Article

A Novel Resolution Scheme of Time-Energy Optimal Trajectory for Precise Acceleration Controlled Industrial Robot Using Neural Networks

Renluan Hou [1], Jianwei Niu [2], Yuliang Guo [1], Tao Ren [1,*], Xiaolong Yu [1], Bing Han [1] and Qun Ma [1]

[1] Hangzhou Innovation Institute, Beihang University, Hangzhou 310051, China; jessierlhou@zju.edu.cn (R.H.); guoyuliang@buaa.edu.cn (Y.G.); yxiaolong@buaa.edu.cn (X.Y.); hbing@buaa.edu.cn (B.H.); maqun@buaa.edu.cn (Q.M.)

[2] State Key Laboratory of Virtual Reality Technology and Systems, School of Computer Science and Engineering, Beihang University, Beijing 100191, China; niujianwei@buaa.edu.cn

* Correspondence: rentaobh@buaa.edu.cn

Abstract: The surging popularity of adopting industrial robots in smart manufacturing has led to an increasing trend in the simultaneous improvement of the energy costs and operational efficiency of motion trajectory. Motivated by this, multi-objective trajectory planning subject to kinematic and dynamic constraints at multiple levels has been considered as a promising paradigm to achieve the improvement. However, most existing model-based multi-objective optimization algorithms tend to come out with infeasible solutions, which results in non-zero initial and final acceleration. Popular commercial software toolkits applied to solve multi-objective optimization problems in actual situations are mostly based on the fussy conversion of the original objective and constraints into strict convex functions or linear functions, which could induce a failure of duality and obtain results exceeding limits. To address the problem, this paper proposes a time-energy optimization model in a phase plane based on the Riemann approximation method and a solution scheme using an iterative learning algorithm with neural networks. We present forward-substitution interpolation functions as basic functions to calculate indirect kinematic and dynamic expressions introduced in a discrete optimization model with coupled constraints. Moreover, we develop a solution scheme of the complex trajectory optimization problem based on artificial neural networks to generate candidate solutions for each iteration without any conversion into a strict convex function, until minimum optimization objectives are achieved. Experiments were carried out to verify the effectiveness of the proposed optimization solution scheme by comparing it with state-of-the-art trajectory optimization methods using Yalmip software. The proposed method was observed to improve the acceleration control performance of the solved robot trajectory by reducing accelerations exceeding values of joints 2, 3 and 5 by 3.277 rad/s^2, 26.674 rad/s^2, and 7.620 rad/s^2, respectively.

Keywords: time-energy optimal trajectory; acceleration control; complex optimization problems; artificial neural networks; industrial robot

check for updates

Citation: Hou, R.; Niu, J.; Guo, Y.; Ren, T.; Yu, X.; Han, B.; Ma, Q. A Novel Resolution Scheme of Time-Energy Optimal Trajectory for Precise Acceleration Controlled Industrial Robot Using Neural Networks. *Actuators* **2022**, *11*, 130. https://doi.org/10.3390/act11050130

Academic Editors: Jinchuan Zheng, Hai Wang, Silu Chen and Ke Shao

Received: 1 April 2022
Accepted: 29 April 2022
Published: 3 May 2022

1. Introduction

As industrial robots are applied in more and more manufacturing scenarios, high efficiency and low energy cost of the motion trajectory are essential to meet the production needs and reduce resource consumption [1–3]. Time-energy cost-optimal trajectory planning for robots given a task path is conducted while surrendering limitations of kinematic and dynamic capabilities for safe operation. Effective trajectory optimization resolution of redundant robots with compliance to single-level physical constraints, such as velocity, is easy to implement. However, existing model-based algorithms for robot path planning may result in joint-angle drift and non-zero final joint velocity and acceleration when systems are subjected to comprehensive velocity, acceleration and torque constraints and controlled at multiple levels, as discussed in [4–6].

Recently, many kinds of solver software, combined with various optimization methods, have been utilized in robot planning research to avoid obtaining infeasible solutions. The objective and constraints input into solver software should follow the specified rules of optimization theories, such as convex optimization (CO), linear programming (LP) [7], quadratic programming (QP) [5], second-order cone programming (SOCP) [3], concave-convex procedure (CCCP) [8], and semi-definite programming (SDP). Each algorithm of the aforementioned software toolkits has its strict solution form and is usually solved by Lagrange duality and the interior point method [9], also known as the barrier method or combined methods. Moreover, the solving process of the above software is implicit, which is not conducive to estimating a global optimum or local minima. Thus, for the trajectory optimization problem of robots with precise multi-level constraints, it is necessary to establish a generalized time-energy consumption optimization model and to develop algorithms with feasible solutions in actual scenes.

To address the issues, this paper proposes a novel multi-objective trajectory optimization approach with the goal of simultaneously minimizing the time and energy consumption of industrial robots. First, we formulate the optimization problem by establishing the weighting function of time and energy consumption in generalized path coordinates. The transformed trajectory variables in the weighting function corresponding to the angle, velocity and acceleration in joint coordinates are considered as generated optimizing variables. Then, we present forward-substitution interpolation functions as basic functions to calculate indirect velocity and acceleration expressions introduced in the discrete optimization model and obtain segmented smooth paths in terms of trajectory variables. Consequently, we impose an iterative metaheuristic scheme to solve trajectory variables in the complex optimization problem based on artificial neural networks, avoiding the fussy conversion of strict convex or linear functions. Moreover, velocity, acceleration and torque constraints controlled in joint performance parameter limits in the overall process of motion are also the focus of the trajectory design and optimization in this paper, according to the actual needs of industrial robots. The proposed trajectory optimization model can adjust the variable values according to the requirement of high-speed motion or low energy consumption. Finally, extensive experiments are conducted to verify the effectiveness of the proposed optimization solution scheme by comparing it with the state-of-the-art trajectory optimization methods using Yalmip software.

2. Related Work

Generally, to solve a trajectory planning problem, joint angles of an industrial robot, given the desired path described in Cartesian space, are first calculated by an inverse kinematic model. Then, trajectory variables, including velocity and acceleration as functions of angle derivative in terms of time, are optimized to achieve the minimum running time or energy consumption with the demands of multi-level control for safe operation. Feng et al. [10] employed fifth-order polynomial expressions to construct joint trajectories and established a time-energy consumption optimization model by introducing velocity variables obtained from the pre-defined trajectory. Since an industrial robot usually has six joints, the optimization objective of all joint angles is a multidimensional expression. For simplification of the objective function, Palleschi et al. [11] described path coordinates as a normalized trajectory variable based on phase plane theory to reduce the high-dimensional state space and solve the tracking problem of a minimum-time path with smooth and continuous accelerations by the SCIP solver of Opti Toolbox. Although software can generate a set of optimized solutions, the implicit solving process is often difficult to converge when multi-level constraints of the robot are required to be controlled.

For the control of robots with optimal trajectory under the multi-level constraints enforced, a conventional solution is the Jacobian pseudo-inverse method, which can be regarded as an analytical solution with a direct correlation between end-effector and joint trajectory variables. Ramezani and Williams [12] present an optimal redundancy resolution technique by using the augmented Jacobian to provide feasible solutions for the minimum objective function. The traditional Jacobian pseudo-inverse scheme is just applicable to trajectory optimization with joint physical limits expressed as equality equations, while inequality constraints can hardly be introduced to this direct solution scheme.

To achieve efficient redundancy resolution for multi-level control of a robot's trajectory, Verscheure et al. [3] transformed the optimization model with consideration of time, energy cost and smoothness into a convex optimization problem, and presented a transcription method to solve a SOCP with inequality constraints using robust numerical algorithms in Yalmip. Nagy and Vajk [7] converted a profile generation model into linear programming (LP) form and solved it with a sequential optimization method. Steinhauser and Swevers [4] presented a two-step iterative learning algorithm that compensates for inevitable model-plant mismatch of time-optimal motion, which improved tracking performance and ensured feasibility based on a sequential convex log barrier method. Zhang et al. [6] formulated the trajectory resolution problem subject to joint angle, velocity, and acceleration constraints as a QP to overcome computationally intensive pseudo-inverse-based schemes. The optimization models referred to require the objective and constraints to be converted to satisfy the strict solving forms, which sometimes induces a failure of duality and obtains results exceeding limits.

Among optimization approaches, metaheuristic algorithms have shown their capabilities for searching for near-global-optimal solutions to numerical real-valued test problems, such as the genetic algorithm (GA) [13,14], particle swarm optimization (PSO) [15], artificial neural networks (ANNs) [16,17], and so forth. Chen et al. [5] formulated a multi-level simultaneous minimization scheme as a dynamical quadratic program (DQP), which was solved by a piecewise linear projection equation neural network. Abu-Dakka et al. [14] constructed smooth joint trajectories with cubic polynomial functions as basic functions in a segmented path to establish an optimization model of minimum time, energy and jerk. Undetermined parameters in basic functions were solved by a parallel-populations genetic algorithm (PPGA) procedure. Based on the multi-objective genetic optimization algorithm NSGA-II, Shi et al. [18] solved the optimization problem in the multi-objective form.

Although the above state-of-the-art metaheuristic algorithms have good effects on solving general optimization problems, some of these methods applied in trajectory planning cannot guarantee the initial and final zero-velocity and zero-acceleration and joint-angle drift sometimes occurs, so are not suitable for actual conditions of the robot operation. In addition to the solution algorithm, the established trajectory optimization model and the form of internal basic function for iteration calculation, dynamic equation expressions, and other factors will also have an impact on the output trajectories results. Thus, it is necessary to establish a multi-objective optimization model with consideration of the applicable conditions in the actual situation.

3. Robot Kinematics and Dynamics Modeling

In this paper, a serial 6-axis robot is considered to establish the time-energy optimization model, the structure of which is shown in Figure 1. The modified Denavit–Hartenberg (MDH) parameters are presented in Table 1. Forward and inverse kinematic models of the specified robot can be established with these parameters.

