periodica polytechnica

Electrical Engineering and Computer Science
56/4 (2012) 95–104
doi: 10.3311/PPee.7163
http://periodicapolytechnica.org/ee

RESEARCH ARTICLE

Joint Constrained Differential Inverse Kinematics Algorithm for Serial Manipulators

Dániel András Drexler / *István* Harmati

Received 2012-05-31, accepted 2013-01-07

Abstract

This paper presents a methodology for avoiding joint limits of robot manipulators in the solution of the differential inverse kinematics problem. A nonlinear transformation is applied to the joint limits, leading to a new kinematics formulation defined in the space of the transformed variables. The introduced transformation ensures that the joint variables stay between joint limits. Optimality conditions are converted into the transformed joint space, and closed-loop differential inverse kinematics is applied. The effectiveness of the introduced method is demonstrated on a simulational example.

Keywords

differential inverse kinematics · joint constraints · joint space transformation

Acknowledgement

This work was supported in part by the Hungarian National Scientific Research Foundation grant OTKA K71762. It is connected to the scientific program of the "Development of quality-oriented and harmonized R+D+I strategy and functional model at BME" project, supported by the New Hungary Development Plan (Project ID: TÁMOP-4.2.1/B-09/1/KMR-2010-0002).

Dániel András Drexler

Department of Control Engineering and Information Technology, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, Magyar tudósok körútja 2., H-1117 Budapest, Hungary
e-mail: drexler@iit.bme.hu

István Harmati

Department of Control Engineering and Information Technology, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, Magyar tudósok körútja 2., H-1117 Budapest, Hungary

1 Introduction

Inverse kinematics is one of the key issues in the motion planning of serial manipulators. Inverse kinematics is the problem of finding the joint variables (joint angles for rotational, and displacement for prismatic joints), that result in the desired end effector position and orientation. In typical industrial applications the manipulators have special geometry, i.e. they have 6 degrees of freedom (DOF), the first 3 joints determining the position of the end effector (regional manipulator), the last 3 joints determining the orientation of the end effector (wrist joint), so the inverse position and inverse orientation problem can be partitioned into two independent systems of equations [14], [15]. These manipulators are thus called decomposable or wrist-partitioned ones, and their inverse kinematics problem can be solved analytically. However, for manipulators with more complex geometry, or more degrees of freedom, the analytical solution to the inverse kinematics problem may not exists. In this case, the inverse kinematics problem is solved on the differential level, using the relationship between the joint velocities and the end effector velocities described by the manipulator Jacobian. The inverse kinematics problem is thus solved in the tangent space of the joint variables, and the joint variables are acquired by integration [7], [4], [5], [6].

However, the joint variables are typically constrained to be in a certain interval, determined by the physical limits of the joints (e.g. the maximum amount of rotation or translation of a rotational or prismatic joint respectively). This implies that only those solutions of the inverse kinematics problem can be accepted, that are in between these joint limits. If the inverse kinematics problem can be solved analytically, then this problem reduces to choosing a solution that does not exceed the joint limits, e.g. [9]. However, if differential inverse kinematics is applied, then the joint variables are acquired by integration of joint velocities, and the joint limits can be easily violated.

The differential inverse can be solved as a quadratic programming problem, by building the joint constraints into the numerical optimization problem [13]. A weighted least squares (WLS) pseudoinverse of the Jacobian can be applied as well [12], and the joints can be kept away from the limits by applying a cost

function that has great values near the joint limits, and by incorporating that cost function into the WLS inverse. The joints can also be driven away from the joint limits using the nullspace motions of the manipulator if the manipulator is redundant [8]. As an alternative solution, the joint limits can be built into the dynamical model of the manipulator, and the planning can be carried out using numerical optimization [11]. Neural networks also provide alternative approaches [10]. These solutions typically use the nullspace of the manipulator, that are generally utilizable only if the manipulator is redundant, and they can not guarantee that the joint limits are not exceeded, or based on numerical methods that are not suitable for real-time applications.

In this article we propose a new methodology that guarantees that the joint limits will not be violated, with good tracking performace, even if some of the joints are at their limits. This is achieved by transforming the joint variables to a fictive joint space, using a special nonlinear transformation. The properties of the transformation guarantee that the joint limits will stay between the limits. The introduced methodology is more than a simple saturation in a sense that the joint limits are reached in a continuous manner due to the properties of the transformation functions. Typical solutions to this problem in the literature use the nullspace of the kinematic mapping, and obtain joint limit avoidance as the result of an optimization task. These approaches however require kinematic redundancy, and they consume the extra degrees of freedom of the manipulator, thus the utilization of the redundancy for other optimization purposes, e.g. obstacle avoidance, becomes impossible. The proposed method ensures that the joint variables acquired as the solution of the differential inverse kinematics problem remain between the joint limits, without explicitly using the nullspace of the manipulator arising from kinematic redundancy. This implies, that this algorithm can be applied to nonredundant manipulators as well, and in case of redundant manipulators, the redundancy is left for other optimization purposes. In order to demonstrate this, we solve the differential inverse kinematics problem of a nonredundant manipulator as an example.

The motion planning is based on the differential geometric model of the robot, thus some basic issues on robot modeling are discussed in Section 2. The nonlinear joint transformation and its effect on the differential inverse kinematics algorithm is discussed in Section 3. An example for the joint transformation and the solution of the inverse kinematics of a common PUMA manipulator is shown in Section 4. The paper ends with the conclusion in Section 5.

2 Preliminaries

Rigid body motions can be characterized by transformations on the Special Euclidean group $SE(3)$, that is a subgroup of $GL(4)$, the group of general 4×4 matrices. Elements of $SE(3)$ are composed of 3×3 orthogonal matrices from the Special Orthogonal group $SO(3)$, i.e. orhogonal matrices with determinant +1, defining the rotation (or orientation), and 3-dimensional vectors from $\mathbb{R}^3$ that define the translation (or position) [2], [3]. An element of $SE(3)$ will be denoted by g and used in the homogeneous form

$$g = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \tag{1}$$

where $R \in SO(3)$ and $p \in \mathbb{R}^3$. The group $SE(3)$ is a Lie group with the Lie algebra elements $se(3)$ together with the Lie bracket $[\cdot, \cdot]$ as binary operation [1]. An element of $se(3)$ is composed of a 3×3 skew-symmetric matrix $\hat{\omega} \in so(3)$ corresponding to the cross product operator of the angular velocity, and a 3-dimensional vector $v \in \mathbb{R}^3$ corresponding to linear velocity. An element of $se(3)$ is called a twist, and is denoted by $\hat{\xi}$ if it is in the matrix form

$$\hat{\xi} = \begin{bmatrix} \hat{\omega} & v \\ 0 & 0 \end{bmatrix} \tag{2}$$

and by ξ if it is in the 6-vector form

$$\xi = \begin{bmatrix} v \\ \omega \end{bmatrix}, \tag{3}$$

where there is an isomorphism between ω and $\hat{\omega}$, defined as

$$\omega = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \rightarrow \hat{\omega} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}. \tag{4}$$

