Robotica (2021), **39**, pp. 1897–1908
doi:10.1017/S0263574721000047

CAMBRIDGE UNIVERSITY PRESS

RESEARCH ARTICLE

An efficient self-motion scheme for redundant robot manipulators: a varying-gain neural self-motion approach

Pengchao Zhang[1,2], Xiaohui Ren[1,3] and Zhijun Zhang[1,4,5,6]*

[1]Key Laboratory of Industrial Automation of Shaanxi Province, Shaanxi University of Technology, Hanzhong, Shaanxi 723000, China, [2]School of Mechanical Engineering, Shaanxi University of Technology, Hanzhong, Shaanxi 723000, China, [3]School of Electrical Engineering, Shaanxi University of Technology, Hanzhong, Shaanxi 723000, China, [4]School of Automation Science and Engineering, South China University of Technology, Guangzhou 510640, China, [5]Guangdong Artificial Intelligence and Digital Economy Laboratory (Pazhou Lab), Guangzhou 510335, China and [6]School of Automation Science and Engineering, East China Jiaotong University, Nanchang 330052, China
*Corresponding author. Email: auzjzhang@scut.edu.cn

Received: 17 October 2019; **Revised:** 17 December 2020; **Accepted:** 8 January 2021; **First published online:** 14 May 2021

Keywords: redundant robot, self-motion, varying-gain recurrent neural network, zeroing neural network, convergence

Abstract

In order to achieve high efficient self-motion for a redundant robot manipulator, a novel quadratic programming and varying-gain recurrent neural network based varying-gain neural self-motion (VGN-SM) approach is proposed and developed. With VGN-SM, the convergence errors can be adaptively and efficiently converged to zero. For comparisons, a traditional fixed-parameter neural self-motion (FPN-SM) approach is also presented. Theoretical analysis shows that the proposed VGN-SM has higher accuracy than the traditional FPN-SM. Finally, comparative experiments between VGN-SM and FPN-SM are performed on a six degrees-of-freedom robot manipulator to verify the advantages of the novel VGN-SM.

1. Introduction

Recently, robot manipulators are widely applied in various fields [1, 2]. A manipulator is redundant when it has more degrees-of-freedom (DOFs) than that needs for accomplishing end-effector tasks [2]. For the nonredundant manipulator, its motion trajectory and joint configuration are unique and only determined by the end-effector primary task, and thus, the nonredundant manipulator does not have extra DOFs to complete secondary tasks [3], such as environmental obstacles avoidance [4, 5], physical limits avoidance [6, 7, 8, 9, 10], and optimizations of other performance indicators [11, 12, 13]. In contrast, the redundant manipulator can not only complete the end-effector primary task but also accomplish secondary tasks. They are widely used in various fields, especially in high-risk situations such as chemical industry [3] and smelting [3, 14, 15]. At present, many researchers conduct research on redundant manipulators and pay more attention to the motion planning and control of redundant manipulators.

The forward kinematic problem of the redundant manipulator is to determine the position of the end-effector by knowing the joint angles of the redundant manipulator. On the contrary, the inverse kinematic problem of the robot manipulator is to determine the joint angles through the known position of the end-effector [10, 16, 17, 18]. For the redundant manipulator, the feasible solutions to the inverse kinematic problem are usually infinite [3]. The optimization algorithm is to find an optimal solution from these infinite solutions. On the basic of accomplishing the end-effector primary task, the optimal solution is able to finish the end-effector secondary tasks.

The self-motion of a robot manipulator is to move the robot manipulator from one joint state to another joint state without moving the end-effector. When the robot manipulator begins to perform a new end-effector task, its initial joint state may not be the expected initial state. Therefore, it needs to

© The Author(s), 2021. Published by Cambridge University Press.

https://doi.org/10.1017/S0263574721000047 Published online by Cambridge University Press

move the joint state of the redundant manipulator to the expected state of the next task. In the periodic repeated movement for a redundant robot manipulator, the manipulator needs to start from the same initial state [19, 20, 21, 22]. However, due to the angle-drift problem of the redundant manipulator, the final state of the joint may be different from the initial state. In order to complete the periodic repetitive motion, the joint state of the robot manipulator needs to be adjusted at the end of each cycle. If the redundant manipulator does not reach the desired state, the accuracy may get worse, and in serious cases, the equipment may be destroyed. Of course, the self-motion of manipulators can be executed by manually adjusting each joint. However, manually adjusting method is obviously ineffective and not convenient and also waste of time [4]. Thereby, it is desirable and incentive to develop efficient self-motion resolution method for self-motion problem of redundant robot manipulators.

