10.24425/119074

Archives of Control Sciences
Volume 28(LXIV), 2018
No. 1, *pages 5–18*

A comparison of methods solving repeatable inverse kinematics for robot manipulators

IGNACY DULEBA and IWONA KARCZ-DULEBA

In this paper two recent methods of solving a repeatable inverse kinematic task are compared. The methods differ substantially although both are rooted in optimization techniques. The first one is based on a paradigm of continuation methods while the second one takes advantage of consecutive approximations. The methods are compared based on a quality of provided results and other quantitative and qualitative factors. In order to get a statistically valuable comparison, some data are collected from simulations performed on pendula robots with different paths to follow, initial configurations and a degree of redundancy.

Key words: manipulator, inverse kinematics, repeatability, optimization, algorithms

1. Introduction

Many robotic tasks at factory floors are performed cyclically in a virtually infinite loop of same actions performed on each consecutive item being processed. Although a production cycle requires a cyclic behavior of the end-effector of a manipulator (in a task-space) only, the cyclicity in a configuration space is also desirable. In this case only once a collision avoidance between a manipulator and its surrounding is to be checked and an optimization of one cycle is automatically extended to any cycle to follow. A practical necessity forced to define an important special kinematic task for robot manipulators called repeatable inverse kinematics [1]. The task relies on planning a loop in a configuration space which corresponds to a given loop in a task-space. Usually an entry configuration to the loop is also given. In robotic literature there exist at least two main approaches to solve the repeatable inverse kinematics task.

The first one have been initialized by Roberts and Maciejewski [12]. The researchers have noticed that an extension of original kinematics into non-redundant one, gives locally a unique solution of inverse kinematics [8]. So, the

The Authors are with Faculty of Electronics, Wroclaw University of Science and Technology, Janiszewski Str. 11/17, 50-372 Wroclaw, Poland.
The corresponding author is Ignacy Duleba, e-mail: ignacy.duleba@pwr.edu.pl
Received 19.04.2017.

repeatability in a configuration space is guaranteed if only the planned trajectory does not pass through any singular configuration. As the extension is free to choose, some extra requirements can be imposed on it to optimize the motion along the loop in a configuration space. A desired trajectory should be short to optimize an energy of motion and make the manipulator to occupy as small volume in a task-space as possible. Locally, around a given configuration, the shortest motion is generated with the Moore-Penrose (pseudo-) inverse kinematics [9]. Therefore, it is natural to extend the original kinematics in such a way that its first columns of the inverse of the augmented Jacobian matrix to be as close to the pseudo-inverse Jacobian matrix as possible over an assumed region in a configuration space [6, 12]. Generally, this approximation task is difficult to solve as it involves unknown partial differentials [10]. However, when a class of functions is assumed (usually small degree polynomials) to describe unknown augmenting functions in a parametric form, the approximation task is reduced into a standard (although multi-dimensional) optimization problem to search for coefficients of the augmenting functions [6]. Recently, a modification of this approach has been proposed aimed at making it task-dependent. This approach is recalled later on as it is one of the methods being compared [3].

The second method to compare is based on principles of continuation methods [11] and propagated in robotics by Quinlan and Khatib [7]. In robotics it has been applied to optimize paths of mobile robots. This method of solving repeatable inverse kinematics construct iteratively a series of loops in a configuration space (Q-loops) which mapped (via forward kinematics) into a task-space converge to a given loop (X-loop) there. The current (initial) Q-loop is trivial as it is composed of a single initial configuration and apparently does not generate a desired X-loop (but only its initial point). In consecutive iterations, a trial point in X-space located on X-loop is selected and, using the Newton algorithm with optimization in a null space of the Jacobian [9], a configuration is computed to reach the point. This configuration is added to designed Q-loop configurations and the configurations, when interpolated and mapped into X-space, give rise another X-loop. If the X-loop is close enough to the given one, the algorithm is completed. Otherwise, more and more configurations are added to Q-loop to get a desired accuracy of tracing the given X-loop. A convergence of the presented procedure is guaranteed as the distance between a current X-loop and the given X-loop decreases monotonically. Details of the method have been presented in [2].

The outline of the paper follows. In Section 2 a repeatable inverse kinematic task is defined and the Newton algorithm of inverse kinematics for redundant manipulators is recalled in two versions: the basic one – without optimization, and extended one – with optimization in a null space of the Jacobian matrix. Both versions are extensively used by methods being compared. Then, two aforementioned approaches to solve the repeatable inverse kinematic tasks are discussed in a nutshell. Complete linear equations for the method of iterative approximations

with general second degree polynomials approximating the Jacobian functions are provided. In Section 3 some factors useful in comparing the two approaches are discussed. Based on them, some advantages and disadvantages of the methods are shown. Simulation results are provided in Section 4. The simulations are performed on models of pendula with the redundancy index modified in order to check various aspects of the algorithm and its parameters. Section 5 concludes the paper.

