A Safety-Focused Admittance Control Approach for Physical Human-Robot Interaction with Rigid Multi-Arm Serial Link Exoskeletons

Jianwei Sun[1], Erik Harrison Kramer[1], and Jacob Rosen[1]

***Abstract*—Ensuring safety in physical human-robot interaction is challenging due to hardware and control architecture differences across robots, and is often implemented as system-dependent ad-hoc approaches. To offer a holistic solution, we present a hardware-independent safety-focused admittance control approach which promotes safety at the reference-generation level. This safety framework can restrict virtual dynamics through soft virtual bounds. Hard bounds are also introduced as a way to impose infinitely stiff soft bounds. As part of the overall approach, we also present a method for serial manipulator and multi-segment entity collision avoidance by using partial Jacobians. In order to demonstrate the methodology's versatility across hardware platforms, we experimentally validate on two robotic systems: (1) the V-Rex, a non-anthropomorphic full-body haptic device composed of five robotic arms interacting with the body at the hands, feet, and pelvis; and (2) the EXO-UL8, an anthropomorphic bimanual upper-limb exoskeleton; which exist on opposite ends of the task/joint space control, non-redundant/redundant, off-the-shelf (industrial)/custom, non-anthropomorphic/anthropomorphic spectra. Experimental results validate virtual dynamics, soft and hard bounds, and multi-arm collision avoidance on both systems. In all cases, both systems respect bound and collision constraints, supporting the approach as a safety-focused admittance control design. Implementation at https://github.com/jianwei-sun/gtfo.**

***Index Terms*—Physical human-robot interaction (pHRI), safety, admittance control, collision-avoidance, exoskeletons, serial link manipulators, multi-arm systems**

I. Introduction

In *physical human-robot interaction* (pHRI), admittance control is often used to generate desired trajectories for controlling how a robot should respond to human-applied forces. In any human-robot interaction, safety is paramount, and must therefore be addressed at not only the hardware level, but also throughout the control. This paper presents a safety-focused admittance control approach for pHRI with rigid multi-arm serial link exoskeletons. Safety in pHRI is a complex problem that has no perfect solution, but has been approached through many ways. Most directly, mechanical safety, such as joint-stops [1], [2], back-drivable actuators [2], e-stops [2], [3], and compliance [4], [5], provide limited degrees of safety. While actuator output can be reduced to levels that minimize harm, doing so sacrifices performance and can result in undesirable behavior [6]. E-stops require human input, which may be delayed during dangerous situations. Compliant and soft interfaces may be mechanically simple to implement, but can significantly complicate the control [4], [5]. Furthermore, these mechanical solutions may not be viable to off-the-shelf manipulators that were not designed with pHRI in mind, and are better suited for custom designs.

Safety research also includes controller-level approaches, which can be implemented entirely in software and/or rely on minor hardware additions. These include: human-tracking through vision [7] or motion-based sensors [8]; monitoring instability through indices or heuristics [9], [10]; saturating or filtering some aspect of human-applied signals [11]–[13]; and so on. While human tracking methods can provide useful real-time information, they may require additional complexity such as more sensors [7], [8], and/or placing sensors directly on the human [14], [15], which detracts from ease of use. Detection of unsafe interaction through performance metrics or heuristics has shown good results [9], [10], but are usually specific to the system and difficult to generalize. Saturation of signals is not necessarily safer because the system can be slower to respond to unsafe situations.

Safety can also be tackled at the reference-generation level, which includes: dynamically tuning admittance control parameters [9], [16]–[18], defining virtual bounding regions [19]–[23], utilizing data-driven methods [18], [22], and collision avoidance [24]–[31]. Collision avoidance involves two aspects: detection (finding intersections between 3D shapes representing the robot's links) and avoidance (using robot kinematics to prevent dangerous motions). Existing simulation environments, such as MuJoCo [32], implement detection by finding intersections between convex hulls of the links. Although most accurate, the computation cost may not be justified in pHRI, which would want large bounding regions around human-operated exoskeletons. Other approaches make this trade-off by using simpler geometries [26], [28], [29]. [26] implements avoidance in task-space, requiring the Jacobian to be invertible which poses a problem for redundant manipulators. In [28] and [29], a series of virtual spheres encapsulate the manipulator, and the velocity between colliding spheres is restricted. However, adjusting the detection radius would modify their number and positions, requiring Jacobian recomputation for each sphere. [30] and [31] restrict the relative velocity between closest points on colliding manipulators. Whereas [30] assumes the closest points can be determined and only restricts the velocity to a fixed value, [31] uses an iterative scheme to compute these points and a quadratic program to restrict the velocity.

Our approach improves upon previous methods and extends the collision avoidance methodology of [26] by using line segments to provide analytic expressions for distances between primitives and utilizing the method from [33] to analytically

[1]The authors are with the Department of Mechanical and Aerospace Engineering at the University of California, Los Angeles (UCLA), USA 90095. {sunjianw1, ehkramer, jacobrosen}@ucla.edu

compute partial Jacobians at any arbitrary collision point.

In this paper, we present a comprehensive safety-oriented admittance control framework for pHRI with rigid multi-arm serial link exoskeletons. We validate this approach on two multi-arm exoskeletons which have significantly different hardware and control. Our approach facilitates admittance control by generating collision-free trajectories that emulate virtual second-order systems, while incorporating soft and hard boundary constraints for operation within a safe workspace.

Within this unified safety framework, we introduce a method for emulating infinitely stiff virtual bounds (sometimes referred to as virtual fixtures in robotic surgery [34]), which avoids issues of stability typically seen with high-stiffness virtual springs. We also present a method for collision avoidance between multi-segment entities by utilizing partial Jacobians. This collision avoidance technique uses the translational components of a robot manipulator Jacobian evaluated at potential collision points to determine safe motion directions, preventing collisions with not only the end effector, but also any point along the manipulator's links.

In summary, our contribution is a broad safety-centric admittance control approach that is comprised of emulating infinitely stiff virtual bounds and avoiding collisions for serial link manipulators. This framework can serve as both a safe self-contained reference generator for admittance control and a versatile lightweight safety intermediary layer, upon which customized applications can be developed. Moreover, individual elements of this framework can be employed modularly to enhance specific safety functionalities of pHRI systems.

II. Methodology

To enable pHRI, admittance control is used to generate motions of a virtual system with some desired dynamics from physical human-applied forces. The robot's controller then tracks these target trajectories, making the robot a physical manifestation of the virtual system. This behavior forms the foundation for advanced features such as safety bounds (limiting movement and speed) and multi-arm collision avoidance.

A. Virtual Dynamics

Emulating how a light-weight system moves is commonly done by propagating a virtual second-order mass-damper model [9], [13], [16], [18]. The second-order dynamics are chosen due to their similarities with physical mechanical systems and our existing intuition about how they react. Furthermore, their parameters are physically intuitive and can be easily tuned. The dynamics in a single dimension are:

$$m\ddot{p}(t) + b\dot{p}(t) = f(t), \tag{1}$$

where $p(\cdot) : \mathbb{R} \to \mathbb{R}$ is the position, $f(\cdot) : \mathbb{R} \to \mathbb{R}$ is the input force, $m \in \mathbb{R}_{>0}$ is the mass, and $b \in \mathbb{R}_{\geq 0}$ is the damping coefficient. The dynamics of vector systems are similar, in that each coordinate implements the scalar dynamics, allowing the system to be used for both task or joint-space admittance control. The state variables' explicit dependency on time will no longer be shown for brevity.

B. Bounds

One approach for safety in pHRI is to define regions in which motions are hindered. To this end, we define a bound, $B \subset \mathbb{R}^n$, as a convex closed subset of the space of virtual positions, where n is the dimension of the virtual system. Convexity is necessary so that projections into B, denoted by $\text{proj}_B(\cdot) : \mathbb{R}^n \to B$, are unique. From practice, we saw that norm bounds and rectangular bounds are most commonly used, so we focus on them. However, the methodology applies to any convex bound.

Moreover, we classify bounds as soft or hard. Soft bounds allow violation but generate a restoring force as a function of how far the virtual position is beyond the bound (typically implemented as a virtual spring-damper system), similar to impedance-based schemes [21], [35] or virtual fixtures [34], [36]. Stiff or rigid boundaries can be simulated using a high stiffness constant for the virtual spring, but such approaches can cause stability issues [34].

To address the limitation of using soft bounds to emulate infinitely stiff boundaries, we introduce hard bounds, which are inviolable at the reference generation level. These bounds constrain the virtual position and velocity directly, avoiding the need to integrate large restoring forces. Hard bounds are particularly useful for defining mechanical endpoints, such as joint limits or physical boundaries. These bounds represent the safe regions in which the pHRI should occur. Thus, the virtual position must always start within the bound for the method to be valid, otherwise the human should not even be interacting with the robot if it is outside safety limits. Using hard bounds alone can lead to abrupt velocity changes at their boundaries when an operator moves rapidly towards them, which may be uncomfortable. Therefore, our approach is designed to allow utilization of both types of bounds simultaneously, with the soft bound contained within the hard bound. This formulation allows for fast motions to be gradually slowed instead of abruptly stopped.

