Stability of the Tracking Problem with Task-Priority Inverse Kinematics ⋆

Mathias Hauan Arbo * **Jan Tommy Gravdahl** **

* *Department of Engineering Cybernetics, Norwegian University of Science and Technology (NTNU), Trondheim, Norway (e-mail: mathias.arbo@ntnu.no).*
** *Department of Engineering Cybernetics, Norwegian University of Science and Technology (NTNU), Trondheim, Norway (e-mail: jan.tommy.gravdahl@ntnu.no).*

Abstract: The multiple task-priority inverse kinematics framework using the Moore-Penrose pseudoinverse has been shown to be asymptotically stable for the regulation problem with certain conditions on the tasks. In this article we present a theorem that extends this to the tracking problem by including an additional criterion that we term *fully represented in the null-space*. We show the effect of this on a simulation with a snake-like robot manipulator with 30 links for 3 compatible tasks, and an example of 2 tasks that are compatible as a regulation problem but incompatible as a tracking problem. As the tracking problem is more affected by the linearization assumption, we also include an example showing that the effect of linearization can be detrimental during tracking.

Keywords: Inverse kinematic problem, Stability analysis, Control (closed-loop), Robot kinematics, Industrial robots, Robot arm

1. INTRODUCTION

From humanoid service robots to industrial manipulators, we often want the robot to follow a reference trajectory. The reference is represented in a task space (f.ex. in Cartesian space) and finding the appropriate robot-centric control setpoints to converge to the reference is referred to as the inverse kinematics problem. Inverse kinematics is a classical problem in robotics, and is fundamental to highly redundant robots such as humanoid or snake robots.

Sciavicco and Siciliano (1986) and Das et al. (1988) present methods of finding joint speed setpoints to achieve a given inverse kinematics. This is commonly called the closed-loop inverse kinematics (CLIK), where the joint speeds are found by inverting differential kinematics. By design, the joint speeds are chosen such that the task errors converge exponentially.

For robots where there are multiple tasks to be achieved, and the robot is redundant with respect to the tasks, Chiaverini (1997) shows that tasks can be combined in priority by placing lower priority tasks in the null-space of the higher priority tasks. Antonelli (2009) presents a set of criteria on the tasks to ensure that we have asymptotic stability of the regulation problem (when the robot is to move to a reference point).

Multiple-task inverse kinematics can be solved in many different ways. We consider the use of pseudo-inverses and null-space operators, but others such as Aertbeliën and Schutter (2014) solves the problem by using a quadratic program. Falco and Natale (2011) consider the discretized version of the problem, and gives a stability proof of the regulation problem.

In this article we consider the tracking problem where the robot is to follow a reference trajectory rather than the regulation problem where it moves to a point. We present a proof with criteria on the tasks to ensure that the tracking problem exhibits asymptotic stability. We give three examples on a 30 link planar snake robot. The first example emphasizes the behavior when we have compatible tasks. The second is a minimal example of tasks that are compatible as a regulation problem but incompatible as a tracking problem. The minimal example can easily happen if a user makes a mistake when designing the tasks. The final example shows how the tracking problem is more sensitive to how long the joint speed setpoint is applied to the robot than the regulation problem.

2. STABILITY

2.1 Description of the Moore-Penrose Pseudoinverse

The Moore-Penrose pseudoinverse, $\boldsymbol{A}^\dagger$ is defined for $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ where $\boldsymbol{A}$ not necessarily has full rank, and it satisfies the following conditions:

$$\boldsymbol{A}\boldsymbol{A}^\dagger\boldsymbol{A} = \boldsymbol{A}, \tag{1}$$

$$\boldsymbol{A}^\dagger\boldsymbol{A}\boldsymbol{A}^\dagger = \boldsymbol{A}^\dagger, \tag{2}$$

$$\left(\boldsymbol{A}\boldsymbol{A}^\dagger\right)^T = \boldsymbol{A}\boldsymbol{A}^\dagger, \tag{3}$$

$$\left(\boldsymbol{A}^\dagger\boldsymbol{A}\right)^T = \boldsymbol{A}^\dagger\boldsymbol{A} \tag{4}$$

⋆ The work reported in this paper was based on activities within centre for research based innovation SFI Manufacturing in Norway, and is partially funded by the Research Council of Norway under contract number 237900.

