applied sciences

MDPI

Article

Finite-Horizon Kinetic Energy Optimization of a Redundant Space Manipulator

Alessandro Tringali [1,*] and Silvio Cocuzza [2]

[1] Department of Design, Manufacturing and Engineering Management, University of Strathclyde, Glasgow G1 1XJ, UK
[2] Department of Industrial Engineering, University of Padova, 35131 Padova, Italy; silvio.cocuzza@unipd.it
* Correspondence: alessandro.tringali@strath.ac.uk or a.tringali@archangel.works

Featured Application: The proposed optimal inverse kinematics method is suitable for space robotics applications, e.g., space servicing and active debris removal (ADR). It is also suitable for industrial robotics applications, e.g., in the planar operations of SCARA robots.

Abstract: The minimization of energy consumption is of the utmost importance in space robotics. For redundant manipulators tracking a desired end-effector trajectory, most of the proposed solutions are based on locally optimal inverse kinematics methods. On the one hand, these methods are suitable for real-time implementation; nevertheless, on the other hand, they often provide solutions quite far from the globally optimal one and, moreover, are prone to singularities. In this paper, a novel inverse kinematics method for redundant manipulators is presented, which overcomes the above mentioned issues and is suitable for real-time implementation. The proposed method is based on the optimization of the kinetic energy integral on a limited subset of future end-effector path points, making the manipulator joints to move in the direction of minimum kinetic energy. The proposed method is tested by simulation of a three degrees of freedom (DOF) planar manipulator in a number of test cases, and its performance is compared to the classical pseudoinverse solution and to a global optimal method. The proposed method outperforms the pseudoinverse-based one and proves to be able to avoid singularities. Furthermore, it provides a solution very close to the global optimal one with a much lower computational time, which is compatible for real-time implementation.

Keywords: robotics; inverse kinematics; redundant manipulator; energy minimization

Citation: Tringali, A.; Cocuzza, S. Finite-Horizon Kinetic Energy Optimization of a Redundant Space Manipulator. *Appl. Sci.* **2021**, *11*, 2346. https://doi.org/10.3390/app11052346

Academic Editor: Oscar Reinoso Garcia

Received: 10 February 2021
Accepted: 4 March 2021
Published: 6 March 2021

1. Introduction

In space robotics missions, both for space servicing or debris capture, the minimization of the robot energy consumption and the minimization of the reactions transferred to the base spacecraft are of crucial importance [1]. Both of these targets are related to a longer system lifetime in orbit: while energy minimization allows longer operations by definition, reactions minimization allows to keep the base spacecraft orientation unchanged, which in turn means fuel savings due to the fact that attitude corrections are not necessary. In the case of trajectory tracking of redundant manipulators, inverse kinematics methods are usually adopted, which however only provide local minima along the end-effector trajectory. Thus, as trajectories increase in length, they fail to follow the lowest energy path in the robot joints space, and tend to be singularity prone.

Differential kinematic inversion has been extensively used for the solution of the real-time motion of redundant manipulators, while minimizing a certain cost function [2–7]. It has been used in plenty of variations, as it has several interesting properties: it has fast execution times, can be used to prioritize specific tasks over others, and it can be used to optimize a variety of different cost functions while driving the robot end-effector through a predefined path. However, this comes at a price: it only provides local optima, which can be quite far from the global one [2], and it is subject to the problem of singularities. In field

applications [8], like space or underwater robotics, avoiding singularities and using as little energy and power as possible is important as robots usually run on limited energy sources, such as batteries or solar panels. In such applications, local inverse kinematics frameworks are not ideal, and, on the other hand, global optimization methods are too computationally demanding. Minimization of energy consumption of robotic systems has been subject of intense research, since it helps reducing operational costs, and allows us to use robots even when resources are limited. Focusing on robot manipulators, many attempts to minimize kinetic energy through inverse kinematics have been focused on minimizing the joint velocities norms. To the best of the authors' knowledge, the first systematic review of this topic has been proposed by Klein and Huang [9], which is a milestone in the research on the use of the pseudoinverse of the Jacobian matrix for robot control. The paper outlines how the pseudoinverse generally keeps the required instantaneous power low, since it minimizes joint velocities, which has been considered as equivalent to approximately minimize the kinetic energy. It also states that the pseudoinverse method, since it minimizes joint velocities, keeps the manipulator away from singularities, which was however proven incorrect by Bailleul et al. [10]. Many of the subsequent works developed their algorithms starting from the concept of weighted pseudoinverse, as presented by Whitney [11]. In particular, several researchers have followed the direction of pointwise minimizing the joint torques as a means to minimize the actuator energy consumption, associating such an algorithm with positive results regarding the minimization of the kinetic energy integral along the whole trajectory. Extensive analysis on this approach is presented by Hollerbach and Suh [12], who however found out that a dynamics-based inverse kinematics resolution method can lead to instabilities on longer trajectories. A more strictly energy-focused approach was followed by Nedungadi et al. [13], who proposed a method for joint torques minimization that holds some global properties. However, it does not allow to minimize kinetic energy when initial conditions on joint displacements are imposed and, although less instable than the previous work, still presents stability issues. The first research providing actuator models for energy consumption minimization was presented by Vukobratovic et al. [14], who introduced the dynamic models of hydraulic and electrical actuators and showed that a major advantage of energy minimization compared to joint velocities minimization is that it tends to move joints with lower inertia more than those with higher one, thus reducing energy consumption. As pointed out by Nenchev [2] in his review, these early studies made it very clear that a trade-off between local optimization performance and global stability is generally unavoidable: pointwise minimization must always be relaxed in order not to get stuck in singularities. Several works have been published on stabilization of joint torques minimization methods: most of them are aimed at minimizing joint torques without considering any robot payload (worthy of notice are [15] by Maciejewski and [16] by Hu et al.); nevertheless, the minimization of torques in the presence of external loads has also been addressed (Woolfrey et al. [7]). However, the solutions they propose are only marginally useful for the methods that minimize kinetic energy. Other recent developments of pseudoinverse-based methods are aimed at addressing specific shortcomings of existing formulations; for example Kelemen et al. [6] presented a solution to minimize computational times by setting a variable priority value for each end-effector task. Alongside pseudoinverse-based methods, other inverse kinematics methods based on Quadratic Programming have been proposed. These are considered promising since they have been proven capable of solving inverse kinematics problems with equality and inequality constraints [17]. These methods, however, have been proposed for joint torques minimization [18] rather than kinetic energy minimization.

Alongside local inverse kinematics methods, offline methods based on optimal control have been pursued. These techniques are slower, but allow for better optima of the cost function, although at the price of much longer computation times. Their cost function is usually a sum of the end-effector position error and a control cost (usually kinetic energy or actuators torque) computed on a specific set of points. The set may correspond to the whole trajectory or to a finite number of points after the last completed integration step.

In [19], Gregory et al. presented an optimal control method for energy optimization along a trajectory subject to holonomic constraints based on Euler–Lagrange equations. However, this method has only proven efficient for non-redundant manipulators. A different optimal control approach, based on Pontryagin maximum principle, is proposed in [20] by Halevi et al. for electromechanical linear actuators. In this work, a mathematical model of the actuators to be included in the cost function is presented, but the adaptability of this approach to manipulators with revolute joints is doubtful. Several other works address the issue of minimizing the energy required to perform the whole robot trajectory using offline methods. Among those, an early attempt was presented by von Stryk et al. [21], featuring an optimal control based minimization of the squared joint velocities along the trajectory, as a means to reduce the kinetic energy in the operations of a non-redundant manipulator. In [22], Saramago et al. investigated an offline path planning method for non-redundant manipulators using a multi-objective cost function involving both trajectory time and mechanical energy. In [23], Field et al. presented a solution of the energy minimization problem based on dynamic programming. Their intuition does not serve the purpose of finding the global optimum, since each one of the iterations only explores a subspace of the search space; however, it manages to avoid mediocre local minima, and keeps the execution time reasonably low avoiding the curse of dimensionality usually involved with dynamic programming. Nurmi et al. [24] recently presented an energy optimal solution for hydraulically powered redundant manipulators based on dynamic programming. Their approach, which has been developed for hydraulic actuators, performs sensibly better than general energy minimizing algorithms in the considered application, proving that actuator models might be beneficial for specific problems. However, the related burden of computational complexity means that this method cannot be used in real time. Recently, Tringali et al. [25] proposed a globally optimal inverse kinematics method for redundant manipulators, which allows for the optimization of different integral cost functions, such as kinetic energy and joint torques norm, can provide solutions with a variety of constraints, both linear and nonlinear, and is suitable for multi-objective optimization. Finally, Faroni et al. [26] presented a fast online predictive method to solve the task-priority differential inverse kinematics of redundant manipulators under kinematic constraints.