1) Soft Bounds: Let $p_k \in \mathbb{R}^n, v_k \in \mathbb{R}^n$ be the position and velocity of the virtual system at the current k^{th} timestep, respectively. Furthermore, let $B \subset \mathbb{R}^n$ be a bound. To emulate a restoring force based on how much B is violated, let $r \in \mathbb{R}^n$ be the vector from the current position to the closest bounded position:

$$r := p_k - \text{proj}_B(p_k), \tag{2}$$

which is well-defined and unique due to the convexity of B. Then, when r is nonzero, let $\hat{r} := \|r\|^{-1} r$, and the restoring force $f_r \in \mathbb{R}^n$ can emulate a virtual spring-damper system:

$$f_r = -k_s r - \max\{v_k \cdot \hat{r}, 0\} d_s \hat{r}, \tag{3}$$

where $k_s \in \mathbb{R}_{\geq 0}$ and $d_s \in \mathbb{R}_{\geq 0}$ are the restoring spring constant and damping coefficient, respectively. Note that all vector norms are the 2-norm, unless otherwise specified. The max function is included so that the restoring force only hinders movement that further violates the bound. The restoring force is then added to equation (1) before the dynamics are propagated.

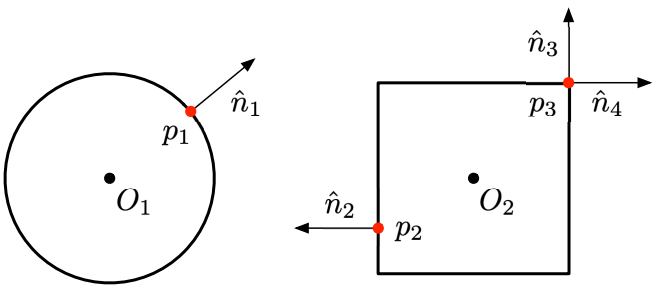


Fig. 1: Convex bounds, such as norms and rectangles, can be used as both soft and hard bounds. In the norm bound, there is only ever a single unit surface normal, so $S_{p_1} = \{\hat{n}_1\}$. However, the rectangular bound is not differentiable at the corners, so $S_{p_2} = \{\hat{n}_2\}$ and $S_{p_3} = \{\hat{n}_3, \hat{n}_4\}$.

2) Hard Bounds: Similar to soft bounds, a hard bound B is also a convex closed subset of the space of virtual positions. However, to ensure that p_k is always contained within B, first assume that the unbounded discretized dynamics of equation (1) would result in $p_{k+1} \notin B$. Then, the virtual position can be updated as $p'_{k+1} := \text{proj}_B(p_{k+1})$, which would ensure containment within B.

For velocities, the components that try to escape B should be removed. Let $s(\cdot) : \partial B \to \mathbb{R}^n$ be a map from points on the boundary of B to its corresponding unit normal vector. Since $s(\cdot)$ may not be defined everywhere on ∂B, e.g., at the corners if B is a rectangle, define the unit surface normal at such a point p as a set, S_p, of all possible limits of $s(\cdot)$ towards p. For bounds where $s(\cdot)$ exists everywhere on ∂B, $|S_p| = 1$. For other bounds, such as the rectangle example, there exist p such that $|S_p| \geq 1$, as shown in Fig. 1.

To ensure that future states of the virtual dynamics remain bounded, update the position and velocity as follows:

$$\tilde{p}_{k+1} = \text{proj}_B(p_{k+1}), \tag{4}$$

$$\tilde{v}_{k+1} = \arg\min_{v} \quad \|v - v_{k+1}\|^2 \tag{5}$$
$$\text{s.t.} \quad \hat{n}^\top v \leq 0, \ \forall \hat{n} \in S_{\tilde{p}_{k+1}}$$

Positions are guaranteed to always remain in B. The update in velocity ensures that components that try to escape the bound are removed, which is necessary for robot systems that only use the virtual velocity as reference.

On each integration step, soft bounds are evaluated before virtual dynamics are propagated with equation (1) in order to integrate the restoring force. Then, hard bounds are enforced with equations (4) and (5) to ensure they are always satisfied. The same hard bound methodology in equations (4) and (5) can be imposed on any higher-order derivatives of position. This allows for further constraints on virtual quantities such as velocity or acceleration.

C. Collisions

Collision avoidance is an integral part of safety in pHRI, especially for multi-arm systems. Any controllable object with which a collision can occur is henceforth denoted a controllable entity. Objects that cannot be controlled, such as the human, obstacles, or static links and frames of a robot, are called uncontrollable entities. Uncontrollable entities can be free or fixed. As mentioned in the literature review, collision avoidance consists of two aspects: detection and avoidance. Detection is concerned with finding entities that are too close, whereas avoidance then restricts the relative motion between the entities from becoming closer.

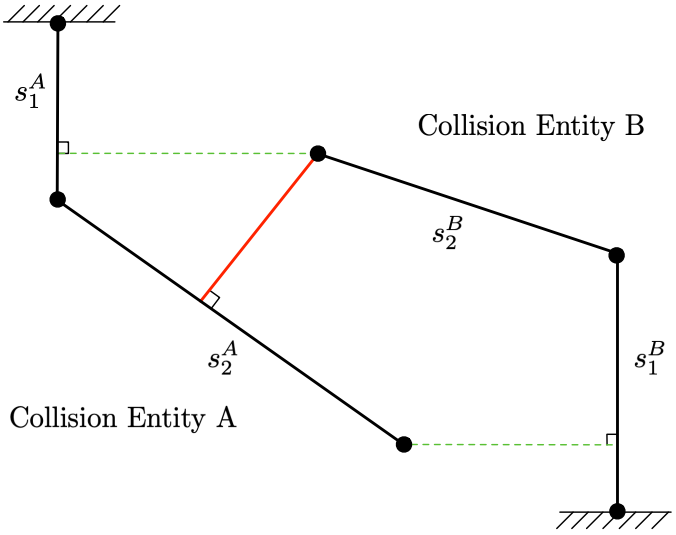


Fig. 2: A sample collision shown between two entities, each of which is represented by three vertices and two corresponding segments: $E_A = \{s_1^A, s_2^A\}$ and $E_B = \{s_1^B, s_2^B\}$. Green dotted lines indicate potential collisions that do not exceed the threshold ϵ, whereas the red line shows the collision. The intersections of the red line with s_2^A and s_2^B constitute the collided sets $\tilde{\mathcal{A}}_A$ and $\tilde{\mathcal{A}}_B$.

1) Detection: Let $V_j^i \in \mathbb{R}^3$, $j \in \{1, \ldots, m_i\}$, $m_i \geq 1$ be called entity vertices, which are a series of user-specified locations along entity i in the system, such that the segment drawn between adjacent points captures the geometry of the corresponding link of the entity, as shown in Fig. 2. m_i is the number of entity vertices on entity i. Let $s_j^i := \{\lambda V_j^i + (1-\lambda)V_{j+1}^i \mid \lambda \in [0,1] \subset \mathbb{R}\}$ be the segment between two adjacent collision points, and $\mathcal{S}$ be the space of segments. Then, let $E_i := \{s_j^i \mid j \in \{1, \ldots, m_i\}\}$ be the set of the aforementioned segments corresponding to entity i. If $m_i = 1$, then $E_i = \{\{V_1^i\}\}$ consists of a single zero-length segment.

Next, for each entity, enumerate the set $\mathcal{C}_i = \{(s_a, s_b) \mid s_a \in E_i, s_b \notin E_i\}$, which is the set of potential collision segments between entity i and every other entity that has collision points. Certain collisions, such as self-collisions between adjacent links or collisions between robots that are physically farther than the limits of their workspace, can be pruned from this set in order to speed up computation. Let the pruned set be $\tilde{\mathcal{C}}_i \subseteq \mathcal{C}_i$.

Let $d(\cdot) : \mathcal{S} \times \mathcal{S} \to \mathbb{R}^3 \times \mathbb{R}^3$ map segments s_a, s_b to points $c_a \in s_a, c_b \in s_b$ such that $\|c_a - c_b\|$ is minimized. To make $d(\cdot)$ well-defined, when s_a and s_b are parallel and proximate, c_a and c_b are chosen as the midpoints of the overlapping region. The definition for $d(\cdot)$ is discussed below. For entity i, let $\mathcal{A}_i := \{d(s_a, s_b) \mid (s_a, s_b) \in \tilde{\mathcal{C}}_i\}$ and compute $\tilde{\mathcal{A}}_i := \{(c_a, c_b) \mid (c_a, c_b) \in \mathcal{A}_i, \|c_a - c_b\| \leq \epsilon_i\}$, called the collided set, where $\epsilon_i \in \mathbb{R}_{\geq 0}$ is a user-specified threshold for detection. Note that $\tilde{\mathcal{A}}_i$ can be computed in parallel. The collided set tells not only whether a collision has occurred,

but also its location and direction.

The aforementioned segment-segment distance function $d(\cdot)$ needs to consider corner cases, such as: zero-length segments and parallel-segments. Let two segments $s_a, s_b \in \mathcal{S}$ have endpoints $a_0, a_1 \in \mathbb{R}^3$ and $b_0, b_1 \in \mathbb{R}^3$, respectively.

