Robotica: (2020) volume 38, pp. 1495–1512. © Cambridge University Press 2019
doi:10.1017/S0263574719001590

Inverse Kinematics of Redundant Manipulators Formulated as Quadratic Programming Optimization Problem Solved Using Recurrent Neural Networks: A Review

Ahmed A. Hassan*, Mohamed El-Habrouk and Samir Deghedie

Department of Electrical Engineering, Faculty of Engineering, Alexandria University, Alexandria, Egypt

(Accepted October 19, 2019. First published online: November 25, 2019)

SUMMARY
The Inverse Kinematics (IK) problem of manipulators can be divided into two distinct steps: (1) *Problem formulation,* where the problem is developed into a form which can then be solved using various methods. (2) *Problem solution,* where the IK problem is actually solved by producing the values of different joint space variables (joint angles, joint velocities or joint accelerations). The main focus of this paper is concentrated on the discussion of the IK problem of redundant manipulators, formulated as a quadratic programming optimization problem solved by different kinds of recurrent neural networks.

KEYWORDS: Inverse kinematics; Redundant manipulators; Quadratic programming; Recurrent neural networks; Bi-criteria kinematic control.

1. Introduction

Inverse kinematics (IK): Position control of manipulators entails solving the IK problem so that, for desired Cartesian coordinates of the end effector, the corresponding joint angles can be computed and used as set points to individual actuator position control.[1] Solving the IK problem for manipulators is a challenging task. The complexity of the problem is caused by the manipulator's geometry and the nonlinear trigonometric equations that describe the mapping between the Cartesian space and the joint space.[2–5]

Traditional methods for solving inverse kinematics: Three traditional methods are used to solve the IK problem: (1) geometric.[6–9] This approach uses trigonometric calculations.[10] This approach is simpler for manipulators with smaller number of degrees of freedom (DOF). For manipulators with larger number of DOF, one possible solution is to split the problem into smaller parts and make calculations with geometrical approach,[11] (2) algebraic.[12–22] Algebraic methods are used to obtain closed-form solutions. The kinematic equations of the manipulator are transformed into higher degree polynomial and then all the roots of the polynomial are determined, and (3) iterative.[23,24] These methods are usually used on manipulators that may not have closed-form IK equations.[25]

* Corresponding author. E-mail: a.hassan0305@gmail.com

Issues pertaining to traditional methods for solving inverse kinematics

1. Prohibitive computational cost because of the high complexity of the geometric structure of manipulators[26,27] and the mathematical structure of the formulation.[28]
2. If the number of DOF increases, traditional methods become complex.[29]
3. The involvement of complex higher order polynomials retarded the wide application of algebraic methods.
4. Robots work in real environments that cannot be modeled accurately using mathematical expressions.[28]
5. Numerical methods are influenced by initial value selection, suffer from limited convergence and unavailability of multiple solutions. Moreover, they do not provide solution if the Jacobian matrix is singular at a particular pose of the manipulator.
6. Numerical methods cannot be used to develop a generalized strategy to obtain IK solutions independent of manipulator's geometry and the number of DOF.[30]

Neural network-based solution of inverse kinematics: Many papers were published about the neural network-based IK solution for manipulators.[2,26,27,31–47] Interest in neural network research was generated to reduce the computational complexity of motion planning and control of manipulators.[48–59] Neural networks are useful for learning the IK of manipulators lacking a well-defined model.[60] Solving the IK using artificial neural networks is useful where less computation times are needed, such as in real-time adaptive manipulator control.[33,61–63] The IK solution using neural networks comes under the class of iterative methods.[40]

Redundant manipulators: Kinematically redundant manipulators are those having more DOF than required to perform a given end-effector primary task.[64–66] A fundamental issue in controlling redundant manipulators is the redundancy resolution problem[65,67,68] (IK problem).[65,68] Since redundant manipulators have more DOF, multiple solutions exist to the IK problem.[69,70] Hence, the redundancy complicates the manipulator's control problem.[68,71–76] By resolving the redundancy properly, it is possible to achieve useful subtasks such as repetitive motion,[46,67] obstacle avoidance,[77–82] joint limits avoidance,[82–85] singularity avoidance,[80,82,86,87] fault tolerance[88,89] and optimization of various performance criteria[90–93] while conducting the end-effector motion task. One example of the performance criteria to be optimized is the norm of joint torques,[94–96] which aims at making a more effective utilization of input power from actuators.

Conventional method for redundancy resolution: The conventional solution of the redundancy problem is obtained by the pseudo-inverse-based formulation (one minimum-norm particular solution plus a homogeneous solution).[71–76,97] However, in this method, joint angle, joint velocity and joint acceleration limits are usually not considered, as these inequality constraints are relatively not easy to be handled by this type of formulation.[98] If $\mathbf{J}$ is rank-deficient, the pseudo-inverse-based methods may encounter the singularity problem. When the end effector traces a closed path, the joint angles may not return back to their initial ones after completing the end-effector task. This is called joint angle drift, or repeatability problem.[67,99] This may induce a problem that the manipulator's behavior is hard to predict. The manipulator's configuration can be readjusted by moving joints without moving the end effector (manipulator self-motion); however, this is not efficient.[67,100] The joint angle drift problem was raised in ref. [91]. A pseudo-inverse-based solution may not be repeatable.[72,75,91] In ref. [101], the necessary condition of repeatability was analyzed, and this condition reveals why the pseudo-inverse-based method is not repetitive in general. A sufficient condition for the drift-free control by the pseudo-inverse-based method was given in ref. [102]. Repetitive control based on the pseudo-inverse method is the exception, rather than the general rule.[103] The extended Jacobian method was presented to solve the nonrepetitive problem in refs. [104, 105]. This method is effective for the repetitive motion planning (RMP) of redundant manipulators. However, singularities may occur at the boundary of conservative regions.[106] The extended Jacobian method is difficult to handle inequality-type constraints such as joint limits. Moreover, the extended Jacobian method usually includes matrix inversion which results in higher computational cost.

The IK problem of manipulators can be divided into two distinct steps: (1) *Problem formulation,* where the problem is developed into a form which can then be solved using various methods. (2) *Problem solution,* where the values of different joint space variables (joint angles, joint velocities

https://doi.org/10.1017/S0263574719001590 Published online by Cambridge University Press

or joint accelerations) are produced. For example, the problem can be formulated as a set of kinematic equations which are reformulated as a higher degree polynomial. Algebraic methods are then used to obtain closed-form solutions by determining the roots of the polynomial. The redundancy problem can be resolved in a more favorable manner via online optimization techniques.[65,67,107–109] These schemes can be reformulated as a quadratic programming (QP) subject to equality, inequality and bound constraints. Such schemes can be solved using many types of recurrent neural networks (RNNs).[110,111] In 1980s, Hopfield and Tank proposed their neural network designed for solving optimization problems.[112] Since then, RNNs are thought to be a powerful alternative to real-time optimization.[69] Unlike feedforward neural networks, most RNNs do not need off-line supervised learning and thus are more suitable for real-time control of redundant manipulators in uncertain environments.[113] In this paper, the main focus is concentrated on the discussion of the IK problem of redundant manipulators formulated as a QP optimization problem solved by different kinds of RNNs.

This paper is organized as follows: Section 2 lists the basic forward and IK equations. Section 3 presents example problem formulations including pseudo-inverse-type formulation. In Section 4, important RNNs used as QP solvers are briefly reviewed. In Section 5, the steps involved in the redundancy resolution problem formulated as a QP problem and solved using different types of RNNs are discussed. In Section 6, the detailed steps of the redundancy resolution problem are described by the aid of example formulation and RNN solvers. Section 7 discusses the results presented in some of the important references. Section 8 concludes the paper.

2. Preliminaries

The forward kinematics equation relating the end-effector position and orientation vector $\mathbf{r} \in \mathbf{R}^m$ in Cartesian space and the joint angle vector $\boldsymbol{\theta} \in \mathbf{R}^n$ for redundant manipulators is described by:[67,99,100,110,111,113,114]

$$\mathbf{r} = f(\boldsymbol{\theta}) \tag{1}$$

The IK problem is to find the joint variables given the desired position and orientation of the end effector through the inverse mapping of (1):[113]

$$\boldsymbol{\theta} = f^{-1}(\mathbf{r}) \tag{2}$$

The direct way to solve (2) is to derive a closed-form solution from (1). Unfortunately, obtaining a closed-form solution is difficult for most manipulators due to the nonlinearity of $f\left(.\right)$. Making use of the linear relation between joint velocity $\dot{\boldsymbol{\theta}}$ and Cartesian velocity $\dot{\mathbf{r}}$ is a common indirect approach to the IK problem.[113] Differentiating (1) w.r.t. time, the relation between $\dot{\mathbf{r}}$ and $\dot{\boldsymbol{\theta}}$ is obtained:[67,99,100,110,111,113,114]

$$\dot{\mathbf{r}} = \mathbf{J}\dot{\boldsymbol{\theta}}, \tag{3}$$

where $\mathbf{J}$ is the Jacobian matrix $\left(\mathbf{J} = \partial f / \partial \boldsymbol{\theta} \in \mathbf{R}^{m \times n}\right)$. This approach begins with the desired velocity of the end effector $\dot{\mathbf{r}}_d$ (based on a planned trajectory and required completion time), then the joint velocity vector $\dot{\boldsymbol{\theta}}$ is computed. By integration of $\dot{\boldsymbol{\theta}}$, the corresponding joint position vector $\boldsymbol{\theta}$ is obtained, which is then used to control the manipulator.[113] Differentiating (3) w.r.t. time yields the relation between the joint acceleration $\ddot{\boldsymbol{\theta}}$ and Cartesian acceleration $\ddot{\mathbf{r}}$:[110,111,114]

$$\mathbf{J}\ddot{\boldsymbol{\theta}} = \ddot{\mathbf{r}} - \dot{\mathbf{J}}\dot{\boldsymbol{\theta}} \tag{4}$$

From the view point of IK (i.e., to solve $\boldsymbol{\theta}$, $\dot{\boldsymbol{\theta}}$ and/or $\ddot{\boldsymbol{\theta}}$ for given $\mathbf{r}$, $\dot{\mathbf{r}}$ and/or $\ddot{\mathbf{r}}$, (1), (3) and (4) are all underdetermined and thus admit infinite solutions.

