Safety Filter for Underactuated Robotic Arms with Velocity Inputs

Hyunseo Jung, *Student Member, IEEE* and Keehoon Kim, *Senior Member, IEEE*

***Abstract*— This paper presents a task space safety filter for velocity-driven underactuated robotic manipulators. Existing control barrier function-based safety filters are mainly developed for fully actuated or kinematically redundant robots, for which any local task space velocity is realizable away from singularities. For underactuated manipulators, however, the feasible task space velocities are restricted to the Jacobian range even at regular configurations, so a command that satisfies a CBF constraint at the optimization stage can lose its safety guarantee after pseudoinverse execution. To address this issue, we explicitly enforce motion feasibility in the safety filter by constraining the filtered velocity to the Jacobian range through an equivalent left-nullspace condition. The resulting formulation unifies obstacle avoidance, general inequality and equality motion constraints, and optional joint velocity limits in a single convex quadratic program. We further discuss feasibility of the QP, and effect of CBF parameters on the QP performance. Experiments on underactuated robotic arm scenarios validate the proposed approach.**

I. Introduction

Velocity-level commands are ubiquitous in reactive manipulation because they can be generated directly in task space by operational-space control, artificial-potential-field methods, and dynamical systems [1]–[4]. In these settings, a high-level module outputs a desired task space velocity, and the robot must realize that command through the Jacobian map from joint velocity to task velocity. This realization step is delicate near singular configurations, where the Moore-Penrose pseudoinverse can amplify joint motion and degrade numerical robustness. Classical remedies such as singularity-robust inverse kinematics and task-priority redundancy resolution regularize this mapping and remain foundational tools for kinematic robot control [5], [6]. By construction, however, they address realizability and conditioning rather than safety-critical state and input constraints.

Control barrier functions (CBFs) provide a principled mechanism for guaranteeing forward invariance of safe sets while modifying a nominal controller as little as possible through a real-time optimization problem [7], [8]. This perspective has rapidly influenced robot manipulation. Recent studies have used CBFs to enforce collaborative-safety regulations [9], operational-space safety under torque saturation [10], time-varying Cartesian output constraints in impedance control [11], obstacle avoidance through differentiable optimization [12], uncertain manipulator safety constraints [13], sampled-data whole-body safety for robotic manipulators [14], and task-consistent operational-space manipulation with many simultaneous constraints [15]. These results make a compelling case for CBF-QP safety filters as a unifying tool for practical manipulator control.

This work was supported by the Industrial Strategic Technology Development Program (RS-2024-00442029) funded by the Ministry of Trade, Industry and Energy (MOTIE, Korea). *(Corresponding author: Keehoon Kim)*

The authors are with the Department of Mechanical Engineering, Pohang University of Science and Technology(POSTECH), Pohang-si 37673, South Korea (e-mail: {hsjung02, khk}@postech.ac.kr).

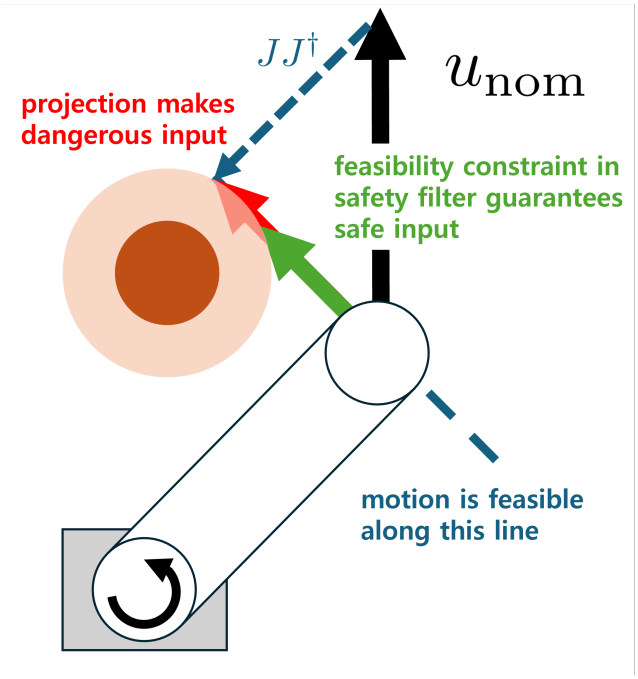


Fig. 1: Our problem statement. Infeasible velocity input from safety filter would result in constraint violation.

At the same time, a parallel literature has shown that operational-space control changes qualitatively once actuation is limited. Underactuated task space behavior arises not only in floating-base or contact-constrained systems, but also in arm manipulators with passive or locked joints, deliberate joint shutdown, actuation failures, and augmented tasks that impose additional objectives on internal links or body points [16]–[18]. Typical robotic fingers are also good examples of underactuation. In such settings, the realizable task space velocities are restricted to $\mathrm{range}(J(q))$ even at regular configurations. This is a structural limitation of the task-actuation pair, not merely a numerical issue associated with singularity.

This observation exposes a gap in the current safety-filter literature. Existing CBF methods typically formulate safety constraints under an implicit assumption that the filtered task space command can be realized by the robot through

the Jacobian map [9]–[11], [13]–[15], [19]. Conversely, underactuated operational-space methods focus on control allocation, projected dynamics, and performance under reduced actuation rather than on CBF-certified safety of the realized task motion [16]–[18]. As a result, a specific and practically important failure mode remains largely untreated: a task space CBF-QP can certify safety for an optimized velocity that does *not* belong to $\mathrm{range}(J)$, while the velocity actually executed after pseudoinverse realization is its projection onto $\mathrm{range}(J)$ and may violate the barrier condition. For underactuated arms, this mismatch can occur even when the Jacobian is well defined. This is why the problem is indeed important: the very scenarios that motivate velocity-level filtering—reactive obstacle avoidance, internal-link constraints, reduced actuation, and augmented tasks—are also the scenarios where infeasible task directions naturally appear.

