Robust Singularity Handling in Operational Space Control

William Chong, Mikael Jorda, and Oussama Khatib

Abstract In the task space control of robots, mapping differential end-effector motion to differential joint motion becomes ill-posed near singularities. Furthermore, discontinuities arising during the transition out of these regions present a significant challenge. While conventional methods often rely on approximate kinematic solutions or avoidance, we propose a dynamics-based approach for open-chain manipulators. This paper introduces a multi-singularity handling methodology within the Constraint-Consistent Whole Body Control framework, enabling smooth navigation through singularity neighborhoods. We present a singularity classification algorithm to identify active singularities in order to execute specific control strategies for accurate task tracking. Additionally, a velocity transition method ensures smooth exit from the singularity region. Experiments demonstrate that the proposed method achieves robust performance and smooth traversal through singularities.

1 Introduction

Kinematic singularities occur when a robot instantaneously loses the ability to translate or rotate about certain directions. Mathematically, the Jacobian

W. Chong
Stanford University, Stanford CA 94305, USA
e-mail: wmchong@stanford.edu

M. Jorda
Stanford University, Stanford CA 94305, USA
e-mail: mjorda@stanford.edu

O. Khatib
Stanford University, Stanford CA 94305, USA
e-mail: khatib@stanford.edu

matrix relating a differential joint motion to a differential end-effector task motion is singular, causing the matrix to be ill-conditioned. This creates issues during end-effector control, such as unbounded joint velocities, poor manipulability, and control instability. To address kinematic singularities for end-effector control, various approaches have been taken. Pseudo, generalized, and singularity-robust inverses of the Jacobian [1, 2, 3, 4] have been used; however, they only provide an approximate solution for inverse kinematics in singularity regions. Other methods focus on avoiding singularities [9, 10]; however, this is limiting, as singularities occur inside the robot's workspace.

Khatib [5, 6] formulated singularity control strategies within the Operational Space Control framework, where the robot is treated as redundant with respect to motion in the singular directions. Building on this methodology, this paper proposes a singularity-handling strategy for improved behavior in singularity neighborhoods.

2 Approach

The Constraint-Consistent Whole Body Control framework [7] is first presented. Given a stack-of-tasks hierarchy, the corresponding whole-body task representation and task Jacobian are represented with $\vartheta_\otimes$ and $J_\otimes$, where $\vartheta_\otimes$ denotes the instantaneous velocity of the whole-body task. For example, given a hierarchy of a constraint task (c) followed by a control task (t), then:

$$\vartheta_\otimes \triangleq \begin{bmatrix} \vartheta_c \\ \vartheta_{t|c} \end{bmatrix}, \quad J_\otimes \triangleq \begin{bmatrix} J_c \\ J_{t|c} \end{bmatrix} \tag{1}$$

where the control task is both projected in the nullspace of the constraint task, and made consistent with the constraint task by projecting the control task into the non-conflicting task space (thus being denoted by the subscript $(t \,|\, c)$).

Given the equations of motion of a robot:

$$A(q)\ddot{q} + b(q, \dot{q}) + g(q) = \Gamma \tag{2}$$

where A is the joint space mass matrix, $b(q, \dot{q})$ are the Coriolis and Centrifugal forces, $g(q)$ is the joint space gravity vector, and Γ is the vector of joint torques, then the whole-body task space dynamics is obtained by projecting the equations of motion using the dynamically-consistent inverse task Jacobian $\overline{J}_\otimes$:

$$\Lambda_\otimes \dot{\vartheta}_\otimes + \mu_\otimes + p_\otimes = F_\otimes \tag{3}$$

$$\begin{aligned} \Lambda_{\otimes} &= \left(J_{\otimes} A^{-1} J_{\otimes}^T\right)^{-1}, \quad \mu_{\otimes} = \overline{J}_{\otimes}^T b - \Lambda_{\otimes} \dot{J}_{\otimes} \dot{q}, \\ p_{\otimes} &= \overline{J}_{\otimes}^T g, \quad F_{\otimes} = \overline{J}_{\otimes}^T \Gamma, \quad \overline{J}_{\otimes}^T = \Lambda_{\otimes} J_{\otimes} A^{-1} \end{aligned} \tag{4}$$

To simplify the presentation, we consider redundant open-chain robots, involving general task frame specifications (6 degrees of freedom). The approach extends to multi-task control of branching open-chain robots with the control framework. The control torque vector required to achieve an end-effector force vector F for a task specification defined with the task Jacobian J is $\Gamma = J^T F$.

