2021 I-RIM Conference
October 8-10, Rome, Italy
ISBN: 9788894580525
DOI:10.5281/zenodo.5900613

Motion Control of Redundant Robots with Generalised Inequality Constraints

Amirhossein Kazemipour* Maram Khatib* Khaled Al Khudir** Alessandro De Luca*

***Abstract*—We present an improved version of the Saturation in the Null Space (SNS) algorithm for redundancy resolution at the velocity level. In addition to hard bounds on joint space motion, we consider also Cartesian box constraints that cannot be violated at any time. The modified algorithm combines all bounds into a single augmented generalised vector and gives equal, highest priority to all inequality constraints. When needed, feasibility of the original task is enforced by the SNS task scaling procedure. Simulation results are reported for a 6R planar robot.**

***Index Terms*—Motion control, Redundant robots, Inequality constraints, Hard limits.**

I. Introduction

Given a m-dimensional primary task to be performed by a robot with n joints, with $n > m$ (redundancy), a standard method to prevent violation of joint/Cartesian inequalities during motion is to resort to some form of artificial potentials [1], pushing away from their limits the joints and the control points on the robot body [2]. However, this method is highly parameter-dependent and may introduce oscillations when activating/deactivating the avoidance task in proximity of the bounds [3]. To milden such undesired behavior, the null-space projection term or the activation function can be designed in an incremental way [4], [5]. Nonetheless, selection of appropriate gains is still needed. Moreover, in case of multiple tasks, incorporating the avoidance behavior in the original Stack of Tasks (SoT) will assign different priorities to each single constraint [4]–[7].

Other numerical approaches incorporate joint space and Cartesian motion limits as inequality constraints using parameter-free optimization, such as Quadratic Programming (QP) [8], [9]. However, these methods are computationally slower than analytical solutions and do not lead to realizable solutions when the original task(s) is not compatible with the set of inequality constraints. The SNS algorithm introduced in [10] links QP to the SoT approach and overcomes these challenges. In the original paper, joint motion limits were considered as hard bounds (i.e., that cannot be relaxed in a least-square sense) and treated out of the SoT. On the other hand, Cartesian (avoidance) constraints were not handled as hard bounds. In [11], an approach has been proposed to include joint and Cartesian inequality constraints in the SoT for torque-controlled manipulators. However, joint limits are always pre-assigned the highest priority over all other constraints. Moreover, all violated constraints are set to their limits at the saturation level, as opposed to what happens in [10]. This is neither necessary nor optimal, and will often lead to high-frequency oscillations at the level of commands.

In this paper, we propose several improvements to the SNS algorithm at the velocity level introduced in [10]. First, both joint and Cartesian inequality (box) constraints are treated as hard bounds. Second, all inequalities are assigned the same priority, i.e., enforced anyway independently of the *end-effector* (EE) task (or simply considered out of the SoT, in case of multiple tasks). Finally, the modifications are made so as to preserve the automatic optimal task scaling of the original SNS approach, which relaxes the primary task (keeping its geometric direction) only when no feasible solution would exist, while guaranteeing satisfaction of all inequality constraints. The resulting algorithm does not need parameter tuning and is faster than QP solvers.

II. Methodology

A. Generalised constraints

Consider a robot with n joints and r generic Cartesian control points distributed on the robot body, each of dimension $d_i \in \{1,2,3\}$, $i = 1,\dots,r$. We define an augmented vector

$$\boldsymbol{a} = \begin{pmatrix} \boldsymbol{q}^T & \boldsymbol{p}_{cp,1}^T & \boldsymbol{p}_{cp,2}^T & \cdots & \boldsymbol{p}_{cp,r}^T \end{pmatrix}^T, \tag{1}$$

where $\boldsymbol{q} \in \mathbb{R}^n$ denotes the joint variables and $\boldsymbol{p}_{cp,i} \in \mathbb{R}^{d_i}$ is the position of the i-th control point, $i = 1,\dots,r$. The joint variables as well as the Cartesian control points will have some desired motion restrictions. Accordingly, we define the augmented matrix