3. Problem Formulation

3.1. Conventional formulation

The IK problem can be formulated as the pseudo-inverse formulation.[75,115–118] The pseudo-inverse-type formulation at the joint velocity level can be:[52,67,100,110]

$$\dot{\boldsymbol{\theta}} = \mathbf{J}^{+}\dot{\mathbf{r}}_d + \left(\mathbf{I} - \mathbf{J}^{+}\mathbf{J}\right)\mathbf{z} \tag{5}$$

https://doi.org/10.1017/S0263574719001590 Published online by Cambridge University Press

where $\mathbf{J}^{+}$ is the pseudo-inverse of $\mathbf{J}$, and $\mathbf{z}$ is an arbitrary vector selected by using some optimization criteria, for example, singularity avoidance, obstacle avoidance and/or task priority control. $\mathbf{J}^{+}\dot{\mathbf{r}}$ is the minimum-norm particular solution and $(\mathbf{I}-\mathbf{J}^{+}\mathbf{J})\mathbf{z}$ is the homogeneous solution.[67] The method in (5) entails the computation of time-varying pseudo-inverse $\mathbf{J}^{+}$.

3.2. Bi-criteria kinematic control

The minimum two-norm solution of joint velocity vector minimizes the sum of squared joint velocities, which does not necessarily minimize the magnitudes of individual joint velocity. It is used as the optimization criterion in many robotic applications, more because of its mathematical tractability than physical desirability.[91] The minimum ∞-norm solution (also called minimum effort or minimum amplitude solution) of joint velocity vector explicitly minimizes the largest component of the joint velocity vector in magnitude and is consistent with the physical limits. Moreover, the minimization of ∞-norm enables better monitoring and control of the magnitude of individual joint velocities.[93,119–121] However, the minimum ∞-norm solutions may encounter a discontinuity problem.[122] This problem exists because of the non-uniqueness possibility. In other words, if the manipulator trajectory orients the solution space so that it is parallel to a hypercube face, the solution may jump from one edge to the other before continuing smoothly on its way.[119] To remedy the discontinuity problem, a balancing scheme is presented in ref. [119] that calculates the minimum ∞-norm and Euclidean-norm solutions separately, and then incorporates the two weighted solutions as the final solution. Compared to the minimum-norm solution, the balancing scheme may at least double the computational time, which may hinder online sensor-based robotic applications. Neural networks are efficient alternatives for real-time solution to such a balanced IK problem.[122] In ref. [122], the following bi-criteria kinematic control problem was considered to avoid discontinuities in minimum effort solution:

$$\text{Minimize } \frac{1}{2}\left[\alpha\left\|\dot{\boldsymbol{\theta}}\right\|_2^2+(1-\alpha)\left\|\dot{\boldsymbol{\theta}}\right\|_\infty^2\right] \qquad \alpha\in(0,\ 1) \tag{6}$$

$$\text{Subject to } \mathbf{J}\dot{\boldsymbol{\theta}}=\dot{\mathbf{r}} \tag{7}$$

$$\dot{\boldsymbol{\theta}}^{-}\le\dot{\boldsymbol{\theta}}\le\dot{\boldsymbol{\theta}}^{+} \tag{8}$$

$$\boldsymbol{\theta}^{-}\le\boldsymbol{\theta}\le\boldsymbol{\theta}^{+} \tag{9}$$

The parameter α is used to diminish the discontinuity, while keeping small the maximal magnitude of minimum effort solution. As $\alpha\to 0$, the bi-criteria solution approaches the infinity-norm solution. As $\alpha\to 1$, the bi-criteria solution approaches the standard two-norm solution. The limited joint range can be formulated in terms of $\dot{\boldsymbol{\theta}}$ using variable bounds. Joint angle limits and joint velocity limits were then combined into a bound constraint. The bi-criteria kinematic control problem (6)–(9) was then expressed as a QP.[122] A dual neural network (DNN) was designed using the Karush–Kuhn–Tucker (KKT) condition and the projection operator for optimal bi-criteria kinematic control of redundant manipulators. Comparison between the minimum effort, bi-criteria and minimum power kinematic control schemes showed that the maximal amplitude and power consumption of bi-criteria solutions are usually between those of the minimum effort and the minimum power solutions. Compared to the minimum Euclidean-norm, the bi-criteria solution always has a smaller maximal magnitude of $\dot{\boldsymbol{\theta}}$

3.3. Drift-free inverse kinematics

Owing to the local nature of general redundancy resolution schemes where only the current values of joint variables are known, minimization of joint displacement between the current and initial states was investigated in refs. [69, 98, 113, 123] to obtain cyclic motion. QP-based RMP schemes could avoid the kinematic singularity problem owing to its inverse-free processing manner (no need to invert $\mathbf{J}$).[100] Three RMP schemes can be used for redundancy resolution:

- The Velocity-Level Repetitive Motion Planning (VRMP) scheme.
  The RMP scheme can be formulated at the joint velocity level.[67,69,98,124,125]

https://doi.org/10.1017/S0263574719001590 Published online by Cambridge University Press

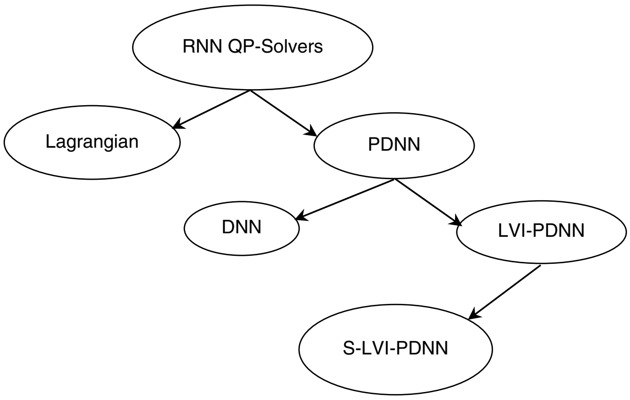

Fig. 1. RNN QP solvers.

- The acceleration-level repetitive motion planning (ARMP) scheme.
  The VRMP scheme may not be applicable to some manipulators which are controlled by joint acceleration or the joint torque. In refs. [110, 111], an ARMP scheme was derived via Zhang's neural-dynamic method.[65] The joint angle, joint velocity and joint acceleration limits are considered. The proposed ARMP scheme is repetitive because it utilizes the drift-free criterion. The ARMP scheme is reformulated as a QP problem in terms of joint acceleration $\ddot{\boldsymbol{\theta}}$
- The jerk-level repetitive motion planning (JRMP) scheme.
  In ref. [126], a cyclic motion planning scheme has been developed to remedy the joint drift phenomenon of redundant manipulators constrained by joint physical limits. The scheme in this paper is resolved at jerk level. The cyclic motion criterion and the avoidance of joint physical limits (joint angle, joint velocity, joint acceleration and joint jerk limits) are considered into the JRMP. The jerk-level scheme is reformulated as a dynamical quadratic program which can be solved by a neural network or a suitable numerical algorithm.

Compared with the extended Jacobian method, the RMP schemes can handle inequality-type constraints. Moreover, these schemes can achieve inversion-free purpose by reformulating them into the QP problem which is solved by RNNs.[111]

4. RNN QP Solvers

Figure 1 shows the functional relationship between different types of RNNs used to solve the IK problem formulated as a QP problem. The primal-dual neural network (PDNN) presents the basic idea from which the DNN, LVI-PDNN and S-LVI-PDNN are derived. The DNN is constructed using the dual decision variables only. The LVI-PDNN is designed based on the QP-LVI conversion, while the S-LVI-PDNN is a simplified version of LVI-PDNN.

Numerical algorithms can be employed for the solution of QPs.[92, 127–131] However, considering subtask criteria and physical constraints, manipulator redundancy resolution becomes time-consuming either by computing the pseudo-inverse-type solution or numerically solving QP problems.[67] The minimal arithmetic operations of a QP numerical algorithm are usually proportional to the cube of the decision variable vector's dimension ($O\left(n^3\right)$ operations).[92, 128–130] Consequently, numerical algorithms may not be efficient for real-time applications.[100, 132] The real-time computational requirement in sensor-based high-DOF robotic systems motivates the emergence of more efficient parallel processing schemes.[67] Parallel-processing computational methods, for example, neural-dynamic and analog solvers,[55, 99, 107, 113, 114, 123, 133–136] can be applied to solve the online QP problem.[100] Neural-dynamic approach is regarded as a powerful alternative for real-time computation in view of its parallel processing nature and convenience of hardware implementation.[54, 113, 123, 135, 137] Many of the neural networks used for kinematic control of manipulators are feedforward networks trained via supervised learning using the backpropagation algorithm. RNNs have also been applied for kinematic control. The dynamical system approach, as a method for solving optimization problems, was first proposed in ref. [138]. Many dynamic solvers based on neural networks were developed.[54, 55, 59, 113, 121, 133, 135, 137–143] Neural networks of other kinds[83, 144] can also be applied

https://doi.org/10.1017/S0263574719001590 Published online by Cambridge University Press

to solve the redundancy resolution problem. In the following sections, the main types of RNNs used as QP solvers are briefly reviewed.

4.1. Lagrangian network

Description: The dynamic equations of the Lagrangian network can be derived using the Lagrangian of the time-varying QP problem[55]

Literature review: A Lagrangian neural network was exploited in ref. [145] to handle general QP problems. In ref. [55], the optimal redundancy resolution is determined by the Lagrangian network through real-time solution to the IK problem formulated as a quadratic optimization problem. The signal for a desired velocity of the end effector is the input of the Lagrangian network, and the joint velocity vector of the manipulator along with the associated Lagrange multipliers are the outputs.

Advantages: The Lagrangian network is shown to be capable of asymptotic tracking for the motion control of kinematically redundant manipulators. It also overcomes the deficiencies of neural techniques that follow a penalty principle to solve constrained optimization problems.

Disadvantages: When solving inequality-constrained QPs, the Lagrange neural network may exhibit the premature defect and the network dimensionality is larger than that of the original problem.

4.2. Primal-Dual Neural Network

Description: The dynamic equations of the PDNN network can be derived using the gradient of the energy function.[121]

Literature review: As an improvement to the model in ref. [54], a two-layered PDNN is presented in ref. [121] to online minimize the ∞-norm of joint velocity. The PDNN[121] was developed for exact solution of constrained QPs, and they handle the primal QP and its dual problem simultaneously by minimizing the duality gap with gradient method. In ref. [146], a PDNN was proposed for generating the asymptotic convergent optimal solutions to strictly convex QP problems.[94] proposed the Lagrangian network and the PDNN for real-time joint torque optimization for kinematically redundant manipulators. For both neural networks, the desired accelerations of the end effector are given as their inputs, and the signals of the minimum driving joint torques are generated as their outputs to drive the manipulator. Both proposed networks are shown to be capable of generating minimum stable driving joint torques. The torques computed by the PDNN never exceed the joint torque limits.

