H}_j^{(h)}) \tag{29}$$

$$= -\max_{h\in N_{h,j}} \frac{A_j^{(h)}\, c_o - b_j^{(h)}}{||A_j^{(h)}||}. \tag{30}$$

F. Environment Modeling

The $\mathcal{SFO}$ constructed in Thm. 3 is composed entirely of spheres, where each sphere is a collision primitive consisting of a point with a safety margin corresponding to its radius. For simplicity, we assume that each obstacle $\mathcal{O}$ and centre of each joint $\mathcal{C}$ is a polytope. Therefore, to make use of this sphere-based representation for trajectory planning, we state the result of [43] [Lem. 11] as an assumption describing the existence of an obstacle and self collision avoidance exact signed distance:

Assumption 8 (Environment Signed Distance Function)**.** *Given the $\mathcal{SFO}$, a convex obstacle polytope $\mathcal{O} \in \mathbb{R}^3$ and arm joint center convex polytope $\mathcal{C} \in \mathbb{R}^3$,as discussed in the previous section $\mathbf{s_d}(\mathcal{SFO}, \mathcal{O})$, computes the exact signed distance between $\mathcal{SFO}$ and $\mathcal{O}$. And similarly $\mathbf{s_d}(\mathcal{SFO}, \mathcal{C})$, computes the exact signed distance between $\mathcal{SFO}$ and $\mathcal{C}$ important for preventing self collision during bimanual manipulation.*

III. Trajectory Optimization Formulation

The method described in [43] computes provably-safe motion plans by solving a nonlinear optimization program in a receding-horizon manner for safe locomotion of high dimensional systems:

$$\min_{k\in K} \quad \texttt{cost}(q_{\text{goal}}, k) \qquad \texttt{(Opt)} \tag{31}$$

$$q_j(T_i;k) \subseteq [q^-_{j,\text{lim}}, q^+_{j,\text{lim}}] \qquad \forall(i,j)\in N_t \times N_q \tag{32}$$

$$\dot q_j(T_i;k) \subseteq [\dot q^-_{j,\text{lim}}, \dot q^+_{j,\text{lim}}] \qquad \forall(i,j)\in N_t \times N_q \tag{33}$$

$$\mathbf{s_d}(\mathcal{SFO}, \mathcal{O}_n) > 0 \qquad \forall n \in N_{\mathcal{O}}, \tag{34}$$

where $\mathcal{O}_n$ is the n^{th} obstacle for $n \in N_{\mathcal{O}}$. Offline, we pre-specify a continuum of trajectories over a compact set $K \subset \mathbb{R}^{n_k}$ such that $n_k \in \mathbb{N}$. Then each trajectory, $q(t;k)$, is defined over a compact time interval T and is uniquely determined by a *trajectory parameter* $k \in K$. The cost function (31) ensures the robot moves towards a user- or task-defined goal q_{goal}. The constraints (32)–(33) ensure that the trajectory remains feasible and does not violate the robot's joint position and velocity limits, respectively. The last constraint (34) guarantees safety by ensuring that the robot's forward occupancy does not collide with any obstacles in the environment.

For bi manual manipulation for computing safe motion plans a separate problem is formulated with self collision avoidance and input parameters constraints, solved using a nonlinear optimization program in a receding-horizon manner as well.

$$\min_{k\in K} \quad \texttt{cost}(q_{\text{goal}}, k) \tag{35}$$

$$q_j(T_i;k) \subseteq [q^-_{j,\text{lim}}, q^+_{j,\text{lim}}] \qquad \forall(i,j)\in N_t \times N_q \tag{36}$$

$$\dot q_j(T_i;k) \subseteq [\dot q^-_{j,\text{lim}}, \dot q^+_{j,\text{lim}}] \qquad \forall(i,j)\in N_t \times N_q \tag{37}$$

$$\mathbf{s_d}(\mathcal{SFO}_1, \mathcal{O}_n) > 0 \qquad \forall n \in N_{\mathcal{O}} \tag{38}$$

$$\mathbf{s_d}(\mathcal{SFO}_2, \mathcal{O}_n) > 0 \qquad \forall n \in N_{\mathcal{O}} \tag{39}$$

$$\mathbf{s_d}(\mathcal{SFO}_1, \mathcal{C}_{2,n}) > r_{1,n} \qquad \forall n \in N_{\mathcal{J}} \tag{40}$$

where $\mathcal{SFO}_1$, $\mathcal{SFO}_2$ represents SFOs of arm1 and arm2 respectively, O_n is the n^{th} obstacle for $n \in N_{\mathcal{O}}$, $\mathcal{C}_{2,n}$ is the n^{th} joint centre of arm2 and $r_{1,n}$ is the radius of each joint approximation for arm1, for $n \in N_{\mathcal{J}}$ The rest of the operations are similar to the nonlinear optimization problem for safe locomotion.

In the remainder of this section, we describe how replaces separate neural networks replaces traditional methods with a fast, neural representation with probabilistic safety-guarantees.

A. Neural Network Models

In this subsection, we describe how individual learns neural network representations: centers and radii of $\mathcal{SFO}$, their gradients, and joint torques with respect to the trajectory parameter.