Kinematic singularities occur when the Jacobian matrix becomes rank deficient at a configuration q. However, robot control becomes difficult in a neighborhood around kinematic singularities. The neighborhood is defined by the domain $D(q) = \{q \,|\, \lambda(\Lambda^{-1}(q)) \leq \lambda_{min}\}$ where $\lambda(\Lambda^{-1}(q))$ are the eigenvalues of the inverse of the task inertia matrix [8]. Small eigenvalues correspond to large task inertia and consequently, large control forces along their eigenvector directions.

Using the sub-scripts ns and s to refer to the non-singular and singular components, the singular value decomposition of J is written as

$$J = U \Sigma V^T = \begin{bmatrix} U_{ns} \; U_s \end{bmatrix} \begin{bmatrix} \Sigma_{ns} & 0 \\ 0 & \Sigma_s \end{bmatrix} \begin{bmatrix} V_{ns} \; V_s \end{bmatrix}^T \tag{5}$$

At singularity, the robot cannot move along the directions spanned by U_s, and V_s gives the corresponding singular joint space basis. Unlike [8], we decompose the task using U_{ns} and U_s instead of the eigenvectors of Λ^{-1}, since the eigenvectors generally do not align with the left singular vectors of J.

When the robot is in a singularity neighborhood, the singular directions are controlled within the nullspace of the non-singular task [6]. The projected non-singular and singular task Jacobians are:

$$J_{ns} = U_{ns}^T J, \quad J_{s|ns} = U_{s|ns}^T J N_{ns}, \quad N_{ns} = I - \overline{J}_{ns} J_{ns} \tag{6}$$

where N_{ns} is the dynamically-consistent nullspace of the non-singular task. In the neighborhood, $\Lambda_{s|ns}^{-1}$ is ill-conditioned from $J_{s|ns}$, causing large forces in the singular task space when using task control. Our approach provides a well-conditioned task representation for control in the nullspace of the non-singular task, combined with control strategies and numerical singularity classification.

2.1 Numerical Singularity Classification

Khatib [6] classified open-chain singularities into two categories in terms of control: Type 1 and 2 (Fig. 1). Type 1 singularities are those at which the

end-effector can be controlled in the singular directions using motion in the nullspace of the non-singular task. Type 2 singularities are those at which motion in the nullspace only affects the singular direction.

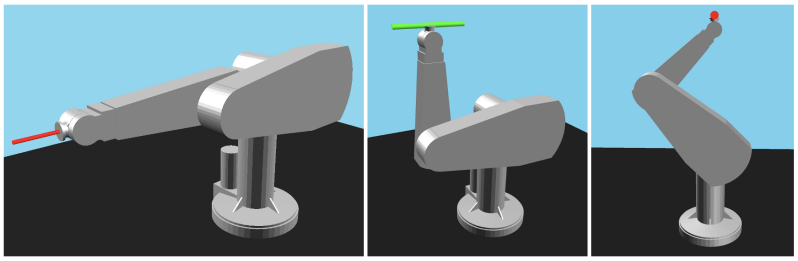

Fig. 1 Type 1 and 2 singularities for the PUMA at various configurations: left — Type 1 (elbow lock), middle — Type 2 (wrist lock), right — Type 2 (overhead lock). Red lines indicate lost linear motion direction; green lines indicate lost angular motion direction.