The differential equation of a general point p rotated by a twist ξ can be described by

$$\dot{p}(t) = \omega \times (p(t) - q) \tag{5}$$

where ω is the unit length axis of rotation, and q is an arbitrary point on the axis of rotation. Introducing the term $v = -\omega \times q$, the differential equation in homogeneous form is

$$\begin{pmatrix} \dot{p} \\ 0 \end{pmatrix} = \begin{bmatrix} \hat{\omega} & v \\ 0 & 0 \end{bmatrix} \begin{pmatrix} p \\ 1 \end{pmatrix} = \hat{\xi} \begin{pmatrix} p \\ 1 \end{pmatrix}. \tag{6}$$

The solution of the linear differential equation (6) with initial condition $p(0)$ is

$$\begin{pmatrix} p(t) \\ 1 \end{pmatrix} = e^{\hat{\xi} t} \begin{pmatrix} p(0) \\ 1 \end{pmatrix} \tag{7}$$

where t is the generalized time parameter. This parameter will be the joint variable in the applications, and will be denoted by θ. Note that all the vector quantities are described in a fixed reference frame (also called reference frame), as it will be throughout the paper.

Each joint of the manipulator can be described by the corresponding twist vector. In order to do so, first choose a fixed reference frame and a reference configuration (also called the home configuration), where the joint variables are zero. For each joint, define the joint twists as follows:

- If joint i is a rotational joint, let ω_i be a unit vector along the joint axis, and q_i an arbitrary point on the joint axis. The term v_i is calculated as $v_i = -\omega_i \times q_i$, and the twist vector is formed as $\xi_i = \begin{bmatrix} v_i^T & \omega_i^T \end{bmatrix}^T$.

- If joint i is a prismatic joint, then $\omega_i = 0$, and let v_i be a unit vector along the joint axis. Then the joint twist is formed as $\xi_i = \begin{bmatrix} v_i^T & 0 \end{bmatrix}^T$.

Let the orientation and position of the end effector in the reference frame and reference configuration be denoted by $g(0) \in SE(3)$. Then the end effector position and orientation in a general θ configuration for an n-DOF manipulator is defined by the product of exponentials formula

$$g(\theta) = e^{\hat{\xi}_1\theta_1} e^{\hat{\xi}_2\theta_2} \dots e^{\hat{\xi}_n\theta_n} g(0). \tag{8}$$

This is also referred to as the forward kinematics map, or the forward kinematics problem, that specifies the relationship between the end effector pose and the joint variables in a fixed reference frame. The differential motion of the end effector can be acquired by differentiating the forward kinematics map, and is formulated as

$$\dot{x} = \sum_{i=1}^{n} \xi_i' \dot{\theta}_i, \tag{9}$$

where $\dot{x}$ is the end effector linear and angular velocity, n is the number of joints of the manipulator, ξ_i' is the joint twist i in the actual configuration $\theta(t)$, that can be acquired from the joint twists in the home configuration (where $\theta = 0$) using the *Adjoint* transformation [2]:

$$\xi_i' = Ad_{e^{\hat{\xi}_1\theta_1} e^{\hat{\xi}_2\theta_2} \dots e^{\hat{\xi}_{i-1}\theta_{i-1}}} \xi_i. \tag{10}$$

The differential motion of the end effector can be expressed in a matrix form as

$$\begin{pmatrix} v_e \\ \omega_e \end{pmatrix} = \begin{bmatrix} \xi_1' & \xi_2' & \dots & \xi_n' \end{bmatrix} \begin{pmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \vdots \\ \dot{\theta}_n \end{pmatrix} = J \begin{pmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \vdots \\ \dot{\theta}_n \end{pmatrix} \tag{11}$$

where v_e is the linear velocity and ω_e is the angular velocity of the end effector, J is the manipulator Jacobian, formed by the joint twists in the actual configuration, and $\dot{\theta}_1, \dot{\theta}_2, \dots, \dot{\theta}_n$ are the joint velocities. The differential inverse kinematics problem of a robot manipulator is based on solving the linear equations defined by (11) to get the joint derivatives for the desired end effector velocity, and integrating the joint velocity vector $\dot{\theta}$ to get the joint variables. If the joint variables are limited, these can not be taken into consideration during integration, so other alternatives need to be investigated to ensure that the joint variables stay between their limit values. The remaining of this paper presents a new methodology to ensure proper joint variable characteristics. It is important to mention that the inversion of (11) may be hard if the Jacobian is singular, however later on we assume that the Jacobian is full rank in all cases.

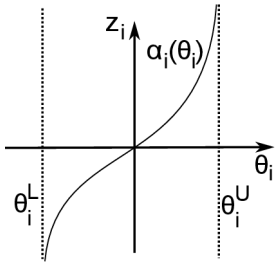


Fig. 1. A possible candidate for function α_i

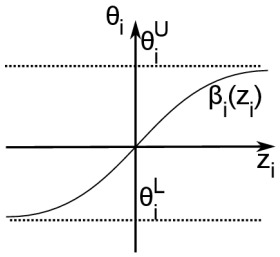


Fig. 2. A possible candidate for function β_i

3 Nonlinear Joint Transformation

In this section a nonlinear joint transformation is introduced in order to redefine the kinematic mapping on the domain constrained by the joint limits. Joint variables (θ_i) are transformed to the fictive joint variables (z_i) using a function that is continuous, monotonously increasing and open on the interval (θ_i^L, θ_i^U), where θ_i^L is the lower limit for joint i and θ_i^U is the upper limit for joint i. Denote this function by α_i, thus

$$z_i = \alpha_i(\theta_i). \tag{12}$$

A possible candidate for such a function can be seen in Figure 1. Since this function is monotonously increasing and continuous on an open interval, its inverse exists. Denote this inverse by β_i:

$$\theta_i = \beta_i(z_i) = \alpha_i^{-1}(z_i). \tag{13}$$

The domain of the function β_i is $[-\infty, \infty]$, however its range is (θ_i^L, θ_i^U) as shown in Figure 2. The main point of the introduced methodology is that the kinematic equations will be written in terms of the fictive variables (z_i), and the integration will be done in the fictive joint space, and finally the real joint variables will be acquired using the β_i functions. Since the range of these β_i functions is $\left(\theta_i^L, \theta_i^U\right)$, the real joint variables will always stay between the joint limits.