1) Neural SFO: Given a family of parameterized trajectories $q(t;k)$ for $k \in K$, we train a neural network to predict the centers $c_{j,i}(q_j(T_i;k))$ and radii $r_{j,i}(q_j(T_i;k))$ for all $(i,j) \in N_t \times N_q$ of the Spherical Forward Occupancy. For simplicity, we have dropped the subscript m to indicate that the center and radii networks only predict the spheres that overapproximate the joint volumes.

Following training, we use [43][Lem. 9, Thm. 10] to construct the remaining spheres that overapproximate the links. The network takes as input $x = (q_0, \dot q_0, k, i)$, which consists of a concatenation of vectors corresponding to the initial joint positions q_0, initial joint velocities $\dot q_0$, trajectory parameter k, and the i^{th} time index. The output of the network is $y = (\hat c_{j,i}, \hat r_{j,i}) \in \mathbb{R}^{4(n_q+1)}$. We discuss specific training details in Sec. IV-A.3.

2) Neural SFO Gradient: To facilitate real-time solutions of $\texttt{(Opt)}$, we also train a neural network to output $\widehat{\frac{\partial c_{j,i}}{\partial k}}$, which is a prediction of the gradient of each joint center with respect to the trajectory parameter k. Because the radii of $\mathcal{SFO}$ do not depend on k, we do not train its corresponding gradient network.

3) Neural Control Parameters: Similar to SFO given, given a family of parameterized trajectories $q(t;k)$ for $k \in K$, we train a neural network to predict the control parameters. The network takes as input $x = (q_0, \dot q_0, \ddot q_0)$, which consists of a concatenation of vectors corresponding to the initial joint positions q_0, initial joint velocities $\dot q_0$, initial joint accelerations $\ddot q_0$. The input although doesn't consider auxiliary joint velocities but are formulated using the initial joint velocities for parameter generation using $PZRNEA$.

The output of the network is $y = (\hat{t}_j, \hat{f}_{j,i}, \hat{m}_{j,i}) \in \mathbb{R}^{4(n_q+1)}$. The model predicts torques, forces and moments out of which currently joint torques will be utilized for constraint evaluation.

B. Conformal Prediction

Next, we describe the use of conformal prediction to formulate probabilistic guarantees for collision avoidance.

1) Calibration: We first construct a calibration set $\mathcal{D}_{cal} = \{(x_d, y_d)\}_{d=1}^{N_{cal}}$ with i.i.d. samples, where $N_{cal} \in \mathbb{N}$. Let $\mathcal{B}_{j,i} = \mathcal{B}(c_{j,i}, r_{j,i})$ be a ground truth sphere and $\hat{\mathcal{B}}_{j,i}(\delta_j) = \mathcal{B}(\hat{c}_{j,i}, \hat{r}_{j,i} + \delta_j)$ be the corresponding prediction. Note that $\mathcal{B}_{j,i}$ corresponds to $S_{j,i}(q(T_i;k))$ in Thm. 3. Then for the calibration step, we define a nonconformity score δ_j for each joint sphere such that

$$\delta_j = \max(||c_{j,i} - \hat{c}_{j,i}||_2 + r_{j,i} - \hat{r}_{j,i},\ 0) \tag{41}$$

is minimum buffer required for $\mathcal{B}_{j,i} \subseteq \hat{\mathcal{B}}_{j,i}(\delta_j)$.

Given the nonconformity scores over the calibration set $\mathcal{D}_{cal}$, the nonconformity score $\delta_{j,d'}$ on a new sample $(x_{d'}, y_{d'})$ follows a gaurantee conditioned on the calibration set $\mathcal{D}_{cal}$ with the probability of $1-\rho$ [38], [46]:

$$\mathbb{P}(\delta_{j,d'} \leq \Delta_{j,1-\hat{\epsilon}} | \mathcal{D}_{cal}) \geq \text{Beta}_{N_{cal}+1-\nu,\nu}(\rho) \tag{42}$$

where $\nu := \lfloor (N_{cal}+1)\hat{\epsilon} \rfloor$, $\text{Beta}_{N_{cal}+1-\nu,\nu}(\rho)$ is the ρ-quantile of the Beta distribution, $\hat{\epsilon}$ is a user-defined variable for desired coverage, and $\Delta_{j,1-\hat{\epsilon}}$ is the $\lceil (N_{cal}+1)(1-\hat{\epsilon}) \rceil$-th quantile of the nonconformity scores over the calibration set $\mathcal{D}_{cal}$. To facilitate the following exposition, let

$$\mathbb{P}(\cdot|\mathcal{D}_{cal}) = \mathbb{P}(\cdot), \tag{43}$$

$$\Delta_{j,1-\hat{\epsilon}} = \Delta_j, \tag{44}$$

and

$$\text{Beta}_{N_{cal}+1-v,v}(\rho) = 1-\epsilon. \tag{45}$$

Then the guarantee in (42) can be written as:

$$\mathbb{P}(\delta_j \leq \Delta_j) \geq 1-\epsilon \tag{46}$$

where $1-\epsilon$ is desired coveraged specified by $\hat{\epsilon}$.