In this paper, a novel inverse kinematics method for redundant manipulators is presented, which is able to provide a solution near to the global optimal one, is able to avoid singularities, and is suitable for real-time implementation. The proposed method is based on the optimization of the kinetic energy integral on a limited subset of future end-effector path points, making the manipulator joints to move in the direction of minimum kinetic energy. Rather than optimizing the cost function at every time step, the cost function is updated at larger intervals, and the appropriate choice of the update interval and the number of future exploration points allows a limited computational cost. The proposed method is tested by simulation for a three degrees of freedom (DOF) planar manipulator in a number of test cases, and its performance is compared to the classical pseudoinverse solution and to the one computed with a global optimal method.

The paper is organized as follows: Section 2 presents and discusses the proposed optimization method and its main parameters; then, Section 3 presents the simulation setup and results and, in particular, a comparison between the simulation results obtained with the proposed method, the classical pseudoinverse solution, and a global optimal method. Finally, Section 4 concludes the paper.

2. Materials and Methods

2.1. Mathematical Formulation

A kinematically redundant manipulator is a manipulator with more DOF than controlled end-effector variables. The tracking problem of a redundant manipulator is the problem of following a reference end-effector trajectory $\boldsymbol{x}_{ref}$, expressed as a function of time t in the Cartesian space. This requires to find the joint variables $\boldsymbol{q}(t)$ such that:

$$\boldsymbol{x}_{ref}(t) = \boldsymbol{x}(\boldsymbol{q}(t)) \tag{1}$$

in which $\boldsymbol{x}$ is the actual end-effector trajectory.

The redundancy implies that the problem expressed by (1) may have an infinite number of solutions in the joints space. The method hereby presented is based on the general solution computed with the Moore–Penrose pseudoinverse:

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{+}\dot{\boldsymbol{x}} + \left(\boldsymbol{I} - \boldsymbol{J}^{+}\boldsymbol{J}\right)\dot{\boldsymbol{q}}_0 \tag{2}$$

in which $\boldsymbol{J}$ is the manipulator Jacobian, $\boldsymbol{J}^{+} = \left(\boldsymbol{J}^{T}\boldsymbol{J}\right)^{-1}\boldsymbol{J}^{T}$ is its Moore–Penrose pseudoinverse, $\dot{\boldsymbol{x}}$ is the desired end-effector velocity, $\boldsymbol{I}$ is the identity matrix, and $\dot{\boldsymbol{q}}_0$ is an arbitrary vector. The first part of (2) is a least-squares solution based on the minimization of the joint velocities norm and is related to the primary task (end-effector trajectory tracking), while the second part is related to a secondary task that can be imposed by exploiting the null-space operator [27] $\left(\boldsymbol{I} - \boldsymbol{J}^{+}\boldsymbol{J}\right)$. The arbitrary vector $\dot{\boldsymbol{q}}_0$ can be chosen so that it results in the minimization of an integral cost function along the specified end-effector trajectory, whereas the precision tracking is provided by the pseudoinverse solution (first term in (2)). A general expression of an integral cost function subject to the kinematic constraint related to the tracking of the desired end-effector trajectory is:

$$\begin{array}{ll} \underset{x}{\text{minimize}} & \int_{t_0}^{t_{fin}} C\left(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, t\right) dt \\ \textit{subject to} & \boldsymbol{x_{ref}}(t) = \boldsymbol{x}(\boldsymbol{q}(t)) \end{array} \tag{3}$$

where $C\left(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, t\right)$ is a cost function to be integrated along the manipulator trajectory, t_0 and t_{fin} are the initial and final times. Final time t_{fin} does not need to be the time when the manipulator motion is completed: it is possible to define a time horizon h', much shorter than the whole motion time, so that the integral cost is optimized in a time window from t_0 to $t_0 + h'$, and then updated during the motion (using each time an updated t_0), until $t_0 + h'$ corresponds to t_{fin}. This method is used here to optimize the kinetic energy integral of a serial manipulator, leading to the following discrete formulation:

$$\begin{array}{lll} \underset{\boldsymbol{q}}{\text{minimize}} & \sum_{i=t_0}^{t_0+h'} \frac{1}{2}\dot{\boldsymbol{q}}_{\boldsymbol{i}}^{T}\boldsymbol{M_i}(\boldsymbol{q_i})\dot{\boldsymbol{q}}_{\boldsymbol{i}}\Delta t & \\ \textit{subject to} & \boldsymbol{x}(\boldsymbol{q_i}) = \boldsymbol{x_{i,ref}} & \textit{for } i = t_0..t_{fin} \end{array} \tag{4}$$

where $\boldsymbol{x_{i,ref}}$ is the reference end-effector position at time i, $\boldsymbol{x}(\boldsymbol{q_i})$ the actual end-effector position at time i, $\boldsymbol{q_i}$ the joints position vector at time i, $\dot{\boldsymbol{q}}_{\boldsymbol{i}}$ the joints velocity vector at time i, $\boldsymbol{M_i}(\boldsymbol{q_i})$ the inertia matrix at time i, and Δt the discrete time step. For finite horizon optimization, as in the case presented here, t_{fin} in (3) is dropped in favour of a closer time $t_0 + h'$, where h' is a finite time horizon.

The integral cost function is used to obtain a velocity vector $\dot{\boldsymbol{q}}_{\boldsymbol{prediction}}$ pointing to the direction in which the kinetic energy integral is minimum for the chosen trajectory. This velocity vector is used as an extra (secondary) task added to the manipulator motion through the Jacobian null space as per (2).

The steps of the proposed method, called Predictive Minimization of Kinetic Energy (PMKE) method, are presented in Table 1.

Table 1. Steps of the Predictive Minimization of Kinetic Energy (PMKE) method.