This paper addresses that gap by developing a task space safety filter for velocity-driven underactuated robotic arms. The key novelty is to enforce motion feasibility *inside* the CBF-QP by constraining the filtered task space velocity to lie in the Jacobian range through an equivalent left-nullspace condition. To the best of our knowledge, this explicit Jacobian-range enforcement is the missing step in prior manipulator safety filters for underactuated task space control [10], [11], [13]–[15], [19]. The strengths of the proposed method are threefold. First, it removes the hidden projection mismatch between optimized and executed task space velocities, so the barrier certificate applies to the motion that the robot actually realizes. Second, it unifies obstacle avoidance, general inequality and equality motion constraints, and optional joint velocity bounds within a single convex optimization problem. Third, it enables analysis that is especially relevant in practice: we establish feasibility under standard initialization, derive a sampling-time-dependent condition on the CBF gain for sample-and-hold implementation, and show that the task space formulation preserves a unique optimal task velocity and is better conditioned than an equivalent joint space reformulation near singular configurations. Simulation studies on an underactuated Panda arm illustrate these properties.

The remainder of the paper is organized as follows. Section II formulates the problem. Section III reviews background material on QPs and CBFs. Section IV presents the proposed safety filter and related analysis. Section V reports simulation results. Section VI concludes the paper.

II. Problem Formulation

Consider the velocity-controlled task space model

$$\dot{x} = u, \tag{1}$$

where $x \in \mathbb{R}^m$ denotes the task state and $u \in \mathbb{R}^m$ denotes the commanded task space velocity. Let $q \in \mathbb{R}^n$ denote the joint configuration. The task map $x = x(q)$ induces the kinematic relation

$$x = x(q), \tag{2a}$$

$$\dot{x} = \frac{\partial x}{\partial q}\dot{q} = J(q)\dot{q}, \tag{2b}$$

where $J(q) \in \mathbb{R}^{m\times n}$ is the task Jacobian. The system is underactuated in task space whenever $\mathrm{rank}\, J(q) < m$, where not every task space velocity is realizable.

Our goal is to obtain a safe task space command $u^\star$ that deviates minimally from the nominal task velocity input u_{nom} while satisfying motion feasibility, obstacle avoidance, joint velocity, and additional motion constraints:

$$\begin{aligned} u^\star = \underset{u\in\mathbb{R}^m}{\mathrm{argmin}} \quad & \frac{1}{2}\left\|u - u_{\mathrm{nom}}\right\|_W^2 \\ \text{subject to} \quad & \text{motion feasibility constraints,} \\ & \text{obstacle avoidance constraints,} \\ & \text{joint velocity constraints,} \\ & \text{additional motion constraints,} \end{aligned} \tag{3}$$

where $\|z\|_W^2 \coloneqq z^T W z$ with $W \succ 0$.

III. Background

A. Quadratic Programming

A quadratic program can be written as

$$\begin{aligned} \text{minimize} \quad & \frac{1}{2}x^T Q x + c^T x \\ \text{subject to} \quad & Ax \preceq b, \end{aligned} \tag{4}$$

where $\preceq$ denotes componentwise inequality. A QP combines a quadratic objective with linear constraints, and efficient solvers make this class of problems suitable for real-time control.

B. Control Barrier Functions

Control barrier functions were introduced for safety-critical control. Let the safe set be the 0-superlevel set of $h : \mathbb{R}^m \to \mathbb{R}$ i.e.,

$$\mathcal{C} = \{x \mid h(x) \geq 0\}, \tag{5}$$

where h is continuously differentiable. The function h is a CBF if there exists an extended class-$\mathcal{K}_\infty$ function α such that

$$\sup_{u\in\mathcal{U}} \Big(\dot{h}(x,u) + \alpha\big(h(x)\big)\Big) \geq 0. \tag{6}$$

Under the standard CBF condition, the safe set $\mathcal{C}$ is forward invariant.

For a control-affine system

$$\dot{x} = f(x) + g(x)u, \tag{7}$$

the CBF condition becomes

$$\sup_{u\in\mathcal{U}} \left(\frac{\partial h}{\partial x}g(x)u + \frac{\partial h}{\partial x}f(x) + \alpha(h(x))\right) \geq 0, \tag{8}$$

which is affine in the control input u.

TABLE I: Linear inequality constraints used in the proposed safety filter.

Constraint	Barrier function	Linear inequality	Notes
Motion feasibility	–	$N^T u \preceq 0,\ -N^T u \preceq 0$	Columns of N form a basis of $\mathrm{null}(J^T)$
Obstacle avoidance	$h(x) = \|x - x_{\mathrm{obs}}\|^2 - r^2$	$-2(x - x_{\mathrm{obs}})^T u \le \alpha(h(x))$	x_{obs}: obstacle center, r: effective obstacle radius
Inequality motion constraint	$h(x) = c(x)$	$-\dfrac{\partial c}{\partial x} u \le \alpha(c(x))$	General state constraint $c(x) \ge 0$
Equality motion constraint	$h_\pm(x) = \varepsilon \mp c(x)$	$\pm\dfrac{\partial c}{\partial x} u \le \alpha(h_\pm(x))$	Relaxed equality constraint $c(x) = 0$ with $0 \le \varepsilon \ll 1$
Joint velocity limit	–	$-J^\dagger u \preceq \dot{q}_{\mathrm{lim}},\ J^\dagger u \preceq \dot{q}_{\mathrm{lim}}$	Optional when joint-rate bounds must be enforced

IV. Proposed Method

We solve (3) with a CBF-QP-based safety filter. Given the nominal task space velocity u_{nom}, the filtered command is obtained from

$$\begin{aligned} u^\star = \underset{u \in \mathbb{R}^m}{\mathrm{argmin}} \quad & \frac{1}{2} \|u - u_{\mathrm{nom}}\|_W^2 \\ \text{subject to} \quad & \dot{h}_i(x,u) + \alpha_i(h_i(x)) \ge 0, \quad i = 1, \dots, k, \\ & \text{other linear inequality constraints.} \end{aligned} \tag{9}$$

If $W \succ 0$, the objective is strictly convex in u. The remaining task is therefore to express each safety or feasibility requirement as a linear inequality in u.