If $a_0 = a_1$ and $b_0 = b_1$, then $d(s_a, s_b) = (a_0, b_0)$.

If only one of the segments has zero-length, say $a_0 = a_1$ without loss of generality, then project a_0 onto s_b and determine its ratio $r_b \in \mathbb{R}$ along s_b:

$$r_b = \frac{(b_1 - b_0)^\top (a_0 - b_0)}{\|b_1 - b_0\|^2}. \tag{6}$$

Then, depending on the value of r_b, assign the point on s_b:

$$d(s_a, s_b) = \begin{cases} (a_0, b_0), & r_b < 0, \\ (a_0, b_0 + r_b(b_1 - b_0)), & 0 \le r_b \le 1, \\ (a_0, b_1), & 1 < r_b. \end{cases} \tag{7}$$

If both segments have nonzero length, then check if they are parallel. If so, i.e., $(a_1 - a_0) \times (b_1 - b_0) = 0$, then find the midpoints of the overlapping region: $a_m \in s_a$ and $b_m \in s_b$. Then, $d(s_a, s_b) = (a_m, b_m)$. Otherwise, compute the mutual-perpendicular segment s_p with endpoints $p_a \in s_a$ and $p_b \in s_b$. This can be done by finding the ratios of p_a along s_a and p_b along s_b. Now, define the normalized cross-product and compute the ratio r_a along segment s_a as:

$$\hat{c} := \frac{a_1 - a_0}{\|a_1 - a_0\|} \times \frac{b_1 - b_0}{\|b_1 - b_0\|}, \tag{8}$$

$$r_a = \frac{1}{\|a_1 - a_0\| \|\hat{c}\|^2} \det \left(\begin{bmatrix} (b_0 - a_0)^\top \\ \frac{(b_1 - b_0)^\top}{\|b_1 - b_0\|} \\ \hat{c}^\top \end{bmatrix} \right), \tag{9}$$

and similarly for r_b. Then, $d(s_a, s_b) = (a_0 + r_a(a_1 - a_0), b_0 + r_b(b_1 - b_0))$. Note that this computation also handles the case when s_a and s_b intersect.

2) Avoidance: Collision avoidance can only be implemented on controllable entities, which are typically robot manipulators. It is implemented in joint-space in order to also consider internal motions of redundant manipulators, which is particularly relevant for exoskeletons. The objective is to restrict motions of the manipulator such that for each $(c_a, c_b) \in \tilde{\mathcal{A}}_i$, the point c_a is prohibited from moving in the direction $c_b - c_a$. Since c_a can be anywhere on the manipulator, the kinematics of the robot are required. However, to be robot-agnostic, our approach requires the translational components of the Jacobian functions of the robot for which it is implemented: $J_j^i(\cdot) : \mathbb{R}^3 \times \mathbb{R}^n \to \mathbb{R}^{3 \times n}$, where the first argument is the location of the point, n is the *degrees-of-freedom* (DoFs) of the robot, and $J_j^i(\cdot)$ is henceforth referred to as the partial Jacobian. The output of $J_j^i(\cdot)$ is the upper three rows (linear motion) of the spatial manipulator Jacobian for an arbitrary point c_a located on segment s_j^i. A partial Jacobian can be constructed online using [33, equation (14)].

Given a desired joint-space velocity $\dot{\theta}_d \in \mathbb{R}^n$, either from the virtual dynamics if they are configured as being in joint-space, or from the robot's inverse kinematics if the virtual dynamics are in task-space, the objective is to restrict the reference joint-space velocity such that the velocity of each collided point c_a does not have positive dot product with $c_b - c_a$. More succinctly, the restricted desired joint-space velocity $\dot{\theta}_r \in \mathbb{R}^n$ can be found by solving the following quadratic program:

$$\begin{aligned} \min_{\dot{\theta}_r} \quad & \|\dot{\theta}_r - \dot{\theta}_d\|^2 \\ \text{s.t.} \quad & (c_b - c_a)^\top J_j^i(c_a, \theta) \dot{\theta}_r \le 0 \\ & \forall (c_a, c_b) \in \tilde{\mathcal{A}}_i \end{aligned} \tag{10}$$

where θ is the physical joint configuration, and each J_j^i is picked such that c_a corresponds to segment s_j^i. Note that each collided point imposes one additional linear constraint on $\dot{\theta}_r$. The optimal $\dot{\theta}_r$ is then sent to the manipulator's controller and also used to update the velocity of the virtual model.

The proposed methodology of constraining potential collision points from moving closer to each other is not inherently restricted to line segments, and can be extended to other geometries in which a similar function to $d(\cdot)$ - which computes the closest points between two objects - can be constructed. For instance, planes can also be used to represent static obstacles such as the floor or walls.

D. Summary

Algorithm 1 outlines the approach's main safety sequence, which runs within the system's control loop.

Algorithm 1 Safety-Focused Admittance Control

for each controllable entity, E_i
  Receive human-applied force measurements f_h
  Compute restoring Soft Bound force f_r (equation 3)
  Sum dynamics input force $f = f_h + f_r$
  Propagate virtual dynamics (equation 1)
  Restrict state via Hard Bounds (equations 4, 5)
  Update entity vertices using physical position
  Compute collided set $\tilde{\mathcal{A}}_i$
  Restrict virtual velocity (equation 10)
  Output virtual state to E_i's controller

III. Experiment Setup

To demonstrate the correctness and versatility of the approach, a series of experiments on two different multi-arm robotic systems: (1) the V-Rex and (2) the EXO-UL8, are conducted. The V-Rex is a non-anthropomorphic exoskeleton consisting of multiple task-space-controlled off-the-shelf industrial serial manipulators, whereas the EXO-UL8 is a custom-built anthropomorphic bimanual upper-limb exoskeleton controlled in joint-space. The existence of the two systems on opposite ends of the off-the-shelf (industrial) vs. custom, task-space vs. joint-space, and non-anthropomorphic/anthropomorphic spectra demonstrates the versatility and generalizability of the approach. The experiments verify and validate both soft and hard bounds, as well as multi-arm collision avoidance, on each of the two systems. Each experiment also implicitly verifies and validates the virtual dynamics, since it is necessary for any motion.

A. Systems

1) V-Rex: The Virtual Reality Exoskeleton (V-Rex) is a full body haptic device designed to provide force feedback to a human interacting with a virtual environment visualized with a *virtual reality* (VR) headset. The system is composed of five task-space-controlled off-the-shelf Kawasaki industrial serial manipulator robots. Two RS-007L robots are gripped by the operator's hands; two BX-100S robots are connected with breakaways to the operator's shoes; and one CX-210L provides gravity offloading through a flying harness. All robot arms have load cells between their end effectors and the human interface (three ATI Omega sensors for the lower limbs and body support plus two ATI Gamma sensors for the upper limbs). Each manipulator takes human-applied wrenches and propagates them into task-space motions of the end effector using the model shown in Fig. 3a.

2) EXO-UL8: The EXO-UL8 is a custom bimanual redundant powered joint-space-controlled upper-limb anthropomorphic exoskeleton with seven DoFs in each arm, built as the latest system in a series of exoskeletons designed for pHRI and robot-assisted rehabilitation [2], [13], [37]–[43]. The *human-in-the-loop* (HITL) requirement has motivated specific hardware designs, such as rotating ring joints to allow shoulder rotation and forearm supination/pronation [2], and anatomically similar joint limits to align the exoskeleton's and operator's arms. pHRI is enabled by joint-space admittance control. Wrenches measured from three ATI Mini40 sensors on each arm are fused and propagate virtual dynamics [37], whose trajectories are tracked by a computed-torque controller [44], [45], as shown in Fig. 3b.

B. Experiments

1) Bounds: Soft and hard bounds are first demonstrated separately on the V-Rex. A single arm of the system is restricted to move in a 2D plane parallel with the ground, as shown in Fig. 4. For both bound types, the operator tries to move the end effector to follow the desired trajectory at a constant speed. The operator starts in the region's center, moves to the bottom-left side, traverses the desired trajectory in a counter-clockwise direction, and then returns to the center. In the soft bounds trial, a square bound of equal size (side length: 37 cm) to the trajectory rotated by 45° is imposed. The bound has parameters: $k_s = 250$ N/m, $d_s = 60$ Ns/m. The hard bound trial utilizes the same bound shape. The manipulator implements the same second-order virtual dynamics parameters $m = 10$ kg, $d = 15$ Ns/m for both trials.

Since the EXO-UL8 implements decoupled 1D virtual dynamics for each joint, hard and soft bounds are simultaneously shown on the elbow (joint 4) and shoulder rotation (joint 3) DoFs. The hard bound restricts the elbow motion to the range of $[0, 70^\circ]$, whereas the soft bound on the shoulder joint becomes active once the motion exceeds $[15^\circ, 60^\circ]$. The parameters of the soft bound are $k_s = 10$ N/m, $d_s = 0$ Ns/m. A movement trajectory, as shown in Fig. 5, is designed to exercise both bound types in the joints.