Since motion in the nullspace for Type 2 singularities only produce internal joint motion, we extend Seng [11] to classify Type 2 singularities by computing the projected second-order task displacement d_s:

$$d_s = u_s^T(v_s^T H(q) v_s) \tag{7}$$

where u_s is the singular direction, v_s is the singular joint space direction, and $H(q)$ is the kinematic Hessian tensor. The term arises from projecting the second-order Taylor expansion term of the forward kinematics onto the singular direction using displacements in the singular joint space, where $d_s = 0$ at a Type 2 singularity. We extend this to the singularity neighborhood, classifying Type 2 singularities as $|d_s| \leq d_{min}$ and Type 1 otherwise, with d_{min} a threshold.

2.1.1 Degenerate Singularities

Degenerate singular values during singularity classification can occur, leading to non-unique left and right singular vectors. We propose a recursive search in the degenerate basis, sequentially optimizing to first find paired (u_s, v_s) for all Type 2 singularities, then for all Type 1 singularities.

Let U_d and V_d, initialized to the degenerate columns of U_s and V_s, track the remaining rank n unclassified degenerate singular task and joint space basis. The optimization to identify Type 2 singularities is:

$$\begin{aligned} \min_x \quad & [d_s(x)]^2 = [u_s(x)^T(v_s(x)^T H(q) v_s(x))]^2 \\ \text{s.t.} \quad & v_s(x) = V_d x, \quad u_s(x) = U_d x, \quad \|x\| = 1 \end{aligned} \tag{8}$$

where x is the coefficient vector to minimize the second-order displacement objective. At zero singular value, u_s is determined from the task displacement from a small virtual displacement in v_s. Since the problem is non-linear, an initial solution estimate is obtained by optimizing an upper bound on the objective:

$$[u_s^T(v_s^T H v_s)]^2 = [x^T U_d^T (x^T V_d^T H V_d x)]^2 \leq \|x\|^2 \|U_d^T (x^T V_d^T H V_d x)\|^2 \quad (9)$$

where the Cauchy–Schwarz inequality is applied. Since the upper bound is quartic in x, a second bound that is quadratic in x is formed for faster computation:

$$\|U_d^T (x^T V_d^T H V_d x)\|^2 = \sum_j (x^T V_d^T (\sum_i (U_d)_{ij} H_i) V_d x)^2 = \sum_j (x^T A_j x)^2 \quad (10)$$

where $A_j = V_d^T (\sum_i (U_d)_{ij} H_i) V_d$, and the final upper bound is given by

$$\sum_j (x^T A_j x)^2 \leq x^T (\sum_j A_j^2) x \quad \text{from} \quad (x^T A_j x)^2 \leq \|A_j x\|^2 = x^T A_j^2 x \quad (11)$$

$(U_d)_{ij}$ is the (i,j)th element of U_d and H_i is the ith slice of the Hessian tensor. The eigenvector corresponding to the smallest eigenvalue of $\sum_j A_j^2$ provides an initial solution for the optimization, which is solved using COBYLA [12].

After solving for x, V_d is updated to the rank $n-1$ basis of V_d projected onto the nullspace of $v_s(x)$ (and similarly for U_d and $u_s(x)$). When the solution returns $|d_s(x)| > d_{min}$, the optimization is changed to maximization to find Type 1 singularities, using the largest eigenvalue of the upper bound for initialization. Type 2 singularities are identified first, since their non-zero second-order displacements along other directions in the degenerate basis could otherwise mischaracterize Type 1 singularities.

2.2 Control Strategy

Singularity strategies are controlled in the null-space of the non-singular task.

- For Type 1 singularities, the end-effector motion in the singular direction is directly controlled by controlling the singular value derivative ($\dot{\sigma}_s$).
- For Type 2 singularities, the singular direction is controlled to rotate away from the control task direction.

The Type 1 singularity strategy also determines whether the task intends to move the robot toward or away from the singularity to adjust the strategy. To eliminate discontinuities after exiting the singularity neighborhood, we propose a transition using goal-position velocity-saturated control with scheduled saturation velocity.