and if $\boldsymbol{A}$ has full rank, we have

$$\boldsymbol{A}\boldsymbol{A}^{\dagger} = \boldsymbol{I}_m. \tag{5}$$

2.2 Single task stability

We consider a general robotic system with n degrees of freedom whose configuration is described by the joint coordinates $\boldsymbol{q} = [q_1, q_2, \ldots, q_n] \in \mathbb{R}^n$. The robot is controlled by the speed of the joints $\dot{\boldsymbol{q}}_{\text{des}}$ and we assume $\dot{\boldsymbol{q}}_{\text{des}}(t) = \dot{\boldsymbol{q}}(t)$. We define a task error, $\tilde{\boldsymbol{\sigma}} : \mathbb{R} \times \mathbb{R}^n \to \mathbb{R}^m$, that should be stabilized to zero. It is a function of time, joint coordinates, and m is the dimension of the task. A typical regulation problem would be:

$$\tilde{\boldsymbol{\sigma}} = \boldsymbol{p} - \boldsymbol{f}_{\boldsymbol{p}}(\boldsymbol{q}) \tag{6}$$

where $\boldsymbol{f}_{\boldsymbol{p}}$ is the forward kinematics, and $\boldsymbol{p}$ is the desired goal state. A typical tracking problem would be:

$$\tilde{\boldsymbol{\sigma}} = \boldsymbol{p}_{\text{ref}}(t) - \boldsymbol{f}_{\boldsymbol{p}}(\boldsymbol{q}) \tag{7}$$

where $\boldsymbol{p}_{\text{ref}}(t)$ is a reference signal that the end-effector tracks.

Looking at the total derivative of $\tilde{\boldsymbol{\sigma}}$ we find the task error is related to the control input $\dot{\boldsymbol{q}}$ by

$$\dot{\tilde{\boldsymbol{\sigma}}}(t, \boldsymbol{q}) = \frac{\partial \tilde{\boldsymbol{\sigma}}}{\partial t}(t, \boldsymbol{q}) + \frac{\partial \tilde{\boldsymbol{\sigma}}}{\partial \boldsymbol{q}}(t, \boldsymbol{q})\dot{\boldsymbol{q}}(t) \tag{8}$$

we define $\boldsymbol{J}(\boldsymbol{q}) = \frac{\partial \tilde{\boldsymbol{\sigma}}}{\partial \boldsymbol{q}} \in \mathbb{R}^{m \times n}$ as the configuration-dependent task error Jacobian. For brevity we simply write $\boldsymbol{J}$ and $\tilde{\boldsymbol{\sigma}}$. We essentially assume that we can linearize the system at this time instance and give the robot a $\dot{\boldsymbol{q}}$ setpoint for a short enough time interval that we have not diverged from this linearization.

To ensure exponential stability of the task error, choose

$$\dot{\tilde{\boldsymbol{\sigma}}} \coloneqq -\Lambda \tilde{\boldsymbol{\sigma}}, \tag{9}$$

where $\Lambda > 0$. From (8), we find

$$\boldsymbol{J}\dot{\boldsymbol{q}} = -\frac{\partial \tilde{\boldsymbol{\sigma}}}{\partial t} - \Lambda \tilde{\boldsymbol{\sigma}}. \tag{10}$$

Note that $\frac{\partial \tilde{\boldsymbol{\sigma}}}{\partial t}$ is a feedforward term of the time derivative of the task error. If $\boldsymbol{J}$ has full rank, we can use the Moore-Penrose inverse of $\boldsymbol{J}$ to find the desired $\dot{\boldsymbol{q}}$

$$\dot{\boldsymbol{q}} = -\boldsymbol{J}^{\dagger}\left(\frac{\partial \tilde{\boldsymbol{\sigma}}}{\partial t} + \Lambda \tilde{\boldsymbol{\sigma}}\right). \tag{11}$$

if the system is redundant with respect to the task, i.e. $n > m$, the solution has a null-space operator $\boldsymbol{N} = (\boldsymbol{I}_n - \boldsymbol{J}^{\dagger}\boldsymbol{J})$ such that

$$\dot{\boldsymbol{q}} = -\boldsymbol{J}^{\dagger}\left(\frac{\partial \tilde{\boldsymbol{\sigma}}}{\partial t} + \Lambda \tilde{\boldsymbol{\sigma}}\right) + \boldsymbol{N}\dot{\boldsymbol{q}}_{\text{null}} \tag{12}$$

where $\dot{\boldsymbol{q}}_{\text{null}} \in \mathbb{R}^n$ can be arbitrarily chosen without affecting the stability of the task. This means that tasks are combined in priority by projecting their desired joint speed, (11), into the null-space of the higher-priority tasks.