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{I} & \boldsymbol{J}_{cp,1}^T & \boldsymbol{J}_{cp,2}^T & \cdots & \boldsymbol{J}_{cp,r}^T \end{pmatrix}^T, \tag{2}$$

where $\boldsymbol{I} \in \mathbb{R}^{n\times n}$ is the identity matrix and $\boldsymbol{J}_{cp,i} \in \mathbb{R}^{d_i\times n}$ is the Jacobian matrix of the i-th control point. Define the position and velocity limits for each joint, $j = 1,\dots,n$, as

$$Q_j^{min} \le q_j \le Q_j^{max}, \qquad V_j^{min} \le \dot{q}_j \le V_j^{max}, \tag{3}$$

and the limits for each control point, $i = 1,\dots,r$, as

$$\boldsymbol{P}_{cp,i}^{min} \le \boldsymbol{p}_{cp,i} \le \boldsymbol{P}_{cp,i}^{max}, \qquad \boldsymbol{V}_{cp,i}^{min} \le \dot{\boldsymbol{p}}_{cp,i} \le \boldsymbol{V}_{cp,i}^{max}. \tag{4}$$

At a generic time instant, the box constraints for the velocity of each component of (1) are given by

$$\begin{aligned} \dot{Q}_{min,j} &= \max\left\{\frac{Q_j^{min} - q_j}{T}, V_j^{min}\right\}, \\ \dot{Q}_{max,j} &= \min\left\{\frac{Q_j^{max} - q_j}{T}, V_j^{max}\right\}, \end{aligned} \tag{5}$$

*Dipartimento di Ingegneria Informatica, Automatica e Gestionale, Sapienza Università di Roma, Via Ariosto 25, 00185 Roma, Italy. Emails: amrkzp@gmail.com, khatib@diag.uniroma1.it, deluca@diag.uniroma1.it.

**School of Mechanical, Aerospace and Automotive Engineering, Coventry University, CV1 5FB Coventry, UK. Email: khaled.alkhudir@coventry.ac.uk.

Algorithm 1 SNS with generalised inequality constraints

```
    $\dot{q}_N \leftarrow 0,\ s^* \leftarrow 0,\ P \leftarrow I,\ A_{lim} \leftarrow$ null, $\dot{a}_N \leftarrow$ null
 2: repeat
        limits_violated ← FALSE
 4:     $\dot{q} \leftarrow \dot{q}_N + (J\,P)^{\#}\,(\dot{x} - J\,\dot{q}_N)$
        $\dot{a} \leftarrow A\,\dot{q}$
 6:     if $\exists\, h \in [1 : n + \Sigma_1^r d_i] : (\dot{a}_h < b_{min,h}) \vee (\dot{a}_h > b_{max,h})$ then
            limits_violated ← TRUE
 8:         $\alpha \leftarrow A\,(J\,P)^{\#}\,\dot{x}$
            $\beta \leftarrow \dot{a} - \alpha$
10:         getTaskScalingFactor($\alpha, \beta$)
            if {task scaling factor} $> s^*$ then
12:             $s^* \leftarrow$ {task scaling factor}
                $\dot{q}_N^* \leftarrow \dot{q}_N,\ P^* \leftarrow P$
14:         end if
            $k \leftarrow$ {the most critical constraint}
16:         $A_{lim} \leftarrow$ concatenate($A_{lim}, A_k$)
            $\dot{a}_N \leftarrow \begin{cases} \text{concatenate}(\dot{a}_N, b_{max,k}) & \text{if } (\dot{a}_h > b_{max,k}) \\ \text{concatenate}(\dot{a}_N, b_{min,k}) & \text{if } (\dot{a}_h < b_{min,k}) \end{cases}$
18:         $P \leftarrow I - (A_{lim})^{\#}\,(A_{lim})$
            if rank($JP$) $< m$ then
20:             $\dot{q} \leftarrow \dot{q}_N^* + (J\,P^*)^{\#}\,(s^*\dot{x} - J\,\dot{q}_N^*)$
                limits_violated ← FALSE
22:         end if
        end if
24:     $\dot{q}_N \leftarrow (A_{lim})^{\#}\,\dot{a}_N$
    until limits_violated = TRUE
26: $\dot{q}_{SNS} \leftarrow \dot{q}$
```