2. Repeatable inverse kinematics: the task and algorithms

2.1. Repeatable inverse kinematics task

Forward kinematics k transforms a configuration $q \in \mathbb{Q}$ into a generalized position $x \in \mathbb{X}$. $\mathbb{Q}/\mathbb{X}$ is a configuration/task-space, respectively. For redundant manipulators, considered later on, $\dim\mathbb{Q} = n, \dim\mathbb{X} = m$, $n > m$ and the redundancy is described by the number $r = n - m$. In the task-space a closed path (X-loop) is given

$$\{\boldsymbol{x}(s), s \in [0, s_{\max}],\ \boldsymbol{x}(0) = \boldsymbol{x}(s_{\max})\}. \tag{1}$$

Repeatable inverse kinematics is aimed at finding a cyclic path $\boldsymbol{q}(\cdot) \in \mathbb{Q}$ (Q-loop) which corresponds to X-loop (1)

$$\forall_{s\in[0,s_{\max}]}\ \boldsymbol{k}(\boldsymbol{q}(s)) = \boldsymbol{x}(s),\ \boldsymbol{q}(0) = \boldsymbol{q}(s_{\max}). \tag{2}$$

Additionally, an entry configuration $\boldsymbol{q}(0)$ to the Q-loop is also given $\boldsymbol{k}(\boldsymbol{q}(0)) = \boldsymbol{x}(0)$. A non-repeatable inverse kinematic task is solved using the Newton algorithm [9]

$$\boldsymbol{q}_{i+1} = \boldsymbol{q}_i + \xi \cdot \boldsymbol{J}^{inv}(\boldsymbol{q}_i)(\boldsymbol{x}(s) - \boldsymbol{k}(\boldsymbol{q}_i)) \left(+ \eta(\boldsymbol{I}_n - \boldsymbol{J}^{\#}(\boldsymbol{q}_i)\boldsymbol{J}(\boldsymbol{q}_i)) \frac{\partial f}{\partial \boldsymbol{q}}\Big|_{\boldsymbol{q}=\boldsymbol{q}_i} \right), \tag{3}$$

where i is the iteration counter, ξ is a small, real and positive parameter and an initial configuration $\boldsymbol{q}_0$ is appropriately selected. The goal points $\boldsymbol{x}(s)$ in the X-space are set for some increasing sequence of the argument s. A resulting configuration, obtained for the previous value of s, serves as the initial configuration $\boldsymbol{q}_0$ of the current iteration (for $s = 0$, $\boldsymbol{q}_0 = \boldsymbol{q}(0)$ is known). For non-redundant kinematics $\boldsymbol{J}^{inv} = \boldsymbol{J}^{-1}$ is an inverse of the the Jacobian matrix $\boldsymbol{J} = \partial\boldsymbol{k}/\partial\boldsymbol{q}$, while for redundant kinematics $\boldsymbol{J}^{inv} = \boldsymbol{J}^{\#} = \boldsymbol{J}^T(\boldsymbol{J}\boldsymbol{J}^T)^{-1}$ is a (Moore-Penrose) pseudo-inverse of $\boldsymbol{J}$ as the matrix is not square in this case. For non-redundant manipulators, the null-space of the Jacobian is typically trivial, thus there is no reason to optimize in the null-space, $\eta = 0$.

In its basic form, $\eta = 0$, the Newton algorithm searches for such a configuration which realizes the point $\boldsymbol{x}(s)$. In its extended version (for redundant manipulators only), $\eta \neq 0$, additionally optimizes a scalar function $f(\boldsymbol{q})$. The Newton algorithm is completed when an error $\|\boldsymbol{k}(\boldsymbol{q}_i) - \boldsymbol{x}(s)\|$ drops below a given threshold value δ. As a rule, the algorithm (3) does not generate a Q-loop when applied to consecutive points $\boldsymbol{x}(s)$ of X-loop (1).

2.2. An algorithm based on the Jacobian augmentation (Method 1)

In this algorithm kinematics $\boldsymbol{k}(\boldsymbol{q})$ is augmented with r components $\boldsymbol{k}_{add}(\boldsymbol{q}) = (k_{m+1}(q), \ldots, k_{m+r}(\boldsymbol{q}))^T$ to form non-redundant kinematics $\boldsymbol{k}_{aug}(\boldsymbol{q}) = (\boldsymbol{k}(\boldsymbol{q})^T, \boldsymbol{k}_{add}(\boldsymbol{q})^T)^T$, the Newton algorithm (3) is applied with $\eta = 0$ and $\boldsymbol{J}^{inv} = \boldsymbol{J}^{-1}_{aug,trunc}$ where $J^{-1}_{aug,trunc}(q_i)$ collects first m columns of the inverse of the augmented Jacobian matrix $\boldsymbol{J}_{aug} = \partial \boldsymbol{k}_{aug}/\partial \boldsymbol{q}$. The algorithm (3) is run for some points $\boldsymbol{x}(s)$ with the variable s increased. An initial configuration $\boldsymbol{q}_0$ for the current task is selected as the final configuration from the previous run. For the first task $\boldsymbol{q}_0 = \boldsymbol{q}(0)$. Resulting configurations are interpolated (usually linearly) to form a final Q-loop.

