On the Application of the Elastic Band Method to Repeatable Inverse Kinematics in Robot Manipulators

Submitted: 20th January 2013; accepted: 4th June 2013

Ignacy Duleba, Michal Opalka

DOI: 10.14313/JAMRIS_4-2013/5

Abstract:
In this paper an idea of the elastic band method was exploited to design a repeatable inverse kinematics algorithm for robot manipulators. The method incorporates an optimization process at many stages of its performance and admits some extensions. Performance of the algorithm was illustrated on models of the three DOF planar pendulum and the PUMA robot. A comparison with a standard pseudo-inverse Jacobian algorithm, which does not preserve repeatability of inverse kinematics, is also provided.

Keywords: *inverse kinematics, repeatability, algorithm*

1. Introduction

Robot manipulators are frequently used to perform repeatable tasks (assembling, drilling, painting) on a factory floor. From a technological point of view, it is desirable to follow exactly the same trajectory by a manipulator when performing a repeatable task. In this case the problem of avoiding obstacles can be significantly reduced as only one trajectory should be tested for the collision avoidance. Therefore, one of the most popular tasks in robot manipulators is a repeatable inverse kinematic task. The task is to find a loop (closed path) in a configuration space that corresponds to a prescribed cycle of the end-effector in a taskspace [1]. This task is meaningful only for redundant manipulators because for non-redundant manipulators a solution of the inverse kinematic task is determined uniquely if only a trajectory does not meet any singular configuration.

A standard approach to construct repeatable inverse kinematic algorithms relies on extending original (redundant) kinematics by some extra functions ($n - m$ items) to get extended kinematics which is non-redundant [10]. Outside singular configurations of the original kinematics, the added functions should preserve full rank of the square Jacobian matrix of the extended kinematics (in other words, they should not introduce extra singular configurations not present in the original kinematics).

This constraint is not too restrictive thus there exist many functions to fulfill the aforementioned minimal requirement. So, naturally, one wants to get an optimal set of functions with respect to a prescribed quality function [5]. This line was exploited in Karpinska's PhD thesis [4].

In this paper we follow a quite different approach that does not add any extra functions and avoids a computationally expensive problem of their optimization. The proposed approach is based on the elastic band method, developed by Quinlan and Khatib [8], which was primarily designed to optimize a path of a mobile robot [2]. The idea of this method is the following: an initial admissible path is assumed, then it is deformed iteratively to optimize a given quality function until a stop condition is satisfied. The deformation was originally based on a potential field method that tried to avoid obstacles, by maximizing a distance from the path to obstacles, and make the path short and smooth. This idea is very close to a more general approach known in mathematics as a continuation method [9]. The continuation method construct a final solution also in an iterative process. The first approximation (maybe not admissible, i.e. not satisfying all requirements imposed on the original task) of the solution should be known (and usually it is simple to determine). Then, a sequence of solutions is constructed which decrease a certain quality function that measures how far the current solution is from that one which satisfies all requirements. In the limit a solution of the original task is determined which satisfies all requirements imposed.

In our adaptation of the elastic band method to repeatable inverse kinematic tasks a role of the deformed path will play a cyclic trajectory in the configuration space. It will be modified in such a way that the modified trajectory remains cyclic and its image, via forward kinematics, is closer to the prescribed cyclic path in the taskspace.

In this paper we use the term trajectory as a synonym for a function in a configuration space parametrized with the same variable that parametrize the prescribed path in the taskspace. Usually the trajectory is described as a function of time.

The paper, being the extended version of [3], is organized as follows. In Section 2 the repeatable inverse kinematic task is defined formally. In this section an algorithm based on the elastic band method to solve the task is discussed with details. Simulation results of the proposed algorithm are collected in Section 3. Section 4 concludes the paper.

2. Elastic band method in repeatable inverse kinematic task

Forward kinematics of a robot manipulator is a mapping [11]

$$k: \; Q \ni q \to x = k(q) \in X \subset SE(3), \qquad (1)$$

where q is a configuration living in the configuration space Q and x denotes a point in the taskspace X. For redundant manipulators dimensionallity of the configuration space is larger than dimensionallity of the taskspace,

$$\dim Q = n > m = \dim X. \tag{2}$$

With the kinematics (1) a Jacobian matrix is associated and given by $J(q) = \partial k/\partial q$.

Let a loop (a closed path)

$$\{x(s), s \in [0, s_{\max}], \;\; x(0) = x(s_{\max})\} \tag{3}$$

be given in the taskspace and also an initial configuration q_0 is known corresponding to the start point of the loop, $k(q_0) = x_0$. The task of repeatable inverse kinematics is to find a trajectory $q(s)$ in the configuration space satisfying

$$\forall_{s\in[0,s_{\max}]}\, k(q(s)) = x(s), \;\; q(0) = q(s_{\max}) = q_0. \tag{4}$$

Additionally, the sum of squares of lengths of the subtrajectories in the configuration space

$$\int_0^{s_{\max}} \left(\frac{\partial q}{\partial s}\right)^T \left(\frac{\partial q}{\partial s}\right) ds = \int_0^{s_{\max}} \langle \frac{\partial q}{\partial s}, \frac{\partial q}{\partial s}\rangle ds. \tag{5}$$

should be minimized. In (5) T stands for transposition, and $\langle\cdot,\cdot\rangle$ introduces the inner product. We prefer to evaluate a trajectory according to Eq. (5) as it is faster to compute than to determine the total length of the trajectory.