Dynamic equations of the robot with several identified parameters are established to express the joint torques ${}^{NE}\tau_i$ according to the iterative Newton-Euler formulation [19]. To establish an accurate dynamic identification model, inertia terms ${}^{r}\tau_i$ and friction terms ${}^{f}\tau_i$ are introduced in the torques expressions [20] as follows:

$$\tau_i = {}^{r}\tau_i + {}^{NE}\tau_i + {}^{f}\tau_i = I_{ai}\ddot{q}_i + {}^{NE}\tau_i + f_{vi}\dot{q}_i + f_{ci}\mathrm{sign}(\dot{q}_i), \tag{1}$$

where $\dot{q}_i$ and $\ddot{q}_i$ denote the velocity and acceleration of joint i, I_{ai} is the inertia moment for rotor and gears of actuator i, f_{vi} and f_{ci} are the viscous and Coulomb friction coefficients. It should be noted that the inertia tensor ${}^{C}\mathbf{I}_i$ related to the center of mass of link in ${}^{NE}\tau_i$ should be converted in terms of the inertia tensor ${}^{A}\mathbf{I}_i$ related to the origin of the joint coordinate by introducing the expression

$$ {}^{C}\mathbf{I}_i = {}^{A}\mathbf{I}_i - m_i[\mathbf{P}_{Ci}^{\mathrm{T}}\mathbf{P}_{Ci}\mathbf{I}_3 - \mathbf{P}_{Ci}\mathbf{P}_{Ci}^{\mathrm{T}}], \tag{2}$$

where ${}^{A}\mathbf{I}_i = \begin{bmatrix} I_{xxi} & -I_{xyi} & -I_{xzi} \\ -I_{xyi} & I_{yyi} & -I_{yzi} \\ -I_{xzi} & -I_{yzi} & I_{zzi} \end{bmatrix}$, m_i is the mass of link i, $\mathbf{I}_3$ is the 3×3 identity matrix, the undetermined vector $\mathbf{P}_{Ci} = \begin{bmatrix} x_{Ci}, & y_{Ci}, & z_{Ci} \end{bmatrix}^{\mathrm{T}}$ locates the center of mass of link i relative to the coordinate system $\{O_i\}$. The vector of the identified parameters for each joint torque τ_i is denoted as

$$\Omega_i = [\, m_i \quad x_{Ci} \quad y_{Ci} \quad z_{Ci} \quad I_{xxi} \quad I_{yyi} \quad I_{zzi} \quad I_{xyi} \quad I_{xzi} \quad I_{yzi} \quad I_{ai} \quad f_{vi} \quad f_{ci}]^{\mathrm{T}} \ , \tag{3}$$

which contains position coordinate components of the center of mass of link (x_{Ci}, y_{Ci}, z_{Ci}), inertia moment components of link i $(I_{xxi}, I_{yyi}, I_{zzi}, I_{xyi}, I_{xzi}, I_{yzi})$.

The differential Equation (1) in terms of 78 dynamic parameters can be linearized and expressed by

$$\boldsymbol{\tau} = \boldsymbol{\omega}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) \cdot \boldsymbol{\Omega}, \tag{4}$$

where $\boldsymbol{\tau} = [\tau_1, \tau_2, \ldots, \tau_6]^{\mathrm{T}}$ is the joint torques vector, $\boldsymbol{\Omega} = [\Omega_1, \Omega_2, \ldots, \Omega_6]^{\mathrm{T}}$ is the standard parameters vector, $\boldsymbol{\omega}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$ is a 6×78 regressor matrix and $\boldsymbol{q}$ is the joint position vector. The above equation can be transformed into a simplified form with a minimal set of 52 identifiable parameters $\boldsymbol{\Omega}_{\min}$ by regrouping the original parameters $\boldsymbol{\Omega}$, a detailed derivation process for which can be seen in [21,22]. Thus, the torque can be simplified as

$$\boldsymbol{\tau} = \boldsymbol{\omega}_{min}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) \cdot \boldsymbol{\Omega}_{\min}, \tag{5}$$

where $\boldsymbol{\omega}_{min}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$ is a subset of the independent columns of $\boldsymbol{\omega}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}})$. We sample several data of torques and joint angles when the robot moves according to an exciting trajectory at $t = t_1, t_2, \ldots, t_N$, and over-determined linear equations are obtained as

$$\Gamma = \Psi \cdot \boldsymbol{\Omega}_{\min} + \varepsilon, \tag{6}$$

where observation matrix is $\Psi = \begin{bmatrix} \boldsymbol{\omega}_{\min}(\boldsymbol{q}(t_1), \dot{\boldsymbol{q}}(t_1), \ddot{\boldsymbol{q}}(t_1)) \\ \boldsymbol{\omega}_{\min}(\boldsymbol{q}(t_2), \dot{\boldsymbol{q}}(t_2), \ddot{\boldsymbol{q}}(t_2)) \\ \vdots \\ \end{bmatrix}$, the torques vector is $\Gamma = \begin{bmatrix} \boldsymbol{\tau}(t_1)^{\mathrm{T}}, & \boldsymbol{\tau}(t_2)^{\mathrm{T}}, & \ldots & \boldsymbol{\tau}(t_N)^{\mathrm{T}} \end{bmatrix}$, and ε is a vector of sampling error. Based on the least square method, the minimum identification parameters set can be obtained by

$$\boldsymbol{\Omega}_{\min} = \left(\Psi^T \Psi\right)^{-1} \Psi^T \Gamma. \tag{7}$$

The calculated minimum set of combined parameters $\boldsymbol{\Omega}_{\min}$ is listed in Table A1 in Appendix A. The dynamic model with an iterative Newton–Euler formulation is obtained by substituting $\boldsymbol{\Omega}_{\min}$ into Equation (1).

Then, we rewrite the dynamic model into the state-space equation form, as follows:

$$\tau = M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) + {}_{f}\tau(\dot{q}), \tag{8}$$

where $M(q)$ is the mass or inertia matrix, $C(q, \dot{q})$ is the vector of Coriolis and centrifugal terms, $G(q)$ is the vector of gravity terms, and ${}^{f}\tau(\dot{q}) = F_c \mathrm{sgn}(\dot{q}) + F_v \dot{q}$ is the vector of friction terms.

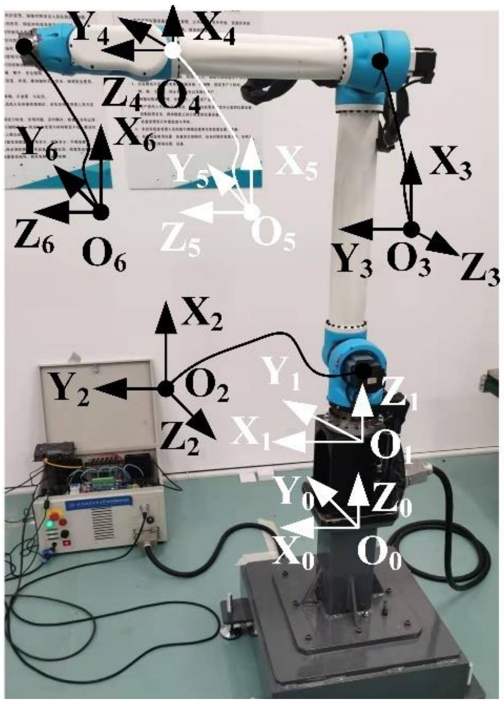

Figure 1. Specified robot structure.

Table 1. Modified Denavit–Hartenberg parameters of the specified robot.

Joint *i*	α_{i-1} **(rad)**	a_{i-1} **(m)**	d_i **(m)**	θ_i **(rad)**
1	0	0	0.435	θ_1
2	−1.571	−0.00022	0	θ_2 (−π/2)
3	0	0.765522	0.060267	θ_3
4	−1.571	0.004315	0.63512	θ_4
5	1.571	0	0	θ_5
6	−1.571	0	0.156	θ_6

4. Problem Formulation

This section formulates the time-energy optimal trajectory planning problem of a robot with precise multi-level constraints. The optimization objective and complex constraints, including velocity, acceleration and torque, are derived based on the phase plane theory and Riemann approximation method. For effective numerical computing, the optimization model introduces the discrete kinematic and dynamic equation results.

4.1. Original Formulation

Minimizing the total motion time t_f of an industrial robot is simply expressed as $\int_0^{t_f} 1dt$. For optimization of the energy cost, the main resistance energy consumption of the servo circuit in each robot joint is modeled by Joule's law, as follows:

$$^{i}e = {}^{i}I_{servo}^{2} \cdot {}^{i}r \cdot t, \text{ for } i = 1, \ldots, 6, \tag{9}$$

where $^{i}I_{servo}$ is electric current and ^{i}r is the resistance. Due to the relevant relations between torque and current, the above equation can be proportional to the expression in terms of torque:

$$^{i}e \propto \tau_i^2 \cdot {}^{i}r \cdot t, \tag{10}$$

For a single servo motor system, the energy consumption model during the running time should be:

$$^{i}E \propto \int_0^{t_f} \tau_i^2 \cdot {}^{i}r dt \propto \int_0^{t_f} \tau_i^2 dt, \tag{11}$$

which omits the coefficient ${}^{i}r$ since it is constant for one joint. Then, energy consumption ${}^{i}E$ is also proportional to the integral of the torque square $\int_0^{t_f} \tau_i^2 dt$. Total energy consumption of a robot can be optimized by minimizing the normalized expression:

$$\min \sum_{i=1}^{6} \int_0^{t_f} \frac{\tau_i^2}{(\overline{\tau}_i)^2} dt \ , \tag{12}$$

which divides the square of maximum torque limit $(\overline{\tau}_i)^2$ for equalized optimization of every joint energy.

For simultaneous minimizing of the time and energy of a robot with multi-level constraints, the following synthetical optimization model is established as

$$\min \int_0^{t_f} 1dt + \beta \sum_{i=1}^{6} \int_0^{t_f} \frac{\tau_i^2}{(\overline{\tau}_i)^2} dt \ , \tag{13}$$

Subjected to

$$\left.\begin{array}{llll} \dot{q}_i(0) = 0, & \dot{q}_i(t_f) = 0, & \ddot{q}_i(0) = 0, & \ddot{q}_i(t_f) = 0, \\ \underline{\dot{q}}_i \le \dot{q}_i(t) \le \overline{\dot{q}}_i, & \underline{\ddot{q}}_i \le \ddot{q}_i(t) \le \overline{\ddot{q}}_i, & \underline{\tau} \le \tau(t) \le \overline{\tau}, & \end{array}\right\}, \text{ for } i = 1, \ldots, 6, \tag{14}$$

where β is the weight coefficient; overline and underline of velocity, acceleration and torque represent the maximum and minimum values.

4.2. Reformulation Based on Phase Plane Theory

Due to the complexity of the multidimensional-optimal problem for an industrial robot with six joints, the optimization objective is calculated using six different trajectories functions until the minimum operation time value for any of all joints is found. For simplification of the objective function, we describe path coordinates as a parameterized trajectory variable based on phase plane theory to reduce the high-dimensional state space to one dimension.

Based on the Pontryagin principle [23], the optimization independent variable t can be replaced by the trip proportion variable assigned to the path curve $s(t)$ through a generalized coordinate transformation $t = \int_0^s \frac{1}{\dot{s}(t)} ds$. Then, the reformulated optimization problem is analyzed in the S-phase plane. Joint position $\boldsymbol{q}$ is a function of s, notated as $q(s)$. Joint velocity and acceleration are transformed as

$$\dot{q}(s) = q'(s)\dot{s}, \ \ddot{q}(s) = q'(s)\ddot{s} + q''(s)\dot{s}^2, \tag{15}$$

where $q'(s) = \frac{\partial q(s)}{\partial s}$, $q''(s) = \frac{\partial^2 q(s)}{\partial s^2}$, $\ddot{s}(t) = \frac{d^2 s}{dt^2}$ is defined as $u(s)$, $\dot{s}(t) = \frac{ds}{dt}$. Moreover, substituting the relations (15) into the dynamic equation (8), yields

$$\boldsymbol{\tau}(s) = M(q(s))q'(s)\ddot{s} + M(q(s))q''(s)\dot{s}^2 + C(q(s),\dot{q}(s))q'(s)\dot{s} + G(q(s)) + F_c\mathrm{sgn}(q'(s)\dot{s}) + F_v q'(s)\dot{s}, \tag{16}$$

where the term $C(q,\dot{q})$ is linear to the joint velocity $\dot{q}_i$ [24], which can be described as $C(q(s),\dot{q}(s)) = C(q(s),q'(s))\dot{s}$. Meanwhile, the trajectory velocity $\dot{s}$ is positive when the robot runs along a path, such that the sign function of $q'(s)\dot{s}$ is equal to $q'(s)$. We define $m(s) = M(q(s))q'(s)$, $c(s) = M(q(s))q''(s) + C(q(s),q'(s))q'(s)$, $g(s) = F_c\mathrm{sgn}(q'(s)) + G(q(s))$ and $f(s) = F_v q'(s)$ for simplification of Equation (16).