Step nr.	Operation
1	A horizon h is selected (h is a positive integer number).
2	A prediction step size ΔT is selected. It is a multiple of the pseudoinverse algorithm step size Δt and it is generally much larger.
3	An update interval size I is selected. It is a multiple of the pseudoinverse step size Δt, but it is generally smaller than ΔT.
4	Expression (4) is optimized with sampling times $t = t_0,\ t_0 + \Delta T, \ldots, t_0 + h\Delta T$. Any deterministic optimisation algorithm can be used, and the Matlab *fmincon* Sequential Quadratic Programming (SQP) function has been used in this work. This provides a prediction of the optimal joint variables for a future window that is defined here as prediction window $T_p = h * \Delta T$.
5	The optimization at point 4 outputs a set of h joint configurations, for the $h + 1$ sampling times $t = t_0,\ t_0 + \Delta T, \ldots, t_0 + h\Delta T$. The set of these vectors is from now on referred as $\left\{\boldsymbol{q_{prediction}}\right\}$. A specific vector in the set is referred with its sampling time as a subscript, with a notation such as $\boldsymbol{q_{h\Delta T}}$ for the sampling time $t_0 + h\Delta T$ ($\boldsymbol{q_0}$ for t_0).
6	A 4th order spline interpolation from $\boldsymbol{q_0}$ to $\boldsymbol{q_{h\Delta T}}$ is computed, which intersect all the points in the set $\left\{\boldsymbol{q_{prediction}}\right\}$. Endpoint conditions of the interpolation are set so that $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$ at $\boldsymbol{q_0}$ are always set equal to the ones reached by the manipulator at time t_0, and are set to zero at $\boldsymbol{q_{h\Delta T}}$. This ensures that the resulting spline has not large derivatives in the interval of interest. The resulting spline is from now on referred as $\boldsymbol{q_{prediction}} = \boldsymbol{f}(t)$.
7	For each step i in the update interval I, the joint velocities are computed as per (2), using the error between the previous joint variables $\boldsymbol{q_{i-1}}$ and the computed value of $\boldsymbol{q_{prediction}}$ divided per Δt as arbitrary vector: $\dot{\boldsymbol{q}}_0 = \frac{\boldsymbol{q_{prediction}} - \boldsymbol{q_{i-1}}}{\Delta \boldsymbol{t}}$
8	Once time I has passed, steps 4 to 7 are repeated for the new update interval, using the last time step of previous interval as t_0.
9	Steps 4–8 are repeated until the amount of time before the end of the trajectory is T_p.
10	The steps left at this point are computed with the last computed $\boldsymbol{q_{prediction}}$.

2.2. Parameters Description

The main parameters involved in the PMKE method are the inverse kinematics discrete time step Δt, the prediction step size ΔT, the horizon h, and the update interval I. This section explains their meaning and the value that has been chosen for the simulations performed for the purposes of this work.

Inverse kinematics discrete time step Δt: this parameter is the time step used to numerically perform the integration of the inverse kinematics equation (2). It has been set to 0.01 s for the implementation hereby presented.

Prediction step size ΔT and horizon h: these altogether determine the prediction window T_p, which is related to the ability of the method to take into account future (up to T_p after the current time t_0) issues that could undermine the quality of the solution. The two parameters must be balanced considering that, for example, for a fixed T_p, having a higher h and a smaller ΔT allows for better resolution of the prediction, making it less likely for the method to pursue local improvements over global ones. However, it also implies longer computational times. For the 3-DOF manipulator model used in the simulations (see Section 3), good results are reached with h = 2. Thus, this value is used to avoid the computational complexity related to higher h values. This means that $\Delta T = T_p/2$, and leaves the question open about how to choose T_p. Increasing it does not change the computational complexity, which is only influenced by h, however a prediction too far in the future might feature a reduced influence on the current path point. Furthermore, it is hard to establish a proper T_p because a value that is good for certain trajectories does not necessarily fit other ones: some of them might benefit from a longer prediction window, while some others might benefit from a shorter one, which makes it difficult to develop a tuning strategy. The value of the prediction window is set to 20% of the total simulation time. This means that, for trajectories with simulation time of 10 s such as the ones used in the following section, the method will perform the optimization of the cost function up to 2 s, corresponding to 200 inverse kinematics time steps. This means that ΔT is equal to 1 s.

Update interval I: the update interval is the variable that decides how often the optimization of the cost function is updated. It should be ideally set as big as the time step Δt, but this would slow down the method. For the simulations presented in this paper, an update interval of 0.5 s is chosen to show that this is sufficient to obtain good results. However, much shorter I intervals are also possible if computational time is not a concern. The computational time required for the PMKE method is discussed in more detail in Section 3.2.

The values of the main parameters of the PMKE method used in this work are summarized in Table 2.

Table 2. Main parameters of the PKME method.

Parameter	Δt [s]	ΔT [s]	h	I [s]
	0.01	1	2	0.5

3. Results and Discussion

3.1. Simulation Setup

A 3-DOF planar manipulator model (with three revolute joints) is considered, which corresponds to a real robot prototype for space applications that has been used in previous works [3–5,25]. The two end-effector Cartesian coordinates are controlled, x and y, thus leaving one degree of kinematic redundancy. Geometrical and inertial characteristics of the manipulator, which is depicted in Figure 1, are presented in Table 3.

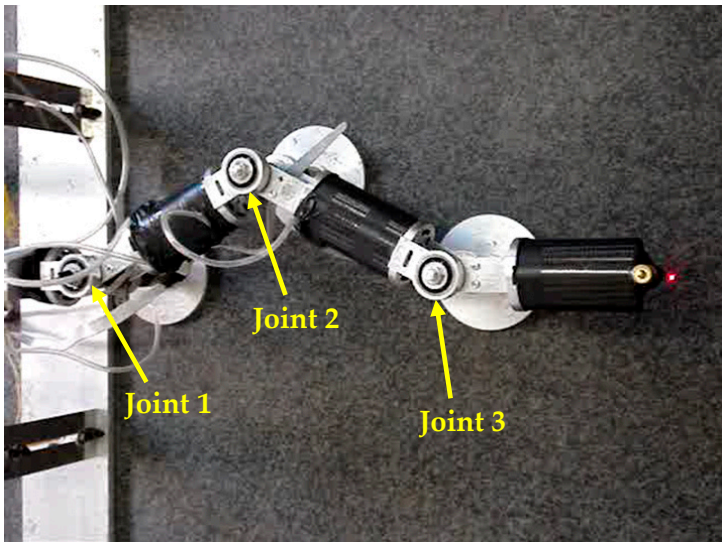

Figure 1. Space robot prototype considered in this work.

Table 3. Manipulator geometrical and inertial parameters.

Link nr.	Mass (kg)	Length (m)	Moment of Inertia (Barycentric) (kgm^2)	Center of Gravity (m)
1	0.615	0.176	0.001811	0.0950
2	0.615	0.176	0.003173	0.0717
3	0.307	0.1375	0.002103	0.0526

A simulator of the manipulator inverse kinematics has been developed using the Matlab programming language, except for the code used for steps 4–5 of the PMKE method described in Section 2.1, which corresponds to the minimization of the kinetic energy integral: the routine for this optimization has been first written in Matlab exploiting the

fmincon function, and then translated into C, compiled, and interfaced to the inverse kinematics simulator.

In order to show the effectiveness of the presented method, four trajectories have been simulated with both the PMKE and a traditional method, based on the Moore–Penrose pseudoinverse and performing the minimization of the joint velocity vectors through local least-squares optimization. This method is called here Least Squares Velocities (LSV) minimization method. Moreover, a further comparison is made with the results computed with a global optimal method [25] recently published by the authors.

Simulations 1–2 feature two short end-effector trajectories aimed at demonstrating the PMKE method general performance. Simulation 3 features a trajectory in which the LSV method encounters a singularity. Finally, simulation 4 features a trajectory whose length causes a significant degradation of the performance of the LSV method. All of the proposed trajectories start from the same end-effector position $\boldsymbol{x_{init}} = [0.468;\ 0]$ m and robot initial configuration $\boldsymbol{q_{init}} = [0;\ 0.327;\ -0.754]$ rad, and run for a total time T = 10 s. The considered trajectories and their characteristic dimensions are presented in Table 4.

Table 4. Simulated trajectories.

Simulation nr.	Shape	Characteristic Dimension	Length [m]
1	Rectilinear	Length	0.10
2	Circular	Radius	0.05
3	Rectilinear	Length	0.40
4	Rectilinear	Length	0.70

The end-effector velocity profiles have been chosen in order to have sufficiently smooth accelerations (continuous and with continuous derivative), as shown in Figure 2. The equation of the derivative of the curvilinear abscissa of the end-effector used is:

$$\frac{ds}{dt} = \begin{cases} 2 * \left(\frac{\frac{L}{2}}{\left(\frac{T}{2}\right)^2}\right) * \left(\frac{t^2}{2} + (2*\beta)^{-2} * \cos((2*\beta)*t) - (2*\beta)^{-2}\right), & t \leq \frac{T}{2} \\ L - \left(2 * \left(\frac{\frac{L}{2}}{\left(\frac{T}{2}\right)^2}\right) * \left(\frac{(T-t)^2}{2} + (2*\beta)^{-2} * \cos\left((2*\beta)*\left(\frac{T}{2} - t\right)\right) - (2*\beta)^{-2}\right)\right), & t > \frac{T}{2} \end{cases} \quad (5)$$

where T is the total motion time, L the overall length of the considered trajectory, and $\beta = \frac{2\pi}{T}$.