Advantages: PDNN uses only simple hardware (adders, multipliers and integrators). Furthermore, the network is guaranteed for converging to exact optimal solution without penalty parameters and proven to be asymptotically stable.[121]

Disadvantages: The dynamic equations of the PDNN are complicated and contain high-order nonlinear terms, with the network size usually larger than the dimensionality of the primal and dual problems.

4.3. Dual Neural Network

Description: As a special case of the PDNN, the DNN[113] was proposed using the dual decision variables only. Different from the PDNN, the DNN is developed directly using KKT conditions and the projection operator to reduce network complexity and increase computational efficiency.

Literature review: In ref. [113], the IK problem of redundant manipulators was formulated as a parametric QP problem at the velocity level with equality constraint solved by a DNN by the real-time computation of the minimum two-norm joint velocity vector. In ref. [114], the real-time torque minimization problem was formulated as a time-varying QP, where the redundancy was resolved at the acceleration level and joint angle limit avoidance is simultaneously taken into account. The problem was solved using a DNN composed of one layer of $m + n$ neurons. The proposed DNN is exponentially convergent to an equilibrium point. In ref. [114], the DNN model in ref. [113] was extended to solve equality and bound-constrained optimization problems and then apply it to minimize the joint torque at the acceleration level with joint limits considered. Following the approach in ref. [113], to resolve the discontinuity deficiency of minimum ∞-norm solution, a DNN was designed in ref. [122] using the KKT condition and the projection operator for online kinematic control of physically constrained redundant manipulators, which is formulated as the inequality-constrained strictly convex QP problem with bi-criteria of the infinity and Euclidean norms. Joint angle and joint velocity limits

https://doi.org/10.1017/S0263574719001590 Published online by Cambridge University Press

are incorporated into the proposed scheme. The DNN is composed of one layer of no more than $3n + m + 1$ neurons without using analog multiplier or penalty parameter, as opposed to the approach in ref. [143]. A DNN was applied to solve the repeatability problem and simulation results based on PA10 manipulator was presented in ref. [69]

Advantages: The DNN in ref. [113] has an architecture of smaller size than that of the Lagrangian network in ref. [55], and it is composed of one layer of n neurons with no analog multipliers or penalty parameters. A circuit realizing the DNN consists of n summers, n integrators and n^2 connections. The DNN is theoretically proven to be exponentially stable.[113] PDNN[94,121,143] has more complicated dynamics and structure in terms of high-order terms and number of neurons/layers. Compared to the PDNN,[121] the dynamic equation of a DNN is piecewise linear without any high-order nonlinear term. Consequently, the architecture of the DNN is much simpler than that of the PDNN and Lagrangian networks.[122] Starting from any initial state, the state vector of the DNN is convergent to an equilibrium point and the output vector converges to the optimal solution of the bi-criteria IK problem.[122]

Disadvantages: The DNN requires online matrix inversion and thus is only able to handle strictly convex QPs and less desirable in terms of computational efficiency and hardware implementation.[123]

4.4. Linear Variational Inequalities-Based Primal-Dual Neural Network

Description: As a QP real-time solver, the linear variational inequalities-based primal-dual neural network (LVI-PDNN) is designed based on the QP-LVI conversion and KKT condition.

Literature review: In refs. [67, 99], LVI-PDNN is used to solve the RMP problem online by minimizing a suitable performance index and incorporating joint angle and joint velocity limits into the problem formulation. The scheme is reformulated as a QP problem and resolved at the velocity level. An LVI-PDNN was developed in ref. [147] with simple piecewise linear dynamics, global (exponential) convergence to optimal solutions, and capability of handling general QP and linear programming problems in an inverse-free manner. In ref. [148], a neural-dynamic-based synchronous optimization scheme of dual redundant manipulators has been proposed. An ARMP optimization criterion has been derived twice. The redundancy resolution problem of the left and right arms are formulated as two QP problems. The two QPs are then integrated into a standard QP problem. An LVI-PDNN was used to solve the QP problem. Considering the differentiation error and the implementation error, a *perturbed* LVI-PDNN has been proposed to solve the QP problem.

Advantages: The LVI-PDNN is capable of handling general QP and linear programming problems in an inverse-free manner, as it does not entail online matrix inversion.[67,99] Different from other neural networks,[69,122,123,149] there is no matrix inversion of expensively $O(n^3)$ operations in the LVI-PDNN approach. The network architecture and computational complexity are thus simpler than other RNNs.[67]

Disadvantages: There are no disadvantages for the LVI-PDNN compared to the other neural networks discussed in this section.

4.5. Simplified-Linear Variational Inequality-Based Primal-Dual Neural Network

Description: The simplified model simplified-linear variational inequality-based primal-dual neural network (S-LVI-PDNN) is obtained by removing the scaling term of the LVI-PDNN dynamic equation [compare (23) and (24)].

Literature review: In ref. [136], to reduce the implementation and computational complexities, an S-LVI-PDNN model is investigated. In ref. [124], an S-LVI-PDNN was presented for online RMP. To do this, a drift-free criterion is exploited in the form of a quadratic function. Joint angle and joint velocity limits are incorporated in the RMP scheme. The scheme is reformulated as a time-varying QP. The S-LVI-PDNN has a simple piecewise linear dynamics and could globally exponentially converge to the optimal solution of strictly convex QPs. The S-LVI-PDNN model is simulated based on PA10 manipulator, and simulation results show the effective remedy of the joint angle drift problem.

Advantages: The S-LVI-PDNN has simpler computational complexity than the LVI-PDNN. In ref. [100], a DNN, LVI-PDNN and S-LVI-PDNN are presented for online RMP of redundant manipulators, and a drift-free criterion is exploited in the form of a quadratic performance index. The scheme also incorporates joint angle and joint velocity limits. The scheme is reformulated as a QP. As QP real-time solvers, the proposed neural networks all have piecewise linear dynamics and can globally exponentially converge to the optimal solution of strictly convex QP. It was shown that

https://doi.org/10.1017/S0263574719001590 Published online by Cambridge University Press

Table I. Number of operations to be performed by different RNN solvers.

	DNN (20)	**LVI-PDNN (23)**	**S-LVI-PDNN (24)**
Additions/subtractions	$10n + 4m$	$6(n + m)$	$4(n + m)$
Multiplications	$5n(n + m)$	$2(n + m)^2$	$(n + m)^2$

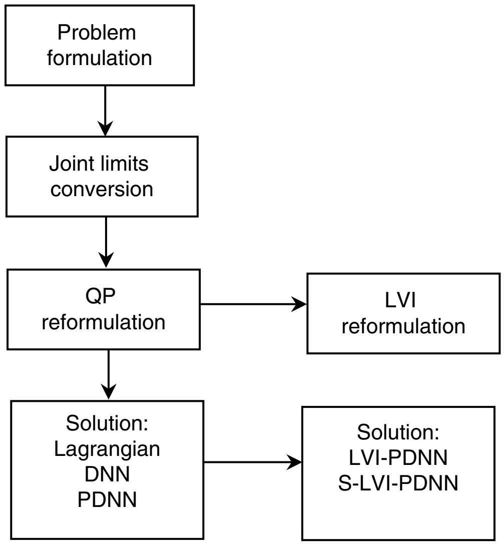

Fig. 2. Methodology for redundancy resolution using RNNs.

the S-LVI-PDNN has the lowest structural and computational complexity among the three networks (see Table I).

Disadvantages: There are no disadvantages for the S-LVI-PDNN compared to the other neural networks discussed in this section.

5. Methodology

Figure 2 shows the main steps involved in the IK problem formulated as a QP problem solved by different types of RNNs.

The procedures start by the initial formulation of the problem, for example, a performance index can be utilized to achieve drift-free motion at the velocity level (VRMP) or acceleration level (ARMP) taking into consideration different joint variables limits (joint angle limits, joint velocity limits and joint acceleration limits). Depending on the level at which the redundancy resolution problem is formulated (velocity level or acceleration level) and the limited joint variables considered, the second step involves the conversion of all joint limits to conform with the same level of the problem, for example, if the problem is formulated at the joint acceleration level, while joint angle, joint velocity and joint acceleration limits are all considered, then the limited joint angle range and joint velocity range have to be converted into an expression based on joint acceleration. In the third step, the problem is reformulated as a QP problem where a new performance index is utilized along with the new joint constraints that resulted from the joint limits conversion conducted in the previous step. At this stage, the problem can be solved using an RNN like DNN or PDNN. The problem can also be solved using LVI-PDNN or S-LVI-PDNN. To do this, a QP-LVI conversion step has to be conducted, where the problem is finally reformulated as a set of LVIs.

6. Case Study: Drift-Free Inverse Kinematics at the Velocity Level Solved by DNN, LVI-PDNN and S-LVI-PDNN

When devising the initial problem formulation, the below points should be considered:

1. The problem is solved at different levels, pertaining to objective function order:
   Velocity level, first order $(\dot{\boldsymbol{\theta}}, \dot{\mathbf{r}})$, acceleration level, second order $(\ddot{\boldsymbol{\theta}}, \ddot{\mathbf{r}})$, jerk level, third order $(\dddot{\boldsymbol{\theta}}, \dddot{\mathbf{r}})$.

https://doi.org/10.1017/S0263574719001590 Published online by Cambridge University Press

2. Performance criteria to be optimized, as defined in the objective function, for example, Joint velocity.
3. Norm of performance criteria to be optimized:
   Single Criterion (two-norm or ∞-norm) or bi-criteria (combined ∞-norm and two-norm).
4. Subtasks considered, as defined in the problem constraints:
   For example, Joint angle limits, Joint angle and joint velocity limits, etc.

Other subtasks can be handled in the same framework (e.g., obstacle avoidance[149]), if the subtasks are formulated in terms of constraints rather than performance indices.

6.1. The VRMP scheme

Taking into account the joint angle and joint velocity limits, the VRMP scheme for physically constrained redundant manipulators can be formulated as[67,99,100,111] (with the same performance index as in refs. [69, 98]:

$$\text{Minimize } \frac{1}{2}(\dot{\boldsymbol{\theta}}+\mathbf{z})^T\left(\dot{\boldsymbol{\theta}}+\mathbf{z}\right), \qquad \text{where} \quad \mathbf{z}=\lambda\left(\boldsymbol{\theta}-\boldsymbol{\theta}(0)\right) \tag{10}$$

$$\text{Subject to } \mathbf{J}\dot{\boldsymbol{\theta}}=\dot{\mathbf{r}} \tag{11}$$

$$\boldsymbol{\theta}^- \le \boldsymbol{\theta} \le \boldsymbol{\theta}^+ \tag{12}$$

$$\dot{\boldsymbol{\theta}}^- \le \dot{\boldsymbol{\theta}} \le \dot{\boldsymbol{\theta}}^+, \tag{13}$$

where λ is a positive design parameter used to scale the magnitude of the manipulator response to joint displacements.