The normalized time-energy consumption optimization model for the robot trajectory is reformulated by substituting expression (16) and $t_f = \int_0^1 \frac{1}{\dot{s}(t)} ds$ as

$$\min \int_0^1 \frac{1}{h(s)} \left[1 + \beta \sum_{i=1}^{6} \frac{\tau_i^2(s)}{(\overline{\tau}_i)^2} \right] ds, \tag{17}$$

where $h(s)$ represents $\dot{s}(t) = \frac{ds}{dt}$.

Since the integral optimization model is difficult to be solved, it is converted to discrete sums based on the Riemann approximation method. Then, the corresponding discretization and normalization are carried out for the time-energy optimization objective with multi-level constraints, as follows:

$$\min \sum_{j=1}^{N-1} \left(\frac{\Delta s_{j+1}}{h(s_{j+1})} \right) \left[1 + \beta \sum_{i=1}^{6} \frac{\tau_i^2(s_{j+1})}{(\overline{\tau}_i)^2} \right] = \sum_{j=1}^{N-1} \Delta s_{j+1} v(s_{j+1}), \tag{18}$$

where s_j is the j-th point proportion of the divided path curve, $\Delta s_{j+1} = s_{j+1} - s_j$. Simultaneously, velocity and acceleration in joint coordinate and torque constraints are discrete as

$$\begin{gathered} {q'}_i(s_1) \cdot h(s_1) = 0,\ {q'}_i(s_N) \cdot h(s_N) = 0,\ {q'}_i(s_1)u(s_1) + {q''}_i(s_1)h(s_1)^2 = 0,\ {q'}_i(s_n)u(s_n) + {q''}_i(s_n)h(s_n)^2 = 0 \\ \underline{\dot{q}}_i \leq {q'}_i(s_j) \cdot h(s_j) \leq \overline{\dot{q}}_i,\ \underline{\ddot{q}}_i \leq {q'}_i(s_j)u(s_j) + {q''}_i(s_j)h(s_j)^2 \leq \overline{\ddot{q}}_i \\ \underline{\tau} \leq \tau(s_j) = m(s_j)u(s_j) + c(s_j)h(s_j)^2 + g(s_j) + f(s_j)h(s_j) \leq \overline{\tau},\ \text{for } i = 1, \ldots, 6,\ j = 1, \ldots, n \end{gathered} \tag{19}$$

where $u(s_1) = 0$, $u(s_{j+1}) = \frac{h(s_{j+1})^2 - h(s_j)^2}{2\Delta s_{j+1}}$ for $j = 1, \ldots, n-1$ is assumed as a piecewise function. In the case of the dramatic increase in velocity and acceleration, the corresponding constraints are limited in the range of $1/(n-1)$ multiple maximum values at the initial and final interval segments.

5. Solution Algorithm

In this section, we present a solving scheme of a time-energy optimal trajectory with a forward-substitution interpolation method and metaheuristic algorithm. Firstly, high-order polynomial interpolation functions are applied as basic functions to calculate indirect kinematic and dynamic expressions introduced in a discrete optimization model and other constraints. Then, we develop a solution algorithm based on artificial neural networks to generate candidate solutions for each iteration without any conversion into strict forms, until a minimum objective function value is achieved.

5.1. Forward-Substitution Interpolation Basic Functions

A path that a robot usually moves around is specified by a series of space position coordinates $P_k = (x_k, y_k, z_k)$. Geometric curve functions, such as the cubic Bezier or B-spline curve, could be used to fit these points, as

$$B_k(S) = \begin{bmatrix} S^3 & S^2 & S & 1 \end{bmatrix} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_k \\ {}^2P_k \\ {}^1P_{k+1} \\ P_{k+1} \end{bmatrix}, S \in [0,1] \tag{20}$$

where control points ${}^2P_k = P_k + 0.15P_{k+1} - 0.15P_{k-1}$, ${}^1P_{k+1} = P_{k+1} + 0.15P_k - 0.15P_{k+2}$. Then the interpolated path points $P_{11}, P_{12}, \ldots$ coordinates are calculated by the cubic Bezier curve function, and we add the number of path points to N. For convenience, we notate points as $Q_1, Q_2, \ldots, Q_l, \ldots, Q_N$. Although the partitioned Bezier curve function describes the discrete interpolated points space coordinates between the specified points along the path. Each joint positions vector $\mathbf{q}(Q_l) = [q_1(Q_l), q_2(Q_l), q_3(Q_l), q_4(Q_l), q_5(Q_l), q_6(Q_l)]^{\mathrm{T}}$ in the joint coordinate system corresponding to the path point can be resolved by the inverse kinematic model with position coordinate (x, y, z) and specified pose orientation vector, as rotations around the x-axis, y-axis, z-axis (rx, ry, rz).

To express the joint trajectory function, isometric distributed generalized path curve parameters $S_1, \ldots, S_N$ are introduced to establish the mapping relation of unevenly distributed trajectory points $\mathbf{q}(Q_1), \ldots, \mathbf{q}(Q_N)$. Figure 2 shows the correspondence relations diagram of path points and generalized path curve parameters. Due to the velocity and acceleration requirements, the third-order polynomial function is the lowest order form we choose to be an interpolated trajectory function in terms of the curve parameter S. Moreover, the initial and final trajectory functions with zero-velocity and zero-acceleration are considered by applying the fifth-order polynomial function. The common fitting method of the fixed segmentation function just uses joint angles of two adjacent points, such as $\mathbf{q}_i(Q_1)$ and $\mathbf{q}_i(Q_2)$, to establish the trajectory function. In this case, the higher derivatives of the adjacent trajectory functions have saltation at the connecting point. To solve this problem, we propose a forward-substitution interpolation as a predefined basic trajectory function expressed as

$$\begin{bmatrix} S_1{}^5 & S_1{}^4 & S_1{}^3 & 1 \\ S_2{}^5 & S_2{}^4 & S_2{}^3 & 1 \\ & \vdots & & \\ S_K{}^5 & S_K{}^4 & S_K{}^3 & 1 \end{bmatrix} \begin{bmatrix} {}^I b_1\left(\frac{1}{S}\right)^5 \\ {}^I b_2\left(\frac{1}{S}\right)^4 \\ {}^I b_3\left(\frac{1}{S}\right)^3 \\ {}^I b_4 \end{bmatrix} = \begin{bmatrix} q_i(Q_1) \\ q_i(Q_2) \\ \vdots \\ q_i(Q_K) \end{bmatrix}, \text{ for } K \le d+1 \cup K-d-l \le 1 \tag{21}$$

$$\begin{bmatrix} S_K{}^5-10S_K{}^2+15S_K & S_K{}^4-6S_K{}^2+8S_K & S_K{}^3-3S_K{}^2+3S_K & 1 \\ S_{K+1}{}^5-10S_{K+1}{}^2+15S_{K+1} & S_{K+1}{}^4-6S_{K+1}{}^2+8S_{K+1} & S_{K+1}{}^3-3S_{K+1}{}^2+3S_{K+1} & 1 \\ & \vdots & & \\ S_N{}^5-10S_N{}^2+15S_N & S_N{}^4-6S_N{}^2+8S_N & S_N{}^3-3S_N{}^2+3S_N & 1 \end{bmatrix} \begin{bmatrix} b_1\left(\frac{1}{S}\right)^5 \\ b_2\left(\frac{1}{S}\right)^4 \\ b_3\left(\frac{1}{S}\right)^3 \\ b_4 \end{bmatrix} = \begin{bmatrix} q_i(Q_K) \\ q_i(Q_{K+1}) \\ \vdots \\ q_i(Q_N) \end{bmatrix}, \tag{22}$$
$$\text{for } K \ge N-d+1 \cup K+d+l \ge N-2$$

$$\begin{bmatrix} S_{K-d-l}{}^3 & S_{K-d-l}{}^2 & S_{K-d-l}{}^1 & 1 \\ S_{K-d-l+1}{}^3 & S_{K-d-l+1}{}^2 & S_{K-d-l+1}{}^1 & 1 \\ & \vdots & & \\ S_{K+d+l}{}^3 & S_{K+d+l}{}^2 & S_{K+d+l}{}^1 & 1 \end{bmatrix} \begin{bmatrix} {}^F b_1\left(\frac{1}{S}\right)^3 \\ {}^F b_2\left(\frac{1}{S}\right)^2 \\ {}^F b_3\left(\frac{1}{S}\right)^1 \\ {}^F b_4 \end{bmatrix} = \begin{bmatrix} q_i(Q_{K-d-l}) \\ q_i(Q_{K-d-l+1}) \\ \vdots \\ q_i(Q_{K+d+l}) \end{bmatrix}, \text{ for other situations,} \tag{23}$$

where ${}^I b_1\left(\frac{1}{S}\right)^5$, ${}^I b_2\left(\frac{1}{S}\right)^4$, ${}^I b_3\left(\frac{1}{S}\right)^3$, ${}^I b_4$, ${}^F b_1\left(\frac{1}{S}\right)^5$, ${}^F b_2\left(\frac{1}{S}\right)^4$, ${}^F b_3\left(\frac{1}{S}\right)^3$, ${}^F b_4$, $b_1\left(\frac{1}{S}\right)^3$, $b_2\left(\frac{1}{S}\right)^2$, $b_3\left(\frac{1}{S}\right)$ and b_4 are the polynomial coefficients, d is the nearest integer less than or equal to $0.1(N-1)$, and half of the added continuous selected points number $l \ge 0$ is decided by the set variation value. Quantity l increases until the root mean squared error of the left-fitting value and the right-truth value of Equations (21)–(23) are larger than the set variation value. The super-positive definite Equations (21)–(23) are solved by the least square method.

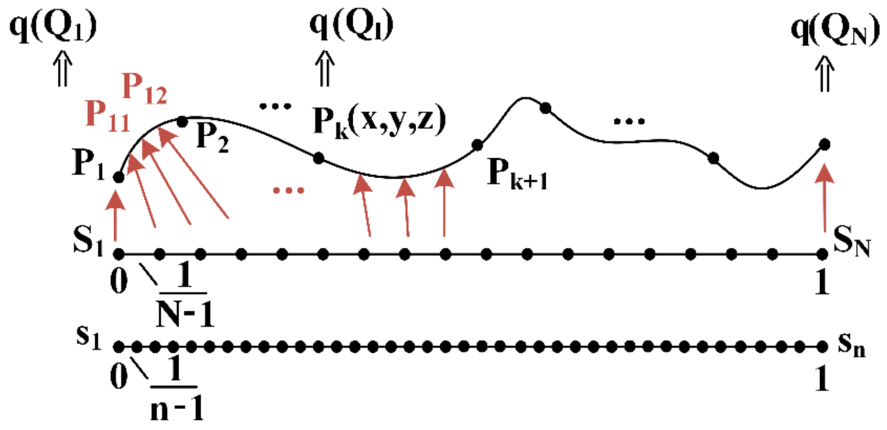

Figure 2. Correspondence relations diagram of path points and generalized path curve parameters.