A. Motion Feasibility

For an underactuated manipulator, the commanded task space velocity must belong to the realizable subspace $\mathrm{range}(J)$. Introducing $\dot{q}$ as an auxiliary optimization variable would enforce feasibility directly, but it also increases the dimension of the QP. Instead, we enforce feasibility in task space through the left nullspace of the Jacobian.

Let $r = \mathrm{rank}\, J(q)$, and let $N(q) \in \mathbb{R}^{m \times (m-r)}$ be a matrix whose columns form a basis of $\mathrm{null}(J(q)^T)$. Then

$$N(q)^T u = 0 \quad \Longleftrightarrow \quad u \in \mathrm{range}(J(q)). \tag{10}$$

Hence, motion feasibility can be written as an equality constraint in the QP. To keep the problem in standard QP form, we implement the equality as the pair of linear inequalities

$$N^T u \preceq 0, \qquad -N^T u \preceq 0. \tag{11}$$

This constraint must be enforced *inside* the safety filter. Suppose instead that a task space QP is solved without motion feasibility, yielding an optimizer output $\hat{u}$, and that the command is then mapped to joint velocity through the Moore-Penrose pseudoinverse,

$$\dot{q} = J^\dagger \hat{u}. \tag{12}$$

The actually executed task space velocity is

$$u_{\mathrm{exec}} = J\dot{q} = JJ^\dagger \hat{u} =: P(q)\hat{u}, \tag{13}$$

where $P(q) = JJ^\dagger$ is the orthogonal projector onto $\mathrm{range}(J)$. If $\hat{u} \notin \mathrm{range}(J)$, then in general

$$\dot{h} = \frac{\partial h}{\partial x} u_{\mathrm{exec}} = \frac{\partial h}{\partial x} P(q)\hat{u} \neq \frac{\partial h}{\partial x}\hat{u}. \tag{14}$$

The CBF condition is enforced on $\hat{u}$, but the physical system evolves with u_{exec}. Consequently, a task space command that is certified as safe by the QP can lose its safety guarantee after pseudoinverse execution. Enforcing $N^T u = 0$ removes this inconsistency by ensuring that the optimized and executed task space velocities coincide.

Example 1 (Loss of CBF satisfaction after pseudoinverse execution)**.** *Consider a 1-DoF planar rotator at $\theta = \pi/4$ whose feasible task space velocities satisfy $u_y = -u_x$. One realization is*

$$J = \begin{pmatrix} -1 \\ 1 \end{pmatrix}, \qquad J^\dagger = (J^T J)^{-1} J^T = \frac{1}{2}\begin{pmatrix} -1 & 1 \end{pmatrix}. \tag{15}$$

Thus, $\mathrm{range}(J) = \{(t, -t) \mid t \in \mathbb{R}\}$ *and motion feasibility reduces to $u_x + u_y = 0$. Suppose an obstacle barrier yields*

$$\frac{\partial h}{\partial x} = \begin{pmatrix} 1 & 0 \end{pmatrix}, \qquad \alpha\big(h(x)\big) = 0.1, \tag{16}$$

so the CBF condition is

$$\frac{\partial h}{\partial x} u + \alpha(h) \ge 0c \quad \Longleftrightarrow \quad u_x \ge -0.1. \tag{17}$$

Let the nominal task space velocity be $u_{\mathrm{nom}} = (0, 1)$, which already satisfies (17)*. If motion feasibility is omitted, the QP returns $\hat{u} = u_{\mathrm{nom}}$. The executed motion is then*

$$\dot{q} = J^\dagger \hat{u} = 0.5, \qquad u_{\mathrm{exec}} = J\dot{q} = \begin{pmatrix} -0.5 \\ 0.5 \end{pmatrix}, \tag{18}$$

which violates the barrier condition because $u_{\mathrm{exec},x} = -0.5 < -0.1$. If motion feasibility is enforced together with (17)*, the closest feasible and safe command becomes $u^\star = (-0.1, 0.1)$, and this command is executed exactly because $u^\star \in \mathrm{range}(J)$.*

For obtaining the N matrix, we can utilize the singular value decomposition. Let the singular value decomposition of the Jacobian be

$$J = U\Sigma V^T, \tag{19}$$

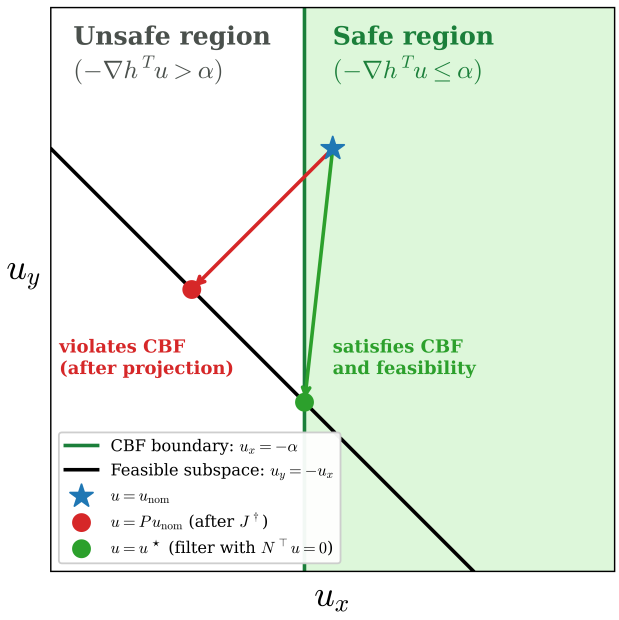


Fig. 2: Geometric interpretation of Example 1. The CBF inequality defines a safe half-space in task velocity space, while motion feasibility restricts the admissible velocity to the one-dimensional subspace $\mathrm{range}(J)$. If feasibility is not enforced in the QP, the optimizer can return an infeasible command $\hat{u}$ that satisfies the CBF inequality; its projection $u_{\text{exec}} = P\hat{u}$ may then violate the barrier condition. Enforcing $N^T u = 0$ directly in the QP yields a feasible and safe command.