6.2. Joint limits conversion

As the redundancy is resolved at the joint velocity level, the limited joint angle range $\left[\boldsymbol{\theta}^-, \boldsymbol{\theta}^+\right]$ in (12) has to be converted into an expression based on joint velocity.[99,107,114,134,150] For example, the following transformation from $\boldsymbol{\theta}$ to $\dot{\boldsymbol{\theta}}$ expressions can be used (i.e., using a dynamic bound constraint):[100]

$$\mu\left(\theta^- - \theta\right) \le \dot{\theta} \le \mu\left(\theta^+ - \theta\right) \tag{14}$$

where the intensity coefficient $\mu > 0$ is used to scale the feasible region of $\dot{\boldsymbol{\theta}}$ caused by the above transformation. The choice of coefficient μ should make sure that the feasible region of $\dot{\boldsymbol{\theta}}$ converted by joint limits (12) is usually not less than the original one made by joint velocity limits (13),[100] $\mu \ge \max_{1\le i\le n}\left\{\left(\dot{\boldsymbol{\theta}}_i^+ - \dot{\boldsymbol{\theta}}_i^-\right)/\left(\boldsymbol{\theta}_i^+ - \boldsymbol{\theta}_i^-\right)\right\}$. Large values of μ may cause quick joint deceleration when the manipulator approaches its joint limits. The physical meaning of the transformation can be explained by taking the right-hand inequality of (14) as an example. As joint variable $\boldsymbol{\theta}$ increases (toward its upper bound $\boldsymbol{\theta}^+$, evidently the upper bound of joint velocity variable $\dot{\boldsymbol{\theta}}$ should accordingly decrease. If $\boldsymbol{\theta}$ reaches its upper limit $\boldsymbol{\theta}^+$, the right-hand inequality of (14) becomes $\dot{\boldsymbol{\theta}} \le 0$ which means that $\boldsymbol{\theta}$ cannot increase any more. Thus, $\boldsymbol{\theta}$ will never exceed its upper limit $\boldsymbol{\theta}^+$. Similarly, $\boldsymbol{\theta}$ will not exceed its lower limit $\boldsymbol{\theta}^-$ either, owing to the left-hand inequality of (14).[100] Equations (13) and (14) can thus be combined into one dynamic bound-constraint $\boldsymbol{\xi}^- \le \dot{\boldsymbol{\theta}} \le \boldsymbol{\xi}^+$ in a unified manner, where the ith elements of $\boldsymbol{\xi}^-$ and $\boldsymbol{\xi}^+$ are defined as:[100] $\xi_i^- = \max\left\{\dot{\theta}_i^-, \mu\left(\theta_i^- - \theta_i\right)\right\}$, $\xi_i^+ = \min\left\{\dot{\theta}_i^+, \mu\left(\theta_i^+ - \theta_i\right)\right\}$.

6.3. QP reformulation

The VRMP scheme (10)–(13) can be reformulated as the following QP problem in terms of joint velocity $\dot{\boldsymbol{\theta}}$:[67,100,111]

$$\text{Minimize } \frac{1}{2}\dot{\boldsymbol{\theta}}^T\mathbf{W}\dot{\boldsymbol{\theta}}+\mathbf{z}^T\dot{\boldsymbol{\theta}} \tag{15}$$

https://doi.org/10.1017/S0263574719001590 Published online by Cambridge University Press

$$\text{Subject to } \ \mathbf{J}\dot{\boldsymbol{\theta}} = \dot{\mathbf{r}} \tag{16}$$

$$\boldsymbol{\xi}^- \leq \dot{\boldsymbol{\theta}} \leq \boldsymbol{\xi}^+, \tag{17}$$

where $\mathbf{W} = \mathbf{I}$, $\boldsymbol{\xi}^-$ and $\boldsymbol{\xi}^+$ are the lower and upper limits, respectively, of the new bound constraint. The velocity-level performance index (15) which results from the simplification of (10) is also called the drift-free criterion at the joint velocity level. Most coefficients in (15)–(17) are time-varying, which entails real-time solution.[67]

6.4. DNN solver

The QP (15)–(17) can be rewritten as:[100]

$$\text{Minimize } \frac{1}{2}\dot{\boldsymbol{\theta}}^T\mathbf{W}\dot{\boldsymbol{\theta}} + \mathbf{z}^T\dot{\boldsymbol{\theta}} \tag{18}$$

$$\text{Subject to } \ \boldsymbol{\gamma}^- \leq \mathbf{E}\dot{\boldsymbol{\theta}} \leq \boldsymbol{\gamma}^+, \quad \boldsymbol{\gamma}^- = \left[\dot{\mathbf{r}}^T, \ \left(\xi^-\right)^T\right]^T, \quad \boldsymbol{\gamma}^+ = \left[\dot{\mathbf{r}}^T, \ \left(\xi^+\right)^T\right]^T, \quad \mathbf{E} = \begin{bmatrix}\mathbf{J}\\ \mathbf{I}\end{bmatrix}$$

At some instant t, QP problem (15)–(17) can be viewed as a parametric optimization problem. By the KKT condition,[130] $\dot{\boldsymbol{\theta}}$ is a solution to (18) IFF there is a dual decision variable vector $\mathbf{u} \in \mathbf{R}^{n+m}$ such that $\dot{\boldsymbol{\theta}} - \mathbf{E}^T\mathbf{u} + \mathbf{z} = 0$ (in view of $\mathbf{W} = \mathbf{I} = \mathbf{W}^{-1}$) and:[100]

$$\begin{cases} \left[E\dot{\theta}\right]_i = \gamma_i^- & \text{if } u_i > 0 \\ \gamma_i^- \leq \left[E\dot{\theta}\right]_i \leq \gamma_i^+ & \text{if } u_i = 0 \\ \left[E\dot{\theta}\right]_i = \gamma_i^+ & \text{if } u_i < 0 \end{cases}$$

which is equivalent to the system of piecewise linear equations $\mathbf{E}\dot{\boldsymbol{\theta}} = P_\Omega\left(\mathbf{E}\dot{\boldsymbol{\theta}} - \mathbf{u}\right)$,[123] where $P_\Omega\left(.\right)$ is a projection operator from $\mathbf{R}^{n+m}$ onto $\Omega = \left\{\mathbf{u} \mid \boldsymbol{\gamma}^- \leq \mathbf{u} \leq \boldsymbol{\gamma}^+\right\} \subset \mathbf{R}^{n+m}$ and the ith output of $P_\Omega\left(\mathbf{u}\right)$ is defined as:[100]

$$\begin{cases} \gamma_i^- & \text{if } u_i < \gamma_i^- \\ u_i & \text{if } \gamma_i^- \leq u_i \leq \gamma_i^+, \quad \forall i \in \{1, \ 2, \ \ldots, \ n+m\} \\ \gamma_i^+ & \text{if } u_i > \gamma_i^+ \end{cases}$$

By the above analysis, the necessary and sufficient condition for solving QP (18) is that $\dot{\boldsymbol{\theta}}$ and $\mathbf{u}$ satisfy $\dot{\boldsymbol{\theta}} - \mathbf{E}^T\mathbf{u} + \mathbf{z} = 0$ and $\mathbf{E}\dot{\boldsymbol{\theta}} = P_\Omega\left(\mathbf{E}\dot{\boldsymbol{\theta}} - \mathbf{u}\right)$. Substituting the former equation into the latter:[100]

$$P_\Omega\left(\mathbf{E}\mathbf{E}^T\mathbf{u} - \mathbf{E}\mathbf{z} - \mathbf{u}\right) = \mathbf{E}\mathbf{E}^T\mathbf{u} - \mathbf{E}\mathbf{z} \tag{19}$$

which gives rise to the dynamic equation of DNN solving QP (18) as well as QP (15)–(17):[100]

$$\dot{\mathbf{u}} = c\left(P_\Omega\left(\mathbf{E}\mathbf{E}^T\mathbf{u} - \mathbf{E}\mathbf{z} - \mathbf{u}\right) - \mathbf{E}\mathbf{E}^T\mathbf{u} + \mathbf{E}\mathbf{z}\right), \tag{20}$$

where c is a design parameter used to scale the convergence rate of neural networks

6.5. LVI reformulation

By duality theory,[130] for the primal QP problem (15)–(17), its dual QP problem can be derived with the aid of dual decision variables. The dual decision variable is usually defined as the Lagrangian multiplier for each constraint such as (16) and (17). QP (15)–(17) can be converted to a set of LVIs. That is, to find a primal-dual equilibrium vector $\mathbf{v}^* \in \Omega = \left\{\mathbf{u} \mid \boldsymbol{\gamma}^- \leq \mathbf{u} \leq \boldsymbol{\gamma}^+\right\}$ such that:[67,99,100]

$$\left(\mathbf{v} - \mathbf{v}^*\right)^T\left(\mathbf{M}\mathbf{v}^* + \mathbf{q}\right) \geq 0, \quad \forall \ \mathbf{v} \in \Omega \tag{21}$$

where the primal-dual decision variable vector $\mathbf{v}$ and its upper/lower bounds $\boldsymbol{\gamma}^\pm$ are defined as (with $\boldsymbol{\varpi} \gg 0 \in \mathbf{R}^m$ to replace the m-dimensional $+\infty$ numerically):

$$\mathbf{v} = \begin{bmatrix}\dot{\boldsymbol{\theta}}\\ \mathbf{y}\end{bmatrix}, \quad \boldsymbol{\gamma}^+ = \begin{bmatrix}\boldsymbol{\xi}^+\\ \boldsymbol{\varpi}\end{bmatrix}, \quad \boldsymbol{\gamma}^- = \begin{bmatrix}\boldsymbol{\xi}^-\\ -\boldsymbol{\varpi}\end{bmatrix} \in \mathbf{R}^{m+n}$$

$\mathbf{y} \in \mathbf{R}^m$ is a dual decision variable vector defined for equality constraint (16), $\boldsymbol{\varpi}$ is sufficiently large to replace $+\infty$

$$\mathbf{M} = \begin{bmatrix} \mathbf{W} & -\mathbf{J}^T \\ \mathbf{J} & 0 \end{bmatrix}, \qquad \mathbf{q} = \begin{bmatrix} \mathbf{z} \\ -\dot{\mathbf{r}} \end{bmatrix} \in \mathbf{R}^{m+n}$$

The LVI problem (21) is equivalent to the system of piecewise linear equations:[69,99,107,114,130,150–155]

$$P_\Omega\left(\mathbf{v} - (\mathbf{M}\mathbf{v} + \mathbf{q})\right) - \mathbf{v} = 0 \tag{22}$$

where $P_\Omega(\,.\,)$ is a projection operator from space $\mathbf{R}^{m+n}$ onto set Ω which are defined to be the same as in the DNN case (except for $\boldsymbol{\gamma}^+$ and $\boldsymbol{\gamma}^-$).