and

$$\dot{P}_{cp,i}^{min} = \max\left\{\frac{P_{cp,i}^{min} - p_{cp,i}}{T}, V_{cp,i}^{min}\right\},$$
$$\dot{P}_{cp,i}^{max} = \min\left\{\frac{P_{cp,i}^{max} - p_{cp,i}}{T}, V_{cp,i}^{max}\right\}, \tag{6}$$

where T is the sampling time. Accordingly, the generalised inequality constraints can be written as the augmentation of joint and Cartesian bounds in (5) and (6) as

$$B_{min} = \begin{pmatrix} \dot{Q}_{min,1}, & \dots & \dot{Q}_{min,n}, & \dot{P}_{cp,1}^{min}, & \dots & \dot{P}_{cp,r}^{min} \end{pmatrix}^T,$$
$$B_{max} = \begin{pmatrix} \dot{Q}_{max,1}, & \dots & \dot{Q}_{max,n}, & \dot{P}_{cp,1}^{max}, & \dots & \dot{P}_{cp,r}^{max} \end{pmatrix}^T. \tag{7}$$

B. Modified SNS algorithm

Consider a single EE velocity task $\dot{x}_d \in \mathbb{R}^m$, with $n > m$, and its Jacobian matrix $J \in \mathbb{R}^{m\times n}$, to be achieved under the generalised hard constraints in (7). The pseudo-code of the modified SNS method is presented in Algorithm 1. If the Cartesian inequality limits in (4) are discarded, the augmented matrix (2) becomes $A = I$ and it is easy to show that Algorithm 1 simplifies to the original SNS algorithm in [10]. Note that at line 15, the most critical constraint corresponds to the smallest scaling factor s_k over all constraints. Also, at line 19, when there is no way to perform the desired task under the hard inequality constrains, we apply an optimal task scaling factor as computed by Algorithm 2, similar to [10].

III. Results

Verification for the proposed algorithm is done through MATLAB simulations, using a 6R planar robot arm ($n = 6$) and considering joint and Cartesian inequality constraints. The

Algorithm 2 Optimal task scaling factor

```
    function GETTASKSCALINGFACTOR($\alpha, \beta$)
 2:     for $h \leftarrow 1 : n + \Sigma_1^r d_i$ do
            $L_h \leftarrow b_{min,h} - \beta_h$
 4:         $U_h \leftarrow b_{max,h} - \beta_h$
            if $\alpha_h < 0 \wedge L_i < 0$ then
 6:             if $\alpha_h < L_h$ then
                    $s_h \leftarrow L_h/\alpha_h$
 8:             else
                    $s_h \leftarrow 1$
10:             end if
            else if $\alpha_h > 0 \wedge U_h > 0$ then
12:             if $\alpha_h > U_h$ then
                    $s_h \leftarrow U_h/\alpha_h$
14:             else
                    $s_h \leftarrow 1$
16:             end if
            else
18:             $s_h \leftarrow 0$
            end if
20:     end for
    return $s$
22: end function
```

EE is required to track a 2D linear path ($m = 2$), see Fig. 1. Therefore, the primary EE velocity task is defined as

$$\dot{x} = \dot{x}_d + K_p(x_d - x_{ee}), \tag{8}$$

with the control gain matrix $K_p = \text{diag}\{50, 50\}$ and the EE position x_{ee} computed by the direct kinematics. The sampling time is $T = 1$ [ms] and the initial robot configuration (in [rad]) is chosen as

$$q_0 = \begin{pmatrix} \frac{\pi}{6} & -\frac{\pi}{6} & -\frac{\pi}{6} & \frac{\pi}{3} & -\frac{\pi}{6} & -\frac{\pi}{6} \end{pmatrix}^T. \tag{9}$$

The limits (3) are equal and symmetric for all joints:

$$Q_j^{max} = -Q_j^{min} = \frac{\pi}{2}\,\text{[rad]}, \quad V_j^{max} = -V_j^{min} = 1\,\text{[rad/s]}. \tag{10}$$