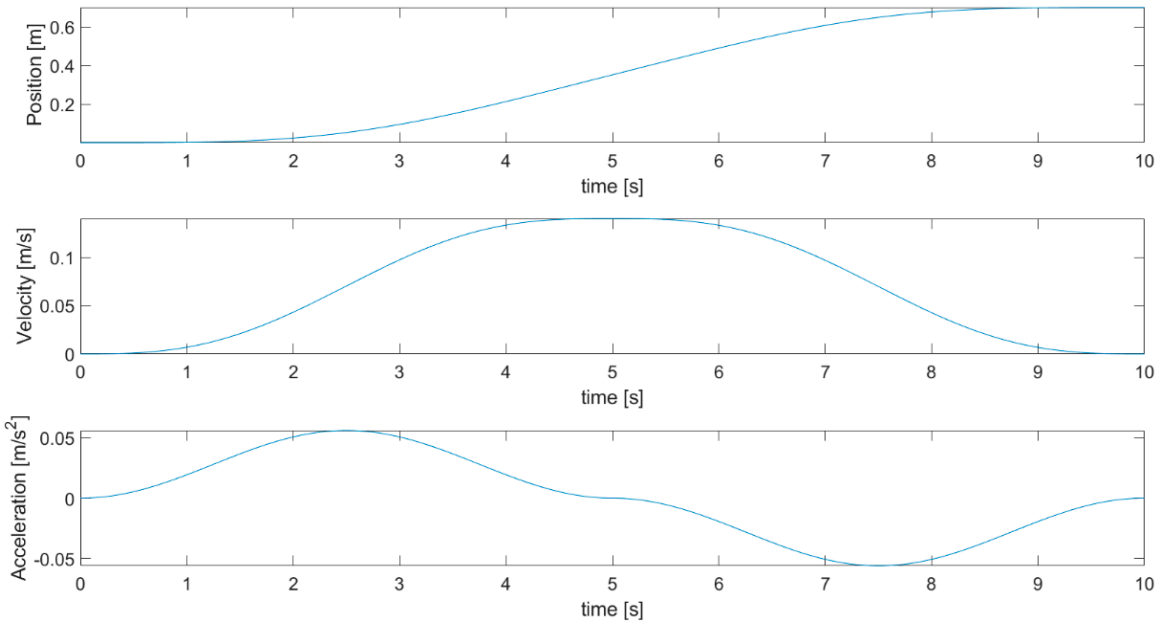

Figure 2. End-effector curvilinear abscissa and its derivatives.

3.2. Simulation Results

Table 5 compares the kinetic energy integral for all the considered methods and trajectories, both in absolute and relative terms. The PMKE method presented in this paper is used as a reference and highlighted in grey. PMKE computational times have also been included, as measured on a machine featuring an Intel i7 ninth generation exa-core processor with 32 GB RAM, SSD mass storage, and using Windows 10 and Matlab version 2019b.

It can be noticed that the LSV method is outperformed by the PMKE one for all the trajectories: non-singular trajectories feature relative differences ranging from 30.1% to 47.4%, while the trajectory that is singular using the LSV method (trajectory 3) is not singular using the PMKE method, resulting in a very high improvement of kinetic energy integral (relative difference of 187.36%). The absolute value of the difference between PMKE solutions and global optimal solutions, on the other hand, is very small (0.13%) for short trajectories and grows with the length of the trajectory (up to 23.6%). This can be explained by observing that, the longer the distance covered by the end-effector, the higher the chance of an energy consuming motion taking place beyond the PMKE prediction window. Kinetic energy integral profiles are shown in detail in Figure 3.

Table 5. Simulation results.

Simulation nr.	PMKE Result [Js]	LSV Result [Js]	Global Result [Js]	PMKE Computational Time [s]
1	5.98×10^{-3}	8.81×10^{-3} (+47.4%)	5.97×10^{-3} (−0.13%)	0.915
2	4.08×10^{-3}	5.37×10^{-3} (+31.7%)	3.87×10^{-3} (−5.03%)	0.934
3	7.57×10^{-3}	2.17×10^{-2} (+187%)	6.09×10^{-3} (−19.5%)	0.961
4	3.97×10^{-2}	5.17×10^{-2} (+30.1%)	3.03×10^{-2} (−23.6%)	0.914

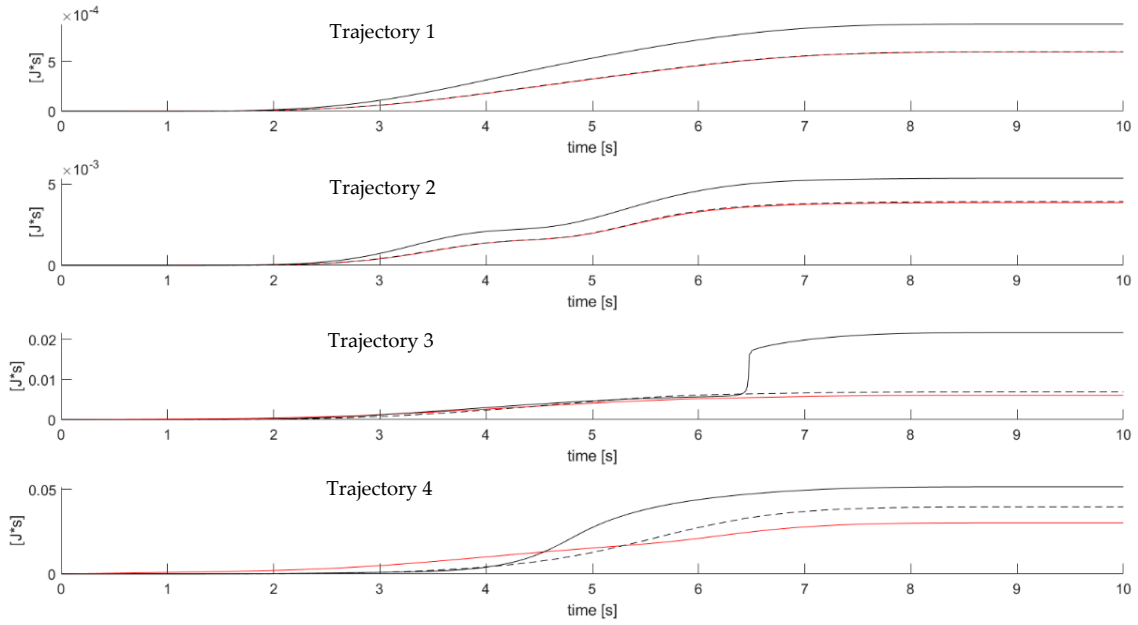

Figure 3. Kinetic energy integral plots for the four considered trajectories. PMKE: dashed black lines; LSV: continuous black lines; global optimal method: red lines.

It has also to be noticed that the execution times of the PMKE method (see Table 5) allow for a real-time kinematic inversion, as it is discussed below. The core of the PMKE

method performs two operations: the optimization (steps 4–5 of the method as described in Section 2.1), and the generation of the coefficients of the 4th order splines used for interpolation (step 6). A hybrid C-Matlab implementation has been deployed to perform these operations. As a result, the optimization of steps 4–5 is performed in C in an average time of ~0.0011 s, while the computation of the spline coefficients of step 6 is performed in Matlab in an average time of ~0.0010 s. The overall average computational time for the two operations is ~0.0021 s, which is compatible with real-time implementation, since the considered time step Δt is equal to 0.01 s. Even faster computation can be obtained by implementing both of these operations in C/C++ and optimizing the code.