2.2.1 Task Representation

The singular joint space is controlled to execute the strategies, defined by the singular joint space (sjs) Jacobian:

$$J_{sjs(i)} = v_{s(i)}^T N_{ns} \tag{12}$$

where $v_{s(i)}$ is the joint space vector that directly maps to motion in $u_{s(i)}$ for the ith singularity. $J_{sjs(i)}$ is well-conditioned since $v_{s(i)}$ is defined in the joint space. Velocity control is used in the strategies with the unit-mass control torque:

$$\Gamma_{sjs(i)}^* = -K_v(\dot{q} - \dot{q}_d) \tag{13}$$

where K_v is the diagonal derivative gain matrix and $\dot{q}_d$ is the desired velocity.

When multiple strategies are active, a combined Jacobian, J_{sjs}, is formed by concatenating all $J_{sjs(i)}$, and the combined unit-mass torque vector, Γ_{sjs}^*, is formed by concatenating all $v_{s(i)}^T \Gamma_{sjs(i)}^*$. The total torque is then computed:

$$\Gamma = \Gamma_{ns} + \Gamma_{sjs}, \quad \Gamma_{ns} = J_{ns}^T F_{ns}, \quad \Gamma_{sjs} = J_{sjs}^T \Lambda_{sjs} \Gamma_{sjs}^* \tag{14}$$

2.2.2 Type 1 Strategy

The strategy controls the singular value derivative through joint velocity to achieve task motion along the singular direction:

$$\nabla_q \sigma_{s(i)} = u_{s(i)}^T H(q) v_{s(i)}, \quad \dot{\sigma}_{s(i)} = (\nabla_q \sigma_{s(i)})^T \dot{q}, \quad \begin{cases} \text{Toward} & \text{if } \dot{\sigma}_{s(i)} < 0 \\ \text{Away} & \text{if } \dot{\sigma}_{s(i)} > 0 \end{cases} \tag{15}$$

Control toward or away from the singularity depends on whether the task intends to approach or retract from it. This is determined by replacing the joint velocity in $\dot{\sigma}_{s(i)}$ with the joint acceleration induced by control torques in the singular space, $\ddot{q}_{s(i)} = A^{-1} \Gamma_{s(i)|ns}$, which represents the instantaneous joint velocity from rest. Then, the corresponding velocity is commanded:

$$\dot{q}_d = (v_{max} \eta_i) \left(\frac{\nabla_q \sigma_{s(i)}}{\|\nabla_q \sigma_{s(i)}\|} \right) \text{sign}(c_i), \quad c_i = \left(\frac{\nabla_q \sigma_{s(i)}}{\|\nabla_q \sigma_{s(i)}\|} \right)^T \frac{\ddot{q}_{s(i)}}{\|\ddot{q}_{s(i)}\|} \tag{16}$$

where $v_{\max}$ is a maximum velocity parameter, η_i scales the maximum velocity, and $\text{sign}(c_i)$ determines the direction along the singular value gradient in order to approach or retract from the singularity based on the task.

Toward the singularity, the velocity scaling factor η_i depends on the singular eigenvalue (λ_i), the singular control force ($F_{s(i)}^* = u_{s(i)}^T F^*$), and the

alignment of the task with the singular value gradient (c_i):

$$\eta_i = \min_{k\in\{1,2,3\}} w_{\beta(k)}(x_k), \qquad x_k = \begin{bmatrix} \lambda_i/\lambda_{min} \\ |F^*_{s(i)}|/F_{max} \\ |c_i| \end{bmatrix}, \quad w_\beta(x) = \frac{1-e^{-\beta \sin(\frac{\pi}{2}x)}}{1-e^{-\beta}} \tag{17}$$

Here, $w_\beta(x)$ is a smooth blending function between $[0, 1]$ with x clamped to $[0, 1]$, with a blending coefficient $\beta(k)$ for each x_k term. The first term in x_k reduces velocity as $\lambda_i \to 0$ to prevent high approach speeds very close to the singularity. The second term reduces velocity as $F^*_{s(i)} \to 0$ to stop motion at zero control force. The final term reduces velocity when the task does not strongly move along the singular value gradient direction.