6.6. *LVI-PDNN solver*

To improve the efficacy of the online solution of QP (15)–(17), an LVI-PDNN is presented in refs. [67, 99] as the QP real-time solver, which is designed based on LVIs.[151,152,154,155] To solve LVI (22) as well as QP (15)–(17), from refs. [55, 69, 114, 123, 156], the following dynamic equation of LVI-PDNN is obtained:[100]

$$\dot{\mathbf{v}} = c\left(\mathbf{I} + \mathbf{M}^T\right)\left(P_\Omega\left(\mathbf{v} - (\mathbf{M}\mathbf{v} + \mathbf{q})\right) - \mathbf{v}\right) = 0 \tag{23}$$

6.7. *S-LVI-PDNN solver*

By removing the scaling term $\left(\mathbf{I} + \mathbf{M}^T\right)$ of LVI-PDNN (23), the simplified model (S-LVI-PDNN) is obtained:[100]

$$\dot{\mathbf{v}} = c\left(P_\Omega\left(\mathbf{v} - (\mathbf{M}\mathbf{v} + \mathbf{q})\right) - \mathbf{v}\right) \tag{24}$$

7. Discussion

Table II summarizes the basic information for different references where the redundancy resolution problem was formulated as a QP problem solved by different types of RNNs.

8. Conclusion

RNNs exhibits many advantages as QP real-time solvers as outlined below:

1. *Path-following task:* In almost all references, the path-following task was completed satisfactorily even in the cases where the solution is not repetitive or the joint limits were violated to a certain extent.
2. *Repetitive motion planning:* In refs. [67, 99, 110, 111], it was shown that the solution becomes repetitive when exploiting the drift-free criterion.
3. *Joint limits avoidance:* In refs. [67, 110, 111, 114], it was shown that, with joint limits considered in the problem formulation, the respective joint variables are kept within their limits during the simulation
4. *Computation time:* In ref. [111], the upper bound of the computation time of the LVI-PDNN is approximately 1.45×10^{-3} s and 9×10^{-3} s for solving the VRMP and ARMP schemes, respectively. In ref. [110], the simulations based on the VRMP and ARMP schemes solved by the S-LVI-PDNN are applied on the 3 DOF planar arm for the same path tracking task. The maximum computation time is 2×10^{-4} s for the VRMP scheme and 5×10^{-4} s for the ARMP scheme which is still very small, indicating that both the VRMP and ARMP schemes solved by the LVI-PDNN or the S-LVI-PDNN are efficient for real-time motion planning and control and are applicable in practical applications.
5. *Convergence:* As QP real-time solvers, the aforementioned neural networks all have piecewise linear dynamics and could globally exponentially converge to the optimal solution of strictly convex QPs.

https://doi.org/10.1017/S0263574719001590 Published online by Cambridge University Press

Table II. Summary of the formulations and RNN solvers used in different references.

Reference	RNN solver	Formulation	Manipulator	Max position error (m)[a]
[55]	Lagrangian	Velocity level	PA-10	$< 10^{-4}$
[121]	PDNN	Velocity level	4 DOF PA-10	Not applicable[b]
[113]	DNN	Velocity level	5 DOF PA-10	$< 2 \times 10^{-16}$ $< 6 \times 10^{-8}$
[114]	DNN	Acceleration level	PUMA 560	$< 1.5\times10^{-4}$
[122]	DNN	Bi-criteria	PA-10	$< 3 \times 10^{-7}$
[99]	LVI-PDNN	VRMP	PUMA 560	$< 4 \times 10^{-8}$
[124]	S-LVI-PDNN	VRMP	PA-10	$< 2\times10^{-6}$
[67]	LVI-PDNN	VRMP	PA-10	$< 1.5 \times 10^{-7}$
[100]	DNN LVI-PDNN S-LVI-PDNN	VRMP	4 DOF	Not applicable[c]
[110][d]	S-LVI-PDNN	ARMP	3 DOF 4 DOF 5 DOF	$< 2.5\times10^{-3}$ $< 5\times10^{-5}$ $< 3.5\times10^{-4}$
[111]	LVI-PDNN	VRMP ARMP	PA-10	$< 1.5 \times 10^{-7}$ $< 1.5 \times 10^{-3}$
[148]	LVI-PDNN *Perturbed* LVI-PDNN[f]	ARMP	Dual PA-10	$< 8\times10^{-4}$[e] $< 10^{-3}$

[a] The values of position error given in the table should be considered as an indication of the performance of the scheme used in the respective reference, not as a measure to compare the relative performance of the schemes employed in different references owing to the fact that these results are produced based on different setups for example, different manipulators, different workspace paths, etc.
[b] The main aim in ref. [121] was reducing the architecture complexity of the neural network and minimizing the joint velocity
[c] The main aim in ref. [100] was comparing between DNN, LVI-PDNN, S-LVI-PDNN used as neural network solvers for the redundancy resolution problem based on QP formulation and showing their efficacy in solving the joint angle drift problem. Table I shows the number of operations to be performed by DNN (20), LVI-PDNN (23) and S-LVI-PDNN (24) per iteration for the same QP-solving and redundancy resolution purposes. The computational cost of the DNN is the highest among the three neural-network solvers, while the S-LVI-PDNN has the lowest structural and computational complexity.[100]
[d] In ref. [110], in addition to the results indicated in the table, for further verification, the ARMP scheme is performed on a six-link planar manipulator. The experiment has demonstrated the physical realizability of the ARMP scheme and the corresponding S-LVI-PDNN solver on the drift-free motion planning and control of the manipulator. Simulations based on PUMA 560 and PA10 were also performed. As synthesized by the ARMP scheme, tracking tasks are completed well, all joint trajectories are closed and all joint variables are kept within their limits.
[e] In ref. [148], simulations were conducted for the dual PA10 manipulators to track three different trajectories. The maximum position error among all trajectories was less than 8×10^{-4} m. In this case, the QP problem has been solved using the standard LVI-PDNN (23).
[f] In ref. [148], considering the differentiation error and the implementation error, a *perturbed* LVI-PDNN has been proposed to solve the QP problem [compare with the standard LVI-PDNN indicated in (23)].

$$\dot{\mathbf{v}} = c\left(\mathbf{I} + \mathbf{M}^T + \triangle\mathbf{D}\right)(P_\Omega(\mathbf{v} - (\mathbf{M}\mathbf{v} + \mathbf{q})) - \mathbf{v}) + \triangle\mathbf{S}, \tag{25}$$

where $\triangle\mathbf{D} \in \mathbf{R}^{(2n+2m)\times(2n+2m)}$ and $\triangle\mathbf{S} \in \mathbf{R}^{(2n+2m)}$ denote the differentiation error matrix and the implementation error vector, respectively. Only one trajectory was tested in this case.

On the other hand, the below points need to be considered:

1. *End-effector orientation:* In refs. [55, 100, 113, 114, 122], only the end-effector positioning is considered, whereas in many real-life applications, it is important to consider the end-effector orientation as well.
2. *Simulation versus Experimental Results:* Most references depend on computer simulation rather than experimental results. Experimental results might differ from simulation results due to the inconstancy between the real environment and the simulated one.
3. *Joint limits:* In some formulations, joint velocity limits and/or joint acceleration limits are not considered when solving the IK problem.

https://doi.org/10.1017/S0263574719001590 Published online by Cambridge University Press