When the path curve proportion variable s chosen is different from the generalized path curve parameter S, we just need to determine which segment s belongs to. The trajectory function $q_i(s_j)$ and derivative $q'_i(s_j)$ and $q''_i(s_j)$ are substituted into the indirect kinematic and dynamic expressions $M(q(s_j))$, $C(q(s),q'(s))$, $G(q(s))$ in the discrete optimization objective expression (17) and constraints Equation (18).

5.2. Dynamic Metaheuristic Optimization Algorithm

After the pre-defined value Δs_j, $\tau(s_j) = m(s_j)u(s_j) + c(s_j)h(s_j)^2 + g(s_j) + f(s_j)h(s_j)$ and $u(s_{j+1}) = \frac{h(s_{j+1})^2 - h(s_j)^2}{2\Delta s_{j+1}}$ is substituted, the discrete optimization objective $v(s_j)$ is a function in terms of variables $h(s_j)$. The sum terms of a normalized torque square in objective expression (17) will induce high-order nonlinear forms of independent variables $h(s_j)$, which is difficult to be solved by the model-based analytical optimization algorithm or solver software without any conversion. Thus, we develop a solution algorithm based on artificial neural networks (ANN) [17] to handle this issue.

In an n-dimensional optimization problem, a pattern solution representing input data in the ANN, is defined as

$$PatternSolution = \begin{bmatrix} h(s_1) & h(s_2) & \cdots & h(s_n) \end{bmatrix}, \tag{24}$$

First, a starting candidate of the pattern solution matrix H is generated, which is randomly generated between the lower and upper bounds of a problem:

$$Population \quad of \quad Pattern \quad Solutions \ H = \begin{bmatrix} {}^1h(s_1) & {}^1h(s_2) & \cdots & {}^1h(s_n) \\ {}^2h(s_1) & {}^2h(s_2) & & \\ & \vdots & & \\ {}^{N_{pop}}h(s_1) & {}^{N_{pop}}h(s_2) & & {}^{N_{pop}}h(s_n) \end{bmatrix}. \tag{25}$$

Cost functions corresponding to pattern solutions are obtained by

$${}^iF = f\left(\ {}^ih(s_1), \ \ {}^ih(s_2) \ \ \cdots \ \ {}^ih(s_n) \ \right) \qquad for \quad i = 1,2\ldots,N_{pop} \quad , \tag{26}$$

where f is the objective function $\sum\limits_{j=1}^{N-1} \Delta s_{j+1} v(s_{j+1})$ in expression (17).

The candidate solution with the minimum objective function value for all pattern solutions is selected as the target solution. This target solution will be updated at each iteration. After determining the target solution H^{Target} among the other pattern solutions, the target weight W^{Target}, corresponding to the target solution, must be selected from the population of weight (weight matrix) by the following expression:

$$\begin{aligned} W(o) &= \begin{bmatrix} W_1 & W_2 & \cdots & W_{N_{pop}} \end{bmatrix} \\ &= \begin{bmatrix} {}^1w_1 \cdots & {}^iw_1 & \cdots & {}^{N_{pop}}w_1 \\ {}^1w_2 \cdots & {}^iw_2 & & {}^{N_{pop}}w_2 \\ & \vdots & & \\ {}^1w_{N_{pop}} \cdots & {}^iw_{N_{pop}} & & {}^{N_{pop}}w_{N_{pop}} \end{bmatrix} \end{aligned} \tag{27}$$

where $W(o)$ is a matrix generating random numbers uniformly between zero to one during iterations, and o is an iteration index. The weight superscript relates to its pattern solution (e.g., ${}^2w$ is related to the second pattern solution) and the weight subscript is shared with the other pattern solutions (e.g., ${}^2w_3$ is shared with the third pattern solution). Every pattern solution has its corresponding weight value which has been involved in generating a new candidate solution. Moreover, the sum of elements in $W(o)$ is 1. After forming $W(o)$, new pattern solutions H^{New} are generated by the following expression:

$$\vec{H}_j^{New}(o+1) = \sum_i^{N_{pop}} {}^iw_j(o) \times \vec{H}_i(o) \qquad for \quad j = 1,2\ldots,N_{pop} \quad , \tag{28}$$

$$\overset{\rightarrow}{H_i}(o+1) = \overset{\rightarrow}{H_i}(o) + \overset{\rightarrow New}{H_i}(o+1) \quad for \quad i = 1, 2 \ldots, N_{pop} \quad . \tag{29}$$

Therefore, the new pattern solution has been updated for iteration $o + 1$. Based on the best weight value called "target weight", the weight matrix should be updated, as follows:

$$\overset{\rightarrow Updated}{W_i}(o+1) = \overset{\rightarrow}{W_i}(o) + 2 \times rand \times \left(\overset{\rightarrow \mathrm{Target}}{W}(o) - \overset{\rightarrow}{W_i}(o) \right) \quad for \quad i = 1, 2 \ldots, N_{pop} \quad , \tag{30}$$

where *rand* is a uniformly distributed random number in the range [0, 1].

In ANN, the bias current is introduced, of which the bias operator modifies a certain percentage of the pattern solutions in the new population of pattern solutions $\overset{\rightarrow New}{H_i}(o+1)$ and updated weight matrix $\overset{\rightarrow Updated}{W_i}(o+1)$. The bias operator in the ANN is another way to explore the search space, which prevents the algorithm from premature convergence and modifies individual numbers in the population. In fact, the bias operation acts as noise to the new pattern solutions and the updated weight matrix.

Then new pattern solutions in the population from their current positions in the search space are transferred to new positions to update and generate better quality solutions toward the target solution by the transfer function operator. The transfer function (TF) operation is defined by the following equation:

$$\overset{\rightarrow *}{H_i}(o+1) = TF(\overset{\rightarrow}{H_i}(o+1)) = \overset{\rightarrow}{H_i}(o+1) + 2 \times rand \times \left(\overset{\rightarrow \mathrm{Target}}{H}(o) - \overset{\rightarrow}{H_i}(o+1) \right) \quad for \quad i = 1, 2 \ldots, N_{pop} \quad , \tag{31}$$

where the *i*-th new pattern solution $\overset{\rightarrow}{H_i}(o+1)$ is transferred to the updated position $\overset{\rightarrow *}{H_i}(o+1)$.

In summary, the optimization problem can be solved by the general behavior of ANN, which can be described by

$$\overset{\rightarrow}{H_i}(o + \Delta o) = f\left(\overset{\rightarrow}{H_i}(o), U(o) \right) \quad for \quad i = 1, 2 \ldots N_{pop} \ , \tag{32}$$

where $\overset{\rightarrow}{H_i}(o + \Delta o)$ and $\overset{\rightarrow}{H_i}(o)$ are the next and current locations of pattern solution *i*-th, respectively. $U(o)$ is a population of pattern solutions with updated weights.

6. Results and Discussion

To assess the general applicability and verify the accuracy of the proposed method, optimal time trajectory and time-energy synthesis optimization results were calculated using simulation and actual experiments with the kinematic and dynamic boundary conditions compared with the state-of-the-art algorithms in [3] using Yalmip software. The boundary conditions are listed in Table 2. A circle path of robot motion was applied to test the effectiveness of the proposed method, as shown in Figure 3. The original coordinates values of the robot end were (0.805133 m, −0.328446 m, 0.201903 m), and the radius of the circle was 0.15 m. The orientations of the robot end in Cartesian space were (3.137975 rad, −0.015916 rad, 1.620108 rad).

Table 2. Absolute value limits of joint angles, velocities and accelerations.

Joint *i*	**1**	**2**	**3**	**4**	**5**	**6**
Velocity (rad/s)	3.877	2.623	3.141	3.964	6.311	6.329
Acceleration (rad/s^2)	9.692	7.658	7.853	9.910	15.777	15.822
Torque (Nm)	193.68	286.284	127.011	100.655	15.9296	15.884

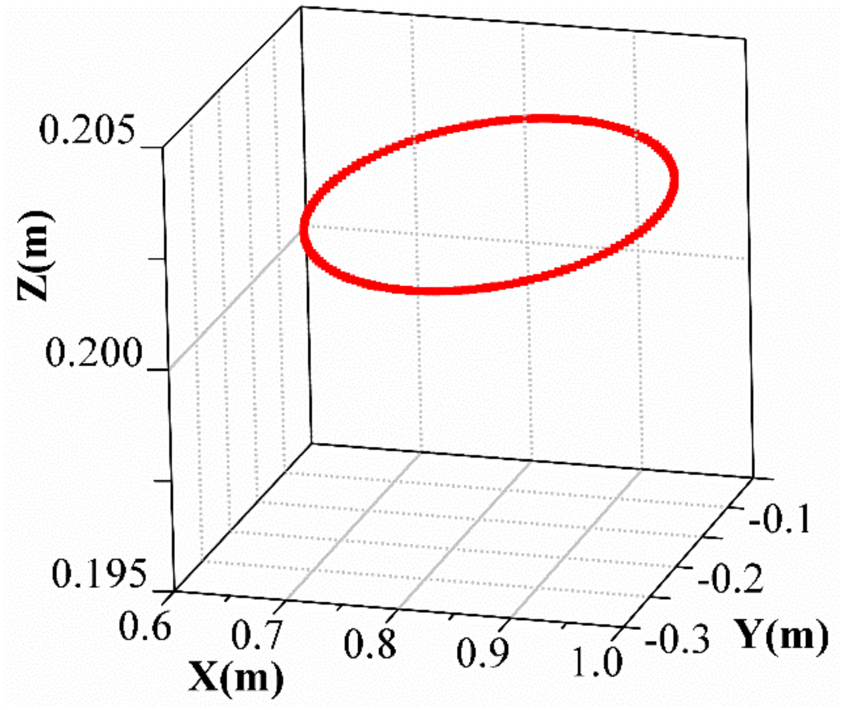

Figure 3. A circle path of robot motion.

In [3], the minimum objective (17) is solved by the conventional optimization method SOCP, which converts the non-convex function $v(s_{j+1}) = \frac{1}{h(s_{j+1})}\left[1+\beta\sum_{i=1}^{6}\frac{\tau_i^2(s_{j+1})}{(\overline{\tau_i})^2}\right]$ into convex form with additional constraint conditions, as follows:

$$\left\| \begin{array}{c} 2 \\ 2\sqrt{\beta}\tau_1(s_{j+1})/\overline{\tau}_1 \\ \vdots \\ 2\sqrt{\beta}\tau_6(s_{j+1})/\overline{\tau}_6 \\ h(s_{j+1}) - v(s_{j+1}) \end{array} \right\| \leq h(s_{j+1}) + v(s_{j+1}). \tag{33}$$

The above fussy conversion of strictly convex functions is avoided by the proposedThe above fussy conversion of strictly convex functions is avoided by the proposed optimization model directly solved by the ANN algorithm in Section 5. Moreover, the second-order polynomial interpolation functions are applied as basic functions to calculate indirect kinematic and dynamic expressions introduced in a discrete optimization model and other constraints in reference [3].