2) Collision Avoidance: While our collision avoidance method applies identically to avoidance of both dynamic free entities and static fixed entities, we demonstrate only with a static entity on each of the two systems for clarity of results. With this methodology, the only difference is that the entity locations are updated on each iteration. For the V-Rex, the fixed entity is physically represented by the right arm oriented with the elbow-to-wrist link nearly vertical. This segment is located at $(0.3\text{m}, 0.014\text{m})$ as shown in images (1) and (2) of Fig. 10. The operator initially aims to move the end effector of the left arm to pass in front of the static right arm. Then, the operator aims to move as far left from the static right arm as they can before returning to the starting location. For ease of data representation, motions of the end effector and elbow-to-wrist link of the left arm are kept in a plane parallel with the ground, similar to the bounds experiment. The collision detection threshold is 0.27m.

For the EXO-UL8, the axes of rotation for shoulder flexion and elbow flexion are first aligned. The other five joints are locked so that the motion of the arm is restricted to the plane perpendicular to these two unrestricted axes of rotation. The quadratic program of equation (10) includes these five locked joints as additional constraints: $0 \leq \dot{\theta}_{r,i} \leq 0, \ i \in \{1, 3, 5, 6, 7\}$. A virtual collision segment is then placed in front of the arm at $(0.4 \text{ m}, -0.43 \text{ m})$ and oriented perpendicular to this plane, as shown in Fig. 6. The operator is instructed to move the arm above the collision segment and then return to the original location. The collision detection threshold is 0.1m.

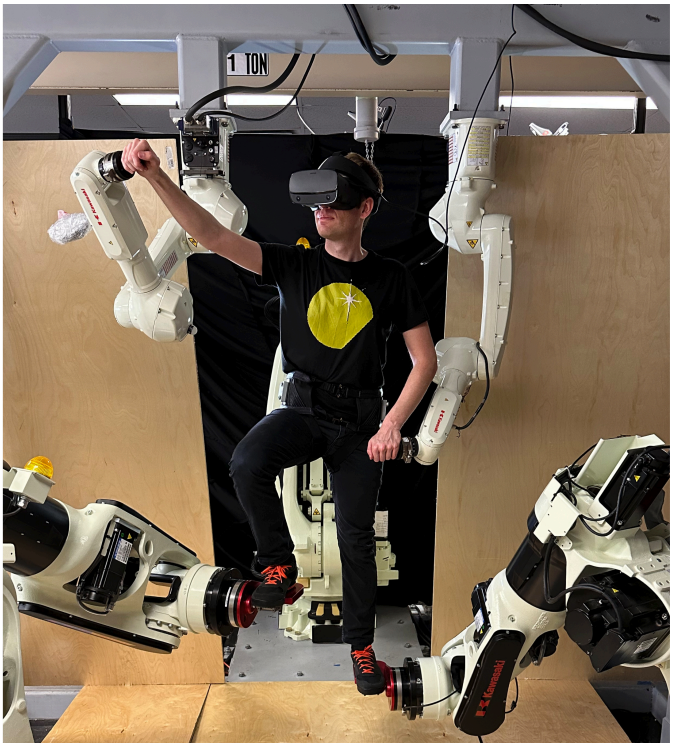
The collision avoidance algorithm requires the free entity positions to be updated on each iteration. For the V-Rex and EXO-UL8, free entities are the robotic manipulators. In each system, the relative locations of the manipulator bases are known, so the absolute locations of any point along the manipulators can be determined through forward kinematics. Thus, no calibration phase is necessary. For fixed entities, such as structural elements of the frame, their static locations are known (from design), so they can be added to the collision algorithm at initialization and do not need to be updated.

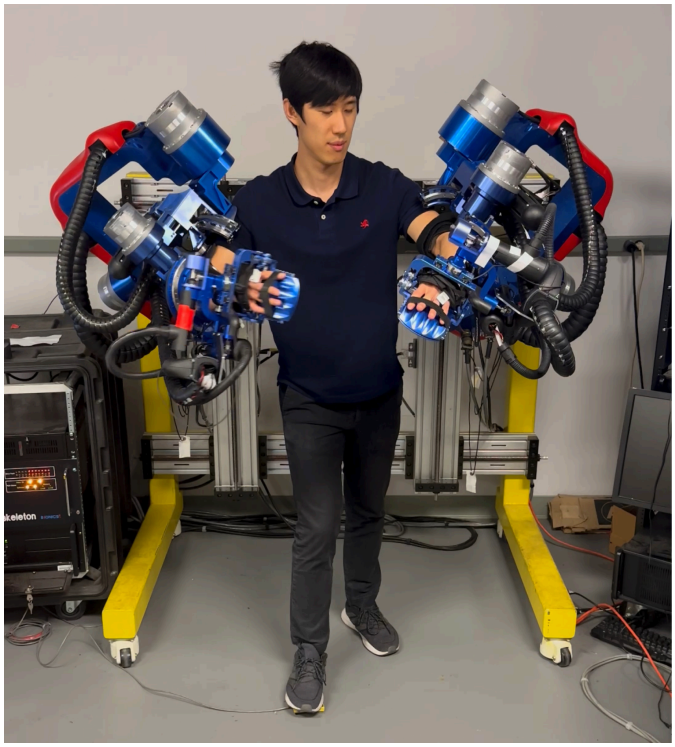
IV. Results

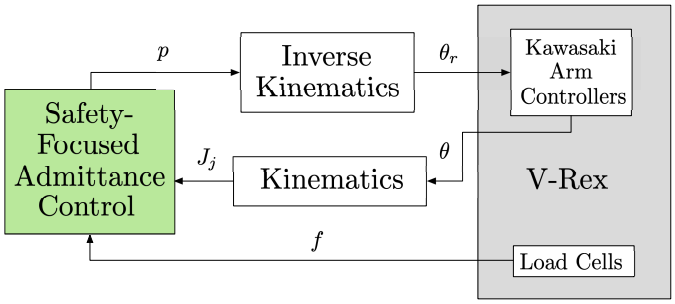
A. Bounds

1) V-Rex: Fig. 7 illustrate the operator's trajectory in the presence of a soft bound. When inside the bound, no restoring forces are present, indicating unrestricted motion. However, upon breaking a bound, restoring forces are generated with a direction to move the operator back into bounds. Notably, all restoring forces are normal to the broken bound surface, and their magnitude increases with distance past the bound, consistent with the virtual spring-damper. The trajectory loop took approximately 18 seconds to complete, resulting in an average speed of 8.2cm/s.

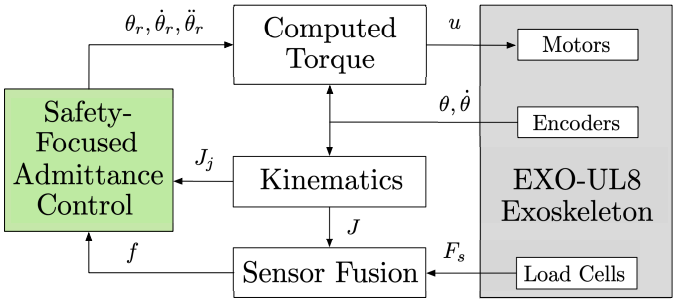
Fig. 8 presents the operator's trajectory in the presence of a hard bound. The trajectory took approximately 12 seconds to complete, resulting in an average speed of 10.2cm/s. The results demonstrate the effectiveness of a hard bound in limiting motion to the bounded region in task space, without destabilizing the system. Despite exerting forces in a direction to break outside the bounds, the virtual position is constrained







(a) The V-Rex is a full body haptic device consisting of five robotic arms, designed for pHRI with virtual environments. Force-torque input from load cells is used to step virtual dynamics and determine the next position reference, p in task space. Physical joint state, θ, is read and utilized to find partial Jacobians, J_j, to adjust reference position for collision avoidance. Utilizing an inverse kinematic solver, all viable joint space configurations are found. A l^2-norm is then applied to find the closest solution to the current joint state, θ. The selected joint state is vetted through a secondary safety check before the joint references θ_r are updated for the Kawasaki arm controller for precise position control. The entire control cycle runs at 500Hz.



(b) The EXO-UL8 is a bimanual upper-limb exoskeleton designed for pHRI and robot-assisted rehabilitation. Human-applied forces are measured by three load cells on each arm located at the: upper-arm (u), lower-arm (l), and wrist (w). The measured wrenches, F_s, $s \in \{u, l, w\}$, are transformed into joint torques through the Jacobian, J, and then fused together into a single joint-space human-applied torque vector, f, with the algorithm in [37]. This torque propagates the virtual dynamics in the admittance control, which also uses the partial Jacobians, J_j for collision avoidance. The virtual states are then tracked by the exoskeleton's computed torque controller, which outputs motor torques, u. The control rate is 1KHz.

Fig. 3: The V-Rex and EXO-UL8 systems exist on opposite ends of the task/joint space control, non-redundant/redundant, off-the-shelf (industrial)/custom, and non-anthropomorphic/anthropomorphic spectra.

and instead slides along the virtual bounding surface in compliance with the applied force. Conversely, when inside the bound, motion remains unrestricted as expected.