The forward kinematics mapping in terms of the fictive z_i joint variables is

$$g(z_1, z_2, \dots, z_n) = e^{\beta_1(z_1)\hat{\xi}_1} e^{\beta_2(z_2)\hat{\xi}_2} \dots e^{\beta_n(z_n)\hat{\xi}_n} g(0) \tag{14}$$

with the $\hat{\xi}_i$ twists being defined in the home configuration. In order to perform differential inverse kinematics, we need to calculate the manipulator Jacobian based on (14), so we need to examine the motions generated by the joints of the manipulator. The velocity of the end effector due to the motion of joint i is defined by [2]

$$V_i = \frac{\partial g(z)}{\partial z_i} \dot{z}_i g^{-1}(z), \tag{15}$$

so the total end effector motion is characterized by

$$\begin{bmatrix} v_e \\ \omega_e \end{bmatrix} = \sum_{i=1}^{n} V_i = \sum_{i=1}^{n} \frac{\partial g(z)}{\partial z_i} \dot{z}_i g^{-1}(z). \tag{16}$$

The derivative of the forward kinematics mapping with respect to z_i is

$$\frac{\partial g(z)}{\partial z_i} = e^{\beta_1(z_1)\hat{\xi}_1} \dots e^{\beta_{i-1}(z_{i-1})\hat{\xi}_{i-1}} \frac{\partial \beta_i(z_i)}{\partial z_i} \hat{\xi}_i e^{\beta_i(z_i)\hat{\xi}_i} \dots e^{\beta_n(z_n)\hat{\xi}_n} g(0) \tag{17}$$

and since

$$g^{-1}(z) = g^{-1}(0) e^{-\beta_n(z_n)\hat{\xi}_n} \dots e^{-\beta_2(z_2)\hat{\xi}_2} e^{-\beta_1(z_1)\hat{\xi}_1} \tag{18}$$

the velocity vector according to (15) is

$$V_i = e^{\beta_1(z_1)\hat{\xi}_1} \dots e^{\beta_{i-1}(z_{i-1})\hat{\xi}_{i-1}} \frac{\partial \beta_i(z_i)}{\partial z_i} \hat{\xi}_i e^{-\beta_{i-1}(z_{i-1})\hat{\xi}_{i-1}} \dots e^{-\beta_1(z_1)\hat{\xi}_1} \dot{z}_i \tag{19}$$

Since $\dfrac{\partial \beta_i(z_i)}{\partial z_i}$ is a scalar valued function, it commutes with the matrix exponentials in (19), and according to (13), $\beta_i(z_i) = \theta_i$ holds, thus (19) can be reformulated as

$$V_i = \frac{\partial \beta_i(z_i)}{\partial z_i} \underbrace{e^{\hat{\xi}_1\theta_1} \dots e^{\hat{\xi}_{i-1}\theta_{i-1}} \hat{\xi}_i e^{-\hat{\xi}_{i-1}\theta_{i-1}} \dots e^{-\hat{\xi}_1\theta_1}}_{Ad_{e^{\hat{\xi}_1\theta_1} \dots e^{\hat{\xi}_{i-1}\theta_{i-1}}} \xi_i} \dot{z}_i \tag{20}$$

The underbrace term in the right-hand side of (20) is the *Adjoint* transformation [2] of the twist ξ_i in the joint configuration θ, thus the velocity can be expressed as

$$V_i = \frac{\partial \beta_i(z_i)}{\partial z_i} Ad_{e^{\hat{\xi}_1\theta_1} \dots e^{\hat{\xi}_{i-1}\theta_{i-1}}} \xi_i \dot{z}_i = \frac{\partial \beta_i(z_i)}{\partial z_i} \xi_i' \dot{z}_i \tag{21}$$

where ξ_i' is the ith column of the manipulator Jacobian in the joint configuration θ. According to these results the mapping between the tangent space of the end effector and the tangent space of the z transformed joint variables is

$$\dot{x} = J(\theta) d\beta(z) \dot{z} \tag{22}$$

where J is the manipulator Jacobian, z is the vector of the transformed joint variables, $\dot{x}$ is the velocity of the end effector, and $d\beta$ is a diagonal matrix formed as

$$d\beta = \begin{bmatrix} \dfrac{\partial \beta_1(z_1)}{\partial z_1} & 0 & \dots & 0 \\ 0 & \dfrac{\partial \beta_2(z_2)}{\partial z_2} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \dfrac{\partial \beta_n(z_n)}{\partial z_n} \end{bmatrix}. \tag{23}$$

Since the functions β_i are strictly monotonously increasing, the derivatives are positive, so it is true for all $i = 1, \dots, n$, that $\partial \beta_i(z_i)/\partial z_i > 0$. Denote the relationship between the time derivatives of z and the end effector velocity by J_c and call it the constrained Jacobian

$$J_c = J d\beta(z). \tag{24}$$

This term is only introduced for notational simplicity, it will only be used in derivations in the remaining of the paper. The effect of the joint transformation on the Jacobian arises as weighting factors for each column of the Jacobian, thus each twist is weighted with the derivative of the inverse transformation function.

3.1 Optimality conditions in the transformed joint space

The solution of the differential inverse kinematics requires the inversion of the Jacobian matrix (11), as it was already stated in Section 2. If the manipulator is kinematically redundant, then the manipulator Jacobian is not square, and a generalized inverse is used. The most typical generalized inverse is the Moore-Penrose pseudoinverse, which chooses a solution from the tangent space of the joint variables that has the least Euclidean norm. However, if the kinematics is expressed in the transformed joint variables, the Moore-Penrose pseudoinverse of the constrained Jacobian does not yield a solution that has the least Euclidean norm in the tangent space of the original joint variables. In this subsection we show how to calculate the pseudoinverse that minimizes the Euclidean norm of the joint velocities for redundant manipulators, if the kinematics is formulated in the transformed joint space.

First of all examine the relationship between the tangent spaces of the real and the transformed joint variables:

$$\dot{\theta} = d\beta(z) \dot{z} \tag{25}$$

that can be verified e.g. by substituting (25) into (22). The optimality condition is given in the tangent space of θ, i.e.

$$\min \left\langle \dot{\theta}, \dot{\theta} \right\rangle, \tag{26}$$

where $< \cdot, \cdot >$ denotes the scalar product. However, the motion planning is carried out in the tangent space of the z transformed variables, so the optimality criterion also has to be transformed to

$$\min \langle d\beta(z)\dot{z}, d\beta(z)\dot{z} \rangle. \tag{27}$$

Since $d\beta(z)$ is a symmetric matrix, this condition can be reformulated as

$$\min \left\langle d\beta^2(z)\dot{z}, \dot{z} \right\rangle. \tag{28}$$

The optimization problem becomes

$$\begin{aligned} &\textit{minimize} && \left\langle d\beta^2(z)\dot{z}, \dot{z} \right\rangle \\ &\textit{subject to} && \\ & && \dot{x} = J_c \dot{z}. \end{aligned}$$