2) Conformalized Reachable Sets: Given (46), the conformalized sphere $\hat{\mathcal{B}}_{j,i}(\Delta_j)$ is guaranteed to enclose the ground truth sphere $\mathcal{B}_{j,i}$ with the following probability:

$$\begin{aligned}
&\mathbb{P}(\mathcal{B}_{j,i} \subseteq \hat{\mathcal{B}}_{j,i}(\Delta_j)) && (47)\\
&\geq \mathbb{P}(\mathcal{B}_{j,i} \subseteq \hat{\mathcal{B}}_{j,i}(\delta_j)) \cdot \mathbb{P}(\hat{\mathcal{B}}_{j,i}(\delta_j) \subseteq \hat{\mathcal{B}}_{j,i}(\Delta_j)) && (48)\\
&= \mathbb{P}(\hat{\mathcal{B}}_{j,i}(\delta_j) \subseteq \hat{\mathcal{B}}_{j,i}(\Delta_j)) && (49)\\
&= \mathbb{P}(\delta_j \leq \Delta_j) \geq 1-\epsilon. && (50)
\end{aligned}$$

This approach can be extended to all joint occupancy spheres to construct a probabilistic safety guarantee to ensure that the ground truth reachable set is collision-free with probability greater than $(1-\epsilon)^{n_q+1}$ over the i^{th} time interval T_i. We summarize this result in the following theorem whose proof can be found in the online appendix[1] :

[1]https://roahmlab.github.io/crows/

Theorem 9. *Given $\mathbb{P}(\mathcal{B}_{j,i} \subseteq \hat{\mathcal{B}}_{j,i}(\Delta_j)) \geq 1-\epsilon$ for $j \in N_q$, the following probability holds under the condition $\mathbf{s_d}(\mathcal{S}\hat{\mathcal{F}}\mathcal{O}_i, \mathscr{O}) > 0$:*

$$\mathbb{P}(\mathbf{s_d}(FO_i, \mathscr{O}) > 0) \geq (1-\epsilon)^{n_q+1} \tag{51}$$

where FO_i denotes the forward occupancy over a single time interval T_i ((2)), $\mathcal{S}\hat{\mathcal{F}}\mathcal{O}_i$ is the Spherical Forward Occupancy constructed from $\hat{\mathcal{B}}_{j,i}(\Delta_j)$ for all $j \in N_q$ over the same time interval T_i by Theorem 3, $\mathscr{O}$ represents a set of obstacles defined as $\{\mathcal{O}_n \mid n \in N_{\mathcal{O}}\}$, and $\mathbf{s_d}(\cdot, \mathscr{O}) = \min_{n \in N_{\mathcal{O}}} \mathbf{s_d}(\cdot, \mathcal{O}_n)$.

3) Conformalized Control Parameters: To extend the probabilistic guarantees to the control domain, we formulate nonconformity scores for the dynamic parameters that govern the robot's motion: torques (τ), forces (f), and moments (m). These parameters are crucial for ensuring reliable execution of planned trajectories while maintaining safety constraints. For the control parameters, we define relative nonconformity scores that capture the normalized deviation between predicted and true values:

$$s_\tau = \frac{|\tau - \hat{\tau}|_2}{|\tau|_2} \tag{52}$$

$$s_f = \frac{|f - \hat{f}|_2}{|f|_2} \tag{53}$$

$$s_m = \frac{|m - \hat{m}|_2}{|m|_2} \tag{54}$$

where $\hat{\tau}$, $\hat{f}$, and $\hat{m}$ represent the predicted values, and $|\cdot|_2$ denotes the Euclidean norm. The normalization by the magnitude of true values ensures scale-invariant conformity measures across different operating conditions. Similar to the spatial occupancy guarantees, we obtain quantile bounds Δ_τ, Δ_f, and Δ_m from the calibration set such that:

$$\mathbb{P}(s_\tau \leq \Delta_\tau) \geq 1-\epsilon_\tau \tag{55}$$

$$\mathbb{P}(s_f \leq \Delta_f) \geq 1-\epsilon_f \tag{56}$$

$$\mathbb{P}(s_m \leq \Delta_m) \geq 1-\epsilon_m \tag{57}$$

where ϵ_τ, ϵ_f, and ϵ_m are user-defined coverage levels for each control parameter. These probabilistic bounds can be incorporated into the optimization problem (62)–(65) by adding the following constraints:

$$|\tau(T_i;k)|_2 \leq \frac{\tau\text{max}}{1+\Delta_\tau} \qquad \forall i \in N_t \tag{58}$$

$$|f(T_i;k)|_2 \leq \frac{f\text{max}}{1+\Delta_f} \qquad \forall i \in N_t \tag{59}$$

$$|m(T_i;k)|_2 \leq \frac{m\text{max}}{1+\Delta_m} \qquad \forall i \in N_t \tag{60}$$

where $\tau_{\max}$, $f_{\max}$, and $m_{\max}$ are the maximum allowable values for each control parameter. The scaling factors $(1+\Delta_{\cdot})^{-1}$ ensure that the predicted values remain within physical limits with the specified probability, accounting for prediction uncertainty. This formulation provides a unified framework for trajectory optimization that considers both spatial safety constraints and control parameter bounds with probabilistic