2) EXO-UL8: The elbow flexion and shoulder rotation reference trajectories are plotted in Fig. 9, with the four waypoints labeled. The trajectory was completed in approximately 20s, resulting in an average speed of $11°/s$. A soft bound, which is only present for the shoulder, produces restoring forces generated by a virtual spring whenever the trajectory exceeds the boundary. Note that the soft bound is only configured with a virtual spring, so the restoring force depends only on position. Hard-bounds were implemented in both DoFs, but only the elbow reached the limits due to the absence of an additional soft-bound. The applied forces at the hard-bounds show that the dynamics were only allowed to propagate tangentially to the bound surface, as designed. This experiment shows a typical application in which a soft bound is placed inside a hard bound. Motions towards bound limits are gently slowed while ensuring that the absolute limits are respected (e.g., at point 3 of Fig. 9).

B. Collision Avoidance

1) V-Rex: The end effector's trajectory and the elbow-to-wrist link's pose during key collision avoidance points are shown in Fig. 10. At the first instance (1) of collision, the end effector trajectory is restricted such that the link will not continue into the collision detection region, despite the operator's force in that direction. Similarly, at the second

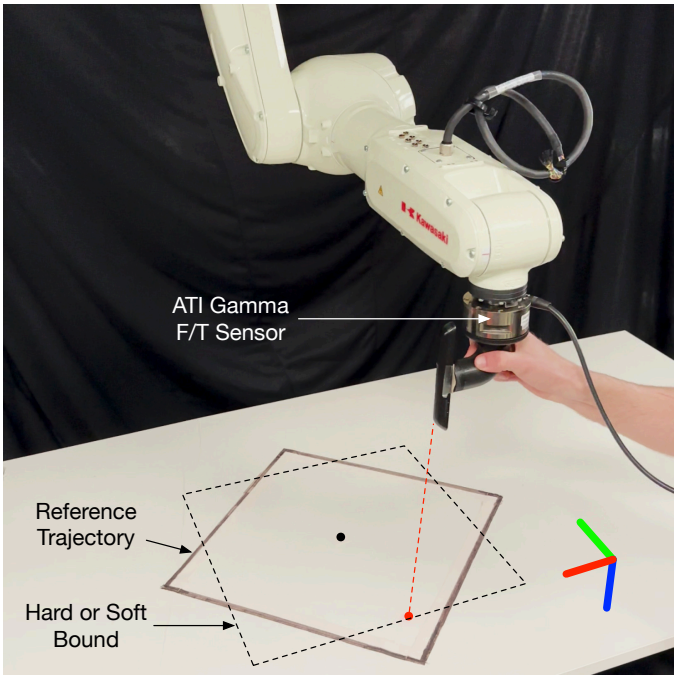


Fig. 4: For bound experiments on the V-Rex, the operator is guided by a black square (solid line) on the surface beneath one of the upper limb arms. A square boundary (shown by the dashed line), either soft or hard, is positioned at a $45°$ angle relative to the trajectory, sharing the same centroid. Starting from the centroid, the operator follows the trajectory to the best of their ability before returning to the starting point. A laser pointer attached to the end effector helps the operator track their progress during the experiment.

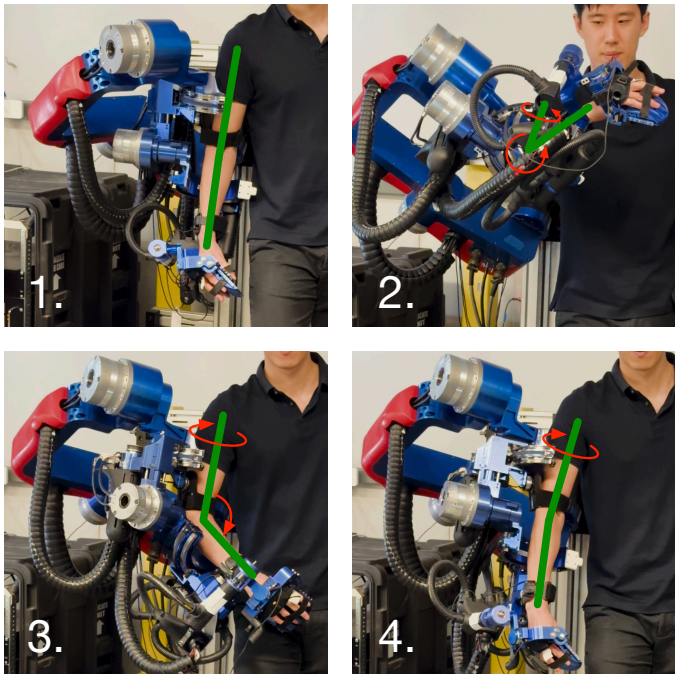


Fig. 5: The bounds experiment on the EXO-UL8 has the operator follow a target trajectory consisting of 4 way-points $(1-4)$. These way-points are strategically placed to engage shoulder rotation (joint 3) and elbow flexion (joint 4) near bound limits. Soft and hard bounds restrict the shoulder and elbow positions, respectively. The operator traverses through the way-points once, pausing at each for approximately 1s. The corresponding motions from the previous way-point are indicated by red arrows, while the configuration of the human arm at each way-point is shown by a green line.

instance (2), the end effector trajectory is restricted; however, the restriction is now in a direction away from the collision detection region, since continued motion in the -X direction would result in the left elbow colliding with the right arm. Unlike the EXO-UL8, the operator's arms are not aligned with the manipulator, which could result in potential collisions between the two. However, the operator can be modeled as an uncontrollable free entity, so that the robotic manipulators would avoid collisions with it in a similar fashion.

2) EXO-UL8: The contour of the collision detection region of Fig. 6 is plotted in the shoulder and elbow flexion joint space in Fig. 11. The trajectory and human-applied forces are overlaid to show the motion being constrained from entering the collision region, even when the operator is pushing into it. Near $(63°, 9°)$, the trajectory doubles back slightly, but collision constraints are respected. Once the operator flexes their elbow, the region is avoided (from $(30°, 38°)$ upwards), and subsequent motion is unaffected. The experiment demonstrates collision avoidance's expected functionality.

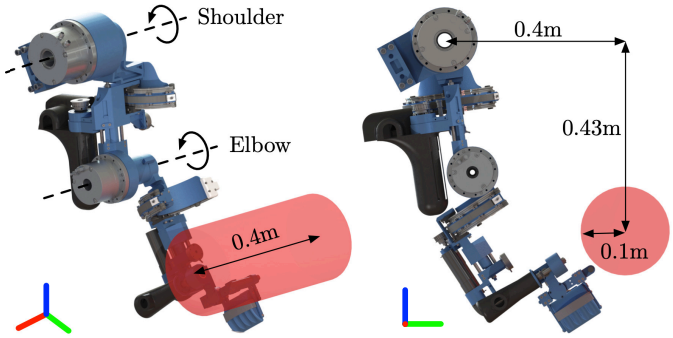


Fig. 6: For the EXO-UL8's collision experiment, the shoulder and elbow joints are aligned and allowed to move, while all other joints are locked. A virtual fixed collision entity is placed in front of the arm and oriented parallel to the joints' axes of rotation. The collision avoidance distance is set to 0.1m.

V. Discussion

A. Comparison to Existing Methods

In general, using runtime as a metric for comparison poses several issues. The implementation has a significant influence; for instance, using a compiled language can be faster than an interpreted one, and parallelism or vectorized operations can make the program even faster. As not all the studies in this comparison [26], [28], [29], [31], [32] provide code implementation, a comparison based on runtime alone may not be meaningful. Furthermore, methods such as [31]'s utilize a neural net to solve the optimization, which could be fast, but also requires more effort to train and setup. Whether this trade-off is justified depends on the particular use case. We have identified the most notable aspects of collision avoidance as: (1) modeling, which uses simple shapes, known as primitives, to represent the manipulators' geometry, (2) enumeration of primitives, which identifies potentially colliding primitives, (3) distance between primitives, which computes the shortest distance between primitives as well as the locations at which this occurs, (4) partial Jacobian, which relates the velocities of these locations to the manipulator's velocity, and (5) optimization solver, which is the method or tool used to complete the computation. A comparison is presented in Table I.

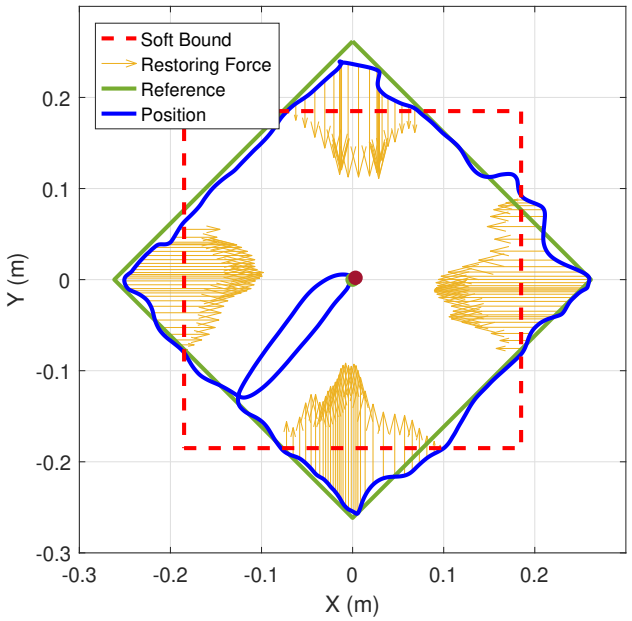


Fig. 7: On the V-Rex, the operator's trajectory (blue line) tries to track the reference (green). However, the presence of a soft bound (dotted red) results in a restoring force (yellow arrow) that amplifies with distance from the bound. As the operator navigates around the corners of the reference, the restoring force shifts from being fully antagonistic to providing some assistance in the direction of motion. This sudden change is reflected in the lower accuracy of the operator's trajectory when compared to the hard bound trials.