The basic algorithm exploited in solving the aforementioned task is the Newton algorithm designed to solve the inverse kinematics when the goal is to reach a single point in the taskspace. The algorithm is described by the iteration process [7, 11] with a possible optimization in the null space of the Jacobian

$$\begin{aligned} q_{i+1} = \;\; & q_i + \xi_1 \cdot J^{\#}(q_i)\,(x_f - k(q_i)) + \\ & + \xi_2 \left(I - J^{\#}(q_i)J(q_i)\right) \tfrac{\partial f(q)}{\partial q}|_{q=q_i}, \end{aligned} \tag{6}$$

where $J^{\#} = J^T(JJ^T)^{-1}$ is the pseudo-inverse of the Jacobian matrix, q_0 is the initial configuration of the manipulator, x_f denotes a current goal point in the taskspace, I is the $(n \times n)$ identity matrix and a scalar function $f(q)$ is to be optimized (the optimization can be switched-off by setting the value of ξ_2 to 0.

Below main steps of the algorithm based on the elastic band method applied to repeatable inverse kinematic task are presented

Step 1 Select an initial loop in the configuration space

$$q(s), s \in [0, s_{\max}], \quad (q(0) = q(s_{\max})).$$

The loop becomes a current trajectory.

Step 2 Define an error function that assigns to a cyclic trajectory $q(\cdot)$ accumulated distance of its image (via forward kinematics) to the traced path $x(\cdot)$ in the taskspace

$$err(q(\cdot)) = \int_{s=0}^{s_{\max}} ||x(s) - k(q(s))||ds, \tag{7}$$

where $||\cdot||$ is an assumed norm.

Step 3 Compute the error (7) for the current trajectory.

Step 4 If the error is below an acceptable threshold, complete the computations and output the resulting current trajectory. Otherwise progress with Step 5.

Step 5 Modify the current trajectory (either at a single point or on an interval) preserving cyclicity of the trajectory with the aim to decreased the value of error (7). The modified trajectory becomes the new current trajectory. Continue with Step 3.

More details concerning the algorithm follow.

- A few terms and notations extensively used later on are introduced:
  - a node point is any point placed on the given path $x(\cdot)$,
  - a node configuration is a configuration which image via forward kinematics is a node point,
  - a node configuration q when included into a trajectory $q(s)$ gets the same argument s as the corresponding node point $x(s)$. The argument s of $x(s)$ is determined uniquely,
  - $(q(s), x(s)) = Newton(q_0, x(s), f(q))$ denotes that the Newton algorithm (6) with the initial configuration q_0, the fixed goal point $x(s)$ (as s is fixed) and minimizing the quality function $f(q)$ was run and produced as its output the pair node configuration - node point $(q(s), x(s))$.
- A selection of the initial (passing through q_0) cyclic trajectory is unrestricted. However, its selection impacts both the speed of convergence as well as its optimality. The best natural choice (and simplest one as well) is a trivial loop, i.e. $\forall s \in [0, s_{\max}]\, q(s) = q_0$.
- Consecutive node configurations are interpolated linearly, although higher order polynomials are also possible.
- An invariant of the algorithm (characteristics that does not change from one iteration to another) is the cyclicity of the current trajectory. Initially there is a trivial loop $q(0) \to q(s_{\max}) = q(0)$. After adding a single node configuration, say $q(s_{\max}/2)$ corresponding to $x(s_{\max}/2)$, the closed loops is the following $q(0) \to q(s_{\max}/2) \to q(s_{\max}) = q(0)$, etc.
- The norm used in Eq. (7) depends on the nature of the taskspace. When all coordinates are positional, the Euclidean norm will be the most proper. When the taskspace includes also angle coordinates some norm inherited from the special Euclidean group $SE(3)$ [12] should be used. To simplify computations, the integral (7) can be replaced by a sum of local errors for arguments s between currently existing node configurations.
- The key step of the algorithm, Step 5, can be implemented at least in two possible ways:

  Scheme 1: on a prescribed path $x(\cdot)$ the furthest node point $x(s^\star)$ from the $x(0)$ is determined.

Then, the Newton algorithm is run

$$(q(s^\star), x(s^\star)) = Newton(q(0), x(s^\star), f(q)).$$

where $f(q) = ||q - q(0)||$. The following computations are defined recursively. Let us assume that a sequence $S = ((q(s_i), x(s_i)), s_i < s_{i+1})$ of pairs node configuration-node points has been already determined. After the first stage we have $S = ((q(0), x(0)), (q(s^\star), x(s^\star)), (q(s_{max}), x(s_{max})))$. Note that the sequence is cyclic (i.e. the last pair in the sequence is the same as the first one; the only difference is that the argument is equal to $s_{\max}$ instead of 0). In each (or some) of the sub-intervals $[s_i, s_{i+1}]$ the furthest node $x(s^\star)$ from the line $\overline{x(s_i)x(s_{i+1})}$ is selected. The Newton algorithm is run

$$(q(s^\star), x(s^\star)) = Newton(q_{ini}, x(s^\star), f(q))$$

initialized at the configuration

$$q_{ini} = \lambda q(s_i) + (1 - \lambda) q(s_{i+1}), \qquad (8)$$

with

$$\lambda = \frac{s^\star - s_i}{s_{i+1} - s_i} \in (0, 1). \qquad (9)$$

q_{ini} is the weighted sum of the two neighboring node configurations while the optimized function is given as

$$f(q) = ||q - q(s_i)|| + ||q - q(s_{i+1})||. \qquad (10)$$

The resulting pair $(q(s^\star), x(s^\star))$ is added to S and the S is sorted to get $\forall i \; s_i < s_{i+1}$. This idea is illustrated graphically in Fig. 1a.