In order to make a more detailed comparison of the results obtained with the PMKE and LSV methods, Figures 4–7 show (top to bottom) the stroboscopic views of the robot motion, joint positions, and joint velocities for both the PMKE (left) and LSV (right) methods. In each stroboscopic plot, the joint 1 (which allows the rotation between link 1 and the base frame) is in the origin and, in the joint positions and velocities plots, joint 1 is shown in blue, joint 2 in red, and joint 3 in yellow.

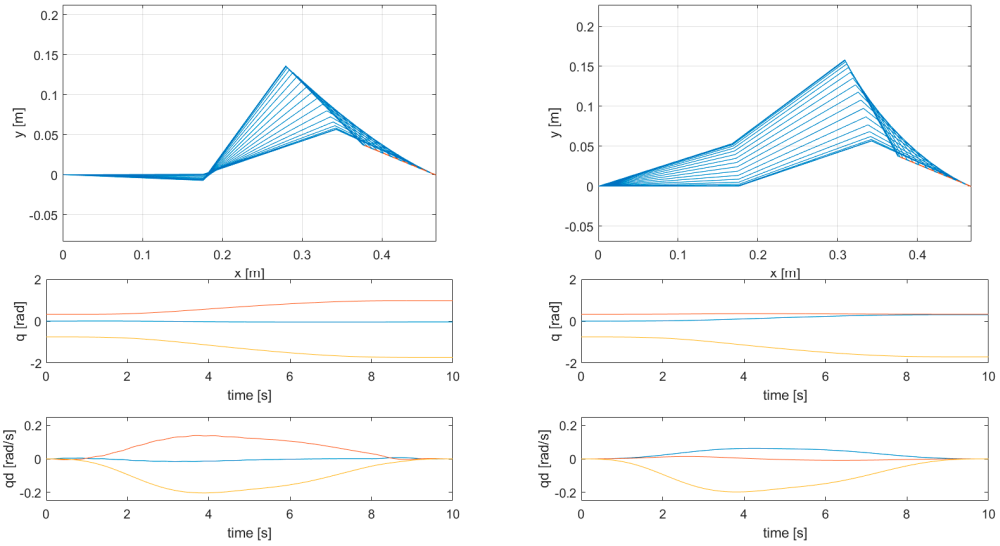

Figure 4. Trajectory 1, comparison between PMKE (**left**) and LSV (**right**). First joint: blue lines; second joint: red lines; third joint: yellow lines.

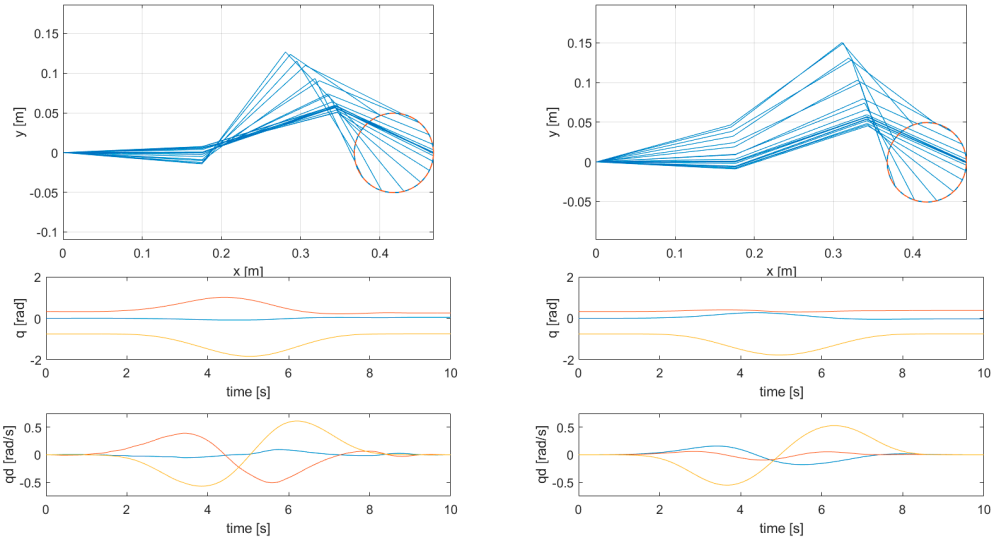

Figure 5. Trajectory 2, comparison between PMKE (**left**) and LSV (**right**). First joint: blue lines; second joint: red lines; third joint: yellow lines.

PMKE results for trajectories 1 and 2 feature a behavior that has been previously noticed when using kinetic energy saving algorithms [5,13]: the joints that are closer to the end-effector feature higher velocities compared to those closer to the base of the manipulator. This behavior is explained by considering that, when a joint is rotated, the same rotation is also transferred to the successive joints; thus, the moment of inertia of a link is constituted by the sum of its own moment of inertia and an additional term for each of the successive joints. While the LSV method simply minimizes the joints velocity norm, the PMKE method, similarly to other methods that minimize kinetic energy, takes into account the fact that the motion of the joints closer to the base are more energetically expensive. Stroboscopic and joint velocities plots in Figures 4 and 5 confirm that the LSV method features a joint motion more uniform than that of the PMKE method. In fact, for trajectory 1, the first joint features a maximum velocity of 0.063 rad/s in the LSV solution and 0.017 rad/s in the PMKE solution, while the third joint reaches up to 0.197 rad/s in the LSV solution and 0.203 rad/s in the PMKE solution. While the maximum velocity of the third joint is around three times the velocity of the first one in the LSV solution, the third joint is more than 10 times faster than the first one in the PMKE solution. Similarly, for trajectory 2, the first joint has a peak velocity of 0.196 rad/s in the LSV solution and a peak velocity of 0.095 rad/s in the PMKE solution, while the third joint velocity reaches 0.529 rad/s in the LSV solution and 0.613 rad/s in the PMKE solution.

This behavior is typical of local kinetic energy minimization algorithms. However, PMKE is able to depart from it when required to obtain improved optimization results. A very clear example of this case is provided by trajectory 3, for which the simulation results are presented in Figure 6. This trajectory is singular using the LSV method, while, on the other hand, the PMKE method is able to avoid the singularity. This remarkable behavior is allowed by the fact that the PMKE method, although generally striving to keep the first joint velocity low, moves it (and the other ones) away from singularities when required, if they fall within its prediction window. The stroboscopic plots in Figure 6 show that, while LSV (right) hits a singularity at t = 6.5 s, PMKE (left) features a different motion altogether, with first and second joints reaching higher velocities (excluding the singularity in LSV): first joint reaches 0.250 rad/s in the PMKE solution and 0.186 rad/s in the LSV solution, while second joint reaches 0.261 rad/s in the PMKE solution and 0.023 rad/s in the LSV solution.