We use the augmented null-space operator to achieve this. The augmented null-space operator describes the null-space formed by the combination of multiple task error Jacobians. That is, for task errors $\tilde{\boldsymbol{\sigma}}_1$ and $\tilde{\boldsymbol{\sigma}}_2$ the augmented Jacobian is defined as

$$\boldsymbol{J}_{1,2} = \begin{bmatrix} \boldsymbol{J}_1 \\ \boldsymbol{J}_2 \end{bmatrix}, \tag{13}$$

and the augmented null-space operator is defined as

$$\boldsymbol{N}_{1,2} = \left(\boldsymbol{I}_n + \boldsymbol{J}_{1,2}{}^{\dagger}\boldsymbol{J}_{1,2}\right). \tag{14}$$

A useful property of the augmented null-space operator is:

$$\boldsymbol{J}_i\boldsymbol{N}_{1,\ldots,k} = \boldsymbol{0}_{m \times n} \tag{15}$$

with $i \in \{1, \ldots, k\}$. For proof see Moe et al. (2016). The desired joint speed for k tasks, is then:

$$\dot{\boldsymbol{q}} = -\sum_{i=1}^{k} \boldsymbol{N}_{1,i-1} J_1{}^{\dagger}\left(\frac{\partial \tilde{\boldsymbol{\sigma}}_1}{\partial t} + \Lambda_1 \tilde{\boldsymbol{\sigma}}_1\right) \tag{16}$$

where we have defined the shorthand $\boldsymbol{N}_{1,0} = \boldsymbol{I}_n$ to simplify our sum expression.

2.3 Task relation definitions

Antonelli (2009) provides three useful definitions for the relation between tasks. Two tasks with Jacobians $\boldsymbol{J}_i$ and $\boldsymbol{J}_j$ are defined to be *annihilating* if:

$$\boldsymbol{J}_i\boldsymbol{J}_j{}^{\dagger} = \boldsymbol{0}_{m \times m}. \tag{17}$$

They are *annihilating in the null-space* of $\tilde{\boldsymbol{\sigma}}_1, \ldots, \tilde{\boldsymbol{\sigma}}_l$ if

$$\boldsymbol{J}_i\boldsymbol{N}_{1,\ldots,l}J_j^{\dagger} = \boldsymbol{0}_{m \times m}, \tag{18}$$

and the two tasks are *independent* if they are not annihilating and

$$\rho(\boldsymbol{J}_i^{\dagger}) + \rho(\boldsymbol{J}_j^{\dagger}) = \rho\left(\left[\boldsymbol{J}_i{}^{\dagger}, \boldsymbol{J}_j{}^{\dagger}\right]\right) \tag{19}$$

where $\rho(\cdot)$ denotes the rank.

We include an additional definition, a tracking task is *fully represented in the null-space* $\boldsymbol{N}_j$ if

$$\boldsymbol{J}_i\boldsymbol{N}_j\boldsymbol{J}_i^{\dagger} = \boldsymbol{J}_i\boldsymbol{J}_i^{\dagger} \tag{20}$$

which for full rank of $\boldsymbol{J}_i$ gives the identity matrix. This new criteria ensures that the time-varying aspect to track in a task can be followed in the null-space it will operate in.

2.4 Tracking Multiple tasks

Theorem 1. Given $k > 1$ tasks, and that

(A) each task has *full rank*,
(B) task i is *independent* of all tasks $1, \ldots, i-1$,
(C) any tasks i and j with $i > j > 1$ are *annihilating* in the augmented null-space $\boldsymbol{N}_{1,\ldots,j-1}$,
(D) if task i is a tracking task, it is *fully represented* in $\boldsymbol{N}_{1,\ldots,i-1}$

then the task errors are asymptotically stable.