Scheme 2: let a sequence of node points be given $(x(s_1), \ldots, x(s_k))$ with $0 < s_i < s_{i+1} < s_{\max}$. Simultaneously, a set of $i \in \{1, \ldots, k\}$ Newton algorithms is run $Newton(q(0), x(s_i), f(q))$ with the quality function, cf. Fig. 2

$$f(q) = ||q - q(s_i^j)|| + ||q - q(s_{i+1}^j)||, \qquad (11)$$

where $q(s_i^j)$ denotes the configuration obtained in the j-iteration of the Newton algorithm run for the i-the node point. Obviously, not all tasks $i \in \{1, \ldots, k\}$ will be completed after the same number of iterations. If the i-th task has been finished after j_i iterations, it is assumed that $\forall j > j_i : q(s_i^j) = q(s_i^{j_i})$. Graphically this idea is visualized in Fig. 1b.

The two schemes rely on the same idea: generate a new configuration to approach the required node point while increasing the length of trajectory as small as possible. However, the first scheme realizes it sequentially while the second one – in parallel.

The procedure of generation new node configurations is repeated until their number is large enough to considered the resulting trajectory to be the solution of the repeatable inverse kinematic task. To this

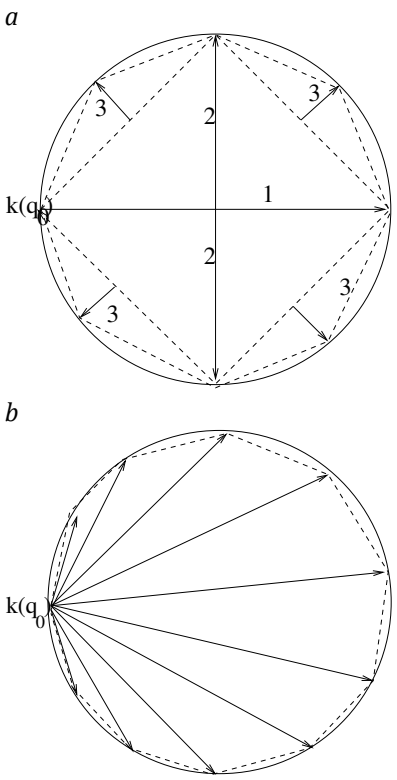

Fig. 1. a – generation of node points with Scheme 1, (numbers correspond to a phase of adding the nodes), b – generation of node points with Scheme 2

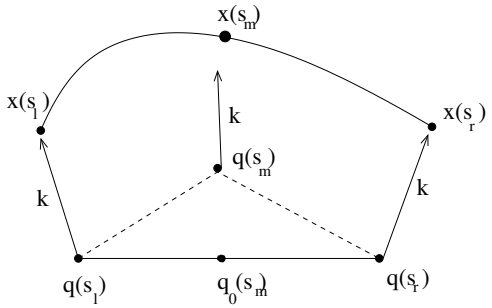

Fig. 2. Evolution of configurations from $q_0(s_m)$ ***to*** $q(s_m)$ ***to reach the point*** $k(q(s_m)) = x(s_m)$

aim let us define for a set of node configurations $S_Q = \{(q(s_i)), = 1, \ldots, K\}$, extracted from the sequence S the distance d

$$d(k(S_Q), x(\cdot)) = \max_{i=1,K-1} ||k\left(\frac{q(s_i)+q(s_{i+1})}{2}\right) - x\left(\frac{s_i+s_{i+1}}{2}\right)||. \qquad (12)$$

If the condition

$$d(k(S_Q), x(\cdot)) \leq \delta, \qquad (13)$$

(where δ denotes an assumed acceptable error) is satisfied, the loop in the configuration space has been found (with the linear interpolation of consecutive node configurations as an output). Otherwise in those segments where condition (13) was violated, computation progresses according to a selected scheme.

One more aspect of the algorithm (6) with optimization in the null space of the Jacobian matrix ($\xi_2 \neq$

0) should be considered because this algorithm is run many times and it greatly impact overall computational complexity.

The simplest choice of coefficients ξ_1, ξ_2 is to set their values small and constant. However, probably this is not the best possible choice. To support this claim let us notice that

1) computational scheme (6) is valid only for a small configuration increase $\Delta q_i = \xi_1 \Delta q_i^1 + \xi_2 \Delta q_i^2$, because Eq. (6) is based on linearization of kinematics around a current configuration
2) the nature of components

$$\Delta q_i^1 = J^{\#}(q_i)(x_f - k(q_i)) \tag{14}$$

and

$$\Delta q_i^2 = \left(I - J^{\#}(q_i)J(q_i)\right) \frac{\partial f}{\partial q}|_{q=q_i} \tag{15}$$

are quite different although both generate values in R^n. They values may strongly depend on a stage of computations. For example the first component (14) takes smaller and smaller values as a point corresponding to a current configuration gets closer and closer to the goal point in the taskspace. The first component (14) is responsible for convergence of the algorithm (6) so it should have got higher priority than the other (15), responsible for optimization of $f(q)$.