where $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$ are orthogonal matrices and $\Sigma \in \mathbb{R}^{m\times n}$ contains the singular values of J. If $\mathrm{rank}(J) = r$, partition U as

$$U = \begin{pmatrix} U_r & U_0 \end{pmatrix}, \tag{20}$$

where the columns of U_r correspond to the nonzero singular values and the columns of U_0 correspond to the zero singular values. Then

$$J^T U_0 = V\Sigma^T U^T U_0 = 0, \tag{21}$$

so the columns of U_0 form a basis of $\mathrm{null}(J^T)$. Therefore, one may choose

$$N = U_0, \tag{22}$$

and the feasibility condition $N^T u = 0$ is equivalent to requiring $u \in \mathrm{range}(J)$. In implementation, singular values smaller than a prescribed numerical threshold can be treated as zero when constructing U_0.

B. Obstacle Avoidance

For a spherical obstacle centered at x_{obs} with effective radius r, consider

$$h(x) = \|x - x_{\text{obs}}\|^2 - r^2. \tag{23}$$

Since $\dot{x} = u$, the barrier derivative is

$$\dot{h} = 2(x - x_{\text{obs}})^T u. \tag{24}$$

Therefore, the CBF condition $\dot{h} + \alpha(h) \ge 0$ is equivalent to the linear inequality

$$-2(x - x_{\text{obs}})^T u \le \alpha(h(x)). \tag{25}$$

C. Additional Motion Constraints

Additional task constraints can also be incorporated into the safety filter. For a general inequality constraint

$$c(x) \ge 0, \tag{26}$$

we choose the barrier function $h(x) = c(x)$. The corresponding CBF condition becomes

$$\dot{h} + \alpha(h) \ge 0 \quad \Longleftrightarrow \quad -\frac{\partial c}{\partial x} u \le \alpha(c(x)). \tag{27}$$

Equality constraints can be handled by introducing two relaxed barrier functions,

$$h_+(x) = \varepsilon - c(x), \qquad h_-(x) = \varepsilon + c(x), \tag{28}$$

where $0 \le \varepsilon \ll 1$ provides a small numerical tolerance. This yields the pair of inequalities

$$\frac{\partial c}{\partial x} u \le \alpha(h_+(x)), \qquad -\frac{\partial c}{\partial x} u \le \alpha(h_-(x)). \tag{29}$$

D. Joint Velocity Limits

Even when the task space command is feasible, the corresponding joint velocity may exceed actuator limits. Using the minimum-norm realization $\dot{q} = J^\dagger u$, joint-rate bounds can be enforced through

$$-\dot{q}_{\text{lim}} \preceq J^\dagger u \preceq \dot{q}_{\text{lim}}, \tag{30}$$

which again defines linear inequalities in the optimization variable u.

E. Feasibility of the Safety Filter

Assume that the current state already lies in the admissible set, so that all barrier and state constraints are satisfied at the current sampling instant. Then the zero input $u = 0$ satisfies every linear inequality in (9), because each barrier inequality reduces to $\alpha(h_i(x)) \ge 0$. Under this standard initialization assumption, the QP is therefore always feasible.

Feasibility, however, is only a minimal requirement. The goal is to keep the admissible set of control inputs as large as possible so that the filter remains close to the nominal command. For this purpose, define

$$\mathcal{U}_{\text{safe}} = \{u \in \mathbb{R}^m \mid u \text{ satisfies (9)}\}, \tag{31a}$$
$$\mathcal{U}_{\text{cbf}} = \{u \in \mathbb{R}^m \mid u \text{ satisfies the CBF constraints}\}, \tag{31b}$$
$$\mathcal{U}_{\text{lin}} = \{u \in \mathbb{R}^m \mid u \text{ satisfies the non-CBF constraints}\}. \tag{31c}$$

Then $\mathcal{U}_{\text{safe}} = \mathcal{U}_{\text{cbf}} \cap \mathcal{U}_{\text{lin}}$. Increasing the CBF gain generally enlarges $\mathcal{U}_{\text{cbf}}$, but in discrete time the gain cannot be arbitrarily large.

Consider a sample-and-hold implementation with sampling period $\Delta t > 0$. Using the first-order approximation

$$h(x_{k+1}) \approx h(x_k) + \dot{h}(x_k, u_k)\Delta t, \tag{32}$$

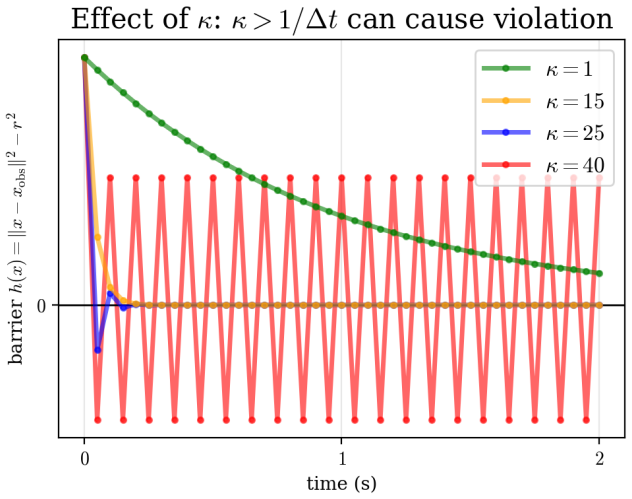


Fig. 3: Sample-and-hold simulation for Example 2 with $\Delta t = 0.05\,\text{s}$. For $\kappa = 1$ and $\kappa = 15$, which satisfy $\kappa \le 1/\Delta t = 20$, the barrier remains nonnegative. For $\kappa = 25$ and $\kappa = 40$, which violate $\kappa \le 1/\Delta t$, the barrier crosses into the unsafe region and oscillatory behavior appears.