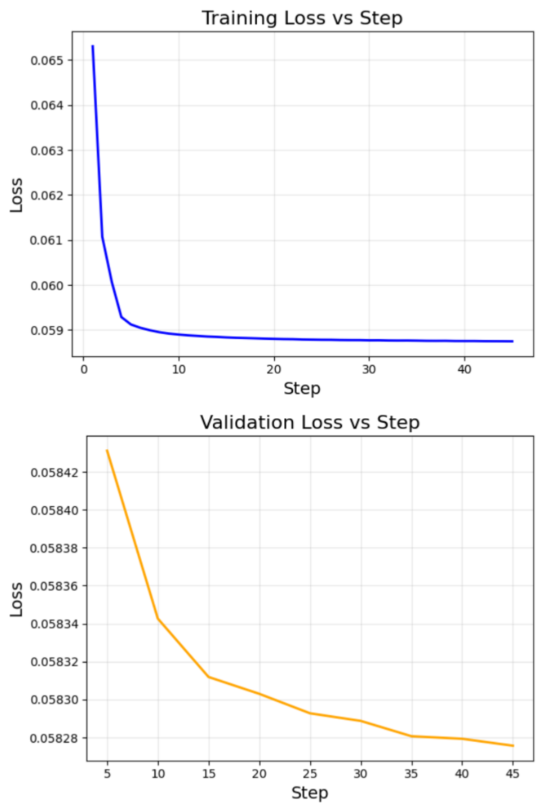

Fig. 4: Training and Validation Plots obtained from SFO training for Digit

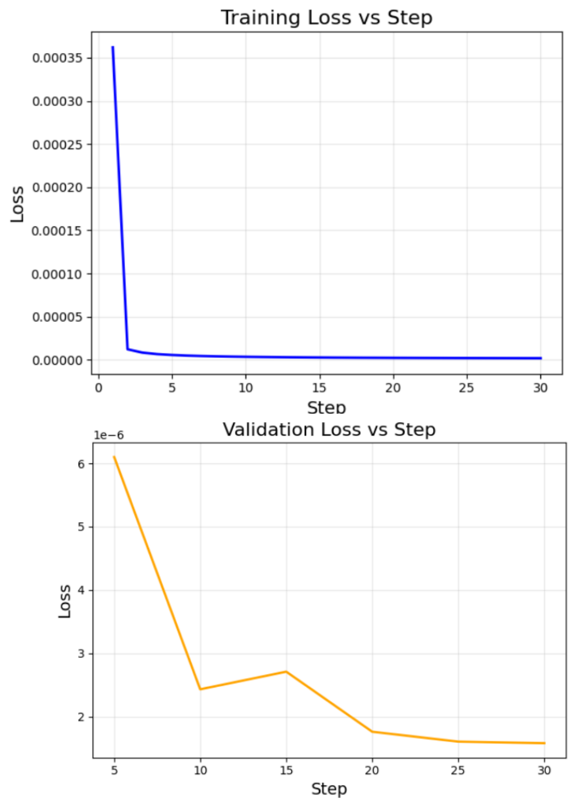

Fig. 5: Training and Validation Plots obtained from SFO training for Arm

guarantees. The joint probability of satisfying all constraints becomes:

$$\mathbb{P}(\text{safe}) \geq (1-\epsilon)^{n_q+1} \cdot (1-\epsilon_\tau) \cdot (1-\epsilon_f) \cdot (1-\epsilon_m) \quad (61)$$

where the first term comes from Theorem 9 for spatial safety, and the remaining terms represent the probability of satisfying control parameter bounds.

C. Online Trajectory Optimization

After training, we generate models to predict the spheres of the $\mathcal{SFO}$. Using this representation, we can reformulate the optimization problem described by (31)–(34) into:

$$\min_{k \in K} \quad \texttt{cost}(q_{\text{goal}}, k) \qquad (\texttt{CROWS-Opt}) \qquad (62)$$

$$q_j(T_i; k) \subseteq [q^-_{j,\text{lim}}, q^+_{j,\text{lim}}] \qquad \forall (i,j) \in N_t \times N_q \qquad (63)$$

$$\dot{q}_j(T_i; k) \subseteq [\dot{q}^-_{j,\text{lim}}, \dot{q}^+_{j,\text{lim}}] \qquad \forall (i,j) \in N_t \times N_q \qquad (64)$$

$$\mathbf{s_d}(\hat{\mathcal{SFO}}_i, \mathscr{O}) > 0 \qquad \forall i \in N_t \qquad (65)$$

where $\hat{\mathcal{SFO}}$ is the conformalized reachable set in Thm. 9.

IV. Results

We demonstrate the performance of these neural networks in simulation. For the locomotion part the neural network is trained to predict the center and radii of joints, which was optimized to some extent. This part has been paused right now due to constraints in the real world implementation. Further we talk about bimanual manipulation implementation, this part is implemented and compared using Bernstein trajectories. This part we train similarly a neural network to predict the centers and radii, along with SDF and input parameter prediction.

A. Implementation Details

A computer with 12 Intel(R) Core(TM) i7-8700K CPU @ 3.70GHz and an NVIDIA RTX A6000 GPU was used for the motion planning experiment in Sec. IV-D. The neural networks are built and trained with Pytorch [47]. The trajectory optimization was solved with IPOPT [48].

1) Simulation and Simulation Environment: We simulate using Digit v2 Humanoid for safe locomotion planning evaluation and single as well as multiple Kinova Gen3 7-DOF serial manipulator [49] for bimanual manipulation implementation. Both robot's collision geometry is provided as a mesh, which we utilize with the trimesh library [50] to check for collisions with obstacles. For simplicity, we only check self-collisions between both arms in bi-manual implementation only and all obstacles are static, axis-aligned cubes. We assume that the start and goal configurations of the robot are collision-free.