Therefore in practical implementation of the Newton algorithm (6) we propose the following procedure: let us assume that the length of admissible change of configuration in a single iteration is equal to Δ. In a current configuration q_i, the components $\Delta q_i^1, \Delta q_i^2$ are computed. Then, the values of ξ_1, ξ_2 are set to get $||\Delta q_i|| = \Delta$. Finally, one dimensional optimization process is invoked with the quality function $f(q_i + \xi_1 \Delta q_i^1 + \xi_2 \Delta q_i^2)$ and the independent variable ξ_2. $\xi_1(\xi_2)$ is computed from

$$||\xi_1 \Delta q_i^1 + \xi_2 \Delta q_i^2|| = \Delta \tag{16}$$

while the convergence condition

$$||k(q_i + \xi_1 \Delta q_i^1 + \xi_2 \Delta q_i^2) - x_f|| < ||k(q_i) - x_f||. \tag{17}$$

must hold. If the condition (17) is not satisfied (i.e. there does not exist admissible ξ_2), the value of Δ should be decreased and once again the procedure is run. Obviously, in a close vicinity of the goal point the value of Δ is decreased obligatory, $\Delta = ||k(q_i) - x_f||$.

Because forward kinematics $k(q)$ as well as the quality function $f(q)$ are continuous functions of q, the aforementioned optimization process is convergent outside singular configurations (where the matrix $J^{\#}$ becomes ill-posed), and the convergence is relatively fast because in one dimensional optimization only computationally cheap kinematics is invoked.

It is worth noticing that in a close vicinity of singular configurations a robust inverse pseudo-inverse should be used. Temporarily, $J^{\#}$ is replaced there by $J^T(JJ^T + \gamma I)^{-1}$ with a small parameter γ. Consequently, the loop in the configuration space can be generated even in this case.

3. Simulations

An illustration of the elastic band method to repeatable inverse kinematic task was performed on two robots. The first model is the 3-dof planar pendulum visualized in Fig. 3 with its kinematics given by

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_1 c_1 + a_2 c_{12} + a_3 c_{123} \\ a_1 s_1 + a_2 s_{12} + a_3 s_{123} \end{bmatrix}, \tag{18}$$

where a_i denote lengths of manipulator's links. A standard robotic convention is exploited to abbreviate trigonometric functions, $c_{12} = \cos(q_1 + q_2)$, $s_{123} = \sin(q_1 + q_2 + q_3)$. In all simulations all lengths of links of the planar pendulum were assumed equal to 1.

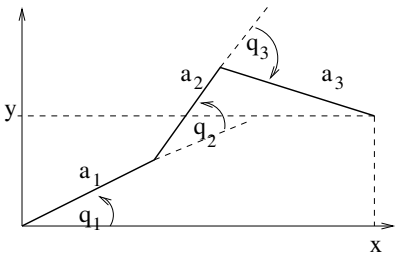

Fig. 3. Kinematic structure of the 3-dof planar pendulum

As the second model, the industrial robot PUMA was chosen (Fig. 4). PUMA's positional kinematics (x, y, z) is given by [6]

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} c_1 A - s_1(d_2 + d_6 s_4 s_5) \\ s_1 A + c_1(d_2 + d_6 s_4 s_5) \\ -a_2 s_2 + d_4 c_{23} + d_6(c_5 c_{23} - c_4 s_5 s_{23}) \end{bmatrix} \tag{19}$$

where $A = a_2 c_2 + d_4 s_{23} + d_6(c_4 s_5 c_{23} + c_5 s_{23})$ and geometric parameters are equal to $a_2 = 0.432\,m$, $d_2 = 0.0745\,m$, $d_4 = 0.432\,m$, $d_6 = 0.056\,m$.

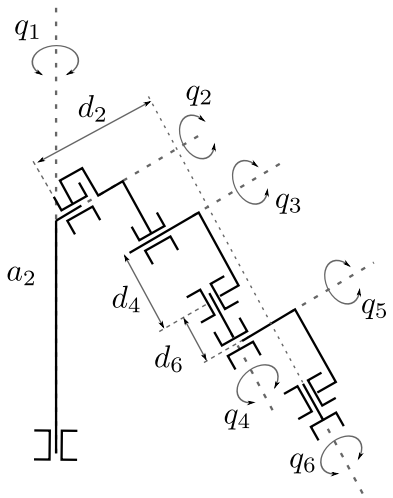

Fig. 4. Kinematic structure of the PUMA robot

In order to compare the performance of both schemes two closed-loops in the taskspace were designed (cf. Fig. 5). The first path (Path 1)

$(x_1(s), y_1(s), z_1(s))$ is a circle

$$\begin{cases} x_1(s) = x_c + R\cos(2\pi s), \\ y_1(s) = y_c + R\sin(2\pi s), \\ z_1(s) = z_c + R\cos(2\pi s), \quad s \in [0,1]. \end{cases} \tag{20}$$

The second path (Path 2) $(x_2(s), y_2(s), z_2(s))$ is a Lissajous-like curve

$$\begin{cases} x_2(s) = x_c + R\sin(2\pi s), \\ y_2(s) = y_c + R\sin(\pi s) - \frac{1}{4}R\cos(8\pi s) \\ z_2(s) = z_c + R\cos(2\pi s), \quad s \in [0,1] \end{cases} \tag{21}$$

For both loops the radius is equal to $R = 0.9$ and the center point is placed at $(x_c, y_c, z_c) = (1,1,0)$. For the 3-dof pendulum the third component of the vector (x, y, z) is neglected everywhere.