For verification of the accuracy and efficiency of the proposed method, the optimization variable results $h(s_j)$ in objective function solved by SDPT3 optimization toolkit in Yalmip and the ANN algorithm, when the basic functions are the second-order polynomial interpolation functions and the initial and final velocity and acceleration boundaries are neglected as in [3], as shown in Figure 4. We select discrete points when the number of s is 20. As can be seen in Figure 4, optimization variable $h(s_j)$ values at different coordinates solved by the proposed method are similar to that of the commercial software toolkit Yalmip, which indicates the high accuracy of our model. The convergence curve of the optimization objective solved by ANN shows the stable minimum value of the objective function is 1.1454 when the number of iterations is more than 330, which is a little less than the minimum objective value, 1.1484, obtained by Yalmip.

To intuitively display the motion of the robot, the joint angles and joint angular velocities and torques of six joints during the trajectory operation with time-energy optimization weight $\alpha = 0$ are obtained according to the method in [3] and our proposed method with bounded velocity, acceleration, torque limits, and initial and final boundaries, as shown in Figures 5 and 6.

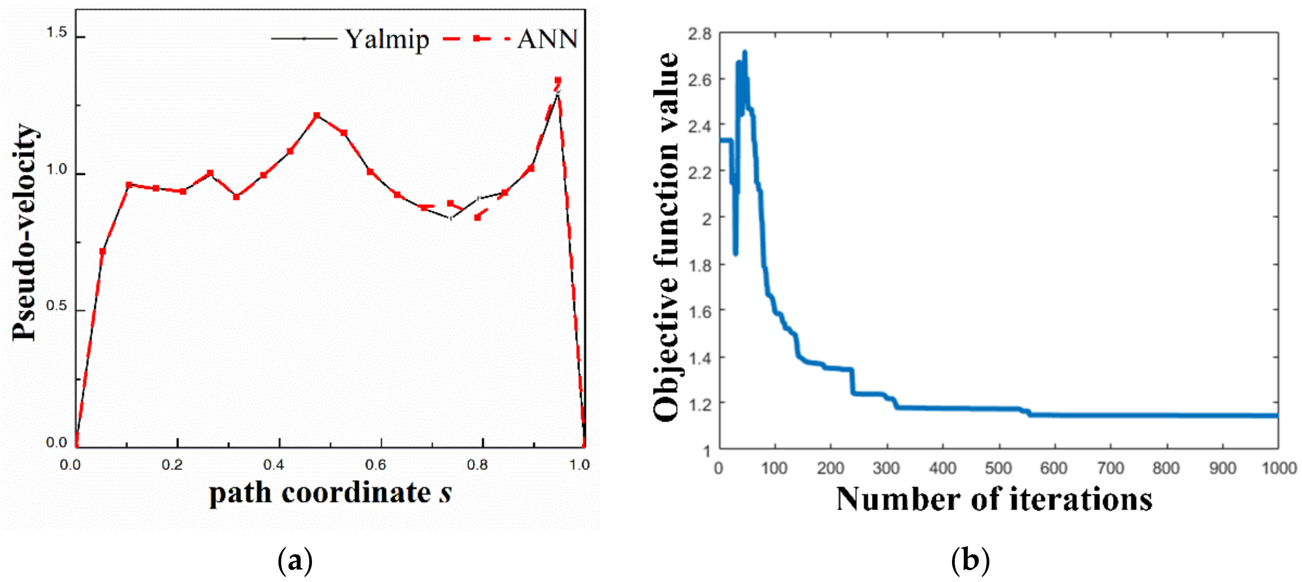

Figure 4. Optimized results for robot joint solved by Yalmip with the method in reference and the proposed method using ANN. (**a**) Optimization variable $h(s_j)$ values at different coordinates; (**b**) Convergence curve of optimization objective with ANN.

The joint angular velocity and acceleration curves start with non-zero values calculated by the method in [3], and accelerations of joints 2, 3 and 5 exceed the limits by 3.277 rad/s^2, 26.674 rad/s^2, 7.620 rad/s^2, respectively. The generated trajectories are hardly used in the actual scene. On the contrary, our result showed a smoother optimization trajectory with initial and final zero velocity and acceleration and all kinematic and dynamic constraints were satisfied within the extreme values.

For verification of the effectiveness of the proposed method, the calculated normalized energy consumption $\sum_{i=1}^{6} \int_0^{t_f} \left(\tau_i^2(s)/\overline{\tau}_i^2\right) dt$ and motion time t_f results obtained by the proposed optimization model are listed in Table 3, when weights β = 0, 0.1, 1 and $10^{0.4}$, of which selected weights β are the same as [3]. In Table 3, the motion time t_f rises and the normalized energy decreases simultaneously when the weight increases, which indicates that the proposed trajectory optimization model can adjust the variable values according to the requirement of high-speed motion or low energy consumption.

The results of the optimization trajectories with different weights β = 0.1 and 1 can be seen in Figures 7 and 8. As can be seen in Figures 6–8, the trajectory acceleration fluctuation at the corresponding singular point decreases gradually with weight increases. This indicates that the design of the objective function is based on the trade-off analysis of time and energy consumption. Therefore, the overall trend of the joint torque curve of the manipulator still tends to be near the safety limit of the rated torque. However, after the energy consumption modeling and weight distribution of the manipulator servo drive control system, the operation trend of the manipulator joint torque curve tends to decrease with increase in the weight. The time near the safety limit of the rated torque gradually decreases, which also means that the energy consumption of the joint servo drive control system has been effectively improved during the corresponding trajectory operation. From the normalized comparison of energy consumption in Table 3 of the manipulator trajectory optimization, it can be seen that modeling the energy consumption index of the robot servo motor and assigning a corresponding weight design plays a significant role in reducing the energy consumption value of the manipulator trajectory operation.

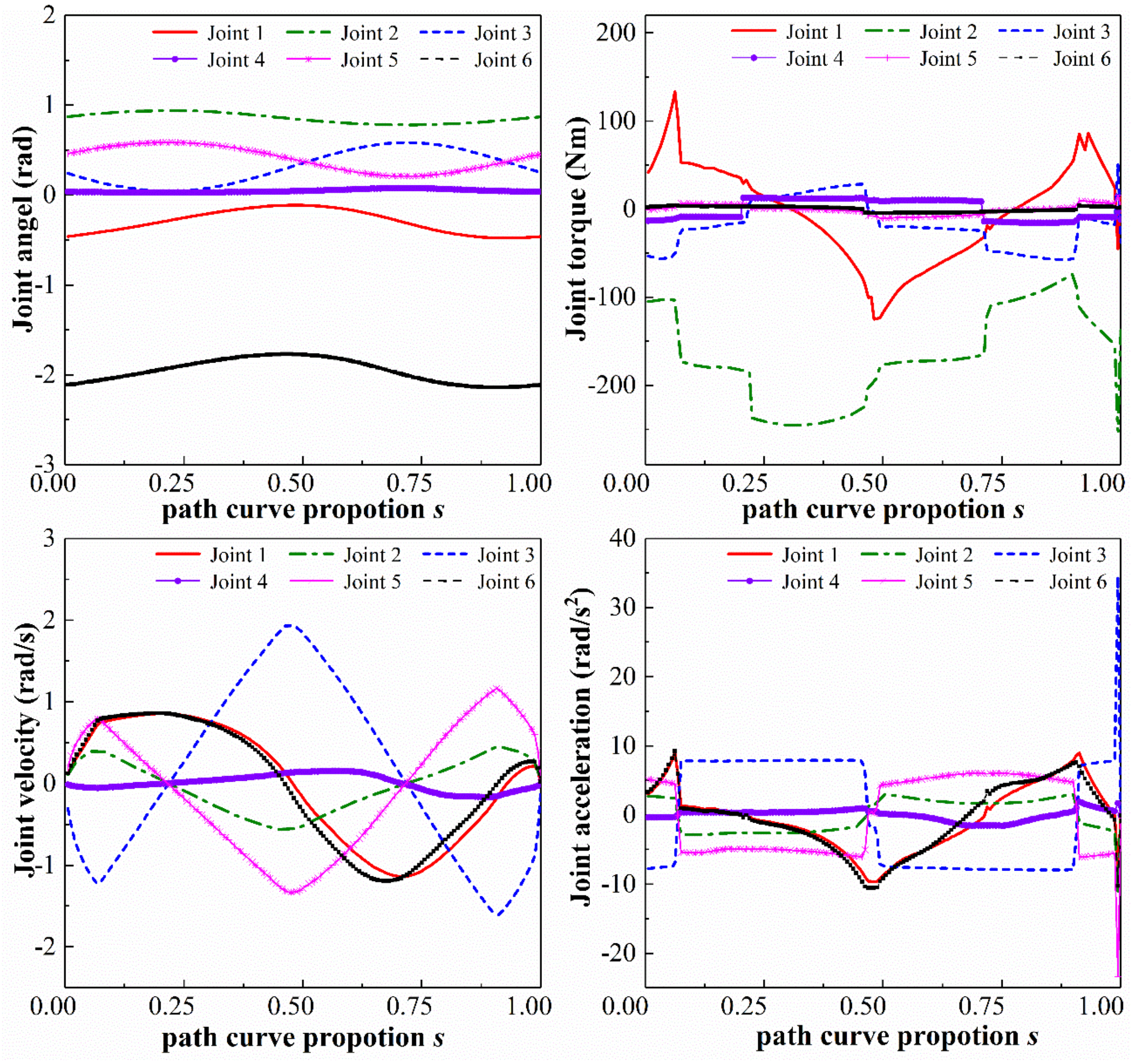

Figure 5. Optimized trajctories for robot joint with the method in reference.