2) Training Details for Digit SFO: Previous to training models, a validation test for SFO generation was carried out using the SPARROWS methodology. Initially a fixed let toe Unified Robot Description Format (URDF) was utilized, but it led significant error accumulation in over-approximation of joints near to the end of kinematic chains. This is due the reason of very long kinematic chains which accumulate error over time. A fixed torso URDF was then employed as it significantly decreased the length of the kinematic chains leading to tight approximations. After experimneting

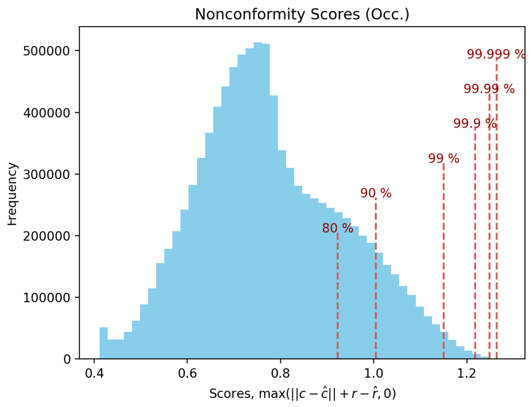

Fig. 6: Non conformity scores of trained model for Digit

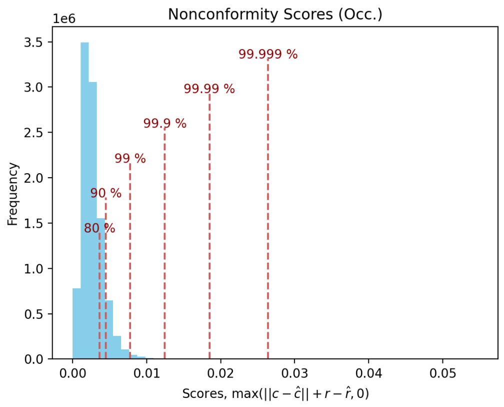

Fig. 7: Non conformity scores of trained model for Arm

and fine-tuning the networks, fully-connected networks were trained to predict the centers, radii, and gradients of the centers with respect to the trajectory parameter. These networks have a width 1024 with 18, 24, and 28 hidden layers, respectively. All the networks used GELU activation function. [51]. The model also dropout and batch normalization layers. As discussed in III-A.1, the networks only predict the centers and radii that over approximate the joint volumes. For the loss function, we used the mean squared error between the target and the prediction. Each network was trained using the AdamW [52] optimizer, uses learning rate scheduler starting with 0.0002, beta (0.8,0.99), and weight decay 0.00001. The training, validation, and calibration datasets consisted of 8e5, 1e5, and 1e5 samples, respectively. Lastly, The model is trained for 45 epochs after which the loss convergence stagnates, depicted in Fig. 4.

3) Training Details for Arm SFO: Using Bernstein trajectories joint reachable sets were created which were utilized to formulate the dataset. As the kinematic chains are simpler in case of arm's the training complexity is decreased compared to Digit resulting faster convergence and usage of small networks. Fully-connected networks were trained to predict the centers, radii, and gradients of the centers with respect to the trajectory parameter. These networks have a width 1024 with 5, 9, and 12 hidden layers, respectively. The radii network uses the ReLU activation function while the others use GELU [51]. As discussed in III-A.1, the networks only predict the centers and radii that overapproximate the joint volumes. The networks also do not predict the center and radius of the base joint sphere as it remains constant for all time. For the loss function, we used the mean squared error between the target and the prediction. Each network was trained using the AdamW [52] optimizer with learning rate 0.0003, beta (0.9,0.999), and weight decay 0.0001. The training, validation, and calibration datasets consisted of 8e5, 1e5, and 1e5 samples, respectively. The training was carried out only for 30 epochs which lead to optimal loss convergence, depicted in Fig. 5. Similarly a model is trained for bi-manual arms.

B. Model Prediction Accuracy

1) Digit SFO model: The mean and maximum errors of the center predictions across all joints are 3.23cm and 10.20cm, respectively. For the radius predictions, the mean and maximum errors are 0.567 m and 1.28m. The nonconformity scores (41) are computed using the calibration set. The 99.9$^{\text{th}}$ percentile of the nonconformity scores for joints 2 through 23 are (0.02, 0.12, 0.13, 0.214, 0.212, 0.354, 0.362,0.422, 0.412, 0.43, 0.404, 0.418, 0.824, 0.432, 0.862, 0.652, 1.252, 0.61, 1.234, 0.798, 1.264, 0.832, 1.224) m, measured from the proximal to distal joint. The overall nonconformity score graph is depicted in Fig. 6. As observable from results the the prediction accuracy of the model is not useful for real-world applications. The model performance can be improved a larger data size and utilizing models like transformers. Currently not optimized further due to the side implementation for bi-manual manipulation.

2) Arm SFO model: The mean and maximum errors of the center predictions across all joints are 0.94cm and 2.80cm, respectively. For the radius predictions, the mean and maximum errors are 0.18 cm and 0.64cm. The median relative error of the neural SFO gradient is 1.47%. The nonconformity scores (41) are computed using the calibration set. The 99.9$^{\text{th}}$ percentile of the nonconformity scores for joints 2 through 8 are (0.02, 0.03, 0.04, 0.54, 0.76, 0.84, 1.02) cm, measured from the proximal to distal joint. In the trajectory optimization `(CROWS-Opt)`, an uncertainty bound of $\hat{\epsilon} = 0.001$ was used. Fig. 7 depicts the overall non conformity scores obtained after SFO training for the arm.