The problem can be solved using Lagrange multipliers. Introduce the Lagrangian function

$$L = \left\langle d\beta^2(z)\dot{z}, \dot{z} \right\rangle + \langle \lambda, J_c\dot{z} - \dot{x} \rangle. \tag{29}$$

Calculating the derivatives of the Lagrangian (29) with respect to $\dot{z}$ and λ and equating them to zero results in

$$\frac{\partial L}{\partial \dot{z}} = 2d\beta^2(z)\dot{z} + J_c^T\lambda = 0 \tag{30}$$

$$\frac{\partial L}{\partial \lambda} = J_c\dot{z} - \dot{x} = 0. \tag{31}$$

Solving (30) for $\dot{z}$ yields

$$\dot{z} = -\frac{1}{2}d\beta^{-2}(z)J_c^T\lambda. \tag{32}$$

Substituting (32) into (31) and solving for λ yields

$$\lambda = -2\left(J_c d\beta^{-2}(z)J_c^T\right)^{-1}\dot{x}. \tag{33}$$

Substituting (33) back into (32) results in

$$\dot{z} = d\beta^{-2}(z)J_c^T\left(J_c d\beta^{-2}(z)J_c^T\right)^{-1}\dot{x}, \tag{34}$$

thus the generalized inverse for J_c that satisfies the optimality criterion (28) is

$$J_c^{\#} = d\beta^{-2}(z)J_c^T\left(J_c d\beta^{-2}(z)J_c^T\right)^{-1}. \tag{35}$$

Substituting the expression for the constrained Jacobian defined by (24), the generalized inverse can be given in terms of the original Jacobian as

$$J_c^{\#} = d\beta^{-1}(z)J^{\#}, \tag{36}$$

where $J^{\#}$ is the Moore-Penrose pseudoinverse of the manipulator Jacobian.

Note that this result can be obtained, if we would calculate the joint velocities in a conventional way as

$$\dot{\theta} = J^{\#}\dot{x}, \tag{37}$$

and transform the joint velocities to the transformed joint space using the inverse of (25):

$$\dot{z} = d\beta^{-1}(z)\dot{\theta}. \tag{38}$$

This means that the differential inverse kinematics can be carried out in a comfortable way (see Figure 3). We can calculate the joint velocities using any conventional method (e.g. (37)), then transform the velocities to the trasformed joint space using (38), acquire the actual values of the z transformed variables through integration, then transform the result back to the real joint space using (13). In Figure 3, the blocks that incorporate joint constraints into the algorithm are depicted as dashdot boxes. Note that the algorithm is the same for nonredundant manipulators as well, the only difference is that the joint velocities are acquired as $\dot{\theta} = J^{-1}\dot{x}$, i.e. the inverse of the Jacobian may be used instead of its pseudoinverse, if the manipulator has the same degrees of freedom as the dimension of the task space (the manipulator is not underactuated).

A straightforward effect of the transformation after the integration is that the joint variables remain in the desired range. However, the effect of the trasformation before the integration should be further analysed. If a joint variable θ_i is near its limit, then the derivative of the function β_i is close to 0. Formally, if $\theta_i \to \theta_i^L$ or $\theta_i \to \theta_i^U$, then $\partial\beta_i/\partial z_i \to 0$. This can be interpreted using (36) as if a joint limit is approached, the effect of the corresponding twist on the end effector velocity in the transformed joint space decreases, and at a joint limit, its effect is almost zero. Practically the twist is turned off, if a joint limit is reached. This is beneficial since the joint limit can not be crossed, and this characteristics also yields that the saturation of the joint variable will be continuous, i.e. the corresponding joint will not stop suddenly when the physical joint limit is reached. However, there are some disadvantages:

1 The differential inverse after the application of the $d\beta^{-1}$ linear transformation may become ill-conditioned every time $\partial\beta_i/\partial z_i$ gets too small for any $i \in \{1,\dots,n\}$. Call such a situation a constrained singularity. Two alternative solutions to this problem are addressed in subsection 3.2.

2 The joint is turned off at the differential level, so it can not generate any motion. This causes the loss of one degree of freedom of the manipulator for each joint that is at its limit, which decreases manipulability. This problem is addressed in subsection 3.3.

3.2 Constrained singularities

This subsection implies a solution to the first problem, i.e. the numerical problems arising in the transformed joint space when a joint limit is reached. The inverse kinematics algorithm uses the inverse of the $d\beta$ matrix before integration, and the numerical problems arise when this matrix becomes ill-conditioned. We call this situation a constrained singularity. Note that in a kinematic singularity, the Jacobian may become ill-conditioned as well, however we suppose that tha Jacobian is full-rank in all cases, and do not deal with this situation in this article.

The $d\beta$ matrix is a diagonal matrix, with the $\partial\beta_i/\partial z_i$ differentials in the diagonal entries, and its inverse is a diagonal matrix as well, and its entries are the reciprocal of the corresponding diagonal elements. In case a joint limit is approached, the corresponding diagonal element in the inverse matrix may become extremely large, that is natural, since small changes in the joint variables near a joint limit result in large changes in the transformed joint variables, as it can be easily verified by examining the transformation function in Figure 1. However, this causes numerical problems in the differential inverse kinematics algorithm. In this article, two different methods for inverting the ill-conditioned $d\beta$ matrix are investigated to overcome the problem of numerical instability:

1 Pseudoinverse based on singular value decomposition (SVD) with truncation at low singular values.

2 Damped pseudoinverse.

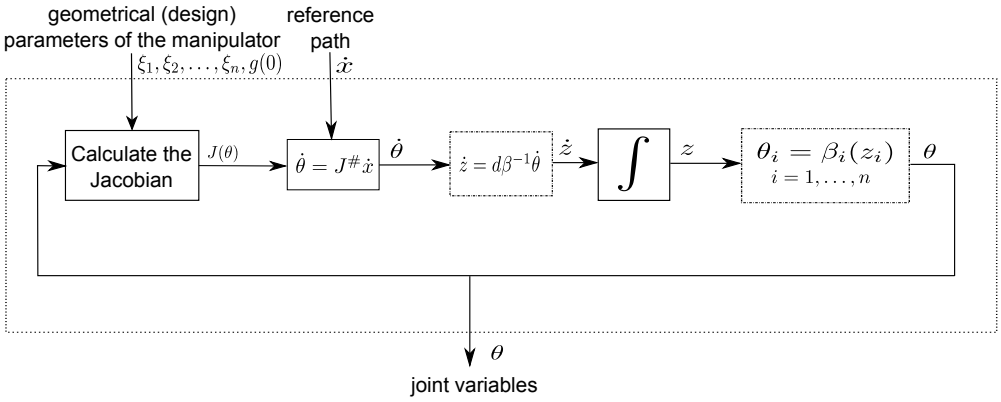


Fig. 3. The differential inverse kinematics algorithm with joint constraints

The SVD of a matrix A is a decomposition

$$A = U\Sigma V^T, \tag{39}$$

where U and V are orthogonal matrices, and Σ is a diagonal matrix with the σ_i singular values in the diagonal elements in descending order, i.e. $\sigma_1 \geq \sigma_2 \geq \ldots \sigma_n \geq 0$. If Σ is full rank and well-conditioned, then the inverse of A can be calculated as