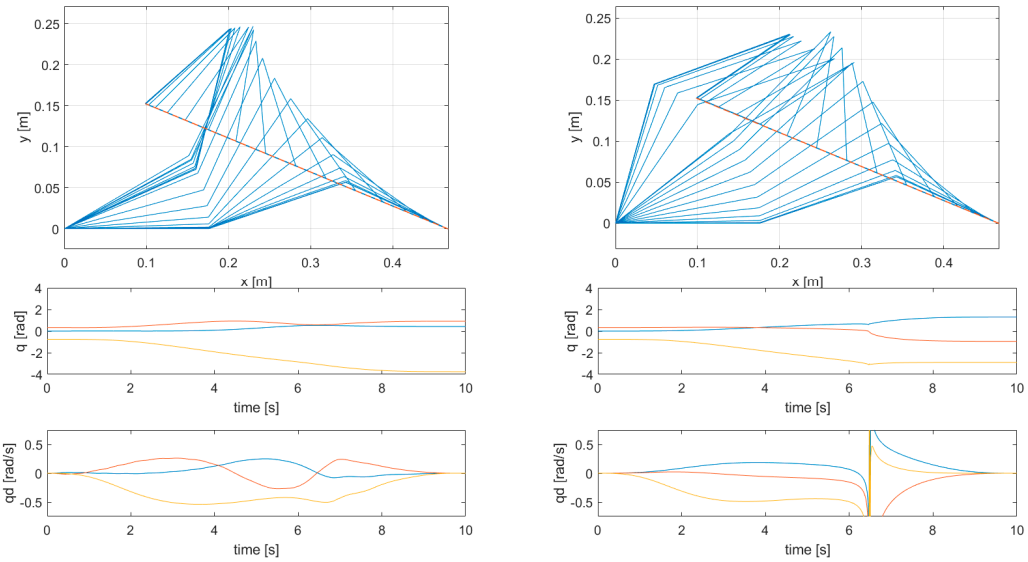

Figure 6. Trajectory 3, comparison between PMKE (**left**) and LSV (**right**). First joint: blue lines; second joint: red lines; third joint: yellow lines. Joint velocities at the singularity not shown in LSV.

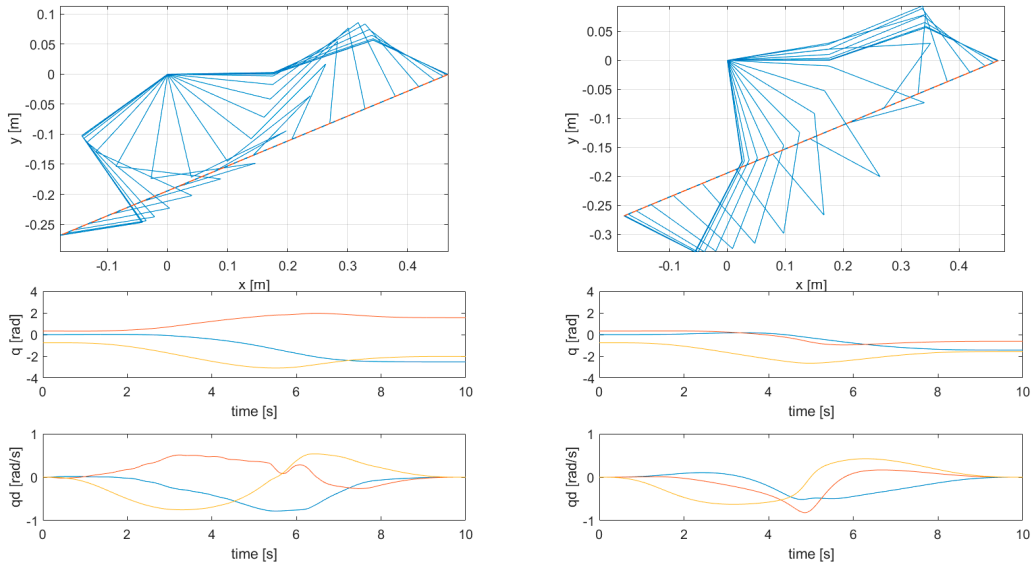

Figure 7. Trajectory 4, comparison between PMKE (**left**) and LSV (**right**). First joint: blue lines; second joint: red lines; third joint: yellow lines.

Finally, the PKME–LSV comparison for trajectory 4 is presented in Figure 7. This is the longest trajectory that has been simulated, and it would be impossible to cover its whole length while also limiting the motion of the first joint. In this case PMKE and LSV feature similar joint velocity profiles, and PMKE features a consistently higher first joint maximum velocity (sitting at 0.780 rad/s, against 0.512 rad/s in the LSV solution). In this case, the PMKE method provides a solution with a lower kinetic energy integral by making the robot to assume configurations that feature lower coefficients of the inertia matrix, compared to those obtained with the LSV method.

In order to verify that the coefficients of the inertia matrix are lower for the PMKE solution, a possible indicator is the maximum eigenvalue of the manipulator inertia matrix, which is known as its spectral radius. In fact, the spectral radius of a square matrix A is defined as:

$$\rho(A) = \max\{|\lambda_1|, \ldots, |\lambda_n|\} \tag{6}$$

where $\lambda_1, \ldots, \lambda_n$ are the eigenvalues of the matrix A. Figure 8 shows the evolution of the spectral radius of the inertia matrix of the robot manipulator along trajectory 4 for both the LSV (dotted line) and PMKE (dashed line) solutions.

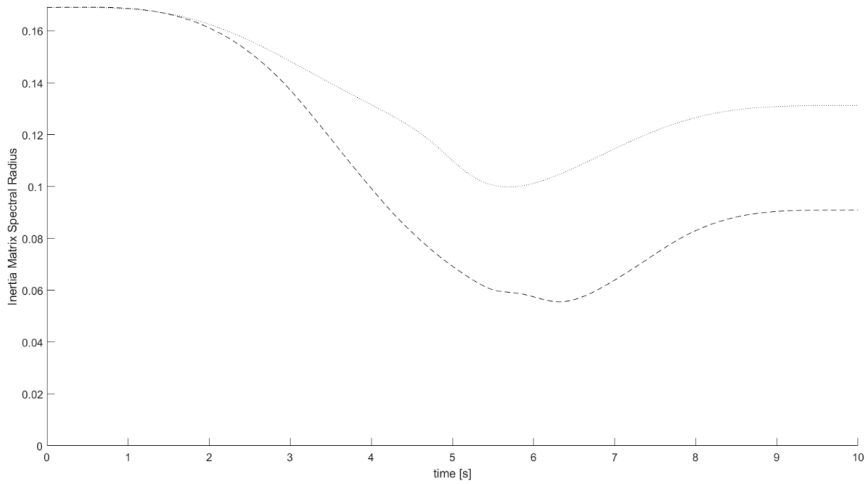

Figure 8. Comparison of the spectral radius of the inertia matrix using PMKE (dashed line) and LSV (dotted line) methods during the execution of trajectory 4.

The figure shows that the maximum eigenvalue of the inertia matrix for the PMKE trajectory is always lower than the maximum eigenvalue for the LSV one, indicating that the matrix coefficients are generally lower. Particularly, the maximum eigenvalue for PMKE is as low as 0.0556 for t = 6.33 s, while for LSV its minimum value is 0.0998 for t = 5.71 s.

While the comparison with the LSV method can help to understand how the PMKE method stands with respect to another local method, it is also worth noticing that the solutions obtained with the PMKE method for the considered trajectories are very similar to the globally optimal ones. Figures 9–12 show the PMKE solutions on the left and the global optimal solutions on the right, both stroboscopic plots and joint variables.

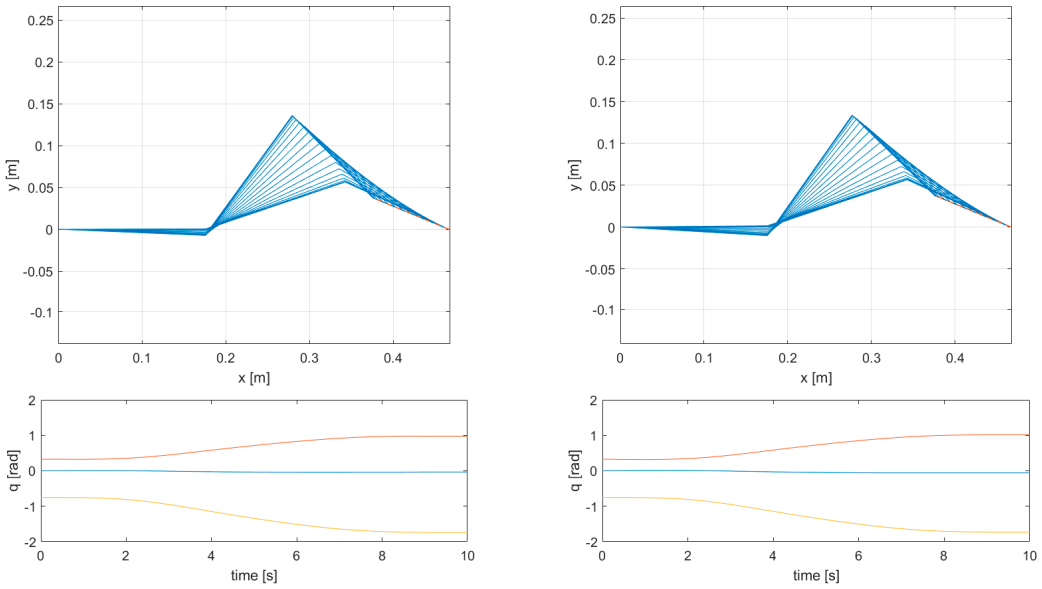

Figure 9. Trajectory 1, comparison between PMKE (**left**) and global optimal solution (**right**). First joint: blue lines; second joint: red lines; third joint: yellow lines.