In this algorithm the only item to be designed is $\boldsymbol{k}_{add}(\boldsymbol{q})$. A pseudo-inverse of the Jacobian matrix locally maximizes a motion of the end-effect for a fixed, small displacement in a configuration space thus it is desirable to search for an augmentation of the matrix approximating the pseudo-inverse solution [6]. Probably, the simplest augmentation of $\boldsymbol{k}(\boldsymbol{q})$ can be performed using the Singular Value Decomposition algorithm [5] which allows to decompose the Jacobian matrix $\boldsymbol{J}$ (at a given configuration the matrix is composed of real numbers)

$$\boldsymbol{J} = \boldsymbol{U}\,[\boldsymbol{D}\ \ \boldsymbol{0}]\boldsymbol{V}^T = [\boldsymbol{U}\boldsymbol{D}\ \ \boldsymbol{0}]\boldsymbol{V}^T \tag{4}$$

into rotation matrices $\boldsymbol{U} \in SO(m)$ ($SO(m)$ denotes an orthogonal group of order m), $\boldsymbol{V} \in SO(n)$, and diagonal matrix $\boldsymbol{D}$ collecting singular values, $\boldsymbol{0}$ is the matrix of the size $(r = n - m) \times n$. It has been checked [3] that the orthonormal augmented Jacobian in the form

$$\boldsymbol{J}_{ortho} = \begin{bmatrix} \boldsymbol{U} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{Rot} \end{bmatrix} \begin{bmatrix} \boldsymbol{D} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{I}_r \end{bmatrix} \boldsymbol{V}^T = \begin{bmatrix} \boldsymbol{U}\boldsymbol{D} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{Rot} \end{bmatrix} \boldsymbol{V}^T = \begin{bmatrix} \boldsymbol{J} \\ \boldsymbol{J}_{add,ortho} \end{bmatrix}, \tag{5}$$

for any $\boldsymbol{Rot} \in SO(r)$, has got first m columns of its inverse matrix exactly the same as the pseudo-inverse $\boldsymbol{J}^{\#}$. Consequently, for a given configuration $\boldsymbol{q}$ and the simplest rotational matrix $\boldsymbol{Rot}$ equal to the identity matrix $\boldsymbol{I}_r$ the values of augmented $\boldsymbol{J}_{add,ortho}$ can be determined. Based on the values, augmenting $\boldsymbol{k}_{add}(\boldsymbol{q})$ can be constructed to have its Jacobian $\boldsymbol{J}_{add}(\boldsymbol{q})$ as close to $\boldsymbol{J}_{add,ortho}(\boldsymbol{q})$ as possible over a given (singular-free) region in a configuration space $\mathbb{Q}$. Usually [6], the region $\boldsymbol{R}$ is assumed to be fixed. Recently, [3] it has been proposed to vary

the region and make it a task-dependent. For computational reasons it will be discretized with a set of configurations. At first, a parametric version of $\boldsymbol{k}_{add}(\boldsymbol{p},\boldsymbol{q})$ is assumed. Then, an iterative algorithm of selecting regions $\boldsymbol{R}$ and optimizing a quality function is run. Let us assume that at the j-th iteration a region $\boldsymbol{R}_j = \{\boldsymbol{q}^1,\ldots,\boldsymbol{q}^{N_j}\}$ is composed of N_j configurations. In the first iteration, $j=1$, $N_j = 1$ and $\boldsymbol{R}_j = \{\boldsymbol{q}(0)\}$. For each item $\boldsymbol{q}^i \in \boldsymbol{R}_j$ a desired $\boldsymbol{J}_{add,ortho}(\boldsymbol{q}^i)$ is computed and the quality function $F(\boldsymbol{p})$ is minimized

$$F(\boldsymbol{p}) = \left\| \frac{\partial \boldsymbol{k}_{add}(\boldsymbol{p},\boldsymbol{q})}{\partial \boldsymbol{q}}\Big|_{\boldsymbol{q}=\boldsymbol{q}^i} - \boldsymbol{J}_{add,ortho}(\boldsymbol{q}^i) \right\|^2, \qquad F(\boldsymbol{p}^\star) = \min_{\boldsymbol{p}} F(\boldsymbol{p}), \tag{6}$$

where $\|\cdot\|$ denotes any matrix norm. The optimal value of $\boldsymbol{p}^\star$ determines $\boldsymbol{k}_{add}(\boldsymbol{p}^\star,\boldsymbol{q})$ and consequently the next current Q-loop. Then the iteration counter j is increased by 1 and the Q-loop, when discretized with N_j points, determines new desired $\boldsymbol{J}_{add,ortho}(\boldsymbol{q}^j)$. The procedure is repeated until a stop condition is satisfied. The simplest and the most natural stop condition is satisfied when a generated Q-loop in the current iteration is not shorter than Q-loop from the previous iteration (at the virtual 0-th iteration, the length of Q-loop is assumed to be infinite).

It is worth noticing that the optimization task (6) in each iteration is decomposed into r independent tasks when consecutive components of $\boldsymbol{k}_{add}(\boldsymbol{p},\boldsymbol{q}) = \{\boldsymbol{k}^c_{add}(\boldsymbol{p},\boldsymbol{q})\}$, $c = 1,\ldots,r$ are determined based on c-th rows of $\boldsymbol{J}_{add,ortho}(\boldsymbol{q}^i)$, $i = 1,\ldots,N_j$.