a conservative sufficient condition for preserving $h(x_{k+1}) \ge 0$ is

$$-\dot{h}(x_k, u_k) \le \frac{h(x_k)}{\Delta t}. \tag{33}$$

Combining this bound with the CBF condition yields

$$\alpha(h) \le \frac{1}{\Delta t} h. \tag{34}$$

For the common choice $\alpha(h) = \kappa h$ with $\kappa > 0$, (34) reduces to

$$\kappa \le \frac{1}{\Delta t}. \tag{35}$$

Example 2 (Excessive CBF gain can break discrete-time safety)**.** *Consider the task space dynamics $\dot{x} = u$ with sample-and-hold control $u(t) = u_k$ for $t \in [k\Delta t, (k+1)\Delta t)$. Let a circular obstacle of radius r be centered at x_{obs}, and define*

$$h(x) = \|x - x_{\text{obs}}\|^2 - r^2. \tag{36}$$

We initialize the system at $x_0 = (r + \delta, 0)$ and choose a nominal command $u_{\text{nom}} = (-0.60, 0)$ directed toward the obstacle. In the simulation, $r = 0.20$, $\delta = 0.02$, and $\Delta t = 0.05$, so that $1/\Delta t = 20$ were used. We tested $\kappa \in \{1, 15, 25, 40\}$. For $\kappa \le 1/\Delta t$, the barrier remained nonnegative. For $\kappa > 1/\Delta t$, one-step barrier violations and oscillatory motion were observed, as shown in Fig. 3.

F. Comparison with a Joint-Space Formulation

The same safety filter can be written in joint space by using $\dot{q}$ as the optimization variable:

$$\begin{aligned} \dot{q}^\star = \underset{\dot{q} \in \mathbb{R}^n}{\text{argmin}} \quad & \frac{1}{2} \|J(q)\dot{q} - u_{\text{nom}}\|_W^2 \\ \text{subject to} \quad & \text{same linear constraints with } u = J(q)\dot{q}. \end{aligned} \tag{37}$$

Although (9) and (37) can produce the same realized task velocity when J is well conditioned, the task space formulation has two important advantages.

First, the task space QP preserves uniqueness of the optimal task space velocity. In (9), the objective is strictly convex in u because $W \succ 0$. Hence, whenever the feasible set is nonempty, the optimizer returns a unique task space command $u^\star$. In contrast, the quadratic term of (37) is

$$\|J\dot{q} - u_{\text{nom}}\|_W^2 = \dot{q}^T J^T W J \dot{q} - 2u_{\text{nom}}^T W J \dot{q} + u_{\text{nom}}^T W u_{\text{nom}}, \tag{38}$$

whose Hessian is $2J^T W J$. This matrix is positive semidefinite, but it is positive definite only when J has full column rank. Near singular configurations, strict convexity is therefore lost and the optimal joint velocity may not be unique.

Second, the task space formulation is better conditioned numerically. Consider a planar 2R manipulator with unit link lengths at the configuration $(q_1, q_2) = (\pi/4, \varepsilon)$. Its Jacobian is

$$J_\varepsilon = \begin{pmatrix} -\frac{\sqrt{2}}{2} - \sin\left(\frac{\pi}{4} + \varepsilon\right) & -\sin\left(\frac{\pi}{4} + \varepsilon\right) \\ \frac{\sqrt{2}}{2} + \cos\left(\frac{\pi}{4} + \varepsilon\right) & \cos\left(\frac{\pi}{4} + \varepsilon\right) \end{pmatrix}. \tag{39}$$

A direct calculation gives

$$J_\varepsilon^T J_\varepsilon = \begin{pmatrix} 2 + 2\cos\varepsilon & 1 + \cos\varepsilon \\ 1 + \cos\varepsilon & 1 \end{pmatrix}, \tag{40}$$

with determinant $\det(J_\varepsilon^T J_\varepsilon) = \sin^2\varepsilon$. Hence the smallest eigenvalue tends to zero as $\varepsilon \to 0$. Since the trace tends to 5, we obtain

$$\lambda_{\min}(J_\varepsilon^T J_\varepsilon) \approx \frac{\varepsilon^2}{5}, \qquad \text{cond}(J_\varepsilon^T J_\varepsilon) \approx \frac{25}{\varepsilon^2} \tag{41}$$

for small ε. By contrast, the Hessian of (9) is simply $2W$, which is independent of J. The Jacobian appears only in linear constraints such as $N^T u = 0$ and, optionally, the joint-velocity bounds. Therefore the task space QP remains well conditioned even near singularity.

In summary, when J is well conditioned, the task space and joint space formulations can yield the same realized task velocity. Near singular configurations, however, the joint space QP becomes numerically ill-defined because its Hessian is $2J^T W J$. To verify this effect, we generated a Panda stress trajectory around a near-singular posture, where $\sigma_{\min}(J)$ reached 5.7×10^{-5} and $\text{cond}(J^T J)$ reached 1.16×10^9. In the corresponding safety-filter comparison, the task space Hessian remained fixed at $2W$ with $\text{cond}(2W) = 1$, whereas the joint space Hessian had a median condition number of 3.12×10^{16}. The joint space solver consequently required substantially more iterations, with a peak of 375 iterations compared with 100 for the task space formulation, and additional near-singularity trials produced approximately a 5% QP accuracy failure rate. Table II summarizes the experimental settings and representative outcomes.