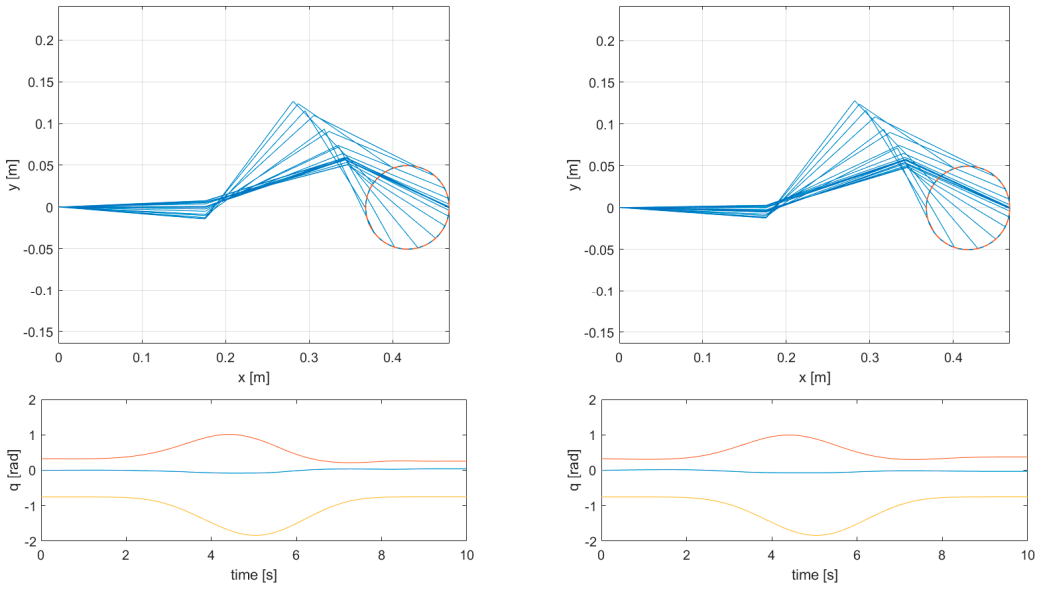

Figure 10. Trajectory 2, comparison between PMKE (**left**) and global optimal solution (**right**). First joint: blue lines; second joint: red lines; third joint: yellow lines.

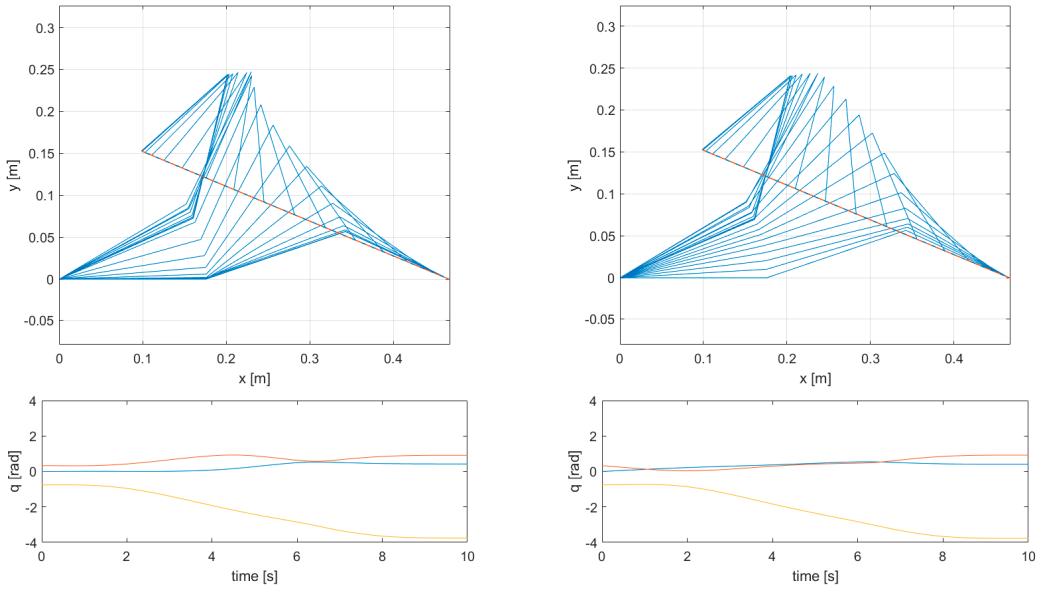

Figure 11. Trajectory 3, comparison between PMKE (**left**) and global optimal solution (**right**). First joint: blue lines; second joint: red lines; third joint: yellow lines.

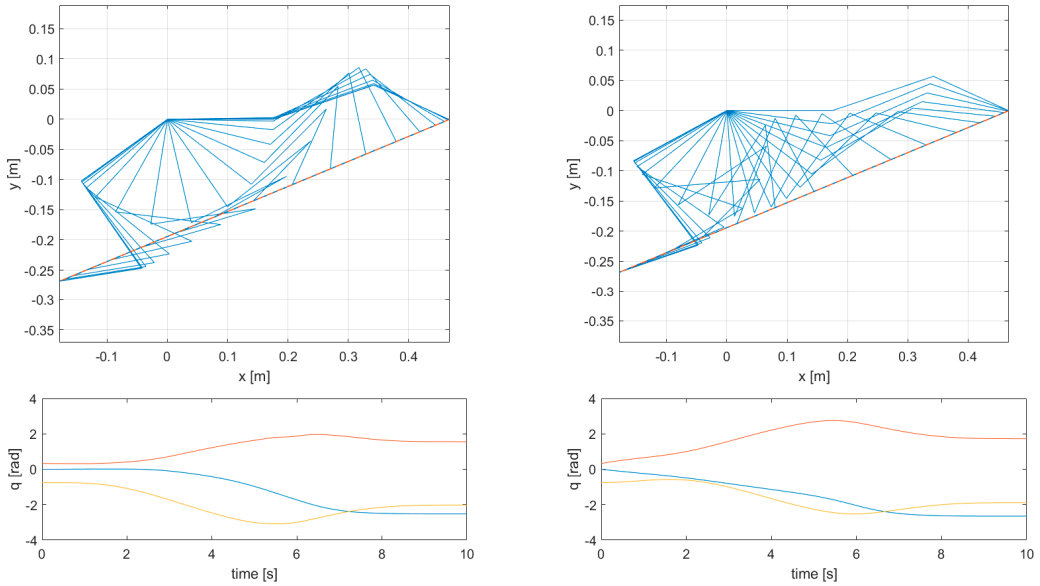

Figure 12. Trajectory 4, comparison between PMKE (**left**) and global optimal solution (**right**). First joint: blue lines; second joint: red lines; third joint: yellow lines.