Now, a single optimization procedure will be described with details (it is repeated $r = n-m$ times for each row of $k_{add}(\boldsymbol{p},\boldsymbol{q})$). Let us express a current row of $k_{add}(\boldsymbol{p},\boldsymbol{q})$) in a parametric, quadratic with respect to $\boldsymbol{q}$, form, and denote it as $h(\boldsymbol{p},\boldsymbol{q})$

$$h(\boldsymbol{p},\boldsymbol{q}) = \sum_{j=1}^{n} p_j q_j + \frac{1}{2}\sum_{j=1}^{n} p_{jj} q_j^2 + \sum_{j=1}^{n}\sum_{k=j+1}^{n} p_{jk} q_j q_k, \tag{7}$$

where $\dim \boldsymbol{p} = n + n(n+1)/2$, $\boldsymbol{p} = (p_1,\ldots,p_n,p_{11},\ldots,p_{nn},p_{12},\ldots,p_{(n-1)n})^T$. The fraction $1/2$ is Eq. (7) is used to simplify consecutive derivations and to present them in a uniform form.

The task to solve is the following: given a set of configurations $\boldsymbol{q}^i$, $i = 1,\ldots,N$, and desired values of rows of the augmented Jacobian matrix $\boldsymbol{v}^i$ at the configurations, find the optimal value of $\boldsymbol{p}$ vector for a linear and a quadratic approximation

$$\widetilde{F}(\boldsymbol{p}) = \sum_{i=1}^{N} \left\| \frac{\partial h(\boldsymbol{p},\boldsymbol{q})}{\partial \boldsymbol{q}}\Big|_{\boldsymbol{q}=\boldsymbol{q}^i} - \boldsymbol{v}^i \right\|^2 = \sum_{i=1}^{N}\sum_{j=1}^{n} \left\| \frac{\partial h(\boldsymbol{p},\boldsymbol{q})}{\partial \boldsymbol{q}_j}\Big|_{\boldsymbol{q}=\boldsymbol{q}^i} - \boldsymbol{v}^i_j \right\|^2. \tag{8}$$

The superscript marks trial (discretization) points and N denotes their number (expression in Eq. (8) is squared to simplify calculations). In [3], explicit closed-form formulas have been presented to get the optimal $\boldsymbol{p}^\star$ with no cross terms present in Eq. (7), i.e. $p_{ij} = 0$ when $i \neq j$. Here, the general case is considered and $\boldsymbol{p}^\star$ can be obtained by a numerical solution of a set of linear equations. A necessary stationary condition for the optimum is

$$\frac{\partial \widetilde{F}(\boldsymbol{p})}{\partial \boldsymbol{p}} = \boldsymbol{0}. \tag{9}$$

Taking into account (7)

$$\frac{\partial h(\boldsymbol{p},\boldsymbol{q})}{\partial q_j} = p_j + \sum_{k=1}^{n} p_{jk} q_k \tag{10}$$

(to simplify notations it is assumed that $p_{ij} = p_{ji}$ if only $j < i$) conditions resulting from Eq. (9) can be divided into three groups. The first equations, are derived for variables p_r, $r = 1,\ldots,n$

$$\frac{\partial \widetilde{F}(\boldsymbol{p})}{\partial p_r} = 0 \quad \Rightarrow \quad N p_r + \sum_{i=1}^{N}\sum_{k=1}^{n} q_k^i p_{rk} = \sum_{i=1}^{N} v_r^i \tag{11}$$

The next equations are obtained for variables p_{rr}, $r = 1,\ldots,n$

$$\frac{\partial \widetilde{F}(\boldsymbol{p})}{\partial p_{rr}} = 0 \quad \Rightarrow \quad \left(\sum_{i=1}^{N} q_r^i\right) p_r + \sum_{i=1}^{N}\sum_{k=1}^{n} q_k^i q_r^i p_{rk} = \sum_{i=1}^{N} v_r^i q_r^i \tag{12}$$

and finally, the equations for remaining variables p_{rs}, $r = 1,\ldots,n$, $s = r+1,\ldots,n$

$$\left(\sum_{i=1}^{N} q_r^i\right) p_r + \left(\sum_{i=1}^{N} q_s^i\right) p_s + \sum_{i=1}^{N}\sum_{k=1}^{n} q_k^i q_r^i p_{rk} + \sum_{i=1}^{N}\sum_{k=1}^{n} q_k^i q_s^i p_{sk} = \sum_{i=1}^{N} (v_r^i q_r^i + v_s^i q_s^i). \tag{13}$$

Based on Eqns. (11), (12), (13), a linear equation

$$\boldsymbol{A}\boldsymbol{p} = \boldsymbol{b} \tag{14}$$

with known $\boldsymbol{A}_{\dim \boldsymbol{p} \times \dim \boldsymbol{p}}$ and $\boldsymbol{b}_{\dim \boldsymbol{p}}$ can be constructed and solved to get $\boldsymbol{p}^\star$ minimizing (8). In order to reduce a computational effort, the following items should be calculated first

$$\sum_{i=1}^{N} q_r^i, \quad \sum_{i=1}^{N} q_r^i q_s^i, \quad \sum_{i=1}^{N} v_r^i, \quad \sum_{i=1}^{N} v_r^i q_r^i, \quad r = 1,\ldots,n, \quad s = r,\ldots,n$$

and substituted into appropriate boxes of the matrix $\boldsymbol{A}$ and vector $\boldsymbol{b}$. For a special, linear case of the augmented kinematic function (10) ($p_{ij} = 0$, $i = 1,\ldots,n$, $j = i,\ldots,n$), $\boldsymbol{p}^\star$ is computed directly from Eq. (11).