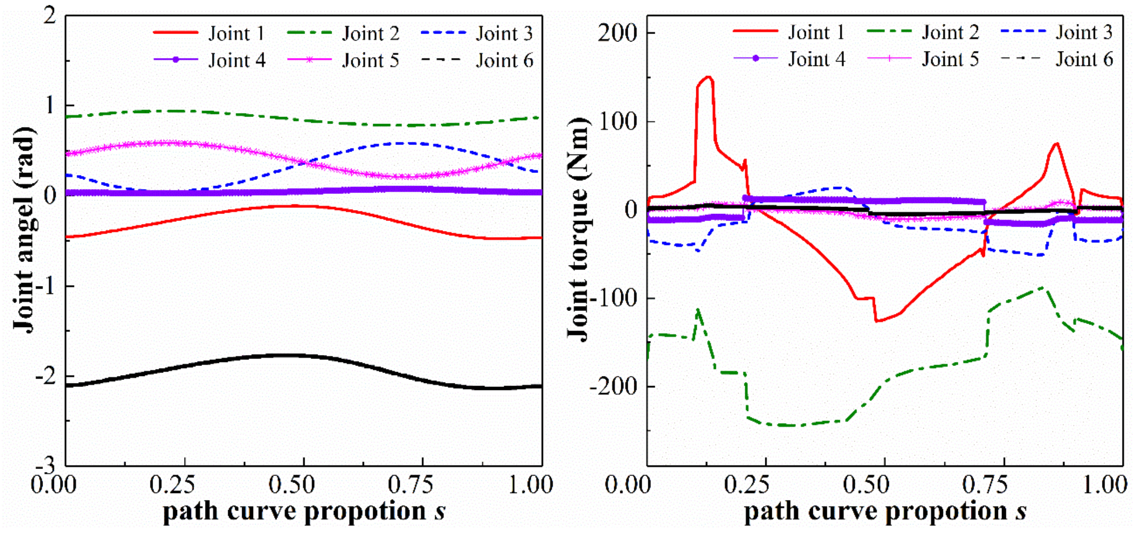

Figure 6. *Cont.*

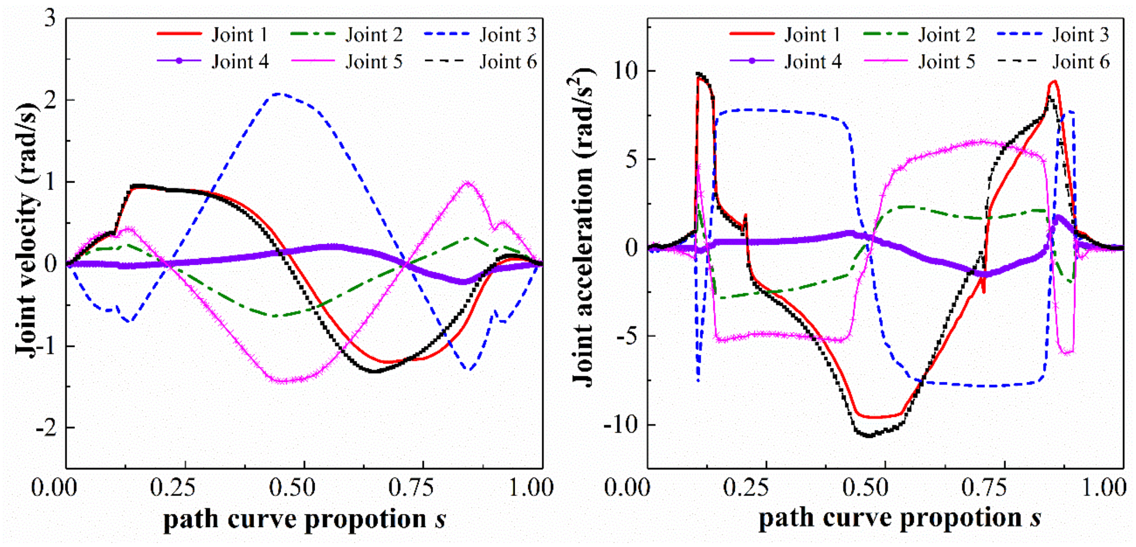

Figure 6. Optimized trajectories for robot joint with the proposed method.

Table 3. Normalized energy consumption and motion time results obtained by the proposed optimization model.

Weight β	Motion Time	Normalized Energy	Maximum Acceleration of Joint 2	Maximum Acceleration of Joint 3	Maximum Acceleration of Joint 5
0	1.8139	0.9998	2.8306	7.8076	6.0125
0.1	1.824	0.9977	2.8291	7.8038	5.9886
1	2.072	0.9641	2.7369	7.7953	5.9441
100.4	3.2915	0.9364	2.6435	7.7955	5.9388

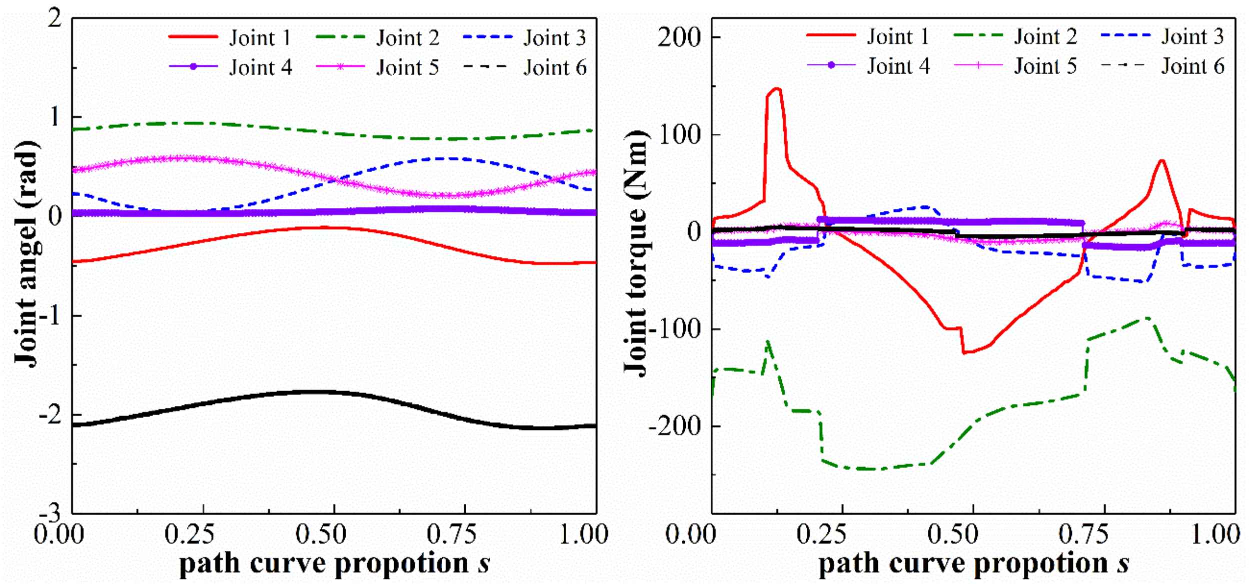

Figure 7. *Cont.*

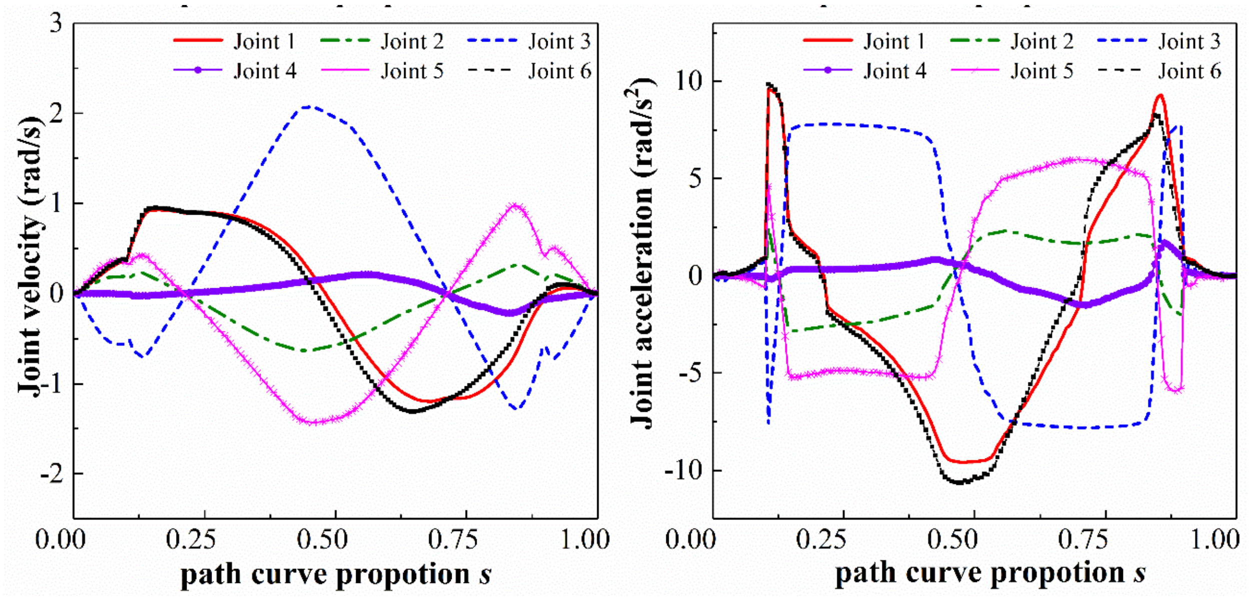

Figure 7. Optimized trajectories for robot joint with the proposed method when α = 0.1.

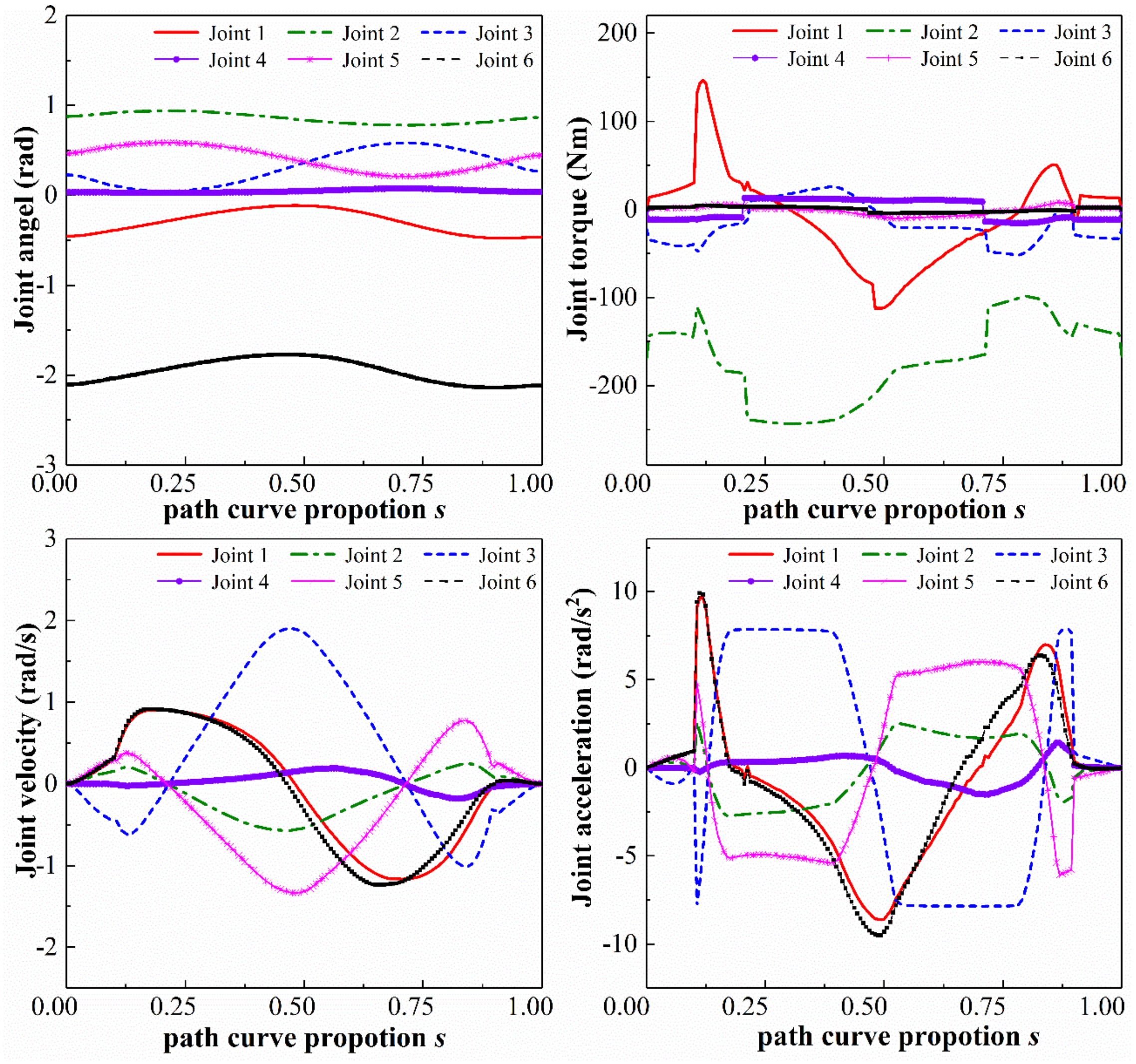

Figure 8. Optimized trajectories for robot joint with the proposed method when α = 1.