Away from singularity, η_i uses the last two terms:

$$\eta_i = \min\Big(w_{\beta(2)}\left(\frac{|F^*_{s(i)}|}{F_{max}}\right), w_{\beta(3)}(|c_i|)\Big) \tag{18}$$

At singularity, any motion in $v_{s(i)}$ moves the task away from it. Thus, to accurately determine if the task intends to retract from the singularity, the direction away from the singularity in singular task space is computed with virtual displacements along $\pm\nabla_q\sigma_{s(i)}$. The alignment of this direction with $F^*_{s(i)}$ indicates desired motion away from the singularity. The posture after crossing the singularity is controlled by selecting the sign of $\nabla_q\sigma_{s(i)}$ in the desired velocity.

2.2.3 Type 2 Strategy

The strategy generates motion in the singular joint space to rotate the singular direction away from the task control direction, causing the task to be controlled in the non-singular directions. The desired velocity is

$$\dot{q}_d = v_{max}\, w_{\beta(4)}\left(\left|u^T_{s(i)}\frac{F^*}{\|F^*\|}\right|\right)\frac{\ddot{q}_{s(i)}}{\|\ddot{q}_{s(i)}\|} \tag{19}$$

The desired velocity scales based on task alignment with the singular direction. The joint velocity direction follows the induced joint acceleration direction from the singular task to ensure joint motion consistency.

2.2.4 Exit Transition

Exiting the neighborhood can cause discontinuities due to accumulated tracking error. To address this, task control switches to goal-position velocity-

saturated control [5], with the saturation velocity V_{max} starting at the exit velocity V_{exit} and ramped to a default velocity V_{def} at rate $\dot{V}_{ramp}$:

$$V_{max}^{(k+1)} = V_{max}^{(k)} + \text{sign}\Big(V_{def} - V_{max}^{(k)}\Big) \min\Big(\dot{V}_{ramp}\,\Delta t,\ \Big|V_{def} - V_{max}^{(k)}\Big|\Big) \quad (20)$$

Once the tracking error is within tolerance, task control returns to normal.

3 Experiments

Experiments were conducted on a physical Franka Panda (Type 1 and 2 singularities) and a simulated PUMA (simultaneous Type 1 and 2). A 6 DOF end-effector task was controlled with 1 kHz torque commands using OpenSai. Results show accurate classifications and smooth tracking through singularity neighborhoods.

3.0.1 Type 1 Singularity

The task commanded linear motions in X, ±Y, and Z from a nominal posture while maintaining its initial orientation, where the goal positions lie outside the reachable workspace. The resulting outstretched postures, shown in Fig. 2, arise from the robot reaching toward the singularity due to the strategy. The figure also shows the task trajectory, the three smallest eigenvalues of Λ^{-1}, and joint 4 torque with and without the strategy, demonstrating smooth performance through the singularity neighborhood.

3.0.2 Type 2 Singularity

Starting from an initial posture (Fig. 3), the task was commanded to follow a sinusoidal trajectory in ±Y while maintaining its initial orientation. In this configuration, maintaining the initial orientation while moving in ±Y induces a Type 2 singularity. The trajectory passes through the singularity, and enters and exits the neighborhood to demonstrate smooth performance. The strategy rotates the joints in the singular joint space to shift the singular direction away from the control direction, enabling accurate tracking.