2.3. An elastic band algorithm of repeatable inverse kinematics (Method 2)

This exemplification of a continuation method generates a final solution, Q-loop, in an iterative process. The first approximation of the Q-loop (not necessarily satisfying all requirements imposed on the original task) should be known. Then, a sequence of solutions is constructed to decrease a quality function which evaluates how far a current solution is from that satisfying all requirements. In the limit, a current Q-loop tends to the solution of the repeatable inverse kinematics.

In the elastic band method of repeatable inverse kinematics, a role of the deformed path will play a cyclic trajectory, Q-loop, in the configuration space. It is modified in such a way that a trajectory remains cyclic and its image, via forward kinematics, is closer to the prescribed X-loop. In the following implementation, a current Q-loop, called L_{disc} is represented as a vector of pairs: (s-variable value, configuration) and interpolated linearly to get the continuous Q-loop, $L_{cont}(s)$, $s \in [0, s_{\max}]$. The vector is ordered with respect to the s-variable. Main steps of the algorithm implementing the elastic band idea follow:

Step 1 Read in initial X-loop (1) and an initial (current) $L_{disc} = \{(0,\boldsymbol{q}(0)),(s_{max},\boldsymbol{q}(0))\}$.

Step 2 Define an error which evaluates how far the X-loop corresponding to current Q-loop is from X-loop (1)

$$err(L_{disc}) = \int_{s=0}^{s_{\max}} \|\boldsymbol{x}(s) - \boldsymbol{k}(L_{cont}(s))\| ds, \tag{15}$$

where $\|\cdot\|$ is a norm in $\mathbb{X} \subset \mathbb{SE}(3)$.

Step 3 Compute error (15) for L_{cont} (derived from L_{disc}) .

Step 4 If the error is below an acceptable threshold ρ, complete the computations and output L_{cont}. Otherwise progress with Step 5.

Step 5 Modify the current Q-loop:

1. On a prescribed path (1) determine the furthest point $\boldsymbol{x}(s^\star)$ from $\boldsymbol{k}(L_{cont}(\cdot))$.
2. For $s^\star$, select from L_{disc} two neighbor configurations $(s_j,\boldsymbol{q}(s_j))$, $(s_{j+1},\boldsymbol{q}(s_{j+1}))$ such that $s_j < s^\star < s_{j+1}$ and denote them as $\boldsymbol{q}^1 = \boldsymbol{q}(s_j)$, $\boldsymbol{q}^2 = \boldsymbol{q}(s_{j+1})$.

3. Run the Newton algorithm (3) with the optimization in the null space of the Jacobian matrix and the following data:
   - the goal point $\boldsymbol{x}(s^\star)$,
   - the initial configuration

$$\boldsymbol{q}_{init} = \lambda \boldsymbol{q}(s_j) + (1-\lambda)\boldsymbol{q}(s_{j+1}), \qquad \lambda = \frac{s^\star - s_j}{s_{j+1} - s_j}, \tag{16}$$

   - the minimized function used in Eq. (3)

$$f(\boldsymbol{q}) = \|\boldsymbol{q} - \boldsymbol{q}^1\|^2 + \|\boldsymbol{q} - \boldsymbol{q}^2\|^2 \tag{17}$$

   which evaluates a squared distance from the current configuration $\boldsymbol{q}$ to a pair of configurations $\boldsymbol{q}^1, \boldsymbol{q}^2$. The aim of the minimization is to minimally stretch a current Q-loop while reaching the current goal $\boldsymbol{x}(s^\star)$.
4. Result of the Newton algorithm run: the pair $(s^\star, \boldsymbol{q}(s^\star))$, satisfying $\boldsymbol{k}(\boldsymbol{q}(s^\star)) = \boldsymbol{x}(s^\star)$, is added into appropriate place in L_{disc} to preserve ordering of the sequence wrt the s-variable.

Step 6 The modified Q-loop becomes a new current trajectory. Continue with Step 3.

3. Comparison of the methods

It is really difficult to compare methods different in nature although aimed at solving the same task. However, two criteria are quite obvious: a solution accuracy and complexity of algorithms implementing the methods. In the theory of computations [13], a computational complexity is measures as a function of a data size. In most of robotic tasks this definition has got a restricted applicability as the number of degrees of freedom (a natural data size) is severely limited and usually small. Another, and practical, measure of algorithms' complexity is their run-time. This criterion has got its own disadvantages as the run-time depends on many factors: hardware-based (processor, ticking time, number of cores, a cache memory available), software-based (a programming language, options of a compiler used) and human-based (skillfulness of the algorithm programmer), an algorithm-based (number of parameters and their settings). Despite all these disadvantages, the run-time qualitatively evaluates algorithms quite well and it will be used later on.