Proof. We have $k > 1$ tasks. For each of these tasks, we desire exponential behavior of the task error derivative

$$\dot{\tilde{\boldsymbol{\sigma}}}_i = \frac{\partial \tilde{\boldsymbol{\sigma}}_i}{\partial t} + \boldsymbol{J}_i\dot{\boldsymbol{q}} \tag{21}$$

$$\dot{\tilde{\boldsymbol{\sigma}}}_i \coloneqq -\Lambda \tilde{\boldsymbol{\sigma}}_i \tag{22}$$

where $i = 1, \ldots, k$. To investigate the stability of all the tasks, we define

$$\overrightarrow{\boldsymbol{\sigma}} = [\tilde{\boldsymbol{\sigma}}_1^T, \ldots, \tilde{\boldsymbol{\sigma}}_k^T]^T \tag{23}$$

$$\frac{\partial \overrightarrow{\boldsymbol{\sigma}}}{\partial t} = [\frac{\partial \tilde{\boldsymbol{\sigma}}_1}{\partial t}^T, \ldots, \frac{\partial \tilde{\boldsymbol{\sigma}}_k}{\partial t}^T]^T \tag{24}$$

Then, by inserting (16) into (21) we obtain the system

$$\dot{\overrightarrow{\boldsymbol{\sigma}}} = - \begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{0}_{m_1 \times m_2} & \dots & \boldsymbol{0}_{m_1 \times m_k} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} & \dots & \boldsymbol{0}_{m_2 \times m_k} \\ \vdots & & \ddots & \\ \boldsymbol{A}_{k1} & \boldsymbol{A}_{k2} & \dots & \boldsymbol{A}_{kk} \end{bmatrix} \overrightarrow{\boldsymbol{\sigma}} + \begin{bmatrix} \boldsymbol{B}_{11} & \boldsymbol{0}_{m_1 \times m_2} & \dots & \boldsymbol{0}_{m_1 \times m_k} \\ \boldsymbol{B}_{21} & \boldsymbol{B}_{22} & \dots & \boldsymbol{0}_{m_2 \times m_k} \\ \vdots & & \ddots & \\ \boldsymbol{B}_{k1} & \boldsymbol{B}_{k2} & \dots & \boldsymbol{B}_{kk} \end{bmatrix} \frac{\partial \overrightarrow{\boldsymbol{\sigma}}}{\partial t} \tag{25}$$

where

$$\boldsymbol{A}_{ij} = \boldsymbol{J}_i \boldsymbol{N}_{1,\dots,j-1} \boldsymbol{J}_j{}^{\dagger} \Lambda_j \tag{26}$$

$$\boldsymbol{B}_{ij} = \begin{cases} \boldsymbol{I} - \boldsymbol{J}_i \boldsymbol{N}_{1,\dots,i-1} \boldsymbol{J}_i{}^{\dagger}, & i = j \\ -\boldsymbol{J}_i \boldsymbol{N}_{1,\dots,j-1} \boldsymbol{J}_j{}^{\dagger}, & i \neq j \end{cases} \tag{27}$$

Note: the upper-trianguluar block terms are zero as a result of (15).

Following the proof in Antonelli (2009), we use the Lyapunov function

$$V = \frac{1}{2} \overrightarrow{\boldsymbol{\sigma}}^T \overrightarrow{\boldsymbol{\sigma}} \tag{28}$$

which is positive for all non-zero $\overrightarrow{\boldsymbol{\sigma}}$. It has a time derivative given by

$$\dot{V} = \overrightarrow{\boldsymbol{\sigma}}^T \dot{\overrightarrow{\boldsymbol{\sigma}}} \tag{29}$$

$$\dot{V} = - \overrightarrow{\boldsymbol{\sigma}}^T \boldsymbol{A} \overrightarrow{\boldsymbol{\sigma}} + \overrightarrow{\boldsymbol{\sigma}}^T \boldsymbol{B} \frac{\partial \overrightarrow{\boldsymbol{\sigma}}}{\partial t} \tag{30}$$

Looking at (26) and (27) we notice that from requirement (C), the off-diagonal elements of $\boldsymbol{A}$ and $\boldsymbol{B}$ are zero matrices of appropriate dimension. From requirement (A) and (D) $\boldsymbol{B}$ disappears entirely, giving

$$\dot{V} = - \overrightarrow{\boldsymbol{\sigma}}^T \boldsymbol{A} \overrightarrow{\boldsymbol{\sigma}}. \tag{31}$$

This means that the tracking problem has been reduced to the regulation problem. As described in Antonelli (2009) the block diagonal elements of $\boldsymbol{A}$ are positive definite from the requirement (B) and (C), and the task errors are asymptotically stable for any $\Lambda_i > 0$ with $i \in 1, \dots, k$. If requirement (D) holds for all tasks, not just the ones that have a time-varying aspect, then we also have $\boldsymbol{A} = \text{diag}(\Lambda_1 \boldsymbol{I}_{m_1}, \Lambda_2 \boldsymbol{I}_{m_2}, \dots, \Lambda_k \boldsymbol{I}_{m_k})$.