This confirms that the task space formulation preserves a unique optimal task command and avoids the singular weighting induced by $J^T W J$. Joint space filters may still be useful when hard joint space constraints, such as velocity or acceleration limits, dominate the design.

TABLE II: Near-singularity stress-test settings and outcomes.

Item	Setting or result
Robot/task	Panda arm; 7 joints; 6-D pose-velocity task
Stress trajectory	12.0 s, $\Delta t = 0.005$ s, 2400 samples
Singularity level	$\min \sigma_{\min}(J) = 5.7 \times 10^{-5}$; $\max \mathrm{cond}(J^T J) = 1.16 \times 10^9$
QP setup	$W = I_6$, $\lvert\dot{q}_i\rvert \leq 0.5$, OSQP tolerance 10^{-6}, no joint-space regularization
Task space result	$\mathrm{cond}(2W) = 1$; peak 100 OSQP iterations; 2500/2500 solved
Joint space result	median $\mathrm{cond}(2J^T W J) = 3.12 \times 10^{16}$; peak 375 OSQP iterations; 2500/2500 solved
Additional trials	about 5% joint-space QP accuracy failures near singularity

G. Implementation on $SE(3)$

For pose tasks on $SE(3)$, let $T \in SE(3)$ denote the task pose and let $x = \log(T)$ be its exponential coordinates. If $\mathcal{V}$ denotes the body twist, then

$$\dot{x} = \mathrm{dexp}_x^{-1} \mathcal{V}. \tag{42}$$

Accordingly, the barrier derivative becomes

$$\dot{h} = \nabla h(x)^T \mathrm{dexp}_x^{-1} \mathcal{V}, \tag{43}$$

so the same QP structure can be used with $\mathcal{V}$ as the optimization variable. Closed-form expressions for the differential of the exponential map are given in [20].

V. Results

We evaluate the proposed method in three representative simulation scenarios using a Franka Emika Panda robot, a 7-degree-of-freedom (DoF) manipulator simulated in PyBullet [21]. This paper does not claim full hardware-level validation. Rather, it isolates and evaluates the proposed kinematic safety filter in a controlled setting that matches the assumptions of the method. Since the main contribution lies in the formulation and behavior of the task space filter regarding underactuation, simulation provides the most direct way to assess that contribution without conflating it with hardware-specific integration issues. Joints and links are indexed from 1 to 7. In every scenario, the task space nominal velocity is generated by a dynamical system and executed for four predefined desired trajectories. Each trajectory is run once from its associated initial point, so each method is evaluated on four runs per scenario.

- **S1)** Joints 3 and 5 are intentionally locked. The task dimension is 6 and the actuation dimension is 5, so the system is underactuated.
- **S2)** Joints 3 and 5 are intentionally locked, and a virtual wall is added as a motion constraint. The task dimension is 6 and the actuation dimension is 5, so the system is again underactuated.
- **S3)** The translational position of link 4 is augmented to the task state. The task dimension is 9 (=6+3), whereas the actuation dimension is 7, so the resulting task is underactuated. Such situations could occur frequently in real world applications, for example the robot is performing a whole-body task or operating in a cluttered environment.

We compare two task space filters: the proposed method, which explicitly enforces motion feasibility, and an ablated baseline that omits this constraint and executes the filtered task space command through the pseudoinverse. Table III reports averages over the four trajectories in each scenario. CBF violation rate is computed by $\frac{\text{number of timesteps with } h < 0}{\text{number of total timesteps}}$.

A. Results for S1

Fig. 5 shows that the two filters remain visually indistinguishable across all four trajectories. CBF value is not plotted for S1 since there is no active constraint. The mean tracking-error curves almost overlap for the entire rollout, and Table III confirms that the average RMS tracking error is identical to four significant digits (0.1171 for both methods).

This behavior is expected because S1 contains no active constraint; thus two filters are equivalent. For all four trajectories, the nominal task command is already sufficiently compatible with the realizable task velocity subspace. As a result, enforcing $N^T u = 0$ does not produce any meaningful loss of tracking performance. The proposed filter does incur a larger QP cost than the baseline (2.26e-3 versus 4.60e-6), but this reflects a modest correction of the task space command rather than degraded realized motion.