References

1. J. F. Gardner, A. Brandt and G. Luecke, "Applications of neural networks for coordinate transformations in robotics," *J. Intell. Rob. Syst.* **8**(3), 361–373 (1993).
2. Z. Bingul, H. M. Ertunc and C. Oysu, "Applying Neural Network to Inverse Kinematic Problem for 6R Robot Manipulator with Offset Wrist," **In**: *Adaptive and Natural Computing Algorithms* (B. Ribeiro, R. F. Albrecht, A. Dobnikar, D. W. Pearson and N. C. Steele, eds.) (Springer, Vienna, 2005).
3. A. T. Hasan, H. M. A. A. Al-Assadi and A. A. M. Isa, "Neural Networks' Based Inverse Kinematics Solution for Serial Robot Manipulators Passing Through Singularities," **In**: *Artificial Neural Networks - Industrial and Control Engineering Applications* (IntechOpen, London, 2011) pp. 459–478.
4. S. Oltean, D. Şoaita, M. Dulau and T. Jovrea, "Aspects of Interdisciplinarity in the MRIP 02 System Design," *International Conference "Optimization of the Robots and Manipulators" (OPTIROB)* (Bren Publishing House, 2007) pp. 121–125.
5. C. Secarã and L. Vlãdãreanu, "Iterative Genetic Algorithm Based Strategy for Obstacles Avoidance of a Redundant Manipulator," *Proceedings of the 2010 American Conference on Applied Mathematics*, Cambridge, USA (2010) pp. 361–366.
6. R. Featherstone, "Position and velocity transformations between robot end-effector coordinates and joint angles," *Int. J. Rob. Res.* **2**(2), 33–45 (1983).
7. G. C. S. Lee, "Robot arm kinematics, dynamics, and control," *IEE Comput.* **15**(12), 62–79 (1982).
8. C. S. G. Lee and M. Ziegler, "Geometric approach in solving inverse kinematics of PUMA robots," *IEEE Trans. Aero. Electron. Syst.* (6), 695–706 (1984).
9. K. Tokarz and S. Kieltyka, "Geometric approach to inverse kinematics for arm manipulator," *ICS'10 Proceedings of the 14th WSEAS International Conference on Systems: Part of the 14th WSEAS CSCC Multi-conference*, Corfu Island, Greece (2010) pp. 682–687.
10. G. Rodriguez, A. Jain and K. Kreutz-Delgado, "A spatial operator algebra for manipulator modelling and control," *Int. J. Rob. Res.* **10**(4), 371–381 (1991).
11. X. Wang and J.-P. Verriest, "A geometric algorithm to predict the arm reach posture of computer-aided ergonomic evaluation," *J. Visual Comput. Animat.* **9**(1), 33–47 (1998).
12. J. Duffy, *Analysis of Mechanism and Robot Manipulators* (John Wiley & Sons, Inc., New York, NY, 1980). ISBN: 0470270020.
13. D. Manocha and J. F. Canny, "Efficient inverse kinematics for general 6R manipulators," *IEEE Trans. Rob. Autom.* **10**(5), 648–657 (1994).
14. R. P. Paul, B. Shimano and G. E. Mayer, "Differential kinematic control equations for simple manipulators," *IEEE Trans. Syst. Man Cybern.* **1**(1), 66–72 (1981).
15. K. S. Fu, R. C. Gonzalez and C. S. G. Lee, *Robotics: Control, Sensing, Vision and Intelligence* (McGraw-Hill, Inc., New York, NY, 1987). ISBN: 0070226253.
16. D. Manocha and Y. Zhu, "A fast Algorithm and System for the Inverse Kinematics of General Serial Manipulators," *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, San Diego, CA, USA, vol. 4 (1994) pp. 3348–3353.
17. R. P. Paul, *Robot Manipulators: Mathematics, Programming and Control* (MIT Press, Cambridge, MA, 1982). ISBN: 026216082X.
18. S. Stiffer, "Algebraic methods for computing inverse kinematics," *J. Intell. Rob. Syst.* **11**(1–2), 79–89 (1994).
19. B. Buchberger, "Gröbner Bases: An Algorithmic Method in Polynomial Ideal Theory," **In**: *Multidimensional Systems Theory* (N. K. Bose, ed.) (Springer, Dordrecht, 1985) pp. 184–232.
20. D. Manocha, *Algebraic and Numeric Techniques in Modeling and Robotics* (University of California at Berkeley, Berkeley, CA, 1992).
21. W. E. Snyder, *Industrial Robots: Computer Interfacing and Control* (Prentice-Hall Industrial Robots Series, Prentice-Hall, Englewood Cliffs, NJ, 1985). ISBN: 9780134631592.
22. J. J. Craig, *Introduction to Robotics - Mechanics & Control,* 3rd edition (Pearson, Prentice Hall, Upper Saddle River, NJ, 2005). ISBN: 0201543613.
23. J. U. Korein and N. I. Balder, "Techniques for generating the goal-directed motion of articulated structures," *IEEE Comput. Graphics Appl.* **2**(9), 71–81 (1982).
24. L.-C. T. Wang and C. C. Chen, "A combined optimization method for solving the inverse kinematics problems of mechanical manipulators," *IEEE Trans. Rob. Autom.* **7**(4), 489–499 (1991).
25. G. Z. Grudić and P. D. Lawrence, "Iterative inverse kinematics with manipulator configuration control," *IEEE Trans. Rob. Autom.* **9**(4), 476–483 (1993).
26. B. Karlik and S. Aydin, "An improved approach to the solution of inverse kinematics problems for robot manipulators," *Eng. Appl. Artif. Intell.* **13**(2), 159–164 (2000).
27. R. Köker, "Reliability-based approach to the inverse kinematics solution of robots using Elman's networks," *Eng. Appl. Artif. Intell.* **18**(6), 685–693 (2005).
28. Z. G. Grudic and P. D. Lawrange, "Iterative inverse kinematics with manipulator configuration control," *IEEE Trans. Rob. Autom.* **9**(4), 476–483 (1993).
29. S. Yahya, M. Moghavvemia and H. A. F. Mohamed, "Geometrical approach of planar hyper-redundant manipulators: Inverse kinematics, path planning and workspace," *Simul. Modell. Pract. Theory* **19**(1), 406–422 (2011).

30. S. S. Chiddarwar and N. R. Babu, “Comparison of RBF and MLP neural networks to solve inverse kinematic problem for 6R serial robot by a fusion approach,” *Eng. Appl. Artif. Intell.* **23**(7), 1083–1092 (2010).
31. W. J. Daunicht, “Approximation of the Inverse Kinematics of an Industrial Robot by DEFAnet,” *Proceedings of the 1991 IEEE International Joint Conference on Neural Networks*, Singapore, vol. 3 (1991) pp. 1995–2000.
32. J. Hakala, G. Fahner and R. Eckmiller, “Rapid Learning of Inverse Robot Kinematics Based on Connection Assignment and Topographical Encoding (CATE),” *Proceedings of the 1991 IEEE International Joint Conference on Neural Networks,* Singapore, vol. 2 (1991) pp. 1536–1541.
33. A. T. Hasan, A. M. S. Hamouda, N. Ismail and H. M. A. A. Al-Assadi, “An adaptive-learning algorithm to solve the inverse kinematics problem of a 6 D.O.F serial robot manipulator,” *Adv. Eng. Software* **37**(7), 432–438 (2006).
34. A. T. Hasan, N. Ismail, A. M. S. Hamouda, I. Aris, M. H. Marhaban and H. M. A. A. Al-Assadi, “Artificial neural network-based kinematics Jacobian solution for serial manipulator passing through singular configurations,” *Adv. Eng. Software* **41**(2), 359–367 (2010).
35. R. Köker, C. Öz, T. Çakar and H. Ekiz, “A study of neural network based inverse kinematics solution for a three-joint robot,” *Rob. Auton. Syst.* **49**(3–4), 227–234 (2004).
36. J. de Lope and M. Santos, “A method to learn the inverse kinematics of multi-link robots by evolving neuro-controllers,” *Neurocomputing* **72**(13–15), 2806–2814 (2009).
37. T. M. Martinetz, H. J. Ritter and K. J. Schulten, “Three-dimensional neural net for learning visuomotor coordination of a robot arm,” *IEEE Trans. Neural Networks* **1**(1), 131–136 (1990).
38. R. V. Mayorga and P. Sanongboon, “Inverse kinematics and geometrically bounded singularities prevention of redundant manipulators: An Artificial Neural Network approach,” *Rob. Auton. Syst.* **53**(3–4), 164–176 (2005).
39. E. Oyama, N. Y. Chong, A. Agah, T. Maeda and S. Tachi, “Inverse Kinematics Learning by Modular Architecture Neural Networks with Performance Prediction Networks,” *Proceedings 2001 ICRA. IEEE International Conference on Robotics & Automation,* Seoul, South Korea, vol. 1 (2001) pp. 1006–1012.
40. S. Tejomurtula and S. Kak, “Inverse kinematics in robotics using neural networks,” *Inf. Sci.* **116**(2–4), 147–164 (1999).
41. P. C. Y. Chen, J. K. Mills and K. C. Smith, “Performance improvement of robot continuous-path operation through iterative learning using neural networks,” *Mach. Learn. J.* **23**(2–3), 191–220 (1996).
42. R. Köker, C. Oz and A. Ferikoglu, “Development of a Vision Based Object Classification System for an Industrial Robotic Manipulator,” *ICECS 2001. 8th IEEE International Conference on Electronics*, Circuits and Systems, Malta, vol. 3 (2001) pp. 1281–1284.
43. B.-L. Lu and K. Ito, “Regularization of Inverse Kinematics for Redundant Manipulators Using Neural Network Inversions,” *Proceedings of ICNN’95 - International Conference on Neural Networks*, Perth, WA, Australia, vol. 5 (1995) pp. 2726–2731.
44. F. Nagata, S. Kishimoto, S. Kurita, A. Otsuka and K. Watanabe, “Neural Network-Based Inverse Kinematics for an Industrial Robot and its Learning Method,” *Proceedings of the 4th IIAE International Conference on Industrial Application Engineering*, Beppu, Oita, Japan (2016).
45. A. R. J. Almusawi, L. C. Dülger and S. Kapucu, “A New Artificial Neural Network Approach in Solving Inverse Kinematics of Robotic Arm (Denso VP 6242),” *Comput. Intell. Neurosci.* **2016**, 5720163 (2016).
46. Z. Zhang, L. Zheng, J. Yu, Y. Li and Z. Yu, “Three recurrent neural networks and three numerical methods for solving a repetitive motion planning scheme of redundant robot manipulators,” *IEEE/ASME Trans. Mech.* **22**(3), 1423–1434 (2017).
47. S. Li, M. Zhou and X. Luo, “Modified primal-dual neural networks for motion control of redundant manipulators with dynamic rejection of harmonic noises,” *IEEE Trans. Neural Networks Learn. Syst.* **29**(10), 4791–4801 (2018).
48. A. Guez and Z. Ahmad, “Solution to the Inverse Problem in Robotics by Neural Networks,” *IEEE 1988 International Conference on Neural Networks, 1988*, San Diego, CA, USA, vol. 2 (1988) pp. 617–624.
49. G. Josin, D. Charney and D. White, “Robot Control Using Neural Networks,” *IEEE 1988 International Conference on Neural Networks, 1988*, San Diego, CA, USA, vol. 2 (1988) pp. 625–631.
50. J. Guo and V. Cherkassky, “A Solution to the Inverse Kinematics Problem in Robotics Using Neural Network Processing,” *International 1989 Joint Conference on Neural Networks, 1989*, Washington, DC, USA, vol. 2 (1989) pp. 299–304.
51. A. Guez and Z. Ahmad, “Accelerated Convergence in the Inverse Kinematics via Multilayer Feedforward Networks,” *International 1989 Joint Conference on Neural Networks, 1989*, Washington, DC, USA, vol. 2 (1989) pp. 341–344.
52. F. J. Arteaga-Bravo, “Multilayer Back-Propagation Network for Learning the Forward and Inverse Kinematics Equations,” *Proceedings of the International Conference on Neural Networks,* Washington, DC, USA (1990) pp. 319–322.
53. S. Lee and R. M. Kil, “Robot Kinematic Control Based on Bidirectional Mapping Neural Network,” *1990 IJCNN International Joint Conference on Neural Networks*, San Diego, CA, USA, vol. 3 (1990) pp. 327–335.
54. H. Ding and J. Wang, “Recurrent neural networks for minimum infinity-norm kinematic control of redundant manipulators,” *IEEE Trans. Syst. Man Cybern. Part A Syst. Humans* **29**(3), 269–276 (1999).

https://doi.org/10.1017/S0263574719001590 Published online by Cambridge University Press