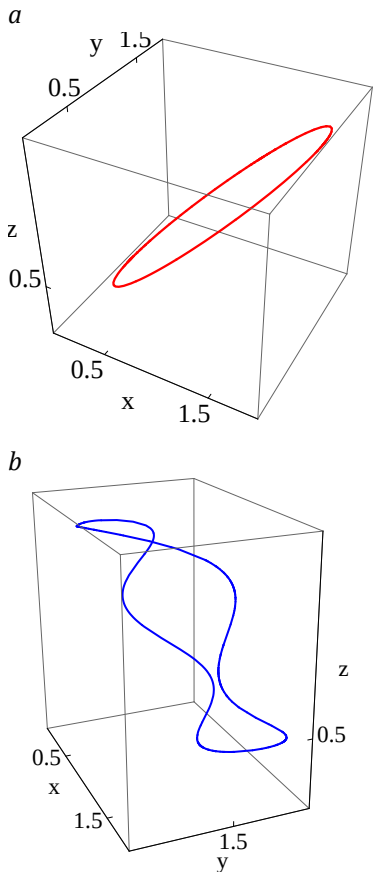

Fig. 5. a – Path 1 – the circle, b – Lissajous-like Path 2

An admissible error to reach each consecutive goal point in the Newton algorithm was set to 0.001, while the maximal change of configuration in a single iteration was equal to $\Delta = 0.5^o$. The parameter ξ_1 of the basic Newton algorithm (6) was derived from Δ, i.e. $\xi_1 = \Delta q_i^1/\Delta$, cf. Eq. (14). When the null-space optimization was preformed (cf. Eqns. (16-17)), scales ξ_1, ξ_2 of the lengths Δq_i^1, Δq_i^2, Eqns. (14-15), were determined in one dimensional optimization process, preserving the final length of configuration change equal to Δ.

Two versions of the elastic band method were run. For Path 1 and Path 2, initial configurations were selected as follows $q_1(0) = (-18.96°, 37.93°, 70.54°)^T$ and $q_2(0) = (-26.05°, 58.80°, 104.92°)^T$, respectively, which corresponds to the initial point of the cyclic path $(x(0), y(0))_{1,2}^T$. The results for the 3-dof pendulum are visualized in Figs. 6-7(b-c) while the stroboscopic view of several postures of the manipulator is depicted in Fig. 8.

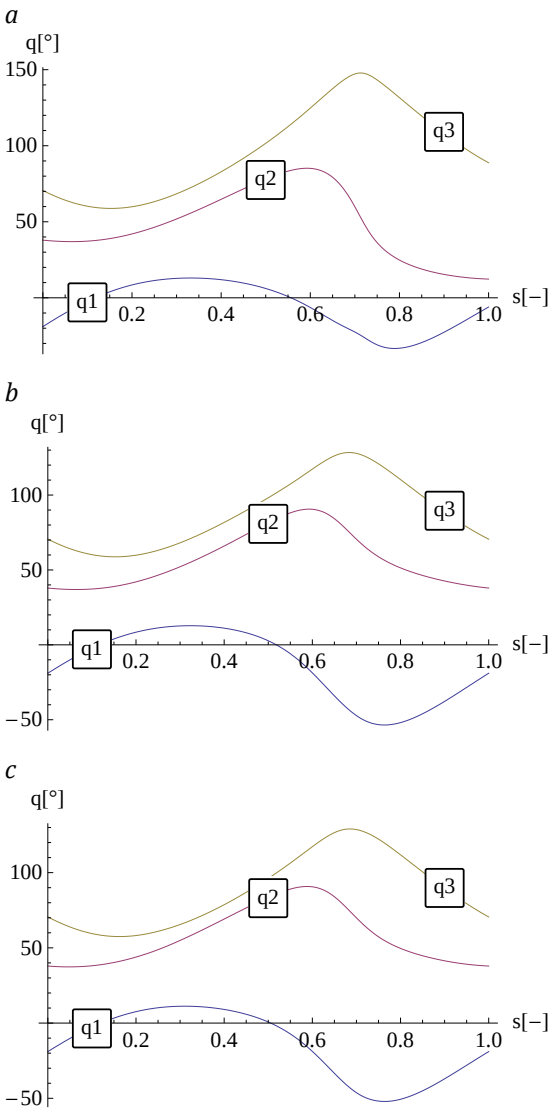

Fig. 6. Trajectories of the 3-dof pendulum generated for Path 1 with: a) the pseudo-inverse Jacobian method, b) the elastic band method (Scheme 1), c) the elastic band method (Scheme 2)

Tab. 1. Numerical results for the 3-dof pendulum

Path 1			
method	lenght $[^o]$	num. of pts.	time $[s]$
pseudo-inverse	253.05	201	0.10
Scheme 1	241.41	201	2.10
Scheme 2	241.08	257	0.50
Path 2			
method	lenght $[^o]$	num. of pts.	time $[s]$
pseudo-inverse	253.78	201	0.11
Scheme 1	254.48	201	2.30
Scheme 2	254.04	257	0.59