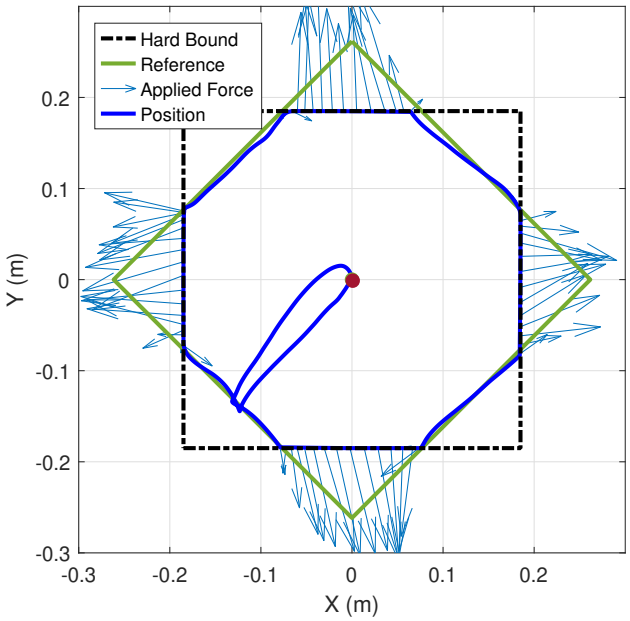


Fig. 8: On the V-Rex, the operator's trajectory (blue line) tracking the reference (green) is impeded by the hard bound (dotted black). At the bound, the position is restricted and slides along the surface, despite the user's exerted force to move outside (blue arrows). Note that for clarity the forces are only shown when the position is at the bound. In contrast, while inside the bound, the virtual dynamics enable the operator to track the trajectory with high fidelity.

Collision avoidance is a complex problem that requires several trade-offs to be made. For instance, [32] uses meshes to represent the manipulator, which can be highly accurate, but comes at the expense of requiring a dedicated library for computing contact. On the other hand, [28] and [29] represent the manipulator as a union of spheres. Although fast to compute pairwise distance, the method cannot find exact collision points, and requires a partial Jacobian to be precomputed for each sphere at setup, detracting from ease of use. The neural net of [31] makes an even bigger trade-off between speed, ease of use, and potentially even correctness.

Our strategy prioritizes generality and ease of use in collision avoidance by using simple geometries and optimizing where possible (e.g., primitive enumeration). In addition to providing detailed specifications, such as OSQP for speed and software compatibility (refer to [46] for its comparison results), our entire approach is implemented as a free and open-source library. This comparison aims to underscore the distinctions in our approach, allowing implementers to make an informed decision for their specific applications.

B. Potential Limitations

When a robot manipulator moves at high speeds, it becomes impractical to immediately stop at a boundary, even if the boundary is a hard bound. The proposed strategy focuses on creating trajectories that prioritize safety, however the robot's ability to precisely track these trajectories is also determined by its controller's capabilities. Incorporating hardware or controller information back into the reference generation can have potential improvements, such as in *model predictive control* (MPC). However, doing so creates an additional feedback loop from state to reference, which may affect overall stability. To ensure generalizability of the approach, we chose not to allow for hardware-level information to influence the reference generation. Doing so would detract the approach's versatility and possibly stability, despite potentially achieving better results on specific systems.

A control rate of 1KHz is commonly recommended for reliable haptic interaction, but is not a strict rule and depends on specific factors such as desired virtual stiffness and the robot's mechanical capabilities. The haptic rendering presented only focused on soft bounds, which were not overly stiff. High-performance Kawasaki industrial manipulators of the V-Rex enabled convincing virtual force emulation at a lower control rate. Therefore, a control rate of 500Hz was chosen for the V-Rex as a suitable compromise between haptic rendering fidelity and CPU usage.

C. Choice of Parameters

The parameters of the V-Rex virtual dynamics simulate a 10 kg object in a damping medium of 15 Ns/m. The mass is not difficult to move in the absence of gravity, and the damping constant was selected as a compromise between transparency and dissipative stability. Soft bounds were configured with a spring constant of 250 N/m and damping of 60 Ns/m, in order to provide a noticeable resistance to the user.

We also chose rectangular and norm bounds for their intuitive nature. Rectangular bounds are easy to understand and configure, while norm bounds also prove useful for velocity by representing speed limits. However, the methodology

Aspect\Study	Bosscher [26]	Liu [28]	Lin [29]	Zhang [31]	Todorov [32]	**Ours**
Modeling (location accuracy)	Spherical shells =	Spheres −	Spheres −	Line segments =	Meshes +	Line segments =
Enumeration of Primitives (complexity)	Pairwise with pruning =	Pairwise −	Pairwise −	Pairwise −	Pairwise with pruning =	Pairwise with pruning =
Distance between Primitives (complexity)	Unspecified Unknown	Analytic =	Analytic =	Iterative −	Iterative −	Analytic =
Partial Jacobian (setup)	Unspecified Unknown	Analytic for each sphere −	Analytic for each sphere −	Unspecified Unknown	N/A	Analytic for any point [33] =
Optimization Solver	Simulink	Unspecified	Simulink	Neural network	Newton	OSQP [46]

TABLE I: The comparison examines the main aspects of collision avoidance using a suitable metric across five different recent publications. In each aspect, the relevant metric (shown in parenthesis) is assessed relative to our proposed approach and reported with one of the symbols: $\{+, -, =\}$, to indicate more performant, less performant, or comparable, respectively. The "unknown" keyword indicates that a comparison to our approach could not be made.

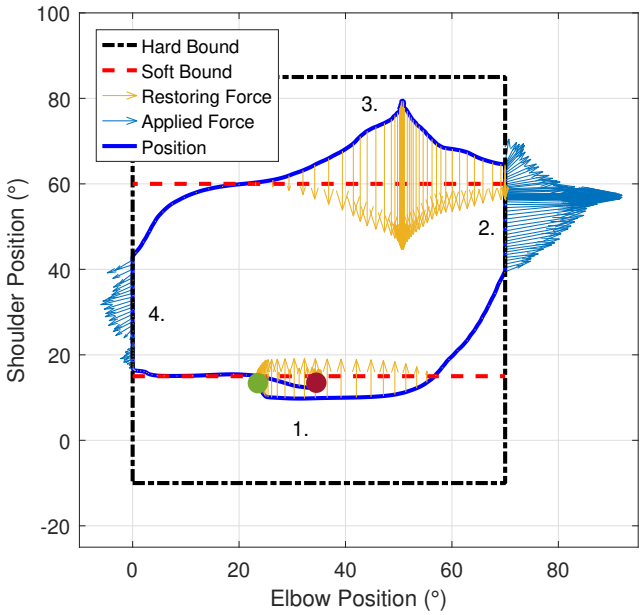


Fig. 9: On the EXO-UL8, the operator's trajectory (blue line) starts at point 1 (green dot) outside the shoulder's soft bound (dotted red line), resulting in restoring forces (yellow arrows). While moving towards point 2, a hard bound (dotted black line) blocks the elbow, despite human-applied forces (blue arrows) pointing out of the bound. The shoulder soft bound applies another restoring force towards point 3. Since the soft bound is configured as a virtual spring, the restoring force is only a function of displacement past the bound. At the intersection of the shoulder upper soft bound and the elbow upper hard bound (between points 2 and 3), the soft bound restoring force appears suddenly. However, this is not the case; as the trajectory slides up along the elbow hard bound, soft bound restoring forces increase continuously, and the largest force arrow visually occludes the smaller arrows underneath.