C. Runtime Comparisons

We compare the runtime performance of bi-manual manipulation with single and multi arm comparison between CROWS and baseline SPARROWS [43] under a planning time limit while varying the number of obstacles. We measure the mean constraint evaluation time as well as the mean planning time. Constraint evaluation includes the time to compute the constraints and the constraint gradients. Planning time includes the time required to construct the reachable sets and the total time required to solve `(Opt)`

Methods	mean constraint eval. time [ms]	
# Obstacles (s)	10	20
CROWS-I	3.7 ± 0.2	4.1 ± 0.1
CROWS-II	**3.9 ± 0.4**	**4.8 ± 0.3**
SPARROWS-I	**3.2± 0.1**	**3.9 ± 0.1**
SPARROWS-II	7.1 ± 0.1	9.4 ± 0.1

TABLE I: Mean runtime for constraint and constraint gradient evaluation across 10 **Random Obstacle Scenarios** under a 0.5s planning time limit.

Methods	mean planning time [s]	
# Obstacles (s)	10	20
CROWS-I	0.14 ± 0.29	0.16 ± 0.10
CROWS-II	0.16 ± 0.20	0.23 ± 0.10
SPARROWS-I	0.16 ± 0.08	0.18 ± 0.08
SPARROWS-II	0.25 ± 0.14	0.31 ± 0.10

TABLE II: Mean per-step planning time across 10 **Random Obstacle Scenarios** under a 0.5s planning time limit.

including constraint gradients evaluations. For each case, the results are averaged over 10 trials. CROWS-I, SPARROWS-I represents for single arm manipulation, and CROWS-II, SPARROWS-II represents for bi-manual manipulation.

Tables I and II summarize the runtime comparisons under a planning time limit of $0.5s$. SPARROWS-I has the lowest mean constraint evaluation time for single arm manipulation, on the other hand, CROWS- II has the lowest mean constraint evaluation time for bi-manual manipulation. In per-step planning time CROWS is the fastest for both manipulation tasks. These results indicate that learned CROWS provides best results for bi-manual manipulation.

D. Motion Planning Experiments

We compare the performance of CROWS to SPARROWS for both single and bi-manual manipulation tasks with **Random Obstacle Scenarios**, containing $n_{\mathcal{O}} = 10$, axis-aligned boxes that are 20cm on each side. For each number of obstacles, we generate 10 scenes with obstacles placed randomly such that the start and goal configurations are collision-free. For both planning tasks, a failure occurs if CROWS or SPARROWS fails to find a plan for two consecutive planning iterations or if a collision occurs. CROWS and SPARROWS are given a planning time limit of 0.5s and a maximum of 150 planning iterations to reach the goal.

Tables and III compare the success rate of CROWS to the baseline(SPARROWS) for **Random Obstacle Scenarios** (Fig. 8), for five scenarios.

SPARROWS achieves the highest success rate on both sets of tasks followed. Note that for both configurations CROWS is competitive with SPARROWS while also remaining collision-free.

V. Conclusion and Future Directions

We implement and analyse advancements in safe motion planning for high-dimensional robotic systems, demonstrating the effectiveness of our approach across two challenging domains: bi-manual manipulation and bipedal locomotion. Through experimental validation, we have shown that our

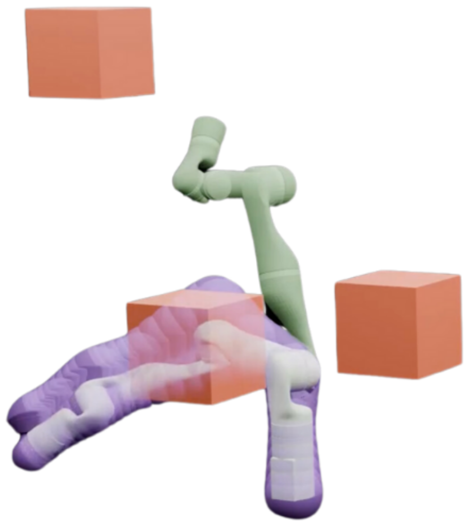
Fig. 8: **Random Obstacle Scenario** for single arm

Methods	# Successes
CROWS-I	5
CROWS-II	4
SPARROWS-I	5
SPARROWS-II	5

TABLE III: Number of successes out of 5 **Random Obstacle Scenarios**.

approach significantly reduces computational overhead compared to traditional methods while preserving rigorous safety constraints.

In the context of bi-manual manipulation, our results demonstrate that learning-based approximations of reachable sets can maintain safety while improving planning efficiency. The method shows particular promise in cluttered environments, where traditional approaches often struggle with computational tractability. The success in this domain has laid the groundwork for the ongoing work for the suction gripper project which could also involve integrating control parameter constraints and visual perception through Gaussian splatting, currently being developed for submission to CoRL 2025. This extension aims to bridge the gap between geometric reasoning and visual understanding, potentially enabling more robust bi-manual manipulation in unstructured environments.