3.0.3 Type 1 and 2 Singularities

To evaluate multi-singularity handling, the PUMA robot was simulated starting from a configuration with one Type 1 and one Type 2 singularity at

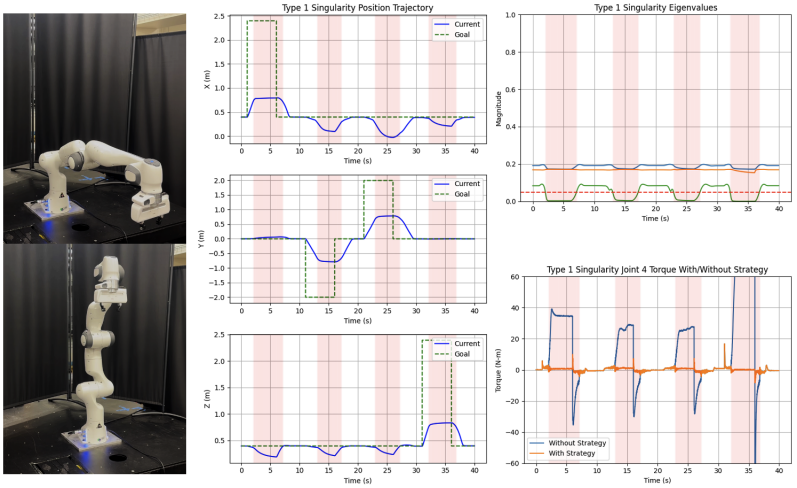

Fig. 2 The left images show two outstretched postures from the experiment, resulting from the Type 1 strategy producing full arm extension. The middle plot shows the position trajectory with steady-state error from attempting to reach a goal outside the workspace. The top right plot shows the three smallest eigenvalues; red regions indicate active singularity strategy (dashed red line is λ_{min}), and the green curve, denoting the smallest eigenvalue, shows the robot reaching and retracting from the singularity. The bottom right plot shows the joint 4 torque, which spikes without the strategy.

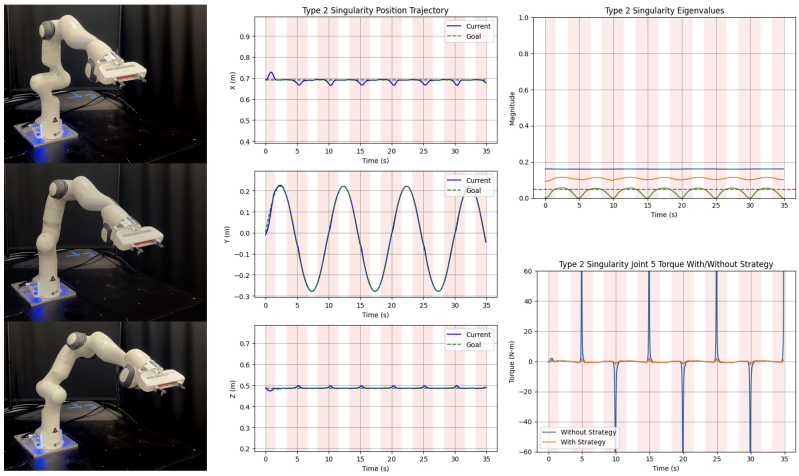

Fig. 3 The left images show the initial posture (top) at a Type 2 singularity, and postures after leaving the singularity (bottom). The middle plot shows accurate tracking through the singularity, with initial and periodic errors from alignment between the singular and control directions, followed by accurate tracking as the strategy shifts the singular direction away. The right plots show the three smallest eigenvalues, indicating traversal through the singularity, and joint 5 torque, which spikes without the strategy.

degenerate, zero singular values (Fig. 4). The results demonstrate accurate classifications and smooth task control due to the strategies.