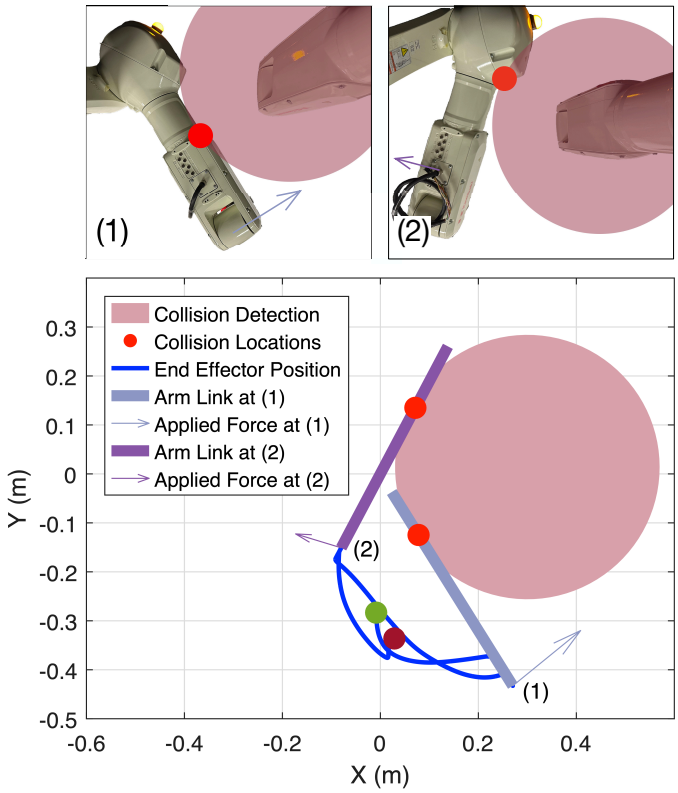


Fig. 10: On the V-Rex, the operator's trajectory (blue line) starts at the green dot and ends at the maroon dot. During the trajectory, two collisions are avoided with a detection radius around the right arm (pink region). A physical representation of the data at the two time instances ((1) and (2)) is shown in the subfigures above. At instance (1), motion is restricted in the direction of the right arm, even though the operator is applying force to continue in that direction. At instance (2), motion is restricted from continuing in the -X direction despite applying force in that direction, because the elbow would collide if the motion were unrestricted.

accommodates any convex bound. In our experiments, position bounds were centered at the robot's starting point for clarity and to allow movement in any direction, but can generally be placed anywhere around the starting point.

D. Safety at Hard Bounds

When the virtual position reaches a hard bound, human-applied forces perpendicular to the bound do not impact motion (see Fig. 8 and Fig. 9). However, these force components are still resisted by the robot, potentially leading to increased motor currents. Safety implications are examined in detail for the shoulder interior/exterior joint of the EXO-UL8, which has the smallest gear ratio. Fig. 12 shows the motor current when the operator pushes against the hard bound at 55°. The current remains within the motor's continuous operating range (RE50 series 578298 from Maxon Motor). Although safety can be addressed at the reference generation level, hardware remains crucial for overall pHRI safety. The proposed methodology,

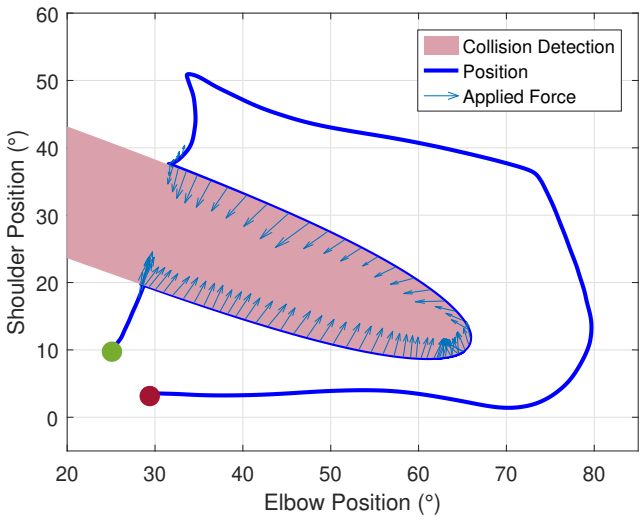


Fig. 11: The virtual obstacle's collision detection region is transformed through inverse kinematics of the EXO-UL8 into the joint-space of the shoulder and elbow flexion DoFs. Even though human-applied forces try to push the trajectory into the region, it respects the collision constraints. Human-applied forces are only plotted at the boundary for clarity.

being hardware agnostic, enhances safety at the reference generation level, without subjecting the robot's controller to any situation more dangerous than tracking a continuous position trajectory. However, controller-level safety must also be ensured by the implementer, otherwise any algorithm would be subject to the same potentially dangerous situations.

VI. Conclusion

This study presents a comprehensive safety-focused admittance control approach, consisting of soft virtual bounding regions, emulation of infinitely stiff bounds, and collision avoidance at any point along the manipulator. Experimental results validate the effectiveness of soft and hard bounds, which produce restoring forces, and rigidly confine trajectories, respectively. Results also validate the multi-arm collision avoidance methodology on both the V-Rex and EXO-UL8 systems by successfully restricting trajectories to prevent any part of the manipulators from colliding, despite the systems' differences in control and redundancy.

Future directions plan to incorporate the human operator as a free entity in the collision avoidance algorithm. This would further improve safety by also ensuring that the robot manipulators cannot collide with the operator. The time-varying location of the operator's body could be determined using a vision-based system, which would likely require calibration to account for varying lighting conditions.

For utilization of our approach in existing and future systems within the pHRI community, an implementation as a free and open-source templated C++ library is available at: https://github.com/jianwei-sun/gtfo. The library requires a C++17 compiler and depends on Eigen [47] and OSQP [46]. Future work completed on this framework will be made available within the open-source library.

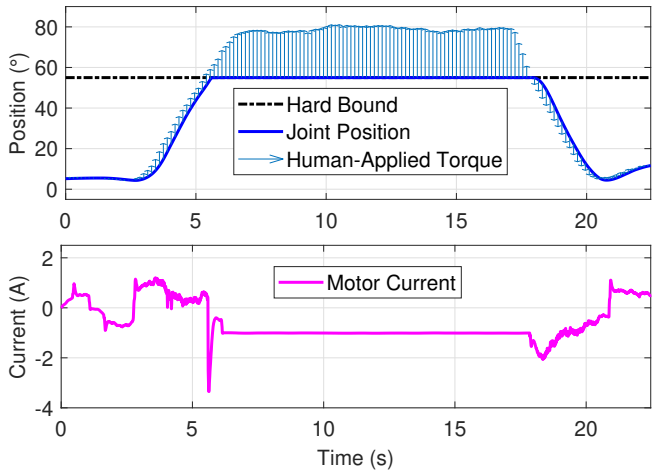


Fig. 12: A hard bound at 55° on the shoulder interior/exterior rotation joint of the EXO-UL8 resists the operator's attempts to push past it. The commanded motor current (-1.01A) remains in its maximum continuous operation range of ±4A. Prior to the position reaching the bound, the motor current varies due to the computed torque controller generating torques that compensate for reaction torques from the other joints' motions. However, at the hard bound, the current appears flat as all joints are stationary. Friction at the joint and low back-drivability limit the variation in the human-applied torque from being easily seen in the current profile. In general, pHRI hardware should ensure that all currents always remain in safe operating ranges, despite adversarial attempts by the operator to cause unsafe situations.

References

[1] J. A. Martinez, P. Ng, S. Lu, M. S. Campagna, and O. Celik, "Design of wrist gimbal: A forearm and wrist exoskeleton for stroke rehabilitation," in *2013 IEEE 13th Int. Conf. on Rehabilitation Robotics (ICORR)*, pp. 1–6, 2013.

[2] J. C. Perry, J. Rosen, and S. Burns, "Upper-limb powered exoskeleton design," *IEEE/ASME Transactions on Mechatronics*, vol. 12, no. 4, pp. 408–417, 2007.

[3] Y. Long, Z. Du, C. feng Chen, W. Wang, L. He, X.-W. Mao, G. Xu, G. Zhao, and W. Dong, "Hybrid control scheme of a hydraulically actuated lower extremity exoskeleton for load-carrying," *Journal of Intelligent & Robotic Systems*, vol. 91, pp. 493–500, 2018.

[4] P. Beyl, K. Knaepen, S. Duerinck, M. Damme, B. Vanderborght, R. Meeusen, and D. Lefeber, "Safe and compliant guidance by a powered knee exoskeleton for robot-assisted rehabilitation of gait," *Advanced Robotics*, vol. 25, pp. 513–535, 2011.

[5] H. Liao, H. H.-T. Chan, F. Gao, X. Zhao, G. Liu, and W.-H. Liao, "Proxy-based torque control of motor-driven exoskeletons for safe and compliant human-exoskeleton interaction," *Mechatronics*, vol. 88, 2022.

[6] M. J. Kim, W. Lee, C. Ott, and W. K. Chung, "A passivity-based admittance control design using feedback interconnections," in *2016 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pp. 801–807, 2016.

[7] D. Liu, J. Xu, C. Chen, X. Long, D. Tao, and X. Wu, "Vision-assisted autonomous lower-limb exoskeleton robot," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 6, pp. 3759–3770, 2021.

[8] Y. Cho, K. Kim, S. Ma, and J. Ueda, "A robotic wearable exoskeleton for construction worker's safety and health," in *Construction Research Congress 2018*, pp. 19–28, 2018.

[9] F. Dimeas and N. Aspragathos, "Online stability in human-robot cooperation with admittance control," *IEEE Transactions on Haptics*, vol. 9, no. 2, pp. 267–278, 2016.

[10] H. Kim and W. Yang, "Variable admittance control based on human–robot collaboration observer using frequency analysis for sensitive and safe interaction," *Sensors*, vol. 21, no. 5, 2021.

[11] A. Vick, D. Surdilovic, and J. Krüger, "Safe physical human-robot interaction with industrial dual-arm robots," in *9th International Workshop on Robot Motion and Control*, pp. 264–269, 2013.

[12] H. Rifaï, S. Mohammed, W. Hassani, and Y. Amirat, "Nested saturation based control of an actuated knee joint orthosis," *Mechatronics*, vol. 23, no. 8, pp. 1141–1149, 2013.