$$A^{-1} = V\Sigma^{-1}U^T, \tag{40}$$

however if Σ is not full rank or ill-conditioned, the pseudoinverse of the matrix A is

$$A^{\#} = V\begin{bmatrix} \hat{\Sigma}^{-1} & 0 \\ 0 & 0 \end{bmatrix}U^T, \tag{41}$$

where $\hat{\Sigma}$ is the minor matrix of Σ with elements $\sigma_i \geq \epsilon$, where ϵ is a parameter characteristic of numerical accuracy and stability. If the matrix A is a diagonal matrix with positive entries (such as the matrix $d\beta$), then its singular values are the diagonal elements, the orthogonal matrices U and V^T are simply permutation matrices, that are used to permute the diagonal elements such that they are in descending order. When the pseudoinverse in calculated, the elements of the matrix are sorted again in their original order. Applying the pseudoinverse technique with truncation at singular values $\sigma_j < \epsilon$ (41) on the diagonal matrix $d\beta$, its pseudoinverse can be defined as

$$d\beta^{\#}_{ii} = \begin{cases} \dfrac{1}{\partial\beta_i/\partial z_i} & \text{if } \dfrac{\partial\beta_i}{\partial z_i} \geq \epsilon \\ 0 & \text{else.} \end{cases} \tag{42}$$

The main disadvantage of this technique is that it practically turns off the joint at differential level if a joint limit is approached, such that when the differential of its inverse transformation function is less than the threshold ϵ. If the joint variable is turned off, it will not be affected by the differential inverse kinematics algorithm, thus no joint motion will be generated for the joint at the limit. However, in subsection 3.3, we propose a solution to this problem.

Another approach to invert the ill-conditioned $d\beta$ matrix is the damped pseudoinverse, that is

$$d\beta^{\#} = (d\beta + \lambda I)^{-1}, \tag{43}$$

where λ is the damping factor, and I is the $n \times n$ identity matrix. If a joint limit is approached, then the corresponding diagonal element in the damped pseudoinverse of the matrix $d\beta$ is upper bounded by the damping factor λ as

$$\sup_{z_i}\left(\frac{1}{\dfrac{\partial\beta_i}{\partial z_i} + \lambda}\right) = \frac{1}{\lambda}. \tag{44}$$

This case the effect of the corresponding joint twist will not be zero, but it will have an influence on the differential inverse with a weighting factor that is upper bounded by $1/\lambda$. This means that there are always motions in the transformed joint space, however these may result in very small motions in the real joint space.

3.3 Regaining manipulability

In the previous subsection, we addressed the numerical problem that arises when the $d\beta$ matrix becomes ill-conditioned. This situation happens, when any of the joint variables gets close to its limit. The manipulator Jacobian is full rank by hypotheses, however the constrained Jacobian becomes ill-conditioned in such situations. We call these situations constrained singularities. Besides numerical problems, considering the inverse kinematics, manipulability also decreases.

Since the constrained Jacobian becomes singular in such situations, the dimension of the nullspace of the constrained Jacobian increases. This implies, that even if the Jacobian is square, the constrained singularity gives rise to nullspace motions. These nullspace motions exist because of the constrained singularity, and are independent of the nullspace motions arising from kinematic redundancy. As a consequence, there are nullspace motions even for nonredundant manipulators in constrained singularities, and for redundant manipulators the dimension of the nullspace motions (selfmotion manifolds) increases as well. In this subsection we utilize the nullspace motions arising from constrained singularities to regain the manipulability of the manipulator. Since this motion is independent

of kinematic redundancies, the proposed algorithm does not utilize the redundancy of redundant manipulators, so the nullspace motions arising from kinematic redundancy may be utilized for other optimization purposes.

The purpose of this subsection is to show how the nullspace motions arising at constrained singularities can be used to move the joint away from the joint limit, if it is needed. This is done by introducing a secondary task vector in the transformed joint space, that tends to drive the joint away from the limit. The secondary task vector is projected to the nullspace using the nullspace projector of the constrained Jacobian. However, this nullspace motion is different from nullspace motions arising from kinematic redundancy in nature, because this is the result of constrained singularities. Thus the nullspace projector has special characteristics as well, as it will be shown.

The nullspace projection method is usually used to make the motion of redundant manipulators satisfy certain optimality criteria. This is achieved by defining a task vector in the joint space denoted by y, and projecting this task vector to the nullspace of the Jacobian. The method is formally the same in this case too, i.e. the calculation of the transformed joint derivatives is

$$\dot{z} = J_c^{\#}\dot{x} + (I - J_c^{\#}J_c)y, \tag{45}$$

where the $J_c^{\#}\dot{x}$ term determines the joint movement for the desired end effector motion and can be calculated with one of the methods described in the previous subsection, and the $(I - J_c^{\#}J_c)$ term is the augmented projector [16] with I as the $n \times n$ identity matrix, that projects the goal vector y defined in the tangent space of z to the nullspace of the mapping $J_c^{\#}\dot{x}$. In this application, vector y may be defined to make the corresponding joint variable move away from the joint limit as

$$y_i = \begin{cases} 0 & \text{if } |z_i| < \gamma_i \\ -z_i\psi_i & \text{if } |z_i| > \gamma_i \end{cases} \tag{46}$$

where γ_i and ψ_i are appropriately chosen constants, and the index i goes from 1 to n. Suppose that we use the SVD technique to calculate the (pseudo)inverse of the constrained Jacobian. Then a suitable choice for γ_i is the solution of $\partial\beta_i(\gamma_i)/\partial z_i = \epsilon_i$, since in this case the null space is only activated to drive the joint away from the limit if the joint movement is turned off.