3. EXAMPLES

To simplify Jacobian and pseudo-inverse calculation, the following examples were implemented in Python with CasADi (Andersson et al., 2018). CasADi is a symbolic and algorithmic differentiation framework for numeric optimization. The system is simulated with Euler's method and a timestep of 0.01 s.

We consider a highly redundant snake-like manipulator with 30 links of unit length with 30 revolute joints. The snake is rigidly attached at the base and moves in the plane. We have $\boldsymbol{q} = [q_1, q_2, \dots, q_{30}]^T \in \mathbb{R}^{30}$ with each subsequent joint angle defined relative the preceding joint. The forward kinematics to link i is then given by

$$\boldsymbol{f}_{\boldsymbol{p}_i}(\boldsymbol{q}) = \begin{bmatrix} \sum_{j=1}^{i} \cos(\sum_{k=1}^{j} q_k) \\ \sum_{j=1}^{i} \sin(\sum_{k=1}^{j} q_k) \end{bmatrix} \tag{32}$$

and the partial derivatives of these are simply

$$\frac{\partial \boldsymbol{f}_{\boldsymbol{p}_i}}{\partial q_l}(\boldsymbol{q}) = \begin{bmatrix} \sum_{j=l}^{i} -\sin(\sum_{k=1}^{j} q_k) \\ \sum_{j=l}^{i} \cos(\sum_{k=1}^{j} q_k) \end{bmatrix} \tag{33}$$

we denote the relative forward kinematics from link i to link j as $\boldsymbol{f}_{p_{i,j}}(\boldsymbol{q})$.

The first example emphasizes how feedforward of the task error derivative reduces the tracking error for tasks that are compatible. The second example shows two tasks that can be combined in the case of a regulation problem but will cause problems as a tracking problem. The third example shows a problem with discretization that is more evident in tracking tasks than in regulation tasks.

3.1 Example 1 - Effect of Feedforward

We have three tasks forming three task errors:

$$\tilde{\boldsymbol{\sigma}}_1(t, \boldsymbol{q}) = \begin{bmatrix} 2\cos(0.1t) + 10 \\ 2\sin(0.1t) + 10 \end{bmatrix} - \boldsymbol{f}_{\boldsymbol{p}_{20}}(\boldsymbol{q}) \tag{34}$$

$$\tilde{\boldsymbol{\sigma}}_2(t, \boldsymbol{q}) = \begin{bmatrix} \cos(0.2t) + 2 \\ \sin(0.2t) + 2 \end{bmatrix} - \boldsymbol{f}_{\boldsymbol{p}_{25,30}}(\boldsymbol{q}) \tag{35}$$

$$\tilde{\boldsymbol{\sigma}}_3(t, \boldsymbol{q}) = q_{21} + q_{22} + q_{23} - \sin(t) \tag{36}$$

the first task is for the 20th link to follow a circle of radius 2 centered around $(10, 10)$, and the second is for the 30th link to follow a circle of radius 1 centered around $(2, 2)$ relative to the frame on the 25th link. The third tasks is for the sum of joints q_{21}, q_{22} and q_{23} to follow a sinusoidal signal. The tasks are annihilating, and fully represented in the necessary null-spaces, and we use gains $\Lambda_1 = \Lambda_2 = \Lambda_3 = 1$.

In Fig.1 we see the norms of the tasks w.r.t time both with and without feedback. Without feedforward, there is a persistent tracking error in all the tasks. In Fig.2 we see the smallest eigenvalue of $\boldsymbol{A}$ and the Frobenius norm of $\boldsymbol{B}$ w.r.t. time, note that $\boldsymbol{A}$ remains positive definite, and $\boldsymbol{B}$ is zero. As $\boldsymbol{B}$ is zero, the tracking tasks are fully represented in the null-space of the higher-priority tasks, and the task errors are asymptotically stable.