We consider $r = 5$ control points (each with $d_i = 1$) along the robot body, located at the joints $j = 2, \dots, 6$. The corresponding limits (4) are equal for all points, and imposed only over the y-direction:

$$P_{cp,i}^{max,y} = 1\,\text{[m]}, \quad P_{cp,i}^{min,y} = -1.1\,\text{[m]},$$
$$V_{cp,i}^{max,y} = -V_{cp,i}^{min,y} = 0.8\,\text{[m/s]}. \tag{11}$$

The EE starts the motion very close to the desired path. As shown in Fig. 2, the positional error converges immediately and remains zero along the whole task. Accordingly, the task scaling is active ($s < 1$) only for few milliseconds at beginning, to comply with the saturated joint and Cartesian velocity limits due to the initial error recovery —see Figs. 3 and 4. Later, the robot is able to perform the complete task perfectly while satisfying all inequality constraints (many of them in saturation). The few discontinuities in the commanded joint velocity in Fig. 3 can be addressed by extending the algorithm to the acceleration level and including suitable joint acceleration limits in the set of constraints.

IV. Conclusion

We have proposed major enhancements to the basic SNS algorithm at the velocity level for redundant robots. Cartesian

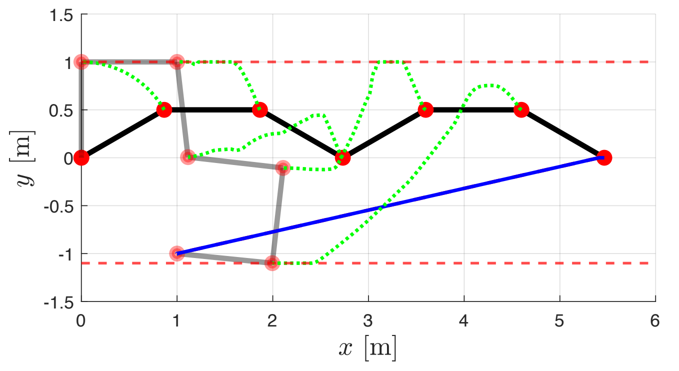
Fig. 1: Initial (black) and final (gray) configurations of the 6R planar arm. The red circles represent the robot joints (and the EE tip). The desired EE path is the blue line, to be traced from right to left. The dashed red lines are the Cartesian position limits. The dashed green lines show the path of control points during task execution.

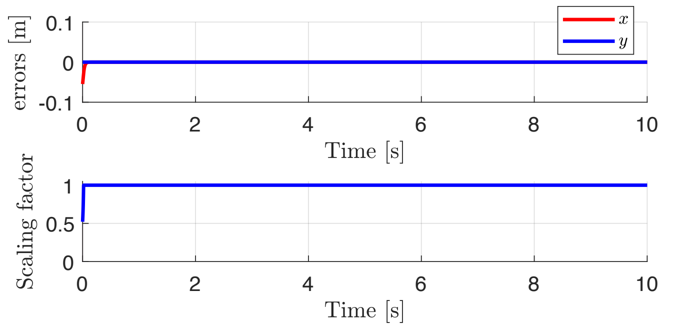
Fig. 2: EE positional errors and related task scaling factor.

inequality constraints are included and treated as hard limits, while preserving all the nice features of the original method. The modified algorithm can be extended to include multiple tasks having different priorities. Moreover, it can be implemented also at the acceleration level, which is beneficial for involving dynamic properties in the resolution of redundancy and is suitable for torque-controlled systems.

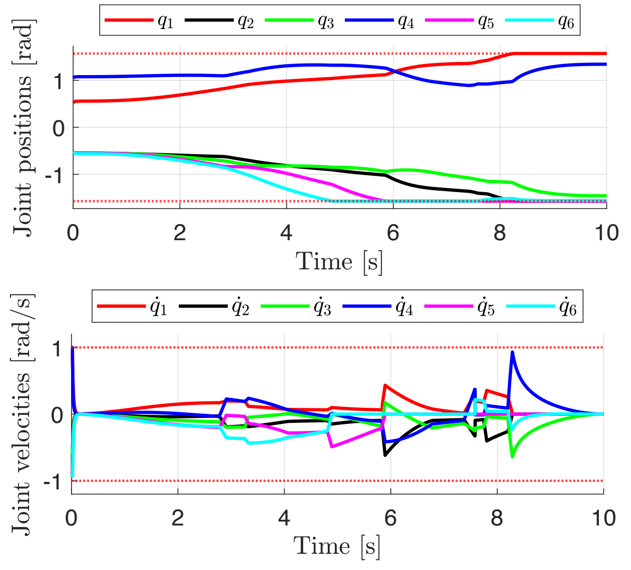
Fig. 3: Evolution of the joints during task execution. The dotted red lines are the bounds on the joint motion.