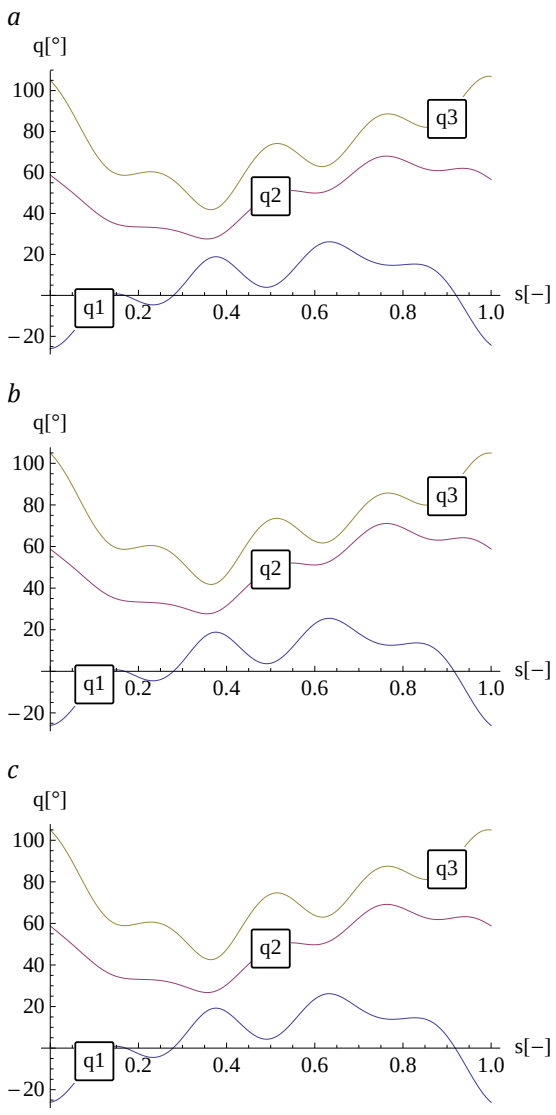

Fig. 7. Trajectories of the 3-dof pendulum generated for Path 2 with: a) the pseudo-inverse Jacobian method, b) the elastic band method (Scheme 1), c) the elastic band method (Scheme 2)

To compare efficiency of the elastic band method with the Jacobian pseudo-inverse method (i.e. the algorithm (6) with $\xi_2 = 0$) some more tests were carried out. The first task of the pseudo-inverse Jacobian method was run with the initial configuration q_0 and the goal $x_f = x(\Delta s)$ where $\Delta s = 0.005$. Consecutive goals were selected as $x_f = x(i\Delta s)$, $i = 2, \ldots, s_{max}/\Delta s$ and the node configuration from the $(i-1)$-st task became an initial one for the i-th task. The pseudo-inverse algorithm does not guarantee repeatability. Final configurations for Path 1 and Path 2 $q_1(1) = (-6.03°, 12.30°, 88.81°)^T$, $q_2(1) = (-24.32°, 56.56°, 106.96°)^T$, respectively, did not meet the initial configurations q_0.

The length of resulting trajectories, computed as $\sum_{k=0}^{K-1} ||q(s_{k+1}) - q(s_k)||$ where K is the final number of node configurations, number of iterations to complete the algorithms and the elapsed time (algorithms were implemented in the Mathematica package and run on 2×3.3 GHz computer) for Scheme 1 and 2,

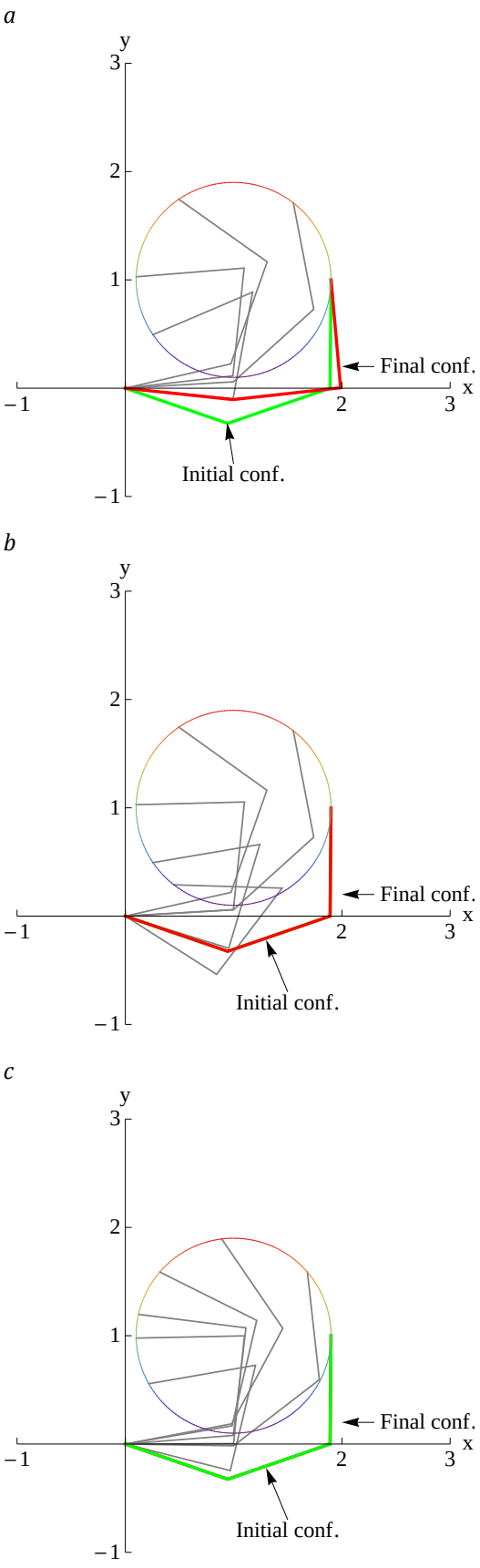

Fig. 8. Stroboscopic view of the 3-dof pendulum following Path 2 with: a) the pseudo-inverse Jacobian method, b) the elastic band method (Scheme 1), c) the elastic band method (Scheme 2)

as well as for the pseudo-inverse Jacobian method, are gathered in Table 1.