However, our work also reveals important limitations and opportunities for future research. In the bipedal locomotion domain, while our method provides theoretical guarantees although not very accurate and tight, the current safety constraints proved challenging to implement in real-world scenarios. This limitation stems primarily from the current limitations in Digit's control architecture, particularly in calculating torso error accurately. The performance of the prediction model can be improved by utilizing a bigger dataset possible with 1e7 samples or more. Additionally, other deep learning model like transformers can also be implemented and tested for improvements.

VI. Learning Outcomes

The comprehensive research experience has yielded valuable expertise across multiple domains of robotics and control systems. Through deep engagement with reachability analysis and safety verification, the project provided thorough understanding of over-approximation techniques and polynomial zonotopes for representing forward occupancy and robot dynamics. The integration of neural networks with traditional robotics approaches enhanced knowledge of modern machine learning methods in robotics applications. Practical experience of trajectory optimization and real-time planning algorithms emerged through practical implementation, while system integration skills developed through combining multiple components into a cohesive framework. The theoretical aspects of the project strengthened understanding of formal methods for proving safety guarantees in robotic systems, particularly in high-dimensional spaces.

References

[1] P. Holmes, S. Kousik, B. Zhang, *et al.*, " Reachable Sets for Safe, Real-Time Manipulator Trajectory Design," in *Proceedings of Robotics: Science and Systems*, Corvalis, Oregon, USA, Jul. 2020.

[2] J. Michaux, P. Holmes, B. Zhang, *et al.*, *Can't touch this: Real-time, safe motion planning and control for manipulators under uncertainty*, 2023.

[3] J. Schulman, Y. Duan, J. Ho, *et al.*, "Motion planning with sequential convex optimization and convex collision checking," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014.

[4] F. Caccavale, "Inverse dynamics control," in *Encyclopedia of Robotics*, M. H. Ang, O. Khatib, and B. Siciliano, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2020, pp. 1–5.

[5] S. Kousik, S. Vaskov, M. Johnson-Roberson, and R. Vasudevan, "Safe trajectory synthesis for autonomous driving in unforeseen environments," in *ASME 2017 Dynamic Systems and Control Conference*, American Society of Mechanical Engineers Digital Collection, 2017.

[6] D. Blackmore and M. Leu, "A differential equation approach to swept volumes," in *[1990] Proceedings. Rensselaer's Second International Conference on Computer Integrated Manufacturing*, May 1990, pp. 143–149.

[7] D. Blackmore and M. Leu, "Analysis of Swept Volume via Lie Groups and Differential Equations," en, *The International Journal of Robotics Research*, vol. 11, no. 6, pp. 516–537, Dec. 1992, Publisher: SAGE Publications Ltd STM.

[8] D. Blackmore, M. C. Leu, and F. Shih, "Analysis and modelling of deformed swept volumes," en, *Computer-Aided Design*, Special Issue: Mathematical methods for CAD, vol. 26, no. 4, pp. 315–326, Apr. 1994.

[9] D. Blackmore, M. Leu, and L. P. Wang, "The sweep-envelope differential equation algorithm and its application to NC machining verification," en, *Computer-Aided Design*, vol. 29, no. 9, pp. 629–637, Sep. 1997.

[10] D. Blackmore, R. Samulyak, and M. C. Leu, "Trimming swept volumes," en, *Computer-Aided Design*, vol. 31, no. 3, pp. 215–223, Mar. 1999.

[11] Lozano-Perez, "Spatial planning: A configuration space approach," *IEEE Transactions on Computers*, vol. C-32, no. 2, pp. 108–120, 1983.

[12] N. Perrin, O. Stasse, L. Baudouin, F. Lamiraux, and E. Yoshida, "Fast humanoid robot collision-free footstep planning using swept volume approximations," *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 427–439, 2012.

[13] M. Campen and L. P. Kobbelt, "Polygonal boundary evaluation of minkowski sums and swept volumes," *Computer Graphics Forum*, vol. 29, 2010.

[14] Y. J. Kim, G. Varadhan, M. C. Lin, and D. Manocha, "Fast swept volume approximation of complex polyhedral models," *Comput. Aided Des.*, vol. 36, pp. 1013–1027, 2003.

[15] A. Gaschler, R. P. A. Petrick, T. Kröger, O. Khatib, and A. Knoll, "Robot task and motion planning with sets of convex polyhedra," in *Robotics: Science and Systems Conference*, 2013.

[16] C. Ekenna, D. Uwacu, S. Thomas, and N. M. Amato, "Improved roadmap connection via local learning for sampling based planners," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 3227–3234.

[17] S. Duenser, J. M. Bern, R. Poranne, and S. Coros, "Interactive robotic manipulation of elastic objects," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 3476–3481.

[18] M. Gaertner, M. Bjelonic, F. Farshidian, and M. Hutter, *Collision-free mpc for legged robots in static and dynamic scenes*, 2021.

[19] B. Brito, B. Floor, L. Ferranti, and J. Alonso-Mora, *Model predictive contouring control for collision avoidance in unstructured dynamic environments*, 2020.

[20] C. Dube, "Self collision avoidance for humanoids using circular and elliptical capsule bounding volumes," in *2013 Africon*, 2013, pp. 1–6.

[21] A. El Khoury, F. Lamiraux, and M. Taïx, "Optimal motion planning for humanoid robots," in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 3136–3141.