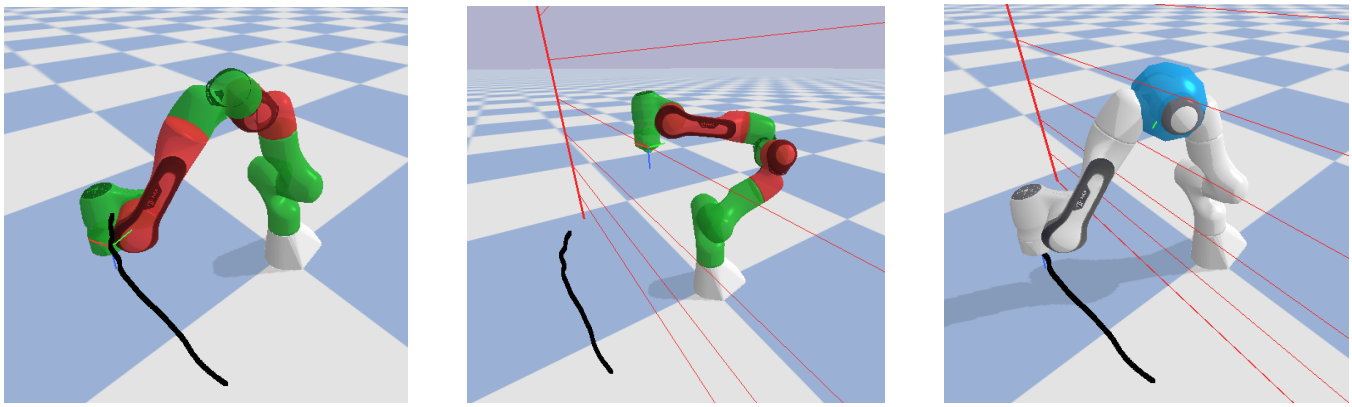
B. Results for S2

The difference between the two filters becomes clear once the virtual wall becomes active. In Fig. 6(a), the solid curves remain close to the boundary for both methods, but the dotted minimum trace of the baseline drops below zero. This indicates that some executed motions violate the wall constraint after pseudoinverse realization even though the optimized task space command satisfied the CBF-QP. At the per-trajectory level, the baseline exhibits persistent barrier violations on all four trajectories, ranging from 0.3% to 91.7% of the rollout, with an average of 25.37%. By contrast, the proposed filter achieves a 0% persistent violation rate in all four runs.

Fig. 6(b) shows that the tracking-error trends of the two methods remain very similar. The average RMS tracking error is 0.2914 for both methods. The proposed filter improves three of the four trajectories slightly and worsens one trajectory slightly, so the aggregate difference is negligible. The main effect of the feasibility constraint in S2 is therefore not a better nominal tracking, but a restoration of safety consistency: the optimized task space command is realizable, so the CBF condition applies to the velocity that the robot actually executes. The modest increase in the QP cost (1.55e-2 versus 1.20e-2) is expected because the optimizer can no longer exploit infeasible task space directions to remain artificially close to the nominal command.

C. Results for S3

S3 is the most demanding scenario because the translational position of link 4 is added to the task state, increasing the task dimension from 6 to 9 while the robot still has only



(a) (b) (c)

Fig. 4: Simulation setups. (a) End-effector tracking with joints 3 and 5 locked. (b) The same task as in (a), with a virtual wall placed between the initial pose and the target trajectory. (c) End-effector tracking with an additional virtual-wall constraint applied to the position of link 4.

TABLE III: Quantitative comparison between the baseline task space filter without motion feasibility and the proposed filter with motion feasibility, averaged over the four predefined trajectories in each scenario.

Scenario	CBF violation rate		Tracking error		QP cost	
	w/o feasibility	w/ feasibility	w/o feasibility	w/ feasibility	w/o feasibility	w/ feasibility
S1	0.00% (no CBF)	0.00% (no CBF)	0.1171	0.1171	4.5963e-06	2.2572e-03
S2	25.37%	0.00%	0.2914	0.2914	1.2045e-02	1.5519e-02
S3	24.70%	0.00%	0.3569	0.0856	7.3275e-04	1.7961e-02

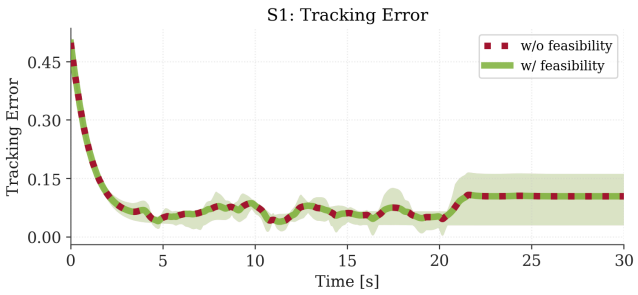
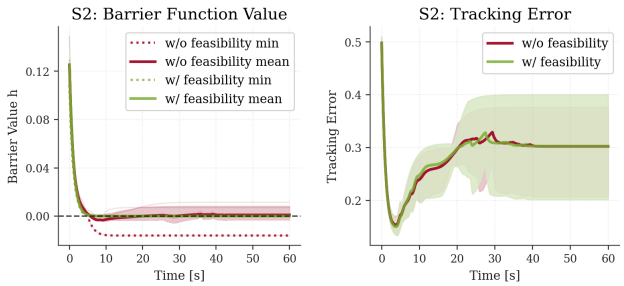


Fig. 5: Results for S1. Mean tracking error over the four predefined trajectories; the shaded band indicates the interquartile range.



(a) (b)

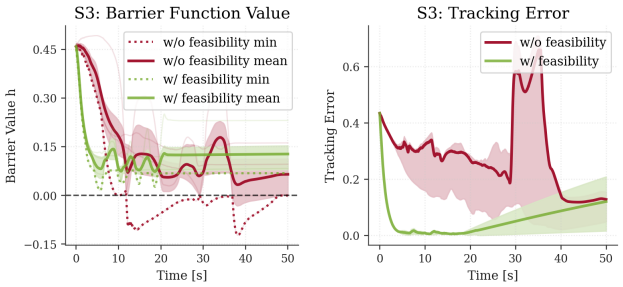
Fig. 6: Results for S2. (a) Barrier values. (b) Tracking error. Solid curves show the mean over the four predefined trajectories and shaded bands indicate the interquartile range; in (a), dotted curves denote the minimum barrier value across trajectories.

7 actuated DoFs. In this setting, underactuation is structural rather than induced only by joint locking. The separation between the two filters is now strong in both safety and tracking.

Fig. 7(a) shows that the baseline minimum barrier becomes negative for a substantial portion of the rollout, whereas the proposed filter maintains a positive margin on every trajectory. Quantitatively, the baseline without motion feasibility constraint has an average persistent violation rate of 24.70%. The proposed filter reduces the persistent violation rate to 0% on all four trajectories and keeps the minimum barrier value strictly positive in every run.

The tracking plot in Fig. 7(b) shows an equally clear performance gap. The average RMS tracking error drops from 0.3569 for the baseline to 0.0856 for the proposed method. Once realizability is enforced inside the QP, the controller expends more task space correction but delivers both safe and better-behaved motion.