[13] J. Sun, P. W. Ferguson, and J. Rosen, "Suppressing delay-induced oscillations in physical human-robot interaction with an upper-limb exoskeleton using rate-limiting," in *2022 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pp. 6695–6701, 2022.
[14] Z. Li, B. Wang, F. Sun, C. Yang, Q. Xie, and W. Zhang, "semg-based joint force control for an upper-limb power-assist exoskeleton robot," *IEEE Journal of Biomedical and Health Informatics*, vol. 18, no. 3, pp. 1043–1050, 2014.
[15] T. D. Lalitharatne, K. Teramoto, Y. Hayashi, and K. Kiguchi, "Evaluation of perception-assist with an upper-limb power-assist exoskeleton using emg and eeg signals," in *11th IEEE Int. Conf. on Networking, Sensing and Control*, pp. 524–529, 2014.
[16] F. Ferraguti, C. Talignani Landi, L. Sabattini, M. Bonfé, C. Fantuzzi, and C. Secchi, "A variable admittance control strategy for stable physical human–robot interaction," *The International Journal of Robotics Research*, vol. 38, 2019.
[17] T. Zhang, M. Tran, and H. Huang, "Admittance shaping-based assistive control of sea-driven robotic hip exoskeleton," *IEEE/ASME Transactions on Mechatronics*, vol. 24, no. 4, pp. 1508–1519, 2019.
[18] A. Sharkawy, P. Koustoumpardis, and N. Aspragathos, "A neural network-based approach for variable admittance control in human–robot cooperation: online adjustment of the virtual inertia," *Intelligent Service Robotics*, 2020.
[19] M. Li, M. Ishii, and R. H. Taylor, "Spatial motion constraints using virtual fixtures generated by anatomy," *IEEE Transactions on Robotics*, vol. 23, no. 1, pp. 4–19, 2007.
[20] D. Bazzi, F. Roveda, A. M. Zanchettin, and P. Rocco, "A unified approach for virtual fixtures and goal-driven variable admittance control in manual guidance applications," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 6378–6385, 2021.
[21] L. Žlajpah and T. Petrič, "Unified virtual guides framework for path tracking tasks," *Robotica*, vol. 38, no. 10, p. 1807–1823, 2020.
[22] W. He, C. Xue, X. Yu, Z. Li, and C. Yang, "Admittance-based controller design for physical human–robot interaction in the constrained task space," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 4, pp. 1937–1949, 2020.
[23] R. Moccia, C. Iacono, B. Siciliano, and F. Ficuciello, "Vision-based dynamic virtual fixtures for tools collision avoidance in robotic surgery," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1650–1655, 2020.
[24] A. De Luca and F. Flacco, "Integrated control for phri: Collision avoidance, detection, reaction and collaboration," in *2012 4th IEEE RAS & EMBS Int. Conf. on Biomedical Robotics and Biomechatronics (BioRob)*, pp. 288–295, 2012.
[25] Z. Li, H. Wei, G. Liu, C. Liu, and D. Chen, "A variable admittance control strategy for self-collision avoidance based on virtual constraints," in *2022 IEEE Int. Conf. on Robotics and Biomimetics (ROBIO)*, pp. 958–963, 2022.
[26] P. Bosscher and D. Hedman, "Real-time collision avoidance algorithm for robotic manipulators," *Industrial Robot*, vol. 38, no. 2, pp. 186–197, 2011.
[27] A. Dietrich, T. Wimböck, H. Täubig, A. Albu-Schäffer, and G. Hirzinger, "Extensions to reactive self-collision avoidance for torque and position controlled humanoids," in *2011 IEEE Int. Conf. on Robotics and Automation*, pp. 3455–3462, 2011.
[28] H. Liu, D. Qu, F. Xu, Z. Du, J. Song, and M. Liu, "Real-time and efficient collision avoidance planning approach for safe human-robot interaction," *Journal of Intelligent & Robotic Systems*, vol. 105, 2022.
[29] H. Lin, Y. Fan, T. Tang, and M. Tomizuka, "Human guidance programming on a 6-dof robot with collision avoidance," in *2016 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pp. 2676–2681, 2016.
[30] H. Kaneko, T. Arai, K. Inoue, and Y. Mae, "Real-time obstacle avoidance for robot arm using collision jacobian," in *1999 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, vol. 2, pp. 617–622 vol.2, 1999.
[31] Z. Zhang, L. Zheng, Z. Chen, L. Kong, and H. R. Karimi, "Mutual-collision-avoidance scheme synthesized by neural networks for dual redundant robot manipulators executing cooperative tasks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 3, pp. 1052–1066, 2021.
[32] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 5026–5033, 2012.
[33] P. Chembrammel and T. Kesavadas, "A new implementation for online calculation of manipulator jacobian," *PLOS ONE*, vol. 14, 2019.
[34] J. J. Abbott, P. Marayong, and A. M. Okamura, "Haptic virtual fixtures for robot-assisted manipulation," in *Robotics Research* (S. Thrun, R. Brooks, and H. Durrant-Whyte, eds.), pp. 49–64, Springer Berlin Heidelberg, 2007.
[35] D. Zhang, G. Yang, and R. P. Khurshid, "Haptic teleoperation of uavs through control barrier functions," *IEEE Transactions on Haptics*, vol. 13, no. 1, pp. 109–115, 2020.
[36] Y. He, Y. Hu, P. Zhang, B. Zhao, X. Qi, and J. Zhang, "Human–robot cooperative control based on virtual fixture in robot-assisted endoscopic sinus surgery," *Applied Sciences*, vol. 9, no. 8, 2019.
[37] J. Sun, Y. Shen, and J. Rosen, "Sensor reduction, estimation, and control of an upper-limb exoskeleton," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1012–1019, 2021.
[38] Y. Shen, J. Sun, J. Ma, and J. Rosen, "Admittance control scheme comparison of exo-ul8: A dual-arm exoskeleton robotic system," in *2019 IEEE 16th Int. Conf. on Rehabilitation Robotics (ICORR)*, pp. 611–617, 2019.
[39] Y. Shen, J. Ma, B. Dobkin, and J. Rosen, "Asymmetric dual arm approach for post stroke recovery of motor functions utilizing the exo-ul8 exoskeleton system: A pilot study," in *2018 40th Annual Int. Conf. of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 1701–1707, 2018.
[40] Y. Shen and J. Rosen, "Chapter 5 - exo-ul upper limb robotic exoskeleton system series: From 1 dof single-arm to (7+1) dofs dual-arm," in *Wearable Robotics* (J. Rosen and P. W. Ferguson, eds.), pp. 91–103, Academic Press, 2020.
[41] M. Simkins, N. Byl, H. Kim, G. Abrams, and J. Rosen, "Upper limb bilateral symmetric training with robotic assistance and clinical outcomes for stroke," *International Journal of Intelligent Computing and Cybernetics*, vol. 9, pp. 83–104, Jan 2016.
[42] H. Kim and J. Rosen, "Predicting redundancy of a 7 dof upper limb exoskeleton toward improved transparency between human and robot," *Journal of Intelligent & Robotic Systems*, vol. 80, pp. 99–119, Dec 2015.
[43] W. Yu and J. Rosen, "Neural pid control of robot manipulators with application to an upper limb exoskeleton," *IEEE Transactions on Cybernetics*, vol. 43, no. 2, pp. 673–684, 2013.
[44] J. Craig, *Introduction to Robotics: Mechanics and Control*. Addison-Wesley series in electrical and computer engineering: control engineering, Pearson/Prentice Hall, 2005.
[45] R. M. Murray, Z. Li, and S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.
[46] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "Osqp: An operator splitting solver for quadratic programs," pp. 339–339, 2018.
[47] G. Guennebaud, B. Jacob, *et al.*, "Eigen v3." http://eigen.tuxfamily.org, 2010.

Jianwei Sun received a B.Sc. in Engineering Science (1T6 + PEY) from the University of Toronto in 2017, a M.Sc. in Electrical Engineering and Information Technology from the Swiss Federal Institute of Technology (ETH Zürich) in 2019, and a Ph.D. in Mechanical Engineering (Bionics Lab) from the University of California, Los Angeles (UCLA) in 2024. His research focuses on control, estimation, and safety for physical human-robot interaction with exoskeletons and haptic robots.

Erik Harrison Kramer received a B.A. in Physics and B.S. in Mechanical Engineering from the University of California, Berkeley in 2015 and a M.S. in Mechanical Engineering from the University of California, Los Angeles (UCLA) in 2018. He is a Ph.D. candidate at the Bionics Lab in the Department of Mechanical and Aerospace Engineering at UCLA and works in collaboration with NASA Jet Propulsion Laboratory on robotic systems for physical emulation of virtual dynamic environments.

Jacob Rosen received a B.Sc. degree in mechanical engineering, and M.Sc. and Ph.D. degrees in biomedical engineering from Tel-Aviv University, Tel-Aviv, Israel, in 1987, 1993, and 1997, respectively. He is a professor in the Department of Mechanical and Aerospace Engineering and the director of the Bionics Lab at the University of California, Los Angeles (UCLA). His research interests focus on medical robotics, biorobotics, human-centered robotics, surgical robotics, wearable robotics, rehabilitation robotics, and neural control.