It is interesting to examine the augmented projector defined by (45) in case of a nonredundant manipulator with $n = 6$, if the SVD pseudoinverse is used to invert the $d\beta$ matrix. In this case, if the manipulator Jacobian is full rank, then $J^{\#}J$ should be the $n \times n$ identity matrix, so the augmented projector should be the zero matrix. However, if a joint limit is approached, and the pseudoinverse is calculated as in (41), then the augmented projector J_c^A is calculated using the constrained Jacobian that is singular, i.e.

$$J_c^A = I - J_c^{\#}J_c = I - d\beta^{\#}J^{\#}Jd\beta. \tag{47}$$

Since J is full rank by hypothesis,

$$J_c^A = I - d\beta^{\#}d\beta. \tag{48}$$

If there are no joint variables near the limits, then J_c^A is zero. However if some of the singular values of the matrix $d\beta$ are truncated when the pseudoinverse is calculated, such that there is a collection of indices $I_s \in \{1, \ldots, n\}$, i.e. $\sigma_{I_s} < \epsilon$, then it can be shown, that the augmented projector takes the form

$$J_c^A = \begin{bmatrix} j_1^A & 0 & \ldots & 0 \\ 0 & j_2^A & \ldots & 0 \\ \vdots & & \ddots & \vdots \\ 0 & \ldots & & j_n^A \end{bmatrix} \tag{49}$$

with $j_i^A = 1$ if $i \in I_s$, and $j_i^A = 0$ if $i \notin I_s$. Note that this solution tries to move the joint away from the limit, and can result in good path tracking performance if the desired path can be achieved while the corresponding joint moves away from the limit. However if the joint variable has to cross the joint limit in order to track the desired path, then this may yield bad path tracking performance, as it is expected, since the desired task can not be executed by the manipulator. In other words, the proposed algorithm works fine if the desired end effector path is consistent with the physical constraints of the manipulator, i.e. the path is realizable.

4 Simulational example

In this section an example is discussed to illustrate the choice of the transformation functions, and their application on the differential inverse kinematics of a PUMA robot arm. We chose a nonredundant robot to illustrate that kinematical redundancy is not neccessary to regain the manipulability of the manipulator at constrained singularities.

First, we introduce the transformation functions that we use to transform the joint variables. The function used to transform the joint variables needs to be continuous, strictly monotonously increasing and onto the whole codomain $\mathbb{R}$ on an open interval of (θ_i^L, θ_i^U). A good candidate for such a function is e.g. the tangent function $tan(\cdot) : x \to tan(x)$, that satisfies these criteria on the open interval $(-\pi/2, \pi/2)$. In order to scale the domain of the $tan(\cdot)$ function to (θ_i^L, θ_i^U), a linear mapping is introduced to map (θ_i^L, θ_i^U) to $(-\pi/2, \pi/2)$:

$$x_i = \frac{\pi\left(2\theta_i - \theta_i^U - \theta_i^L\right)}{2\left(\theta_i^U - \theta_i^L\right)}, \tag{50}$$

thus the transformation function is

$$\alpha_i(\theta_i) = \tan\left(\frac{\pi\left(2\theta_i - \theta_i^U - \theta_i^L\right)}{2\left(\theta_i^U - \theta_i^L\right)}\right), \tag{51}$$

and the inverse function is

$$\beta_i(z_i) = \frac{\theta_i^U - \theta_i^L}{\pi}\tan^{-1}(z_i) + \frac{\theta_i^U + \theta_i^L}{2}. \tag{52}$$

The differential of the inverse function is

$$\frac{\partial\beta_i(z_i)}{\partial z_i} = \frac{\theta_i^U - \theta_i^L}{\pi}\frac{1}{1 + z_i^2}. \tag{53}$$

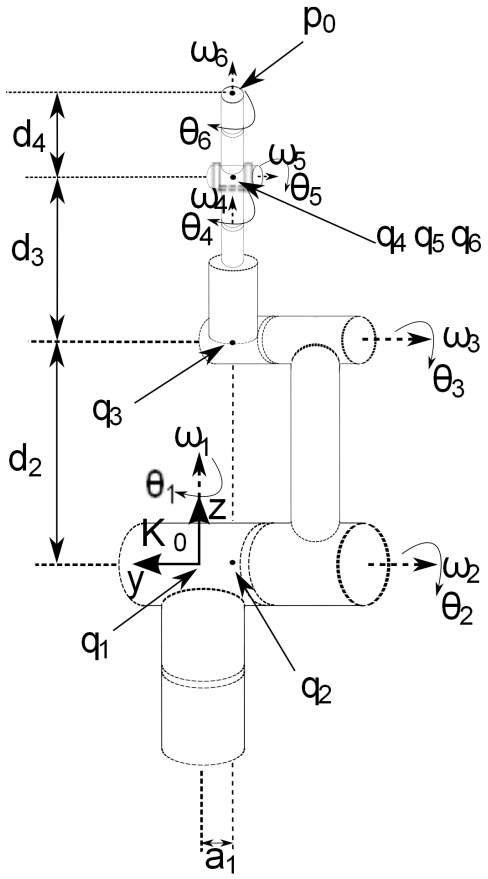


Fig. 4. Geometric parameters of the PUMA arm

It can be easily verified that the derivative of the inverse function $d\beta_i/dz_i \to 0$ if $z_i \to \pm\infty$, thus it has the characteristics discussed in the previous section. The examined manipulator is a PUMA manipulator with an extended configuration chosen as the reference configuration as in Figure 4. The reference frame is K_0, that is a right-handed orthogonal frame (the x axis is not depicted on the figure). The geometric quantities of the twists at the home configuration are

$$\omega_1 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \tag{54}$$

$$\omega_2 = \begin{bmatrix} 0 & -1 & 0 \end{bmatrix}^T \tag{55}$$

$$\omega_3 = \begin{bmatrix} 0 & -1 & 0 \end{bmatrix}^T \tag{56}$$

$$\omega_4 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \tag{57}$$

$$\omega_5 = \begin{bmatrix} 0 & -1 & 0 \end{bmatrix}^T \tag{58}$$

$$\omega_6 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T \tag{59}$$

$$q_1 = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T \tag{60}$$

$$q_2 = \begin{bmatrix} 0 & -a_1 & 0 \end{bmatrix}^T \tag{61}$$

$$q_3 = \begin{bmatrix} 0 & -a_1 & d_2 \end{bmatrix}^T \tag{62}$$

$$q_4 = \begin{bmatrix} 0 & -a_1 & d_2 + d_3 \end{bmatrix}^T \tag{63}$$

$$q_5 = \begin{bmatrix} 0 & -a_1 & d_2 + d_3 \end{bmatrix}^T \tag{64}$$

$$q_6 = \begin{bmatrix} 0 & -a_1 & d_2 + d_3 \end{bmatrix}^T \tag{65}$$

and the orientation and position of the end effector in the home configuration is

$$g(0) = \begin{bmatrix} I & p_0 \\ 0 & 1 \end{bmatrix}, \tag{66}$$

with $p_0 = \begin{bmatrix} 0 & -a_1 & d_2 + d_3 + d_4 \end{bmatrix}^T$, $a_1 = 0.5\,m$, $d_2 = 2\,m$, $d_3 = 1.5\,m$ and $d_4 = 0.5\,m$. These distances are rough, however this is only an example for visualization purposes. The distance unit will be in meters throughout the paper, and the unit of angles is radian. The orientation of the end effector is described by a 3×3 identity matrix in the home configuration in the reference frame, i.e. the axes of the frame attached to the end effector are parallel to the axes of the reference frame. The joint twists in the home configuration are thus

$$\xi_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^T \tag{67}$$

$$\xi_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & -1 & 0 \end{bmatrix}^T \tag{68}$$

$$\xi_3 = \begin{bmatrix} 2 & 0 & 0 & 0 & -1 & 0 \end{bmatrix}^T \tag{69}$$

$$\xi_4 = \begin{bmatrix} -0.5 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^T \tag{70}$$

$$\xi_5 = \begin{bmatrix} 3.5 & 0 & 0 & 0 & -1 & 0 \end{bmatrix}^T \tag{71}$$

$$\xi_6 = \begin{bmatrix} -0.5 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^T. \tag{72}$$