The other criterion, the accuracy of solution, seems to be easier to evaluate at the first sight. However, the accuracy of a solution depends on some parameters

fixed while solving a given task. Besides common input data for both algorithms: a model of manipulator, an initial X-loop and configuration $\boldsymbol{q}(0)$ there are also method-dependent parameters. For Method 1: a threshold value ρ to stop computations, cf. (15), a threshold value δ to stop the Newton algorithm (3) and a method to solve two dimensional (ξ, η) optimization problem in the algorithm. For Method 2: base functions (polynomials, harmonic) and their degree used to express augmented functions and a threshold value δ to stop the basic, $\eta = 0$, Newton algorithm (3) are to be selected.

It is worth noticing that the two evaluation criteria are tightly related. A shorter run-time usually means a worse accuracy. However other criteria should not be missed. Method 2 does not add any extra functions and avoids a computationally expensive problem of their optimization. This method also does not introduce any extra singular configurations (to those defined by the condition $rank(J(\boldsymbol{q})) < m$) (Method 1 does it, as singular configurations are characterized by $rank(J_{aug}(\boldsymbol{q})) < n$ and include all singular configurations of Method 2). In the Newton algorithm (3) inverses of the Jacobians are performed on $(m \times m)$ matrices for Method 2 and on $(n \times n)$ matrices for Method 1. For a significant redundancy value, $r = n - m$, the factor is important as the Newton algorithm is run frequently.

Based on the discussed criteria Method 2 is better suited to solve the repeatable inverse kinematics than Method 1. Moreover, Method 2 does not involve iterative processing and it is much easier to implement.

4. Simulations

In order to evaluate the proposed methods, planar pendula manipulators up-to five degrees of freedom (DoF) are selected, Fig. 1. For the manipulators a high degree of redundancy is obtained easily and their task-space (assuming an unlimited motion at joints) is easy to find. It facilitates determining an admissible

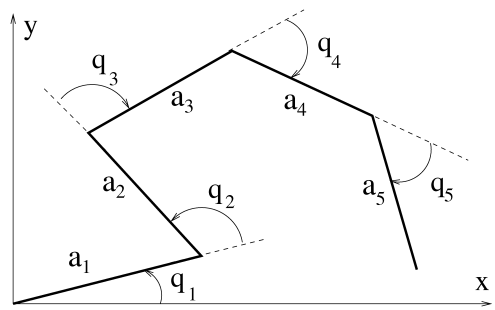

Figure 1: Planar pendula up to 5 DoFs: a configuration, parameters, the task-space

X-loops within a task-space. Forward positional kinematics of the pendula are given by

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} k_1(\boldsymbol{q}) \\ k_2(\boldsymbol{q}) \end{bmatrix} = \begin{bmatrix} a_1c_{1,1} + a_2c_{1,2} + a_3c_{1,3} + a_4c_{1,4} + a_5c_{1,5} \\ a_1s_{1,1} + a_2s_{1,2} + a_3s_{1,3} + a_4s_{1,4} + a_5s_{1,5} \end{bmatrix}, \tag{18}$$

where $\boldsymbol{a} = (a_i, i = 1, \ldots, n)$ is a vector of links' lengths. The lengths are set to the value of 1 in all simulations. A standard robotic convention is used to abbreviate trigonometric functions, $c_{1,i} = \cos\left(\sum_{j=1}^{i} q_j\right)$, and $s_{1,i} = \sin\left(\sum_{j=1}^{i} q_j\right)$. While running the Newton algorithm (3), it is assumed that a total change of a configuration $\|\boldsymbol{q}_{i+1} - \boldsymbol{q}_i\|$ at a single iteration is limited to 3° to preserve accuracy of the linearization of kinematics and the threshold value to stop the algorithm is set to $\delta = 0.005$. Two typical, practical and easy to parameterize X-loops are considered:

1) a circle centered at (x_c, y_c) with the radius R

$$\boldsymbol{x}(s) = x_c + R\cos(2\pi s), \qquad y(s) = y_c + R\sin(2\pi s), \quad s \in [0,1], \tag{19}$$

2) a rectangle with edges parallel to x-y axes characterized by its left-bottom corner (x_c, y_c) and edge lengths $\Delta x, \Delta y$. X-loop $\boldsymbol{x}(s)$, $s \in [0,1]$ traces the rectangle anti-clockwise where s denotes a linearly scaled current X-loop length.

Simulations are performed for planar pendula with the smallest, $n = 3$, and significant, $n = 5$, redundancy and circular and rectangular paths of two different lengths. In order to avoid an impact of a selected entry configuration $\boldsymbol{q}_0$ ($\boldsymbol{k}(\boldsymbol{q}_0) = \boldsymbol{x}(0)$) on final results and to get a statistical insight into obtained data, a number of tasks with varied $\boldsymbol{q}_0$ is considered. The entry configurations have been determined with algorithm described in [4]. For each task a mean and standard deviation of a Q-loop length are calculated. Additionally, the shortest Q-loop, L_{best}, the number of total tasks and failed tasks are counted. It should be noticed that admissibility of entry configuration $\boldsymbol{q}_0$ does not necessary mean admissibility of a Q-loop (as the Newton algorithm (3) may fail due to approaching any singular configuration at some iteration). Algorithms of the methods were implemented in Mathematica and run on a computer equipped with Intel Core2 2.66 GHz processor. The mean and dispersion of run-time were computed. All the resulting data together with parameters of tested X-loops are gathered in Table 1. Based on the data collected in Table 1 the following remarks can be formulated.