The results seem to be rather unexpected as more constrained method (the repeatable inverse) generated shorter (or almost the same length) trajectories as unconstrained method (pseudo-inverse). To explain this phenomenon let us notice that the pseudo-inverse method optimizes a change of configuration locally (from one iteration to another) while the elastic band method exploits global distribution of consecutive node points. For this particular robot and Path 1, it appears that the global knowledge in constrained task impacts the final trajectory more than a local optimization in unconstrained task. It can be easily noticed, cf. Table 1) that the pseudo-inverse method without optimization within the null space of the Jacobian matrix is the fastest among methods tested and Scheme 2 generated results much faster than Scheme 1.

Finally, two schemes were tested on the PUMA robot using cyclic Path 1 and Path 2. Once again, a comparison with the pseudo-inverse Jacobian method was preformed. For Path 1, the initial configuration was chosen as $q(0) = (-0.84°, 62.40°, -67.05°, 46.58°, 44.06°, 45°)^T$ while the final configuration was computed as $q(1) = (9.29°, 66.17°, -64.37°, -24.97°, 15.93°, 45°)^T$. The pseudo-inverse method run for Path 2 also did not satisfy the repeatability condition as $q(0) = (-11.25°, 57.87°, -78.52°, 41.93°, 34.17°, 45°)^T$ and $q(1) = (-6.67°, 61.80°, -77.17°, 43.16°, 21.58°, 45°)^T$. Numerical characteristics were gathered in Tab. 2 while the computed trajectories were plotted in Figs. 9-10. As expected, the pseudo-inverse method did not generate a closed-loop trajectory in the configuration space. When an elastic band method was applied, for both of the schemes, the resulting trajectory remained cyclic. The ordering of the tested methods, with respect to the elapsed time to complete prescribed tasks, observed for the model of 3-dof pendulum remains valid also for the PUMA robot. One can notice that the length of trajectories for the 3-dof planar pendulum is almost the same for both initial paths given in R^2 while lengths of trajectories for the PUMA robot differ significantly where paths are given in R^3. This observation should attract attention of path designers to appropriately locate the path within a taskspace (usually the shape of the path is determined by technological requirements, but its location not necessary).

4. Conclusions

In this paper an adaptation of the elastic band method to repeatable inverse kinematic task was presented. Advantages of the proposed approach include: an application of known and simple methods (the Newton algorithm with an optimization in the null space of the Jacobian matrix) simplicity of kinematic computations which allow to implement this algorithm also in on-line regime, admittance of many variants of changing the current cyclic trajectory as well as optimization at many stages of the algorithm (the Newton algorithm, selection the number and distribution of node points). This algorithm admits also extension in optimizing the initial node configuration (in the current version is was assumed as a given initial data). One more way to extend the proposed algorithm is to adapt it to obstacle-cluttered environements, possibly,

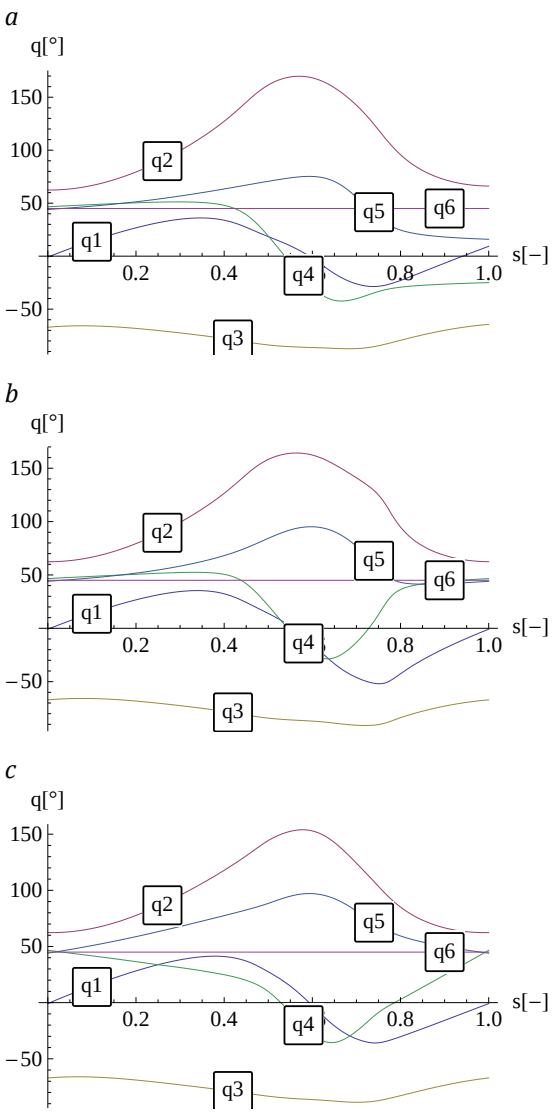

Fig. 9. Trajectories of the PUMA robot generated for Path 1 with: a) the Jacobian pseudo-inverse method, b) the elastic band method (Scheme 1), c) the elastic band method (Scheme 2)