Let the joint limits of the manipulator be

$$\theta_1^U = \frac{\pi}{2} \tag{73}$$

$$\theta_1^L = -\frac{\pi}{2} \tag{74}$$

for the first joint variable, and

$$\theta_i^U = \frac{2\pi}{3} \tag{75}$$

$$\theta_i^L = -\frac{2\pi}{3} \tag{76}$$

for the remaining joint variables, i.e. $i = 2, \ldots, 6$.

It is trivial from the attributes of the transformation functions that the joint limits can not be exceeded with the algorithm defined in this paper. However, the tracking performance may be bad in constrained singularities, so these situations have great interest. The simulation should be carried out with a desired path that can only be achieved if some of the joint limits have to be reached during the path tracking, to examine the behaviour of the algorithm at constrained singularities.

However, it is hard to define a path directly that has the desired attributes, thus we define the path indirectly in the following way. First, we define the path at the joint space, since in the joint space we can easily define a joint trajectory that takes values at the joint limits. However, we need the end effector trajectory for the simulation, not the joint trajectories. Thus we take the reference path described in the joint space, and calculate the reference path in the task space using the forward kinematics equations of the manipulator, and use the reference path in the task space for simulation purposes.

Fig. 5. The tracking error for the end effector position

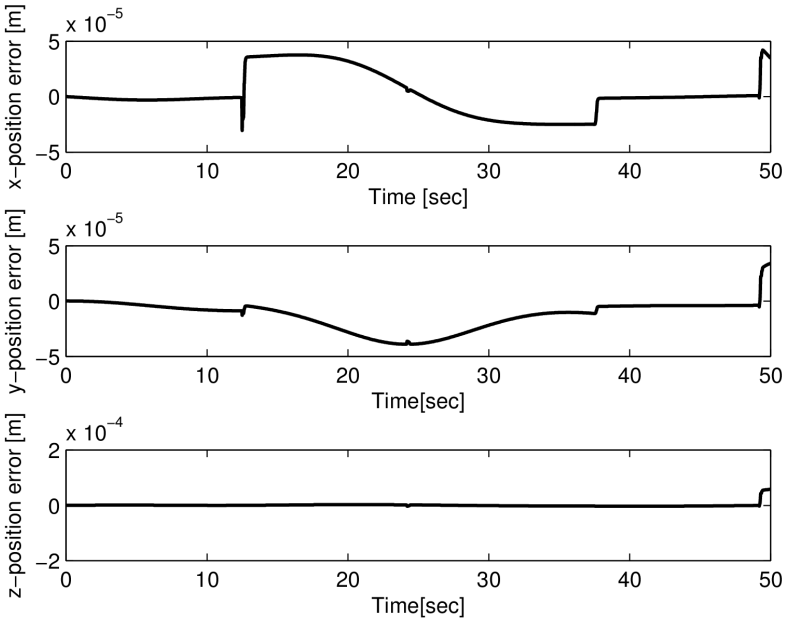
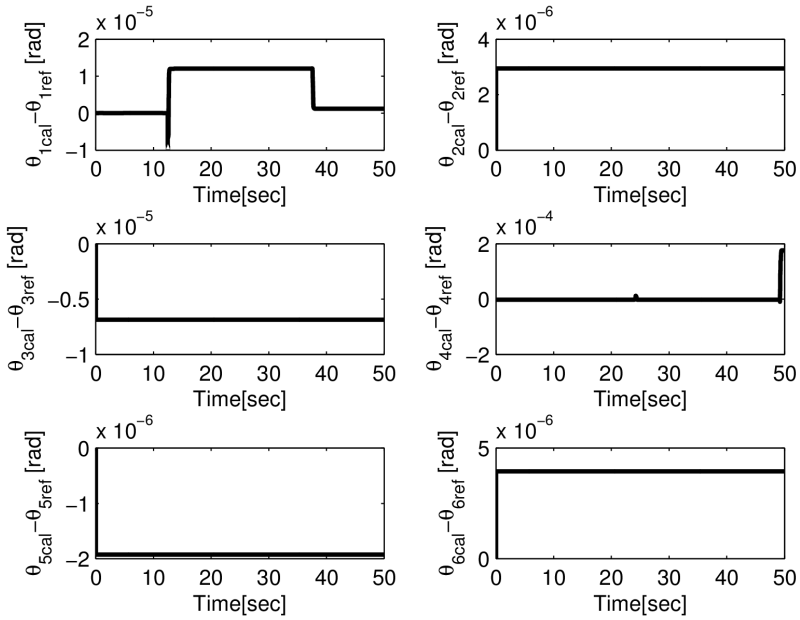


Fig. 6. The difference between the calculated trajectories and reference trajectories of the joint variables



The reference path described as joint trajectories is

$$\theta_1(t) = \frac{\pi}{2}\sin\left(\frac{2\pi t}{T_{max}}\right) \tag{77}$$

$$\dot{\theta}_1(t) = \frac{\pi^2}{T_{max}}\cos\left(\frac{2\pi t}{T_{max}}\right) \tag{78}$$

$$\theta_2(t) = \frac{\pi}{3} \tag{79}$$

$$\dot{\theta}_2(t) = 0 \tag{80}$$

$$\theta_3(t) = \frac{\pi}{3}\sin\left(\frac{2\pi t}{T_{max}}\right) \tag{81}$$

$$\dot{\theta}_3(t) = \frac{2\pi^2}{3T_{max}}\cos\left(\frac{2\pi t}{T_{max}}\right) \tag{82}$$

$$\theta_4(t) = \frac{2\pi}{3}\cos\left(\frac{2\pi t}{T_{max}} + 0.1\right) \tag{84}$$

$$\dot{\theta}_4(t) = -\frac{4\pi^2}{3T_{max}}\sin\left(\frac{2\pi t}{T_{max}} + 0.1\right) \tag{85}$$

$$\theta_5(t) = \frac{\pi}{3} \tag{86}$$

$$\dot{\theta}_5(t) = 0 \tag{87}$$

$$\theta_6(t) = 0 \tag{88}$$

$$\dot{\theta}_6(t) = 0 \tag{89}$$

with $T_{max} = 50\,sec$. This joint trajectory results in a motion where the joint variables 1 and 4 must reach both their upper

and lower limits. The reference end effector path and velocity is calculated from the reference joint trajectory using the forward kinematics equations (8) and the differential kinematics equations (9) respectively. The reference end effector path and velocity is then used as the input of the differential inverse kinematics algorithm at the simulation. The initial joint configuration is the same as the reference joint trajectory at $t = 0$.