[22] M. Zucker, N. Ratliff, A. D. Dragan, *et al.*, "Chomp: Covariant hamiltonian optimization for motion planning," *The International Journal of Robotics Research*, vol. 32, no. 9-10, pp. 1164–1193, 2013.

[23] A. T. Le, G. Chalvatzaki, A. Biess, and J. Peters, *Accelerating motion planning via optimal transport*, 2023.

[24] B. Sundaralingam, S. K. S. Hari, A. Fishman, *et al.*, *Curobo: Parallelized collision-free minimum-jerk robot motion generation*, 2023.

[25] N. Kochdumper and M. Althoff, "Sparse polynomial zonotopes: A novel set representation for reachability analysis," *IEEE Transactions on Automatic Control*, vol. 66, no. 9, pp. 4043–4058, 2020.

[26] Z. Brei, J. Michaux, B. Zhang, P. Holmes, and R. Vasudevan, "Serving time: Real-time, safe motion planning and control for manipulation of unsecured objects," *IEEE Robotics and Automation Letters*, vol. 9, no. 3, pp. 2383–2390, 2024.

[27] Y. Kwon, J. Michaux, S. Isaacson, *et al.*, *Conformalized reachable sets for obstacle avoidance with spheres*, 2024.

[28] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *international conference on machine learning*, PMLR, 2016, pp. 1050–1059.

[29] A. Shamsi, H. Asgharnezhad, M. Abdar, *et al.*, "Improving mc-dropout uncertainty estimates with calibration error-based optimization," *arXiv preprint arXiv:2110.03260*, 2021.

[30] B. Lütjens, M. Everett, and J. P. How, "Safe reinforcement learning with model uncertainty estimates," in *2019 International Conference on Robotics and Automation (ICRA)*, IEEE, 2019, pp. 8662–8668.

[31] E. Daxberger, A. Kristiadi, A. Immer, R. Eschenhagen, M. Bauer, and P. Hennig, "Laplace redux-effortless bayesian deep learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 20 089–20 103, 2021.

[32] H. Ritter, A. Botev, and D. Barber, "A scalable laplace approximation for neural networks," in *6th international conference on learning representations, ICLR 2018-conference track proceedings*, International Conference on Representation Learning, vol. 6, 2018.

[33] L. Goli, C. Reading, S. Sellán, A. Jacobson, and A. Tagliasacchi, "Bayes' rays: Uncertainty quantification for neural radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 20 061–20 070.

[34] K. Chua, R. Calandra, R. McAllister, and S. Levine, "Deep reinforcement learning in a handful of trials using probabilistic dynamics models," *Advances in neural information processing systems*, vol. 31, 2018.

[35] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," *Advances in neural information processing systems*, vol. 30, 2017.

[36] A. N. Angelopoulos and S. Bates, "A gentle introduction to conformal prediction and distribution-free uncertainty quantification," *arXiv preprint arXiv:2107.07511*, 2021.

[37] G. Shafer and V. Vovk, "A tutorial on conformal prediction.," *Journal of Machine Learning Research*, vol. 9, no. 3, 2008.

[38] A. Dixit, Z. Mei, M. Booker, M. Storey-Matsutani, A. Z. Ren, and A. Majumdar, "Perceive with confidence: Statistical safety

assurances for navigation with learning-based perception," in *8th Annual Conference on Robot Learning*, 2024.

[39] M. Zhao, R. Simmons, H. Admoni, and A. Bajcsy, "Conformalized teleoperation: Confidently mapping human inputs to high-dimensional robot actions," *arXiv preprint arXiv:2406.07767*, 2024.

[40] L. Lindemann, M. Cleaveland, G. Shim, and G. J. Pappas, "Safe planning in dynamic environments using conformal prediction," *IEEE Robotics and Automation Letters*, 2023.

[41] J. Lekeufack, A. N. Angelopoulos, A. Bajcsy, M. I. Jordan, and J. Malik, "Conformal decision theory: Safe autonomous decisions from imperfect predictions," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2024, pp. 11 668–11 675.

[42] A. Lin and S. Bansal, "Verification of neural reachable tubes via scenario optimization and conformal prediction," in *6th Annual Learning for Dynamics & Control Conference*, PMLR, 2024, pp. 719–731.

[43] J. Michaux, A. Li, Q. Chen, C. Chen, B. Zhang, and R. Vasudevan, "Safe planning for articulated robots using reachability-based obstacle avoidance with spheres," *ArXiv*, vol. abs/2402.08857, 2024.

[44] M. Spong, S. Hutchinson, and M. Vidyasagar, "Robot modeling and control," 2005.

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

[46] V. Vovk, "Conditional validity of inductive conformal predictors," in *Asian conference on machine learning*, PMLR, 2012, pp. 475–490.

[47] A. Paszke, S. Gross, S. Chintala, *et al.*, "Automatic differentiation in pytorch," 2017.

[48] A. Wächter and L. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, pp. 25–57, Mar. 2006.

[49] Kinova, *User Guide - KINOVA Gen3 Ultra lightweight robot*. 2022.

[50] Dawson-Haggerty et al., *Trimesh*, version 3.2.0, 2019.

[51] D. Hendrycks and K. Gimpel, *Gaussian error linear units (gelus)*, 2023.

[52] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2017.