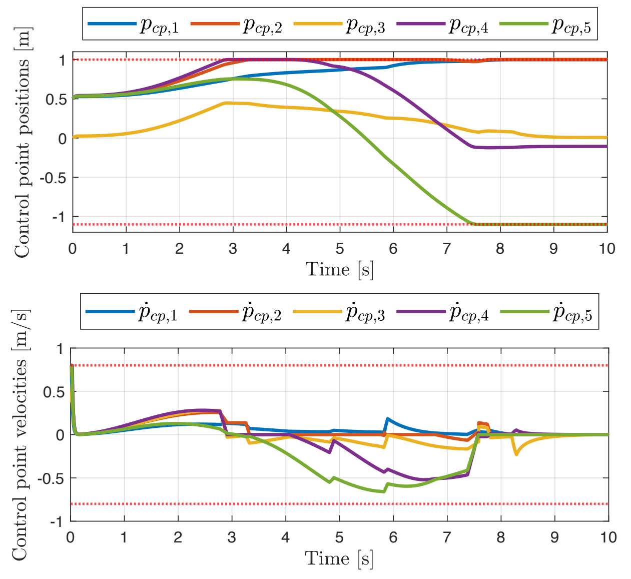
Fig. 4: Evolution of the control points along the y-direction. The dotted red lines are the Cartesian bounds on the motion of the control points.

References

[1] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Autonomous Robot Vehicles* (I. Cox and G. Wilfong, eds.), pp. 396–404, Springer, 1986.

[2] M. Khatib, K. Al Khudir, and A. De Luca, "Task priority matrix at the acceleration level: Collision avoidance under relaxed constraints," *IEEE Robotics and Automation Lett.*, vol. 5, no. 3, pp. 4970–4977, 2020.

[3] M. Khatib, K. Al Khudir, and A. De Luca, "Task priority matrix under hard joint constraints," in *Proc. 2nd Italian Conf. on Robotics and Intelligent Machines*, pp. 173–174, 2020.

[4] N. Mansard, O. Khatib, and A. Kheddar, "A unified approach to integrate unilateral constraints in the stack of tasks," *IEEE Trans. on Robotics*, vol. 25, no. 3, pp. 670–685, 2009.

[5] E. Simetti and G. Casalino, "A novel practical technique to integrate inequality control objectives and task transitions in priority based control," *J. of Intelligent & Robotic Systems*, vol. 84, no. 1, pp. 877–902, 2016.

[6] L. Sentis and O. Khatib, "Synthesis of whole-body behaviors through hierarchical control of behavioral primitives," *Int. J. of Humanoid Robotics*, vol. 2, no. 4, pp. 505–518, 2005.

[7] L. Sentis and O. Khatib, "A whole-body control framework for humanoids operating in human environments," in *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 2641–2648, 2006.

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

[9] E. M. Hoffman, A. Laurenzi, L. Muratore, N. G. Tsagarakis, and D. G. Caldwell, "Multi-priority Cartesian impedance control based on quadratic programming optimization," in *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 309–315, 2018.

[10] F. Flacco, A. De Luca, and O. Khatib, "Control of redundant robots under hard joint constraints: Saturation in the null space," *IEEE Trans. on Robotics*, vol. 31, no. 3, pp. 637–654, 2015.

[11] J. D. M. Osorio, F. Allmendinger, M. D. Fiore, U. E. Zimmermann, and T. Ortmaier, "Physical human-robot interaction under joint and cartesian constraints," in *Proc. 19th Int. Conf. on Advanced Robotics*, pp. 185–191, 2019.