- The elastic band method, Method 2, runs much faster than both versions of the augmented Jacobian method, Method 1 lin/quad, cf. $\overline{T}$ rows. It needs

Table 1: Initial data and statistical results for Method 1 (approximations with with linear and quadratic functions), and Method 2 (elastic band)

		Task number							
		1	2	3	4	5	6	7	8
init data	DoFs	3				5			
	path	circle		rectangle		circle		rectangle	
	(x_c, y_c)	$(1.5, 0)$	$(1.5, 0)$	$(1, -0.5)$	$(1, -1)$	$(2.5, 0)$	$(2.5, 0)$	$(1, -1)$	$(1, -1)$
	R	0.5	1			0.75	1.5		
	$\Delta x,\ \Delta y$			0.5, 1	0.5, 2			0.5, 1	0.5, 3
Method	total	122	78	226	178	234	86	1024	1024
elastic (2)	failed	0	0	0	0	0	0	0	0
	$\overline{T}$ [s]	0.05	0.085	0.042	0.07	0.074	0.11	0.048	0.11
	σ_T [s]	0.003	0.005	0.005	0.005	0.004	0.006	0.005	0.01
	$\overline{L}$ [°]	163.8	310.6	155.6	237.9	144.3	265.9	110.9	241
	σ_L [°]	19.8	36.9	21.7	22.3	22.2	30.9	18.4	31.2
	L_{best} [°]	128.7	270.8	124.5	201.7	101.4	214	68.1	154
linear (1 lin)	failed	8	52	6	2	38	36	73	304
	$\overline{T}$ [s]	0.48	0.63	0.09	0.23	0.72	0.95	0.24	0.31
	σ_T [s]	0.3	0.18	0.08	0.24	0.67	0.82	0.2	0.39
	$\overline{L}$ [°]	169.3	313.3	155.3	237.2	148.6	282.6	109.5	248.5
	σ_L [°]	25.7	37	23.8	24.7	26	40.8	19.2	36.4
	L_{best} [°]	128.2	269.7	122.2	199.2	101	217.8	66.3	156
quadrat (1 quad)	failed	5	52	12	11	54	48	116	404
	$\overline{T}$ [s]	0.85	0.67	0.42	0.83	1.3	1.24	0.99	1.54
	σ_T [s]	0.66	0.52	0.45	0.86	0.84	0.75	0.85	2.31
	$\overline{L}$ [°]	169.7	313	155.8	236.7	150.3	289.2	109.8	252.4
	σ_L [°]	26.6	37.5	24.6	24.9	26.3	42	19	35.6
	L_{best} [°]	128.2	269.1	122.1	199.2	101.2	223.3	66.2	164

only one iteration to generate a resulting Q-loop while the two versions may require a few iterations to converge. Moreover, using the Newton algorithm (3) Method 2 inverts matrices of smaller sizes than the other methods.

- Method 2 is more robust to initial data (the number of failed tasks is significantly smaller than for Methods 1) as it does not introduce any extra singularities. Method 1lin seems to be less sensitive to failures than

Method 1quad as it is easier to fail due to ill-conditioned matrix $\boldsymbol{A}$ in Eq. (14) than ill-conditioning of Eq. (11).

- The best, shortest Q-loops generated with Method 1 and 2 were more or less the same, cf. L_{best}. Exemplary stroboscopic views of the shortest Q-loops for Task 8 are presented in Figure 2. It is worth to observe that the shortest Q-loops are realized with short motions in q_1 coordinate. This motion is transferred, via a long kinematic chain, into large displacements of the end-effector to trace the given rectangle.

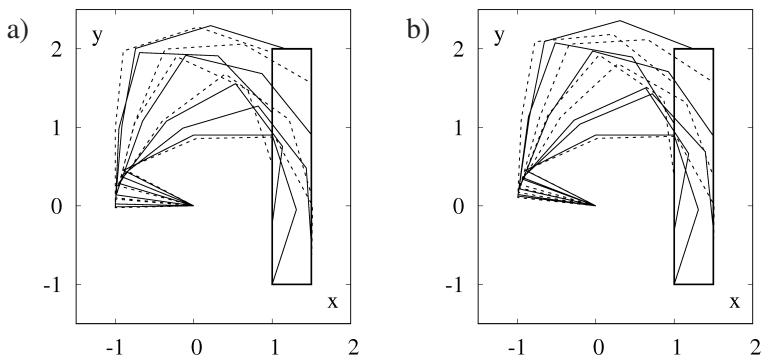

Figure 2: The stroboscopic view of the shortest Q-loops for an exemplary rectangular path of the 5DoF planar pendulum (Task 8) obtained with: a) Method 2, b) Method 1 lin

- It does not seem that the quadratic approximation improves quality of the resulting Q-loop comparing to a linear approximation. It should be pointed out that the optimization in Methods 1 is a path-dependent process (at any iteration Q-loop is deformed differently by the two versions of Method 1).