***Tab. 2.* Numerical results for the PUMA robot**

Path 1			
method	lenght $[^o]$	num. of pts.	time $[s]$
pseudo-inverse	347.59	201	0.39
Scheme 1	384.79	201	9.06
Scheme 2	344.26	257	1.62
Path 2			
method	lenght $[^o]$	num. of pts.	time $[s]$
pseudo-inverse	236.62	201	0.37
Scheme 1	249.12	201	7.14
Scheme 2	237.84	257	1.47

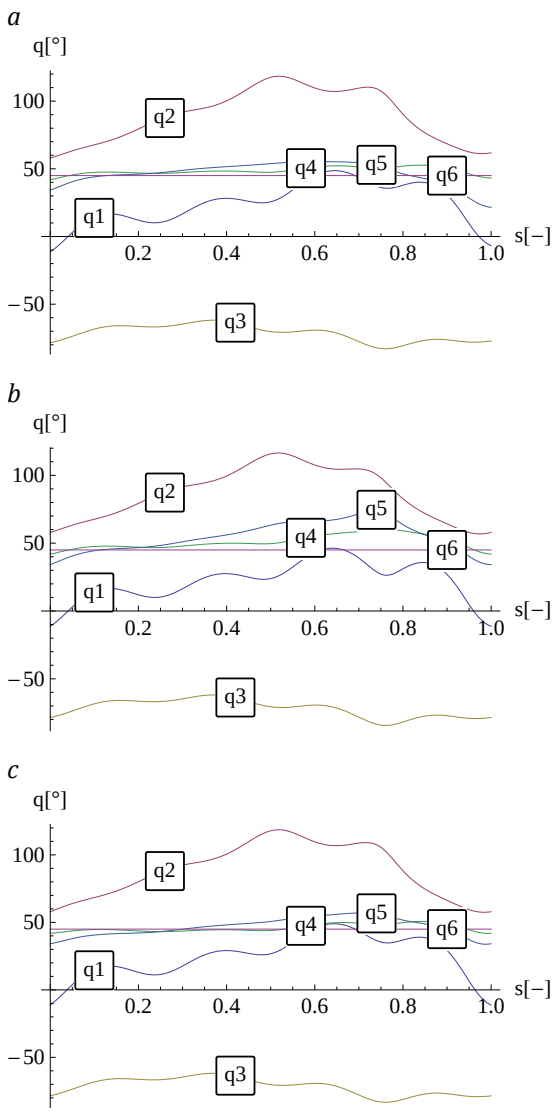

Fig. 10. Trajectories of the PUMA robot generated for Path 2 with: a) the Jacobian pseudo-inverse method, b) the elastic band method (Scheme 1), c) the elastic band method (Scheme 2)

via redefining quality function f to punish approaching to obstacles.

AUTHORS

Ignacy Duleba* – Institute of Computer Engineering, Control and Robotics Wroclaw University of Technology, 50–372 Wroclaw, Janiszewski St. 11/17, e-mail: ignacy.duleba@pwr.wroc.pl.

Michal Opalka – Institute of Computer Engineering, Control and Robotics Wroclaw University of Technology, 50–372 Wroclaw, Janiszewski St. 11/17, e-mail: michal.opalka@pwr.wroc.pl.

*Corresponding author

REFERENCES

[1] Chiaverini S., Oriolo G., Walker I.D., *Handbook of Robotics*, chapter *Kinematically redundant manipulators*, Springer Verlag, Berlin, 2008, 245–268.

[2] Duleba I., *Methods and algorithms of motion planning for manipulators and mobile robots*, EXIT, Academic Science Publisher, Warsaw, 2001, in Polish.

[3] Duleba I, Opałka M., *Using the elastic band method to repeatable inverse kinematic algorithm for manipulators*, Science Works, Warsaw Univ. of Technology, Electronics series, 2012, vol. 182, 2012, 351–356, in Polish.

[4] Karpińska J., *Approximation of algorithms for robot motion planning*. PhD thesis, Wroclaw University of Technology, 2012, in Polish.

[5] Klein C.A., Chu-Jeng C., Ahmed S., "A new formulation of the extended Jacobian method and its use in mapping algorithmic singularities for kinematically redundant manipulators", *IEEE Trans. Robot. Autom.*, vol. 11, 1995, 50–55.

[6] Lee C., *Robot arm kinematics, dynamics, and control*, Computer, vol. 15 (12), 1982, 62–80.

[7] Nakamura Y., *Advanced Robotics: Redundancy and Optimization*. Addison Wesley, New York, 1991.

[8] Quinlan S., Khatib O., "Elastic bands: Connecting path and control", *IEEE Int. Conf. on Robotics and Automation*, vol. 2, 1993, 802–807.

[9] Richter S., DeCarlo R., "Continuation methods: Theory and applications", *IEEE Tran. on Automatic Control*, vol. 28, no. 6, 1983, 660–665.

[10] Roberts R., Maciejewski A.A., "Repeatable generalized inverse control strategies for kinematically redundant manipulators", *IEEE Tran. on Automatic Control*, vol. 38, 1993, 689–699.

[11] Spong M., Vidyasagar M., *Introduction to robotics. Robot Dynamics and Control*. MIT Press, Cambridge, 1989.

[12] Tchon K., Duleba I., "Definition of a kinematic metric for robot manipulators", Journal of Robotic Systems, vol. 11, no. 3, 1994, 211–221.