To test the actual optimization effect, an experiment of the proposed method was implemented with the robot system, as shown in Figure 9. The path time of the optimal trajectory when $\alpha = 10^{0.4}$ was 3.2915 s which ensured the joint velocity in the range of the velocity set limits. The energy consumption of the robot running optimal trajectory was 0.357 Wh. The experimental sampling data of six joint angles, torques and velocities of time-energy consumption optimal trajectory are shown in Figure 10. The torque values of the servo motor of joints were within the torque performance constraints. The safety range of the accelerations of robot joint was reached at several discrete solutions, and the obtained solutions were used in an actual scene, which demonstrated the effectiveness of the above algorithm and the constraint design.

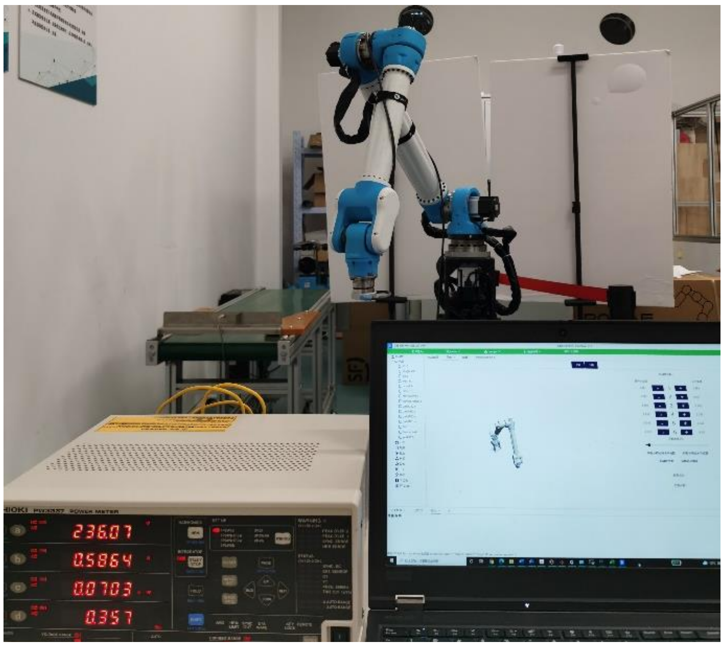

Figure 9. Experiment platform of six-joints robot.

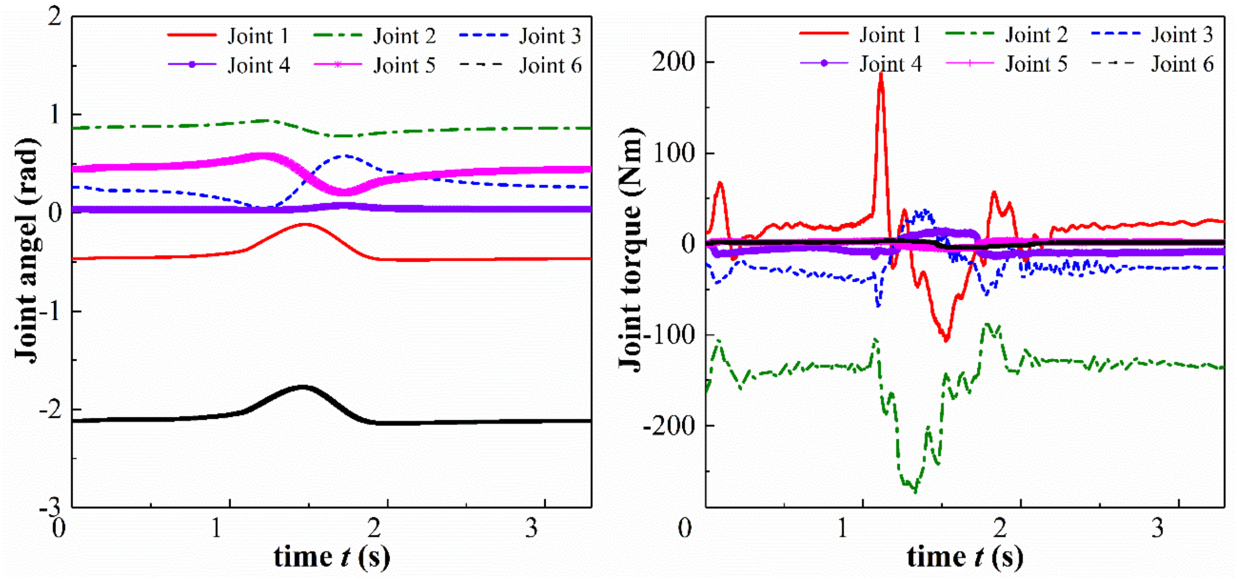

Figure 10. *Cont.*

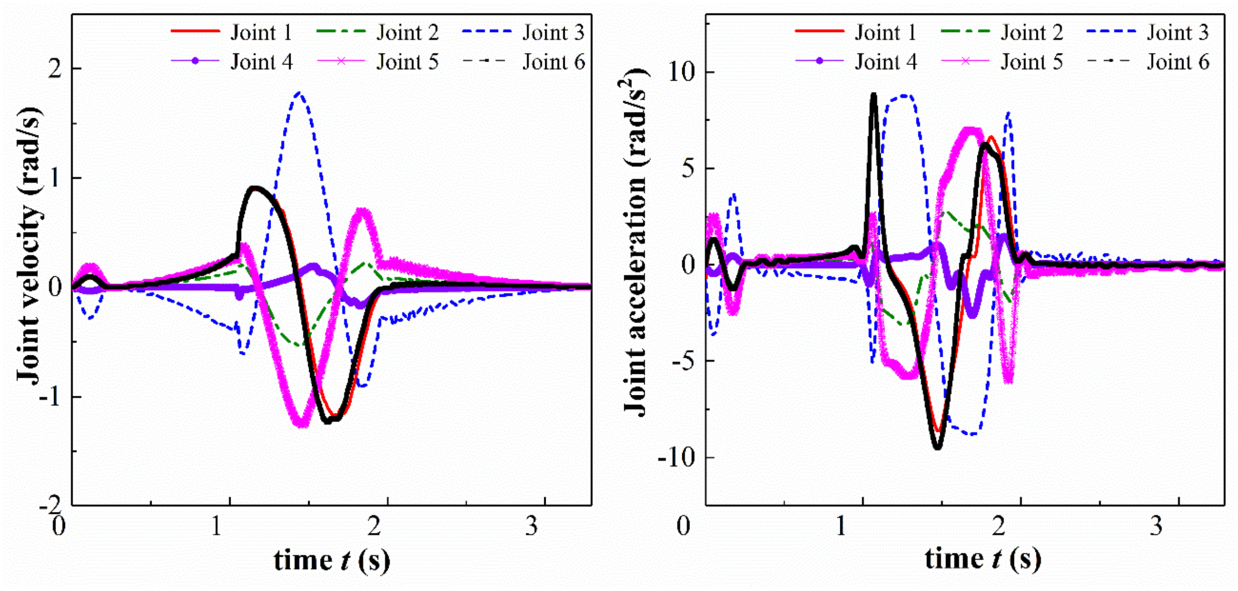

Figure 10. Experiment data of optimal trajectory with $\alpha = 10^{0.4}$.

7. Conclusions

To describe the optimization problem of robot trajectory planning, a time-energy optimization model based on the ANN algorithm was proposed in this paper. The energy consumption model of servo resistance loss was constructed in addition to motion time. For a six degrees of freedom industrial robot, the kinematics and dynamics impact on the coupling constraint condition were considered, including velocity, acceleration and torque. The optimization model was discretized based on the Riemann approximation method. Based on the established kinematic and dynamic model of the robot, a basic discretization model in terms of the generalized path variable mapping was constructed. Forward-substitution interpolation functions were presented as basic functions for the insurance of the initial and final zero-velocity and zero-acceleration of indirect kinematic expressions introduced in the discrete optimization model. Finally, the trajectory optimization parameters and the comprehensive tradeoff time-resistance energy loss index with multi-level performance constraints were solved by a numerical iterative solution strategy based on neural networks. The simulation and actual experiments were implemented with different optimization weights. The proposed method could enhance the acceleration control performance of the solved robot trajectory by reducing accelerations exceeding values of joint 2, 3 and 5 by 3.277 rad/s^2, 26.674 rad/s^2, 7.620 rad/s^2, respectively. Moreover, comparison results between our method and recent optimization methods showed that the improved basic function contributes to the smoothness of the optimization trajectory and guarantees zero velocity and acceleration at the starting and ending points.

Author Contributions: Conceptualization, J.N.; methodology, R.H.; software, B.H., X.Y. and Q.M.; validation, T.R.; formal analysis, Y.G.; writing—original draft preparation, R.H.; writing—review and editing, T.R.; funding acquisition, J.N. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Key Research and Development Program of Zhejiang province, No. 2020C01026.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

Nomenclature

α_{i-1}	Link twist	${}^{NE}\tau_i$	Joint torques derived by the iterative Newton–Euler formulation
a_{i-1}	Link length	I_{ai}	Inertia moment for rotor and gears of actuator i
d_i	Link offset	f_{vi}, f_{ci}	Viscous and Coulomb friction coefficients
θ_i	Joint angle	${}^{C}\mathbf{I}_i$	Inertia tensor related to center of mass
m_i	Link mass	${}^{A}\mathbf{I}_i$	Inertia tensor related to the origin of joint coordinate
τ_i	Joint torque	I_{xxi}, I_{yyi}, I_{zzi}, I_{xyi}, I_{xzi}, I_{yzi}	Inertia moment components of link i
${}^{f}\tau_i$	Friction torque	$\boldsymbol{\omega}(q,\dot{q},\ddot{q})$	Regressor matrix
${}^{r}\tau_i$	Inertia torque	$\boldsymbol{\omega}_{min}(q,\dot{q},\ddot{q})$	Subset of the independent columns of $\boldsymbol{\omega}(q,\dot{q},\ddot{q})$
q	Joint position vector	$\mathbf{P}_{Ci} = [x_{Ci}, y_{Ci}, z_{Ci}]^{\mathrm{T}}$	Coordinates of center of mass
$\dot{q}_i$	Velocity	Ω_i	Standard parameters for joint i
$\ddot{q}_i$	Acceleration	$\boldsymbol{\Omega}_{\min}$	Minimal set vector of identifiable parameters
Γ	Torques vector	$M(q)$	Mass or inertia matrix
t	Time	$C(q,\dot{q})$	Coriolis and centrifugal vector
t_f	Total motion time	${}^{i}e$	Resistance energy consumption of the servo circuit in robot joint
$G(q)$	Gravity	β	Weight coefficient of energy consumption
${}^{i}I_{servo}$	Electric current	s	Trip proportion assigned to path curve
${}^{i}r$	Resistance	s_j	The j-th point proportion of the divided path curve
b	Polynomial coefficients	$u(s)$	Second order derivative of s
Ψ	Observation matrix	$h(s)$	First order derivative of s
N	Point number	$B_k(S)$	Cubic Bezier curve function
rx,ry,rz	Pose orientation vector	l	Half of the added continuous selected points number
v(sj)	Discrete optimization objective	d	The nearest integer less than or equal to 0.1(N−1)
${}^{i}F$	Cost functions	H	Population of pattern solution
o	Iteration index	$W(o)$	Population of weight
TF	Transfer function	*rand*	Random number

Appendix A

The minimum identification parameter set is shown in Table A1.