For trajectories 1 and 2, the similarity can be easily noticed in both the stroboscopic and joint variables plots. The longer trajectories (3 and 4) feature some difference in terms of joint positions; however, the solutions are still very similar as it can be noticed in the stroboscopic plots. A complete overlap is not possible with the limited prediction window of the PMKE method: the longer the trajectory is, the higher the chance that local effects will influence the final energetic cost. However, the PMKE method is able to compute the best joint velocities to perform kinetic energy minimization with the chosen prediction window, and these are very similar to the ones of the global optimal solution.

4. Conclusions

In this paper, a novel inverse kinematics method for redundant manipulators, called Predictive Minimization of Kinetic Energy (PMKE) method, is presented, which is able

to provide a solution near to the global optimal one, is able to avoid singularities, and is suitable for real-time implementation. The PMKE method has been tested by simulation for a three degrees of freedom (DOF) planar manipulator in a number of test cases, and its performance has been compared to the classical pseudoinverse solution and to the one computed with a global optimal method. The simulations showed that the proposed method outperforms the pseudoinverse-based one and is suitable to avoid singularities. Furthermore, it provides a solution very close to the global optimal one with a much lower computational time, which is compatible for real-time implementation. The optimization of different cost functions (e.g., torques norm integral), the simulation of the performance in 3D and higher-DOF manipulators, a more optimized C/C++ implementation of the method to allow for faster computation, and the implementation in a real robot prototype will be part of future work.

Author Contributions: Conceptualization, S.C. and A.T.; methodology, S.C. and A.T.; software, A.T.; investigation, S.C. and A.T.; writing—original draft preparation, S.C. and A.T.; writing—review and editing, S.C. and A.T.; supervision, S.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Flores-Abad, A.; Ma, O.; Pham, K.; Ulrich, S. A review of space robotics technologies for on-orbit servicing. *Prog. Aerosp. Sci.* **2014**, *68*, 1–26. [CrossRef]
2. Nenchev, D.N. Redundancy resolution through local optimization: A review. *J. Robot. Syst.* **1989**, *6*, 769–798. [CrossRef]
3. Cocuzza, S.; Pretto, I.; Debei, S. Least-Squares-Based Reaction Control of Space Manipulators. *J. Guid. Control. Dyn.* **2012**, *35*, 976–986. [CrossRef]
4. Cocuzza, S.; Pretto, I.; Debei, S. Novel reaction control techniques for redundant space manipulators: Theory and simulated microgravity tests. *Acta Astronaut.* **2011**, *68*, 1712–1721. [CrossRef]
5. Cocuzza, S.; Tringali, A.; Yan, X.-T. Energy-efficient motion of a space manipulator. In Proceedings of the 67th International Astronautical Congress, Guadalajara, Mexico, 26–30 September 2016.
6. Kelemen, M.; Virgala, I.; Lipták, T.; Miková, L.; Filakovský, F.; Bulej, V. A novel approach for a inverse kinematics solution of a redundant manipulator. *Appl. Sci.* **2018**, *8*, 2229. [CrossRef]
7. Woolfrey, J.; Lu, W.; Liu, D. A Control Method for Joint Torque Minimization of Redundant Manipulators Handling Large External Forces. *J. Intell. Robot. Syst. Theory Appl.* **2019**, *96*, 3–16. [CrossRef]
8. Sarkar, N.; Podder, T.K. Coordinated motion planning and control of autonomous underwater vehicle-manipulator systems subject to drag optimization. *IEEE J. Ocean. Eng.* **2001**, *26*, 228–239. [CrossRef]
9. Klein, C.A.; Huang, C.H. Review of Pseudoinverse Control for Use with Kinematically Redundant Manipulators. *IEEE Trans. Syst. Man Cybern.* **1983**, *SMC-13*, 245–250. [CrossRef]
10. Baillieul, J.; Hollerbach, J.M.; Brockett, R. Programming and control of kinematically redundant manipulators. In Proceedings of the 23rd Conference on Decision and Control, Las Vegas, NV, USA, 12–14 December 1984; pp. 768–774.
11. Whitney, D. Resolved Motion Rate Control of Manipulators and Human Prostheses. *IEEE Trans. Man Mach. Syst.* **1969**, *10*, 47–53. [CrossRef]
12. Hollerbach, J.M.; Suh, K.C. Redundancy Resolution of Manipulators through Torque Optimization. *IEEE J. Robot. Autom.* **1987**, *3*, 308–316. [CrossRef]
13. Nedungadi, A.; Kazerouinian, K. A local solution with global characteristics for the joint torque optimization of a redundant manipulator. *J. Robot. Syst.* **1989**, *6*, 631–654. [CrossRef]
14. Vukobratovic, M.; Kircanski, M. A Dynamic Approach to Nominal Trajectory Synthesis for Redundant Manipulators. *IEEE Trans. Syst. Man Cybern.* **1984**, *103*, 580–586. [CrossRef]
15. Maciejewski, A.A.; Klein, C.A. Numerical Filtering of the Operation of Robotic Manipulators through Kinematically Singular Configurations. *J. Robot. Syst.* **1988**, *5*, 527–552. [CrossRef]
16. Hu, B.; Teo, C.L.; Lee, H.P. Local Optimization of Weighted Joint Torques for Redundant Robotic Manipulators. *IEEE Trans. Robot. Autom.* **1995**, *11*, 422–425. [CrossRef]
17. Kanoun, O.; Lamiraux, F.; Wieber, P. Kinematic control of redundant manipulators: Generalizing the task priority framework to inequality tasks. *IEEE Trans. Robot.* **2011**, *27*, 1–9. [CrossRef]
18. Zhang, Y.; Ge, S.S.; Lee, T.H. A unified quadratic-programming-based dynamical system approach to joint torque optimization of physically constrained redundant manipulators. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2004**, *34*, 2126–2132. [CrossRef]

19. Gregory, J.; Olivares, A.; Staffetti, E. Energy-optimal trajectory planning for robot manipulators with holonomic constraints. *Syst. Control Lett.* **2012**, *61*, 279–291. [CrossRef]
20. Halevi, Y.; Carpanzano, E.; Montalbano, G. Minimum Energy Control of Redundant Linear Manipulators. *J. Dyn. Syst. Meas. Control* **2014**, *136*, 1–8. [CrossRef]
21. von Stryk, O.; Schlemmer, M. Optimal Control of the Industrial Robot Manutec r3. *Comput. Optim. Control* **1994**, *115*, 367–382.
22. Saramago, S.F.P.; Steffen, V., Jr. Trajectory Modeling of Robot Manipulators in the Presence of Obstacles. *J. Optim. Theory Appl.* **2001**, *110*, 17–34. [CrossRef]
23. Field, G.; Stepanenko, Y. Iterative Dynamic Programming: An Approach to Minimum Energy Trajectory Planning for Robotic Manipulators. In Proceedings of the 1996 IEEE Intemational Conference on Robotics and Automation, Minneapolis, MN, USA, 22–28 April 1996; pp. 2755–2760.
24. Nurmi, J.; Mattila, J. Global energy-optimal redundancy resolution of hydraulic manipulators: Experimental results for a forestry manipulator. *Energies* **2017**, *10*, 647. [CrossRef]
25. Tringali, A.; Cocuzza, S. Globally optimal inverse kinematics method for a redundant robot manipulator with linear and nonlinear constraints. *Robotics* **2020**, *9*, 61. [CrossRef]
26. Faroni, M.; Beschi, M.; Pedrocchi, N.; Visioli, A. Predictive Inverse Kinematics for Redundant Manipulators with Task Scaling and Kinematic Constraints. *IEEE Trans. Robot.* **2019**, *35*, 278–285. [CrossRef]
27. Nakamura, Y.; Hanafusa, H.; Yoshikawa, T. Task-Priority Based Redundancy Control of Robot Manipulators. *Int. J. Robot. Res.* **1987**, *6*, 32–42. [CrossRef]