3.2 Example 2 - Incompatible Tasks

A simple mistake during task design is to take what should be one task, and splitting it into two. Here we consider the case where we control the x and y position of the end-effector as two different tasks. This gives the task errors

$$\tilde{\sigma}_x(t, \boldsymbol{q}) = 2\cos(0.1t) + 10 - \sum_{j=1}^{30} \cos(\sum_{k=1}^{j} q_k) \tag{37}$$

$$\tilde{\sigma}_y(t, \boldsymbol{q}) = 2\cos(0.1t) + 10 - \sum_{j=1}^{30} \sin(\sum_{k=1}^{j} q_k) \tag{38}$$

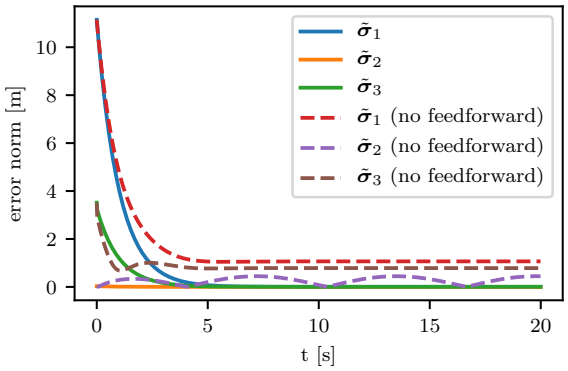

Fig. 1. Norm of task errors for example 1. With compatible tasks, feedforward removes tracking error.

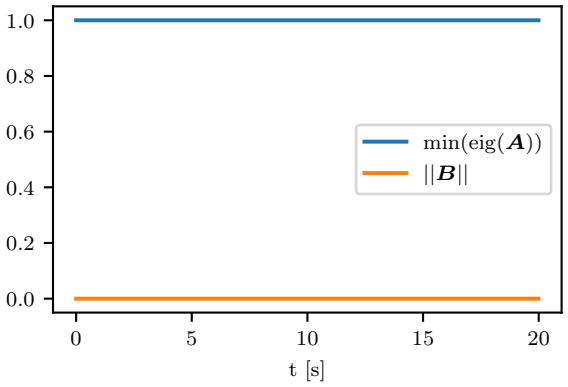

Fig. 2. Minimum eigenvalue of $\boldsymbol{A}$ and Frobenius norm of $\boldsymbol{B}$ for example 1. Only the case with feedforward is given as the no feedforward case is similar.

for which the partial derivatives are

$$\frac{\partial \tilde{\sigma}_x}{\partial q_i} = \sum_{j=i}^{30} \sin(\sum_{k=1}^{j} q_k) \tag{39}$$

$$\frac{\partial \tilde{\sigma}_y}{\partial q_i} = -\sum_{j=i}^{30} \cos(\sum_{k=1}^{j} q_k) \tag{40}$$

In Appendix A we show that as a regulation problem this is asymptotically stable, but not as a tracking problem. In Fig.3 we see the norm of the task errors. The high-priority $\tilde{\sigma}_x$ converges, but $\tilde{\sigma}_y$ does not as it is affected by the time-varying reference signal. In Fig.4 we see that the smallest eigenvalue of $\boldsymbol{A}$ is positive, and that $\boldsymbol{B}$ is non-zero.

3.3 Example 3 - Linearization

CLIK approaches generally assume that $\dot{\boldsymbol{q}} = \dot{\boldsymbol{q}}_{\text{desired}}$, calculates $\dot{\boldsymbol{q}}_{\text{desired}}$, and then applies $\dot{\boldsymbol{q}}_{\text{desired}}$ to the robot for a time interval. This means that the time interval we apply $\dot{\boldsymbol{q}} = \dot{\boldsymbol{q}}_{\text{desired}}$ to the robot should be sufficiently small for the linearization assumption to hold. In Fig.5 we show the norm of the full task error vector in Example 1 for varying timestep sizes. In Fig.6 we show the same tasks but with a constant reference point instead of a reference trajectory.

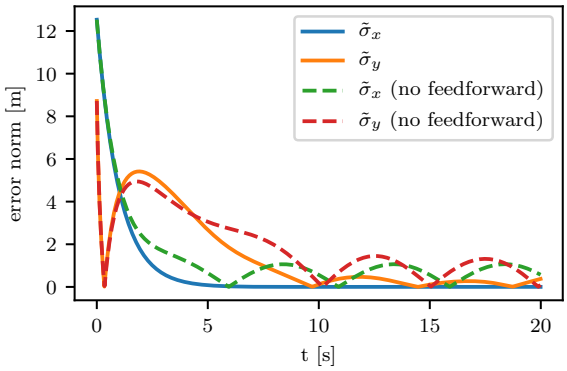

Fig. 3. Norms of the task errors for Example 2. Note that $\tilde{\sigma}_x$ converges but $\tilde{\sigma}_y$ does not.