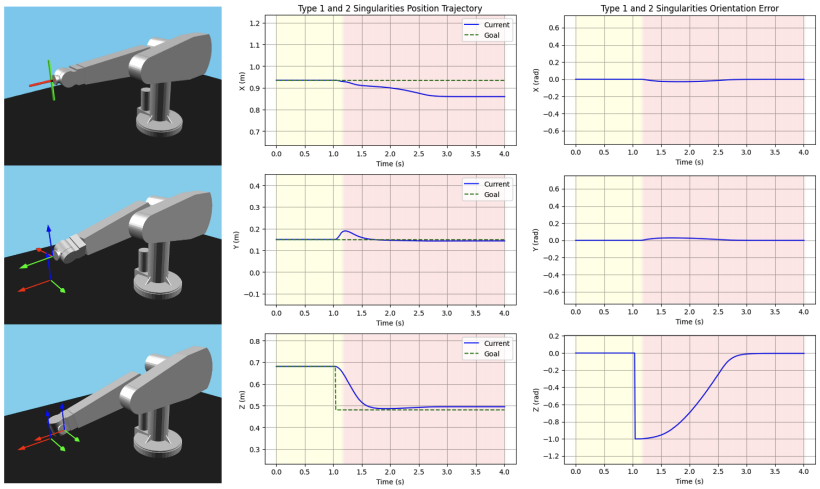

Fig. 4 The left images show Type 1 (red) and Type 2 (green) singularities in the initial posture, and additional postures during control to a goal pose. The plots show smooth task motion in the singularity neighborhood; position converges to a steady-state error from trying to reach a goal position outside of the workspace (Type 1 strategy leading to full arm extension) and orientation error converges to zero (Type 2 strategy). Yellow indicates that both Type 1 and 2 strategies are active, and red indicates only Type 1.

References

1. Y. Nakamura and H. Hanafusa, "Inverse kinematic solutions with singularity robustness for robot manipulator control," *ASME J. Dyn. Syst., Meas., Control*, vol. 108, no. 3, pp. 163–171, 1986.
2. J. M. Hollerbach and K. C. Suh, "Redundancy resolution of manipulators through torque optimization," in *Proc. IEEE Int. Conf. Robot. Autom.*, 1985, pp. 1016–1021.
3. S. Chiaverini, B. Siciliano, and O. Egeland, "Controlling a 6-DOF robot manipulator near kinematic singularities," in *Proc. 3rd Int. Symp. Exp. Robot.*, Kyoto, Japan, 1993.
4. Y. Nakamura, *Kinematical studies on the trajectory control of robot manipulators*, Ph.D. thesis, Kyoto Univ., Japan, 1985.
5. O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE J. Robot. Autom.*, vol. 3, no. 1, pp. 43–53, 1987.
6. K.-S. Chang and O. Khatib, "Manipulator control at kinematic singularities: A dynamically consistent strategy," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Pittsburgh, PA, 1995, pp. 84–88.

7. O. Khatib, M. Jorda, J. Park, L. Sentis, and S.-Y. Chung, "Constraint-consistent task-oriented whole-body robot formulation," *Int. J. Robot. Res.*, vol. 41, no. 13–14, pp. 1079–1098, 2022.
8. H. Han and J. Park, "Robot control near singularity and joint limits using continuous task transition," *Int. J. Adv. Robot. Syst.*, vol. 10, no. 10, 2013.
9. V. Kurtz, P. M. Wensing, and H. Lin, "Control barrier functions for singularity avoidance in passivity-based manipulator control," in *Proc. IEEE Conf. Decis. Control (CDC)*, Austin, TX, 2021, pp. 6125–6130.
10. L. Petrović, F. Marić, I. Marković, J. Kelly, and I. Petrović, "Trajectory optimization with geometry-aware singularity avoidance for robot motion planning," in *Proc. Int. Conf. Control, Autom. Syst. (ICCAS)*, Jeju, Korea, 2021, pp. 1760–1765.
11. J. Seng, K. A. O'Neil, and Y. C. Chen, "Escapability of singular configurations for redundant manipulators via self-motion," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Pittsburgh, PA, 1995, vol. 3, pp. 78–83, doi:10.1109/IROS.1995.525865.
12. M. J. D. Powell, "A direct search optimization method that models the objective and constraint functions by linear interpolation," in *Proc. 1994 Cambridge Workshop on Applied Math. Programming and Modelling*, Cambridge Univ. Press, 1994, pp. 51–67.