55. J. Wang, Q. Hu and D. Jiang, "A Lagrangian network for kinematic control of redundant robot manipulators," *IEEE Trans. Neural Networks* **10**(5), 1123–1132 (1999).
56. G. A. Bekey, "Robotics and Neural Networks," **In**: *Neural Networks for Signal Processing* (Prentice-Hall, Englewood, 1992) pp. 161–185.
57. Y. Li and N. Zeng, "A Neural Network Based Inverse Kinematics Solution in Robotics," **In**: *Neural Networks in Robotics* (Springer, Boston, MA, 1993).
58. S. Lee and R. M. Kil, "Redundant arm Kinematic Control with Recurrent Loop," *Proceedings of 32nd IEEE Conference on Decision and Control,* San Antonio, TX, USA, vol. 2 (1993) pp. 1109–1115.
59. Z. Mao and T. C. Hsia, "Obstacle avoidance inverse kinematics solution of redundant robots by neural networks," *Robotica* **15**(1), 3–10 (1997).
60. V. R. de Angulo and C. Torras, "Learning inverse kinematics: Reduced sampling through decomposition into virtual robots," *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **38**(6), 1571–1577 (2008).
61. C. M. Clark and J. K. Mills, "Robotic system sensitivity to neural network learning rate: Theory, simulation, and experiments," *Int. J. Rob. Res.* **19**(10), 955–968 (2000).
62. S. Alavandar and M. J. Nigam, "Neuro-Fuzzy based approach for inverse kinematics solution of industrial robot manipulators," *Int. J. Comput. Commun. Contr.* **3**(3), 224–234 (2008).
63. M. L. Husty, M. Pfurner and H. P. Schrocker, "A new and efficient algorithm for the inverse kinematics of a general serial 6R manipulator," *Mech. Mach. Theory* **42**(1), 66–81 (2007).
64. L. Sciavicco and B. Siciliano, *Modeling and Control of Robot Manipulators* (Springer-Verlag, London, UK, 2000). ISBN: 978-1-4471-0449-0.
65. Y. Zhang, Analysis and Design of Recurrent Neural Networks and their Applications to Control and Robotic Systems *PhD dissertation* (Chinese University of Hong Kong, 2003). ISBN: 0-493-98186-1.
66. J. Chen and H. Y. K. Lau, "A Reinforcement Motion Planning Strategy for Redundant Robot Arms Based on Hierarchical Clustering and K-Nearest-Neighbors," *2015 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Zhuhai (2015) pp. 727–732.
67. Y. Zhang, X. Lv, Z. Li, Z. Yang and K. Chen, "Repetitive motion planning of PA10 robot arm subject to joint physical limits and using LVI-based primal–dual neural network," *Mechatron.* **18**(9), 475–485 (2008).
68. E. Conkur, "Path following algorithm for highly redundant manipulator," *Rob. Auton. Syst.* **45**(1), 1–22 (2003).
69. Y. Zhang, J. Wang and Y. Xia, "A dual neural network for redundancy resolution of kinematically redundant manipulators subject to joint limits and joint velocity limits," *IEEE Trans. Neural Networks* **14**(3), 658–667 (2003).
70. O. Hock and J. Šedo, "Forward and inverse kinematics using pseudoinverse and transposition method for robotic arm DOBOT," *IntechOpen* (2017).
71. R. R. Ulrey, A. A. Maciejewski and H. J. Siegel, "Parallel Algorithms for Singular Value Decomposition," *Proceedings of 8th IEEE International Conference of Parallel Processing Symposium*, Cancun, Mexico (1994) pp. 524–533.
72. A. Liegeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *IEEE Trans. Syst. Man Cybern.* **7**(12), 868–871 (1977).
73. M. Goel, A. A. Maciejewski, V. Balakrishnan and R. W. Proctor, "Failure tolerant teleoperation of a kinematically redundant manipulator: An experimental study," *IEEE Trans. Syst. Man Cybern. Part B Syst. Humans* **33**(6), 758–765 (2003).
74. O. Khatib and A. Bowling, "Optimization of the Inertial and Acceleration Characteristics of Manipulators," *Proceedings of IEEE International Conference on Robotics and Automation*, Minneapolis, MN, USA, vol. 4 (1996) pp. 2883–2889.
75. A. De Luca, L. Lanari and G. Oriolo, "Control of Redundant Robots on Cyclic Trajectories," *Proceedings 1992 IEEE International Conference on Robotics and Automation,* Nice, France, vol. 1 (1992) pp. 500–506.
76. J. D. English and A. A. Maciejewski, "On the implementation of velocity control for kinematically redundant manipulators," *IEEE Trans. Syst. Man Cybern. Part A Syst. Humans* **30**(3), 233–237 (2000).
77. A. A. Maciejemski and C. A. Klein, "Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments," *Int. J. Rob. Res.* **4**(3), 109–117 (1985).
78. M. Pourazady and L. Ho, "Collision avoidance control of redundant manipulators," *Mech. Mach. Theory* **26**(6), 603–611 (1991).
79. X. Wang, C. Yang, J. Chen, H. Ma and F. Liu, "Obstacle avoidance for kinematically redundant robot," *IFAC-PapersOnLine* **48**(28), 490–495 (2015).
80. V. V. M. J. S. Chembuly, "An Optimization Based Inverse Kinematics of Redundant Robots Avoiding Obstacles and Singularities," *AIR '17 Proceedings of the Advances in Robotics*, New Delhi, India (2017).
81. F. Blanchini, G. Fenu, G. Giordano and F. A. Pellegrino, "A convex programming approach to the inverse kinematics problem for manipulators under constraints," *Eur. J. Contr.* **33**, 11–23 (2017).
82. M. Kelemen, I. Virgala, T. Lipták, L. Miková, F. Filakovský and V. Bulej, "A novel approach for a inverse kinematics solution of a redundant manipulator," *Appl. Sci.* **8**(11), 2229 (2018).
83. S. F. M. Assal, K. Watanabe and K. Izumi, "Neural network-based kinematic inversion of industrial redundant robots using cooperative fuzzy hint for the joint limits avoidance," *IEEE/ASME Trans. Mechatron.* **11**(5), 593–603 (2006).

84. B. Allotta, V. Colla and G. Bioli, "Kinematic control of robots with joint constraints," *ASME J. Dyn. Syst. Meas. Contr.* **121**(3), 433–442 (1999).
85. J. Wan, H. T. Wu, R. Ma and L. Zhang, "A study on avoiding joint limits for inverse kinematics of redundant manipulators using improved clamping weighted least-norm method," *J. Mech. Sci. Technol.* **32**(3), 1367–1378 (2018).
86. S. Miyata, S. Miyahara and D. Nenchev, "Analytical formula for the pseudoinverse and its application for singular path tracking with a class of redundant robotic limbs," *Adv. Rob.* **31**(10), 509–518 (2017).
87. H. Zhou and K.-L. Ting, "Path generation with singularity avoidance for five-bar slider-crank parallel manipulators," *Mech. Mach. Theory* **40**(3), 371–384 (2005).
88. L. Jin, B. Liao, M. Liu, L. Xiao, D. Guo and X. Yan, "Different-level simultaneous minimization scheme for fault tolerance of redundant manipulator aided with discrete-time recurrent neural network," *Frontiers in Neurorobotics* **11** (2017).
89. M. Hassan and L. Notash, "Optimizing fault tolerance to joint jam in the design of parallel robot manipulators," *Mech. Mach. Theory* **42**(10), 1401–1417 (2007).
90. K.-T. Oen and L.-C. T. Wang, "Optimal dynamic trajectory planning for linearly actuated platform type parallel manipulators having task space redundant degree of freedom," *Mech. Mach. Theory* **42**(6), 727–750 (2007).
91. C. A. Klein and C. H. Huang, "Review of pseudoinverse control for use with kinematically redundant manipulators," *IEEE Trans. Syst. Man Cybern.* (2), 245–250 (1983).
92. F.-T. Cheng, R.-J. Sheu, T.-H. Chen and F.-C. Kung, "The Improved Compact QP Method for Resolving Manipulator Redundancy," *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'94),* Munich, Germany, vol. 2 (1994) pp. 1368–1375.
93. A. S. Deo and I. D. Walker, "Minimum effort inverse kinematics for redundant manipulators", *IEEE Trans. Rob. Autom.* **13**(5), 767–775 (1997).
94. W. S. Tang and J. Wang, "Two recurrent neural networks for local joint torque optimization of kinematically redundant manipulators," *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **30**(1), 120–128 (2000).
95. A. Nedungadi and K. Kazerounian, "A local solution with global characteristics for joint torque optimization of a redundant manipulator," *J. Rob. Syst.* **6**(5), 631–654 (1989).
96. S. Ma, "A new formulation technique for local torque optimization of redundant manipulators," *IEEE Trans. Ind. Electron.* **43**(4), 462–468 (1996).
97. B. Liao and W. Liu, "Pseudoinverse-type bi-criteria minimization scheme for redundancy resolution of robot manipulators," *Robotica* **33**(10), 2100–2113 (2015).
98. F. Cheng, T. Chen and Y. Sun, "Resolving manipulator redundancy under inequality constraints," *IEEE Trans. Rob. Autom.* **10**(1), 65–71 (1994).
99. Y. Zhang, X. Lv, Z. Li and Z. Yang, "Repetitive Motion Planning of Redundant Robots Based on LVI-based Primal-dual Neural Network and PUMA560 Example," **In**: Life System Modeling and Simulation. LSMS 2007. *Lecture Notes in Computer Science* (Springer, Berlin, Heidelberg, 2007) pp. 536–545,
100. Y. Zhang, Z. Tan, K. Chen, Z. Yang and X. Lv, "Repetitive motion of redundant robots planned by three kinds of recurrent neural networks and illustrated with a four-link planar manipulator's straight-line example," *Rob. Auton. Syst.* **57**(6–7), 645–651 (2009).
101. T. Shamir and Y. Yomdin, "Repeatability of redundant manipulators: Mathematical solution of the problem," *IEEE Trans. Autom. Contr.* **33**(11), 1004–1009 (1988).
102. T. Shamir, "The singularities of redundant robot arms," *Int. J. Rob. Res.* **9**(1), 113–121 (1990).
103. R. G. Roberts and A. A. Maciejewski, "Nearest optimal repeatable control strategies for kinematically redundant manipulators," *IEEE Trans. Rob. Autom.* **8**(3), 327–337 (1992).
104. J. Baillieul, "Avoiding Obstacles and Resolving Kinematic Redundancy," *Proceedings 1986 IEEE International Conference on Robotics and Automation,* San Francisco, CA, USA (1986) pp. 1698–1704,
105. C. A. Klein, C. Chu-Jenq and S. Ahmed, "A new formulation of the extended Jacobian method and its use in mapping algorithmic singularities for kinematically redundant manipulators," *IEEE Trans. Rob. Autom.* **11**(1), 50–55 (1995).
106. C. A. Klein, C. Chu-Jenq and S. Ahmed, "Use of an Extended Jacobian Method to Map Algorithmic Singularities," *Proceedings of the IEEE International Conference on Robotics and Automation*, Atlanta, GA, USA, vol. 3 (1993) pp. 632–637.
107. Y. Zhang, S. S. Ge and T. H. Lee, "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.* **34**(5), 2126–2132 (2004).
108. Z. Hou, L. Cheng and M. Tan, "Multicriteria optimization for coordination of redundant robots using a dual neural network," *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **40**(4), 1075–1087 (2010).
109. Y. Zhang, "Towards Piecewise-linear Primal Neural Networks for Optimization and Redundant Robotic," *2006 IEEE International Conference on Networking, Sensing and Control*, Ft. Lauderdale, FL (2006) pp. 374–379.
110. Y. Zhang, H. Wu, Z. Zhang, L. Xiao and D. Guo, "Acceleration-level repetitive motion planning of redundant planar robots solved by a simplified LVI-based primal-dual neural network," *Rob. Comput. Integr. Manuf.* **29**(2), 328–343 (2013).
111. Y. Zhang, H. Wua, D. Guo, L. Xiao and H. Zhu, "The link and comparison between velocity-level and acceleration-level repetitive motion planning schemes verified via PA10 robot arm," *Mech. Mach. Theory* **69**, 245–262 (2013).