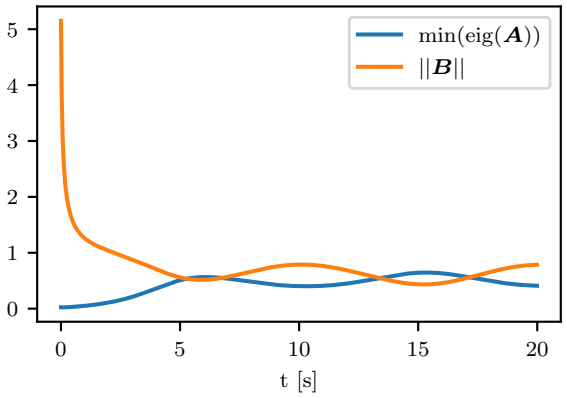

Fig. 4. Minimum eigenvalue of $\boldsymbol{A}$ and Frobenius norm of $\boldsymbol{B}$ for example 2. Only the case with feedforward is given as the no feedforward case is similar. Note that the minimum eigenvalue remains positive, but $\|\boldsymbol{B}\|$ is non-zero.

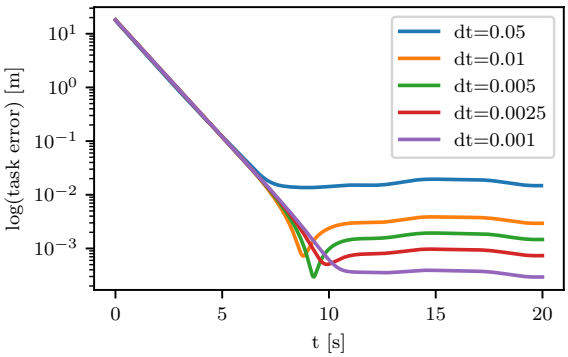

Fig. 5. Norms of the full task error vector from (34)-(36) for varying timestep lengths.

4. CONCLUSION

When performing tracking tasks, it is not as easy to combine tasks as when handling a regulation problem. In this article we presented a theorem showing sufficient conditions on the tasks to ensure asymptotic stability of the task errors for task-priority inverse kinematics for tracking problems. The new requirement when compared

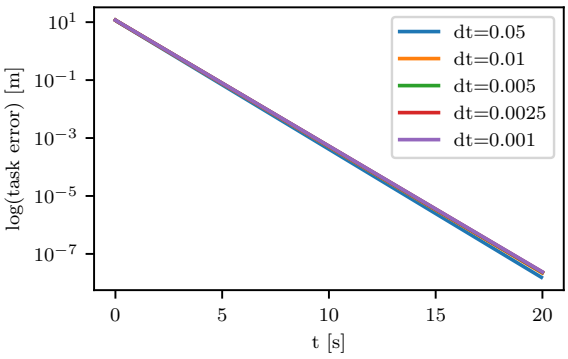

Fig. 6. Norms of the full task error vector from (34)-(36) for varying timestep lengths and a constant $t = 0$.

to the regulation problems is that tracking tasks are fully represented in the null-space of the previous tasks. This was shown with two examples, one showing compatible tasks where the tracking error was removed by the feedforward signal, and one showing a minimal example of incompatible tasks that could arise by simple user mistakes. The third example emphasized the dangers of linearization and discretization as a reminder that this becomes a larger issue when handling tracking problems. The errors observed in the third example are to be expected in other CLIK approaches where we assume the same linearity in the task error Jacobian. As the error observed is in the order of millimeters or even centimeters, this could negatively affect industrial use-cases such as seam-following as the error may exceed the desired tolerances if not accounted for.

The multiple task-priority inverse kinematics framework has been extended to set-based tasks in Moe et al. (2016). Extending the results of this article to set-based tasks will allow us to design control systems where we can ensure convergence while avoiding obstacles moving in known paths, or when the robot must remain inside a time-varying workspace.

The linearization issues may be addressed by extending the work of Falco and Natale (2011) to consider the tracking problem in the discretized version of the approach.

REFERENCES

Aertbeliën, E. and Schutter, J.D. (2014). etasl/etc: A constraint-based task specification language and robot controller using expression graphs. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1540–1546.

Andersson, J.A.E., Gillis, J., Horn, G., Rawlings, J.B., and Diehl, M. (2018). CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*.

Antonelli, G. (2009). Stability analysis for prioritized closed-loop inverse kinematic algorithms for redundant robotic systems. *IEEE Transactions on Robotics*, 25(5), 985–994.