- On average, Method 2 gives shorter Q-loops than Method 1, cf. $\overline{L}$ and with a little bit smaller dispersion σ_L. The histogram of Q-loop length differences for Method 1lin and Method 2, Δ, solving Task 8 is presented in Fig. 3. It appear that in rare cases Method 1 given significantly longer paths than Method 2 ($\Delta > 60^\circ$).

- Run-time of Method 2 is proportional to the total length of X-loop. For Method 1 this regularity does not hold as the optimization process is path-dependent in the Q-space.

- For both methods, an efficiency of planning the shortest Q-loop strongly depends on the initial configuration $\boldsymbol{q}(0)$, cf. L_{best}, $\overline{L}$. This dependence is amplified by increasing redundancy $r = n - m$ as in this case there is more

initial configurations to choose. The longest Q-loop solving Task 8 with Method 2 was $L_{worst} = 338.4^\circ$ while for Method 1lin – $L_{worst} = 391.2^\circ$. The values are significantly larger than average ones, cf. Table 1.

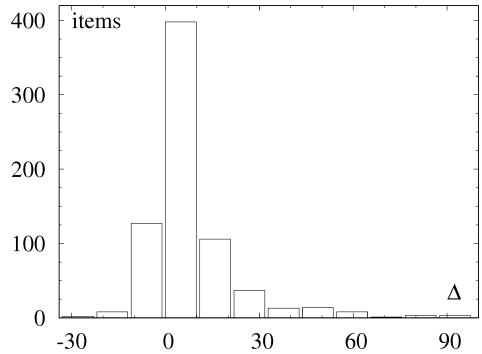

Figure 3: Histogram of differences of Q-loop lengths Δ generated with Method 1 lin and Method 2 while solving Task 8 with the same and varied initial configurations. The total number of tasks simultaneously solved 720

5. Conclusions

In this paper two methods of repeatable inverse kinematic task are compared. The first one solved the task via approximating a pseudo-inverse Jacobian matrix with linear and quadratic polynomials and exploited an iteratively modified task-dependent region in the configuration space to evaluate the quality function (length of a resulting Q-loop minimized). The second one adapted an elastic band method to the title task. Simulations and theoretical considerations allow to draw a conclusion that the elastic band method is more reliable than approximation techniques in solving the repeatable inverse kinematic task and it has got several advantages over the approximation methods (significantly shorter run-time, easy implementation, a smaller impact of configuration singularities on solvability of the task).

References

[1] S. CHIAVERINI, G. ORIOLO and I.D. WALKER: Handbook of Robotics, chapter Kinematically redundant manipulators, 245–268, Springer Verlag, Berlin, 2008.

[2] I. DULEBA and M. OPAŁKA: On application of elastic band method to repeatable inverse kinematics in robot manipulators, *Journal of Automation, Mobile Robotics and Intelligent Systems*, **7**(4), (2013), 5–12.

[3] I. DULEBA and I. KARCZ-DULEBA: Suboptimal approximations in repeatable inverse kinematics for robot manipulators *Bull. Pol. Ac.: Tech.*, **65**(2), (2017), 209–217.

[4] I. DULEBA and I. KARCZ-DULEBA: A suboptimal solution of repeatable inverse kinematics in robot manipulators with a free entry configuration, Proc. of Mediterranean Conf. on Control and Automation, Athens, (2016), 563–568.

[5] G.H. GOLUB and C. REINSCH: Singular value decomposition and least squares solutions, *Numerische Mathematik*, **14**(5), (1970), 403–420.

[6] J. KARPIŃSKA: Approximation of algorithms for robot motion planning, PhD. dissertation, Wroclaw University of Technology, (2012), in Polish.

[7] S. QUINLAN and O. KHATIB: Elastic bands: Connecting path and control, Prof. IEEE Int. Conf. on Robotics and Automation, Vol. 2 (1993), 802–807.

[8] C.A. KLEIN, C. CHU-JENG and S. AHMED: A new formulation of the extended Jacobian method and its use in mapping algorithmic singularities for kinematically redundant manipulators, *IEEE Trans. on Robot. Autom.*, **11**, (1995), 50–55.

[9] Y. NAKAMURA: Advanced robotics: redundancy and optimization, Addison Wesley, New York, 1991.

[10] J.E. RATAJCZAK, K. TCHON and M. JANIAK: Approximation of Jacobian inverse kinematics algorithms: differential geometric vs. variational approach, *Journal of Intelligent and Robotic Systems*, **68**(3/4), (2012), 211–224.

[11] S. RICHTER and R. DECARLO: Continuation methods: theory and applications, *IEEE Trans. on Automatic Control*, **28**(6), (1983), 660–665.

[12] R. ROBERTS and A.A. MACIEJEWSKI: Repeatable generalized inverse control strategies for kinematically redundant manipulators, *IEEE Trans. on Automatic Control*, **38**, (1993), 689–699.

[13] M. SIPSER: Introduction to the Theory of Computations, 2nd ed., Thomson Course Technology, 2006.