Table A1. Elements of the calculated minimum identification set $\boldsymbol{\Omega}_{\min}$.

$\boldsymbol{\Omega}_{\min}(1)$ 14.2652	$\boldsymbol{\Omega}_{\min}(2)$ 7.7347	$\boldsymbol{\Omega}_{\min}(3)$ 13.4288	$\boldsymbol{\Omega}_{\min}(4)$ 19.2053	$\boldsymbol{\Omega}_{\min}(5)$ 0.3683	$\boldsymbol{\Omega}_{\min}(6)$ −13.2957	$\boldsymbol{\Omega}_{\min}(7)$ 20.0815
$\boldsymbol{\Omega}_{\min}(8)$ −0.6178	$\boldsymbol{\Omega}_{\min}(9)$ 0.0673	$\boldsymbol{\Omega}_{\min}(10)$ 1.0250	$\boldsymbol{\Omega}_{\min}(11)$ 33.5249	$\boldsymbol{\Omega}_{\min}(12)$ 26.4618	$\boldsymbol{\Omega}_{\min}(13)$ −0.0689	$\boldsymbol{\Omega}_{\min}(14)$ 4.9279
$\boldsymbol{\Omega}_{\min}(15)$ 4.0606	$\boldsymbol{\Omega}_{\min}(16)$ 4.6718	$\boldsymbol{\Omega}_{\min}(17)$ 0.3442	$\boldsymbol{\Omega}_{\min}(18)$ 1.4541	$\boldsymbol{\Omega}_{\min}(19)$ −0.8636	$\boldsymbol{\Omega}_{\min}(20)$ −0.4512	$\boldsymbol{\Omega}_{\min}(21)$ 10.7137
$\boldsymbol{\Omega}_{\min}(22)$ 12.0429	$\boldsymbol{\Omega}_{\min}(23)$ 0.0897	$\boldsymbol{\Omega}_{\min}(24)$ 0.0388	$\boldsymbol{\Omega}_{\min}(25)$ −0.0721	$\boldsymbol{\Omega}_{\min}(26)$ 0.1035	$\boldsymbol{\Omega}_{\min}(27)$ −0.0151	$\boldsymbol{\Omega}_{\min}(28)$ 0.7826
$\boldsymbol{\Omega}_{\min}(29)$ −0.4236	$\boldsymbol{\Omega}_{\min}(30)$ −0.2792	$\boldsymbol{\Omega}_{\min}(31)$ 8.4568	$\boldsymbol{\Omega}_{\min}(32)$ 11.0075	$\boldsymbol{\Omega}_{\min}(33)$ 0.0457	$\boldsymbol{\Omega}_{\min}(34)$ 0.1336	$\boldsymbol{\Omega}_{\min}(35)$ −0.0042
$\boldsymbol{\Omega}_{\min}(36)$ 0.3912	$\boldsymbol{\Omega}_{\min}(37)$ 0.0167	$\boldsymbol{\Omega}_{\min}(38)$ 0.1285	$\boldsymbol{\Omega}_{\min}(39)$ −0.3073	$\boldsymbol{\Omega}_{\min}(40)$ −0.6151	$\boldsymbol{\Omega}_{\min}(41)$ 1.4333	$\boldsymbol{\Omega}_{\min}(42)$ 1.9285
$\boldsymbol{\Omega}_{\min}(43)$ −0.0537	$\boldsymbol{\Omega}_{\min}(44)$ −0.0283	$\boldsymbol{\Omega}_{\min}(45)$ 0.0152	$\boldsymbol{\Omega}_{\min}(46)$ −0.1652	$\boldsymbol{\Omega}_{\min}(47)$ 0.0156	$\boldsymbol{\Omega}_{\min}(48)$ −0.0056	$\boldsymbol{\Omega}_{\min}(49)$ 0.0453
$\boldsymbol{\Omega}_{\min}(50)$ 0.1991	$\boldsymbol{\Omega}_{\min}(51)$ 1.009	$\boldsymbol{\Omega}_{\min}(52)$ 2.0133				

References

1. Verscheure, D.; Demeulenaere, B.; Swevers, J.; De Schutter, J.; Diehl, M. Time-Energy Optimal Path Tracking for Robots: A Numerically Efficient Optimization Approach. In Proceedings of the 2008 10th IEEE International Workshop on Advanced Motion Control, Trento, Italy, 26–28 March 2008; pp. 727–732.
2. Riazi, S.; Wigström, O.; Bengtsson, K.; Lennartson, B. Energy and Peak Power Optimization of Time-Bounded Robot Trajectories. *IEEE Trans. Autom. Sci. Eng.* **2017**, *14*, 646–657. [CrossRef]
3. Verscheure, D.; Demeulenaere, B.; Swevers, J.; De Schutter, J.; Diehl, M. Time-Optimal Path Tracking for Robots: A Convex Optimization Approach. *IEEE Trans. Autom. Control.* **2009**, *54*, 2318–2327. [CrossRef]
4. Steinhauser, A.; Swevers, J. An Efficient Iterative Learning Approach to Time-Optimal Path Tracking for Industrial Robots. *IEEE Trans. Ind. Inform.* **2018**, *14*, 5200–5207. [CrossRef]
5. Chen, D.; Li, S.; Li, W.; Wu, Q. A Multi-Level Simultaneous Minimization Scheme Applied to Jerk-Bounded Redundant Robot Manipulators. *IEEE Trans. Autom. Sci. Eng.* **2020**, *17*, 463–474. [CrossRef]
6. Zhang, Y.; Li, S.; Gui, J.; Luo, X. Velocity-Level Control With Compliance to Acceleration-Level Constraints: A Novel Scheme for Manipulator Redundancy Resolution. *IEEE Trans. Ind. Inform.* **2018**, *14*, 921–930. [CrossRef]
7. Nagy, Á.; Vajk, I. Sequential Time-Optimal Path-Tracking Algorithm for Robots. *IEEE Trans. Robot.* **2019**, *35*, 1253–1259. [CrossRef]
8. Yuille, A.L.; Rangarajan, A. The Concave-Convex Procedure. *Neural Comput.* **2003**, *15*, 915–936. [CrossRef] [PubMed]
9. Verscheure, D.; Diehl, M.; De Schutter, J.; Swevers, J. On-Line Time-Optimal Path Tracking for Robots. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009; pp. 599–605.
10. Feng, Y.; Ji, Z.; Gao, Y.; Zheng, H.; Tan, J. An Energy-Saving Optimization Method for Cyclic Pick-and-Place Tasks Based on Flexible Joint Configurations. *Robot. Comput. -Integr. Manuf.* **2021**, *67*, 102037. [CrossRef]
11. Palleschi, A.; Garabini, M.; Caporale, D.; Pallottino, L. Time-Optimal Path Tracking for Jerk Controlled Robots. *IEEE Robot. Autom. Lett.* **2019**, *4*, 3932–3939. [CrossRef]
12. Ramezani, N.; Williams, M.-A. Smooth Robot Motion with an Optimal Redundancy Resolution for PR2 Robot Based on an Analytic Inverse Kinematic Solution. In Proceedings of the 2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids), Seoul, Korea, 3–5 November 2015; pp. 338–345.
13. He, T.; Zhang, Y.; Sun, F.; Shi, X. Immune Optimization Based Multi-Objective Six-DOF Trajectory Planning for Industrial Robot Manipulators. In Proceedings of the 2016 12th World Congress on Intelligent Control and Automation (WCICA), Guilin, China, 12–15 June 2016; pp. 2945–2950.
14. Abu-Dakka, F.J.; Assad, I.F.; Alkhdour, R.M.; Abderahim, M. Statistical Evaluation of an Evolutionary Algorithm for Minimum Time Trajectory Planning Problem for Industrial Robots. *Int. J. Adv. Manuf. Technol.* **2017**, *89*, 389–406. [CrossRef]
15. Tang, X.; Shi, C.; Deng, T.; Wu, Z.; Yang, L. Parallel random matrix particle swarm optimization scheduling algorithms with budget constraints on cloud computing systems. *Appl. Soft Comput.* **2021**, *113*, 107914. [CrossRef]
16. Villarrubia, G.; De Paz, J.F.; Chamoso, P.; la Prieta, F.D. Artificial Neural Networks Used in Optimization Problems. *Neurocomputing* **2018**, *272*, 10–16. [CrossRef]
17. Sadollah, A.; Sayyaadi, H.; Yadav, A. A Dynamic Metaheuristic Optimization Model Inspired by Biological Nervous Systems: Neural Network Algorithm. *Appl. Soft Comput.* **2018**, *71*, 747–782. [CrossRef]
18. Shi, X.; Fang, H.; Pi, G.; Xu, X.; Chen, H. Time-Energy-Jerk Dynamic Optimal Trajectory Planning for Manipulators Based on Quintic NURBS. In Proceedings of the 2018 3rd International Conference on Robotics and Automation Engineering (ICRAE), Guangzhou, China, 17–19 November 2018; pp. 44–49.
19. Craig, J. *Introduction to Robotics: Mechanics and Control*, 3rd ed.; Pearson Education: Hoboken, NJ, USA, 2005.
20. Khalil, W.; Gautier, M.; Lemoine, P. Identification of the Payload Inertial Parameters of Industrial Manipulators. In Proceedings of the 2007 IEEE International Conference on Robotics and Automation, Rome, Italy, 10–14 April 2007; pp. 4943–4948.
21. Jin, J.; Gans, N. Parameter Identification for Industrial Robots with a Fast and Robust Trajectory Design Approach. *Robot. Comput. Integr. Manuf.* **2015**, *31*, 21–29. [CrossRef]
22. Gautier, M. Numerical Calculation of the Base Inertial Parameters of Robots. *J. Robot. Syst.* **1991**, *8*, 485–506. [CrossRef]
23. Dmitruk, A.V.; Kaganovich, A.M. The Hybrid Maximum Principle Is a Consequence of Pontryagin Maximum Principle. *Syst. Control. Lett.* **2008**, *57*, 964–970. [CrossRef]
24. Sciavicco, L.; Siciliano, B. *Dynamics. Modelling and Control of Robot Manipulators*; Sciavicco, L., Siciliano, B., Eds.; Springer: London, UK, 2000; pp. 131–183, ISBN 978-1-4471-0449-0.