Chiaverini, S. (1997). Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators. *IEEE Transactions on Robotics and Automation*, 13(3), 398–410.

Das, H., Slotine, J.E., and Sheridan, T.B. (1988). Inverse kinematic algorithms for redundant systems. In *Proceedings. 1988 IEEE International Conference on Robotics and Automation*, 43–48 vol.1.

Falco, P. and Natale, C. (2011). On the stability of closed-loop inverse kinematics algorithms for redundant robots. *IEEE Transactions on Robotics*, 27(4), 780–784.

Moe, S., Antonelli, G., Teel, A.R., Pettersen, K.Y., and Schrimpf, J. (2016). Set-based tasks within the singularity-robust multiple task-priority inverse kinematics framework: General formulation, stability analysis, and experimental results. *Frontiers in Robotics and AI*, 3, 16.

Sciavicco, L. and Siciliano, B. (1986). Coordinate transformation: A solution algorithm for one class of robots. *IEEE Transactions on Systems, Man, and Cybernetics*, 16(4), 550–559.

Appendix A. STABILITY OF EXAMPLE 2 AS A REGULATION PROBLEM

We have the partial derivatives (39) and (40) that form the Jacobians $\boldsymbol{J}_x \in \mathbb{R}^{1\times n}$ and $\boldsymbol{J}_y \in \mathbb{R}^{1\times n}$. We investigate the stability of the regulation problem by looking at $\boldsymbol{A}$. As $\boldsymbol{A}$ is lower-triangular, we are only interested in A_{11} and A_{22}:

$$A_{11} = \Lambda_1 \tag{A.1}$$

and

$$A_{22} = \boldsymbol{J}_y(I - \boldsymbol{J}_x{}^\dagger \boldsymbol{J}_x)\boldsymbol{J}_y{}^\dagger \Lambda_2 \tag{A.2}$$

$$A_{22} = (1 - \boldsymbol{J}_y \boldsymbol{J}_x{}^\dagger \boldsymbol{J}_x \boldsymbol{J}_y{}^\dagger)\Lambda_2 \tag{A.3}$$

We observe that

$$\boldsymbol{J}_y \boldsymbol{J}_x{}^\dagger = \frac{1}{\|\boldsymbol{J}_x\|^2} \boldsymbol{J}_y \boldsymbol{J}_x^T \tag{A.4}$$

$$\boldsymbol{J}_y \boldsymbol{J}_x{}^\dagger \boldsymbol{J}_x \boldsymbol{J}_y{}^\dagger = \frac{1}{\|\boldsymbol{J}_x\|^2 \|\boldsymbol{J}_y\|^2} \left(\boldsymbol{J}_y \boldsymbol{J}_x^T\right)^2 \tag{A.5}$$

and that the tasks are not annihilating by design. By Cauchy-Schwarz, assuming $\boldsymbol{J}_y$ and $\boldsymbol{J}_x$ are non-zero (we avoid singularities), we have that

$$1 \geq \frac{\left(\boldsymbol{J}_x \boldsymbol{J}_y^T\right)^2}{\|\boldsymbol{J}_x\|^2 \|\boldsymbol{J}_y\|^2} \tag{A.6}$$

which gives:

$$\text{eig}(\boldsymbol{A}) = (\Lambda_1, \alpha\Lambda_2) \tag{A.7}$$

where $\alpha \geq 0$ as long as $\boldsymbol{J}_y \neq 0$, $\boldsymbol{J}_x \neq 0$ and the two are linearly independent. This means that given any arbitrary $\Lambda_1 > 0$ and $\Lambda_2 > 0$ the regulation problem is asymptotically stable. The tracking problem on the other hand is not. We have

$$\boldsymbol{B} = \begin{bmatrix} 0 & 0 \\ -\beta & 1-\alpha \end{bmatrix} \tag{A.8}$$

where

$$\beta = \frac{\boldsymbol{J}_y \boldsymbol{J}_x^T}{\|\boldsymbol{J}_x\|^2} \tag{A.9}$$

If $\boldsymbol{J}_x$ and $\boldsymbol{J}_y$ are linearly independent, then $\beta = 0$ but $1 - \alpha \neq 0$. If $\boldsymbol{J}_x$ and $\boldsymbol{J}_y$ are dependent, then $1 - \alpha = 0$, but $\beta \neq 0$. The example shows how we can have stability of the regulation problem while not guaranteeing stability of the tracking problem with pseudo-inverse based closed-loop inverse kinematics.