The differential inverse kinematics algorithm is defined in the z transformed joint variables as

$$\dot{z} = d\beta^{\#} J^{\#} \dot{x}_{ref} + J_c^A y, \tag{90}$$

where the pseudoinverse of $d\beta$ is calculated as in (42) with $\epsilon_i = 10^{-10}$, $i = 1, \ldots, n$, $\dot{x}_{ref}$ is the reference end effector velocity, J_c^A is calculated as in (49), and y is calculated as in (46) with $\psi_i = 1$, $i = 1, \ldots, n$.

The simulation was carried out using the reference path and velocity generated from the reference joint trajectory. The simulation time was $T_{max} = 50\ sec$.

The position error of the end effector is in Figure 5. The minimum and maximum values of the path tracking error in the different coordinates were:

$$e_{x,min} = 3.32 \cdot 10^{-10}\, m \tag{91}$$

$$e_{x,max} = 4.19 \cdot 10^{-5}\, m \tag{92}$$

$$e_{y,min} = 1.25 \cdot 10^{-11}\, m \tag{93}$$

$$e_{y,max} = 3.9 \cdot 10^{-5}\, m \tag{94}$$

$$e_{z,min} = 1.52 \cdot 10^{-10}\, m \tag{95}$$

$$e_{z,max} = 5.8 \cdot 10^{-5}\, m. \tag{96}$$

The tracking error was maximal in the x and z coordinates when joint variable 4 reached its upper limit, and maximal in the y coordinate when joint variable 4 reached its lower limit, thus the tracking error increased significantly at constrained singularities. However, the values of the path tracking error are sufficiently low even in such situations.

The joint trajectory resulting as the output of the differential inverse kinematics algorithm is similar to the reference joint trajectory. The difference between the calculated and the reference joint trajectories can be seen in Figure 6.

The simulation showed that the proposed algorithm has good path tracking performace, even if some of the joint limits is reached, thus the problems caused by constrained singularities can be solved with the methods proposed in this paper.

5 Conclusion

We proposed a method to incorporate joint constraints into the differential inverse kinematics algorithm of robot manipulators, by introducing a nonlinear transformation on the joint variables. This transformation ensures, that the joint limits are not exceeded, however, it inherently has some disadvantages, i.e. at joint limits singularities arise. We call such situations constrained singularities. We have analysed this phenomenon, and gave solutions to the problems caused by constrained singularities. A simulation was carried out to demonstrate the effectiveness of the algorithm.

The algorithms are defined as general as it is possible considering the inverse kinematics problems of serial manipulators practically arising in robotics. The problem can be solved with any joint transformation function that has the desired characteristics, the tangent function was only used as an example. The differential inverse kinematics can also be modified for planar, inverse position, or inverse orientation problems, the proposed methods and statements will remain true in these cases as well.

References

1 **Postnikov M**, *Lectures in Geometry, Semester V, Lie Groups and Lie Algebras*, MIR Publishers Moscow, 1986.

2 **Murray RM, Sastry SS, Zexiang L**, *A Mathematical Introduction to Robotic Manipulation*, CRC Press, 1994.

3 **J.M. Selig**, *Geometric Fundamentals of Robotics (Second Edition)*, Springer, 1996.

4 **Caccavale F, Chiaverini S, Siciliano B**, *Second-Order Kinematic Control of Robot Manipulators with Jacobian Damped Least-Squares Inverse: Theory and Experiments*, IEEE/ASME Transactions on Mechatronics, **2**(3), (1997), 188–194.

5 **Tan J, Xi N, Wang Y**, *A singularity-free motion control algorithm for robot manipulators–a hybrid system approach*, Automatica, **40**(7), (2004), 1239 - 1245.

6 **Chiaverini S**, *Singularity-Robust Task-Priority Redundancy Resolution for Real-Time Kinematic Control of Robot Manipulators*, IEEE Transactions on Robotics and Automation, **13**(3), (1997), 398–410.

7 **Sciavacco L, Siciliano B**, *A Solution Algorithm to the Inverse Kinematic Problem for Redundant Manipulators*, IEEE Transactions on Robotics and Automation, **4**(4), (1988), 403-410.

8 **Lee H-Y, Yi B-J, Choi Y**, *A realistic Joint Limit Algorithm for Kinematically Redundant Manipulators*, In: Proceedings of International Conference on Control, Automation and Systems 2007, October, 2007, pp. 47–50.

9 **Shimizu M, Yoon W-K, Kitagaki K**, *A Practical Redundancy Resolution for 7 DOF Redundant Manipulators with Joint Limits*, In: Proceedings of 2007 IEEE International Conference on Robotics and Automation, Roma, Italy, April, 2007, pp. 4510 -4516.

10 **Assal SFM, Watanabe K, Izumi K**, *Cooperative Fuzzy Hint Acquisition for Avoiding Joint Limits of Redundant Manipulators*, In: Proceeings of The 30th Annual Conference of the IEEE Industrial Electronics Society, Busan, Korea, November, 2004, pp. 169 -174.

11 **Kim JH, Yang J, Abdel-Malek K**, *A novel formulation for determining joint constraint loads during optimal dynamic motion of redundant manipulators in DH representation*, Multibody System Dynamics, **19**(4), (2008), 427-451.

12 **Chan TF, Dubey RV**, *A Weighted Least-Norm Solution Based Scheme for Avoiding Joint Limits for Redundant Joint Manipulators*, IEEE Transactions on Robotics and Automation, **11**(2), (1995), 286 - 292.

13 **Zhang Y, Guo D, Cai B, Chen K**, *Remedy scheme and theoretical analysis of joint-angle drift phenomenon for redundant robot manipulators*, Robotics and Computer-Integrated Manifacturing, **27**, (2011), 860 - 869.

14 **Spong MW, Hutchinson S, Vidyasagar M**, *Robot Dynamics and Control*, John Wiley & Sons. Inc., 2006.

15 **Siciliano B, Sciavicco L, Villani L, Oriolo G**, *Robotics - Modelling, Planning and Control*, Springer, 2009.

16 **Rózsa P**, *Introduction to Matrix Theory (in Hungarian)*, Typotex, 2009.