Overall, the experiments support the central claim of the paper. Across all 12 scenario-trajectory cases, the proposed filter achieves zero persistent barrier violations. The baseline violates the barrier in all four S2 runs and in two of the four S3 runs. These results are consistent with the analysis in Section IV-A: when the nominal task already lies close to $\mathrm{range}(J)$, adding the feasibility constraint is essentially



(a) (b)

Fig. 7: Results for S3. (a) Barrier values. (b) Tracking error. Solid curves show the mean over the four predefined trajectories and shaded bands indicate the interquartile range; in (a), dotted curves denote the minimum barrier value across trajectories.

neutral, whereas in wall-constrained or augmented-task settings it removes the mismatch between the optimized and executed task velocities. In the structurally underactuated S3 case, this correction improves tracking as well as safety.

VI. Conclusion

This paper presented a task space safety filter for velocity-driven underactuated robotic arms. The main contribution is to enforce motion feasibility directly inside the CBF-QP by constraining the filtered task space velocity to lie in the Jacobian range through an equivalent left-nullspace condition. This resolves the mismatch between the optimized task space command and the velocity actually executed after pseudoinverse realization, which is the main source of hidden safety loss in underactuated settings. The resulting formulation accommodates obstacle avoidance, general inequality and equality motion constraints, and optional joint velocity bounds within a single convex optimization problem.

We also discussed feasibility of the filter, derived a sampling-time-dependent upper bound on the CBF gain for sample-and-hold implementation, and explained why the task space formulation is better conditioned than a joint space formulation near singular configurations. Simulation results showed that the proposed filter preserves nominal performance when no safety constraint is active and prevents barrier violations when wall-type constraints become active. In the more demanding augmented-task scenario, it also improves practical tracking behavior.

References

[1] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal on Robotics and Automation*, vol. 3, no. 1, pp. 43–53, 1987.

[2] ——, "Real-time obstacle avoidance for manipulators and mobile robots," *The International Journal of Robotics Research*, vol. 5, no. 1, pp. 90–98, 1986.

[3] S. M. Khansari-Zadeh and A. Billard, "Learning stable non-linear dynamical systems with gaussian mixture models," *IEEE Transactions on Robotics*, vol. 27, no. 5, pp. 943–957, 2011.

[4] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: Learning attractor models for motor behaviors," *Neural Computation*, vol. 25, no. 2, pp. 328–373, 2013.

[5] Y. Nakamura and H. Hanafusa, "Inverse kinematic solutions with singularity robustness for robot manipulator control," *Journal of Dynamic Systems, Measurement, and Control*, vol. 108, no. 3, pp. 163–171, 1986.

[6] S. Chiaverini, "Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators," *IEEE Transactions on Robotics and Automation*, vol. 13, no. 3, pp. 398–410, 1997.

[7] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Transactions on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2017.

[8] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *2019 18th European Control Conference (ECC)*, 2019, pp. 3420–3431.

[9] F. Ferraguti, M. Bertuletti, C. T. Landi, M. Bonfè, C. Fantuzzi, and C. Secchi, "A control barrier function approach for maximizing performance while fulfilling to iso/ts 15066 regulations," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5921–5928, 2020.

[10] M. A. Murtaza, S. Aguilera, V. Azimi, and S. Hutchinson, "Real-time safety and control of robotic manipulators with torque saturation in operational space," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 702–708.

[11] H. Wang, J. Peng, F. Zhang, H. Zhang, and Y. Wang, "High-order control barrier functions-based impedance control of a robotic manipulator with time-varying output constraints," *ISA Transactions*, vol. 129, pp. 361–369, 2022.

[12] B. Dai, R. Khorrambakht, P. Krishnamurthy, V. Gonçalves, A. Tzes, and F. Khorrami, "Safe navigation and obstacle avoidance using differentiable optimization based control barrier functions," *IEEE Robotics and Automation Letters*, vol. 8, no. 9, pp. 5376–5383, 2023.

[13] D. Zeng, Y. Jiang, Y. Wang, H. Zhang, and Y. Feng, "Robust adaptive control barrier functions for input-affine systems: Application to uncertain manipulator safety constraints," *IEEE Control Systems Letters*, vol. 8, pp. 279–284, 2024.

[14] Y. Xiong, D. H. Zhai, and Y. Xia, "Robust whole-body safety-critical control for sampled-data robotic manipulators via control barrier functions," *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 16 050–16 061, 2025.

[15] D. Morton and M. Pavone, "Safe, task-consistent manipulation with operational space control barrier functions," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2025, pp. 187–194.

[16] M. Mistry and L. Righetti, "Operational space control of constrained and underactuated systems," in *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, 2011.

[17] Y. Lee, N. Tsagarakis, and J. Lee, "Study on operational space control of a redundant robot with un-actuated joints: Experiments under actuation failure scenarios," *Nonlinear Dynamics*, vol. 105, no. 1, pp. 331–344, 2021.

[18] X. Chu, Y. Tang, A. M. Giordano, T. Chen, and K. W. S. Au, "Operational space control for planar pa^{N-1} underactuated manipulators using orthogonal projection and quadratic programming," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 12 853–12 859.

[19] J. Lin, D.-H. Zhai, Y. Xiong, and Y. Xia, "Safety control for ur-type robotic manipulators via high-order control barrier functions and analytical inverse kinematics," *IEEE Transactions on Industrial Electronics*, vol. 71, no. 6, pp. 6150–6160, 2024.

[20] J. Kim, M. Sung, Y. Choi, J. Park, and W. K. Chung, "Impedance control design framework using commutative map between se(3) and $\mathfrak{se}(3)$," *IEEE Transactions on Robotics*, 2025.

[21] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics, and machine learning," 2016, software available at http://pybullet.org.