112. J. Hopfield and D. Tank, " "Neural" computation of decision in optimization problems," *Biol. Cybern.* **52**(3), 141–52 (1985).
113. Y. Xia and J. Wang, "A dual neural network for kinematic control of redundant robot manipulators," *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **31**(1), 147–154 (2001).
114. Y. Zhang and J. Wang, "A dual neural network for constrained joint torque optimization of kinematically redundant manipulators," *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **32**(5), 654–662 (2002).
115. W. S. Owen, E. A. Croft and B. Benhabib, "A multi-arm robotic system for optimal sculpting," *Rob. Comput. Integr. Manuf.* **24**(1), 92–104 (2008).
116. C. A. Klein and S. Ahmed, "Repeatable pseudoinverse control for planar kinematically redundant manipulators," *IEEE Trans. Syst. Man Cybern.* **25**(12), 1657–1662 (1995).
117. L. Huang, "An Extended form of Damped Pseudoinverse Control of Kinematically Redundant Manipulators," *1997 IEEE International Conference on Systems, Man, and Cybernetics Computational Cybernetics and Simulation*, Orlando, FL, USA, vol. 4 (1997) pp. 3791–3796.
118. C. A. Klein and K.-B. Kee, "The nature of drift in pseudoinverse control of kinematically redundant manipulators," *IEEE Trans. Rob. Autom.* **5**(2), 231–234 (1989).
119. I. A. Gravagne and I. D. Walker, "On the structure of minimum effort solutions with application to kinematic redundancy resolution," *IEEE Trans. Rob. Autom.* **16**(6), 855–862 (2000).
120. J. Lee, "A structured algorithm for minimum |∞-norm solutions and its application to a robot velocity workspace analysis," *Robotica* **19**(3), 343–352 (2001).
121. W. S. Tang and J. Wang, "A recurrent neural network for minimum infinity-norm kinematic control of redundant manipulators with an improved problem formulation and reduced architecture complexity," *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **31**(1), 98–105 (2001).
122. Y. Zhang, J. Wang and Y. Xu, "A dual neural network for Bi-Criteria Kinematic control of redundant manipulators," *IEEE Trans. Rob. Autom.* **18**(6), 923–931 (2002).
123. Y. Zhang and J. Wang, "A dual neural network for convex quadratic programming subject to linear equality and inequality constraints," *Phys. Lett. A* **298**(4), 271–278 (2002).
124. Y. Zhang, Z. Tan, Z. Yang, X. Lv and K. Chen, "A Simplified LVI-based Primal-dual Neural Network for Repetitive Motion Planning of PA10 Robot Manipulator Starting from Different Initial States," *2008 IEEE International Joint Conference on Neural Networks International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, Hong Kong (2008) pp. 19–24.
125. G. R. Pennock and C. C. Squires, "Velocity analysis of two 3-R robots manipulating a disk," *Mech. Mach. Theory* **33**(1–2), 71–86 (1998).
126. M. Yang, Y. Zhang, H. Huang, D. Chen and J. Li, "Jerk-level Cyclic Motion Planning and Control for Constrained Redundant Robot Manipulators using Zhang Dynamics: Theoretics," *2018 Chinese Control and Decision Conference (CCDC)*, Shenyang (2018) pp. 450–455.
127. S. Amaei, "An algorithm for solving indefinite quadratic programming problems," *J. Appl. Comput. Math.* **7**(1), (2018).
128. Optimization Toolbox for Use with MATLAB (Version 2.3, The MathWorks Inc., 2003).
129. W. Li and J. Swetits, "A new algorithm for solving strictly convex quadratic programs," *SIAM J. Optim.* **7**(3), 595–619 (1997).
130. M. S. Bazaraa, H. D. Sherali and C. M. Shetty, *Nonlinear Programming—Theory and Algorithms* (Wiley, New York, 1993).
131. T. V. Dang, T. Tran and K. V. Ling, "Numerical Algorithms for Quadratic Programming in Model Predictive Control - An Overview," *ISSAT MCSE'15,* Danang, Vietnam (2015).
132. L. Xiao, Y. Zhang, B. Liao, Z. Zhang, L. Ding and L. Jin, "A velocity-level Bi-Criteria optimization scheme for coordinated path tracking of dual robot manipulators using recurrent neural network," *Front. Neurorobo.* **11** (2017).
133. Y. Xia, "A new neural network for solving linear programming problems and its application," *IEEE Trans. Neural Networks* **7**(2), 525–529 (1996).
134. J. Wang and Y. Zhang, "Recurrent neural networks for real-time computation of inverse kinematics of redundant manipulators," **In**: *Machine Intelligence* (Quo Vadis, World Scientific, Singapore, 2004).
135. C. Mead, *Analog VLSI and Neural Systems*, 1st edition (Addison-Wesley VLSI Systems Series, Addison-Wesley, January 1989). ISBN: 978-0201059922.
136. Y. Zhang, Z. Li, H. Tan and Z. Fan, "On the Simplified LVI-based Primal-dual Neural Network for Solving LP and QP Problems," *2007 IEEE International Conference on Control and Automation*, Guangzhou (2007) pp. 3129–3134.
137. J. Wang, "Recurrent neural networks for computing pseudoinverses of rank-deficient matrices," *SIAM J. Sci. Comput.* **18**(5), 1479–1493 (1997).
138. I. B. Pyne, "Linear programming on an electronic analogue computer," *Trans. Am. Inst. Electr. Eng. Part I Commun. Electro.* **75**(2), 139–143 (1956).
139. D. W. Tank and J. J. Hopfield, "Simple 'neural' optimization networks: an A/D converter, signal decision circuit, and a linear programming circuit," *IEEE Trans. Circuits Syst.* **33**(5), 533–541 (1986).
140. S. S. Ge, T. H. Lee and C. J. Harris, *Adaptive Neural Network Control of Robotic Manipulators*, World Scientific Series in Robotics and Intelligent Systems (World Scientific, Singapore, 1998).
141. C. Diorio and R. P. N. Rao, "Neural circuits in silicon," *Nature* **405**, 891–892 (2000).
142. S. Wu, L. He, M. Zou, J. Li and Y. Zhang, "Time-varying Linear Programming via LVI-PDNN with Numerical Examples," *2016 IEEE 13th International Conference on Signal Processing (ICSP)*, Chengdu, China (2016).

143. H. Ding and S. K. Tso, "A fully neural-network-based planning scheme for torque minimization of redundant manipulators," *IEEE Trans. Ind. Electron.* **46**(1), 199–206 (1999).
144. J. Liu, Y. Wang, S. Ma and B. Li, "Shape Control of Hyper-redundant Modularized Manipulator Using Variable Structure Regular Polygon," *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sendai, vol. 4 (2004) pp. 3924–3929.
145. S. Zhang and A. Constantinides, "Lagrange programming neural networks," *IEEE Trans. Circuits Syst. II Analog Digital Signal Process.* **39**(7), 441–452 (1992).
146. W. Tang and J. Wang, "A Primal-dual Neural Network for Kinematic Control of Redundant Manipulators Subject to Joint Velocity Constraints," *ICONIP'99. ANZIIS'99 & ANNES'99 & ACNN'99. 6th International Conference on Neural Information Processing*, Perth, WA, Australia, vol. 2 (1999) pp. 801–806.
147. Y. Zhang, J. Yin and B. Cai, "Infinity-norm acceleration minimization of robotic redundant manipulators using the LVI-based primal-dual neural network," *Rob. Comput. Integr. Manuf.* **25**(2), 358–365 (2009).
148. Z. Zhang, Q. Zhou and W. Fan, "Neural-dynamic based synchronous-optimization scheme of dual redundant robot manipulators," *Front. Neurorob.* **12** (2018).
149. Y. Zhang and J. Wang, "Obstacle avoidance of kinematically redundant manipulators using a dual neural network," *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **34**(1), 752–759 (2004).
150. Y. Zhang, "Inverse-free computation for infinity-norm torque minimization of robot manipulators," *Mechatron.* **16**(3–4), 177–184 (2006).
151. B. He and H. Yang, "A neural network model for monotone linear asymmetric variational inequalities," *IEEE Trans. Neural Networks* **11**(1), 3–16 (2000).
152. M. C. Ferris and J. S. Pang, "Complementarity and Variational Problems: State of the Art," *Proceedings in Applied Mathematics*. Society for Industrial & Applied, March 1997. ISBN: 978-0898713916.
153. Y. Zhang, "On the LVI-based Primal–dual Neural Network for Solving Online Linear and Quadratic Programming Problems," *Proceedings of the 2005 American Control Conference*, Portland, OR, USA, vol. 2 (2005) pp. 1351–1356.
154. Y. Xia and J. Wang, "A recurrent neural network for solving linear projection equations," *Neural Netw.* **13**(3), 337–350 (2000).
155. D. Kinderlehrer and G. Stampacchia, *An Introduction to Variational Inequalities and their Applications*, vol. 88, 1st edition (SIAM, Philadelphia, 1980). ISBN: 9780080874043.
156. Y. Zhang, C. Yi and W. Ma, "Simulation and verification of Zhang neural network for online time-varying matrix inversion," *Simul. Modell. Pract. Theory* **17**(10), 1603–1617 (2009).