In the past few years, some self-motion schemes have been presented. In ref. [23], the concept of self-motion manifold was presented. Meanwhile, it is concluded that in the case of robots with revolute joints, the number of disjoint self-motion manifolds cannot exceed the maximum number of inverse-kinematic solutions of a non-redundant manipulator of the same class. In ref. [24], a scheme based on gradient projection method for determining the magnitude of the self-motion to optimize a performance criterion for a redundant manipulator was presented. In ref. [25], a new collision-avoidance method by controlling the self-motion of manipulator for redundant manipulator was presented. In ref. [26], the quadratic-optimization-based self-motion planning method with no target configuration was proposed. However, the robot manipulator joints can only move as close to the median joint angle as possible, instead of any expected state which satisfies the self-motion condition. In recent years, many experts and scholars have applied quadratic programming (QP) technology to the motion planning of robotic manipulator. In ref. [27], a self-motion planning scheme of redundant robot manipulators based on QP is proposed, at the same time, a primal dual neural network based on linear variational inequalities is proposed to solve the QP problem. However, this method will produce a nonzero initial velocity of angular joint, which cannot be realized in reality. In ref. [28], Zhang *et al.* proposed a varying-gain recurrent neural network (VG-RNN) method to solve time-varying convex QP problem, and it was successfully applied to the motion planning of the robot manipulator. Inspired by the VG-RNN, a new varying-gain neural self-motion (VGN-SM) approach for solving the self-motion problem is proposed in this paper.

The rest of the paper is organized as follows. In Section 2, the varying-gain neural self-motion (VGN-SM) approach is proposed, and the stability of VGN-SM approach is proved. In Section 3, computer simulations on a 6 DOFs manipulator are conducted. Section 4 gives a conclusion of the paper. The main contributions of this article are listed as follows:

(1) A novel VGN-SM approach is proposed and investigated to solve self-motion problem of the redundant robot manipulator. As far as the authors know, it is the first time to apply the varying-gain neural network to solve the self-motion problem.
(2) The theoretical analysis on convergence rate of the proposed VGN-SM approach is given in detail. It is proved that the proposed VGN-SM approach has higher convergence rate than that of the fixed-parameter neural self-motion (VPN-SM) approach.

2. Problem formulation and neural network

In this section, according to the previous research on redundant manipulators [2, 3], a novel VGN-SM approach is designed and analyzed. The schematic diagram of the whole manipulator control system with VGN-SM is shown in Fig. 1. At the same time, the FPN-SM approach is presented and analyzed for comparisons.

2.1. QP scheme formulation

The forward-kinematic equation of the robot manipulator is described as

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

https://doi.org/10.1017/S0263574721000047 Published online by Cambridge University Press

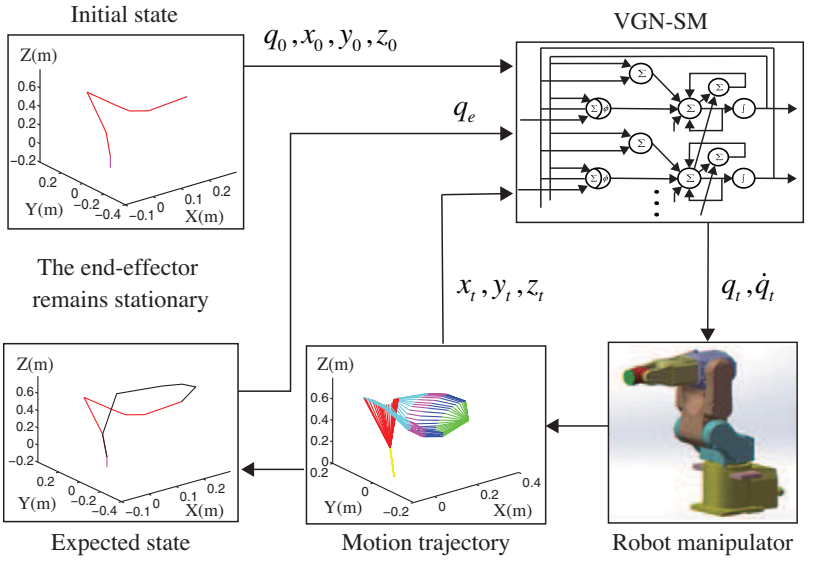


Figure 1. *Schematic diagram of the whole manipulator control system with VGN-SM.*

where $r(t) \in R^m$ and $\theta(t) \in R^n$ denote the end-effector position vector and joint angle vector, respectively. For the self-motion problem of a robot manipulator, $r(t)$ is a constant sequence vector; $f(\cdot)$ is a smooth nonlinear function, which can be obtained if the robot structure is given; n is the dimension of the robot joint space; and m is the dimension of end-effector Cartesian space. Because of $n > m$, for Eq. (1), it is difficult to get an analytical solution to $\theta(t)$ when $r(t)$ is given. Considering the self-motion constraint of the robot manipulator, the differential of Eq. (1) can be expressed as

$$J(\theta(t))\dot{\theta}(t) = 0, \tag{2}$$

where $\dot{\theta}(t) \in R^n$ denotes the joint velocity vector; $J(\theta)$ is the Jacobian matrix, which is defined as $J(\theta) = \partial f(\theta)/\partial\theta \in R^{m\times n}$. Considering the self-motion joint drift criterion $\eta(\theta(t) - \theta_e)$, a QP-based self-motion scheme can be defined as

$$\text{min.} \qquad \|(\dot{\theta}(t) + \eta(\theta(t) - \theta_e))\|_2^2/2, \tag{3}$$

$$\text{s.t.} \qquad J(\theta(t))\dot{\theta}(t) = 0, \tag{4}$$

where $\|\cdot\|_2$ denotes the two-norm of a vector; $\theta(t)$ denotes the actual joint angle vector of the redundant manipulator; θ_e denotes the expected joint state. $\eta > 0$ is a scalar coefficient which is designed to scale the magnitude of manipulators' response to the joint drift $\theta(t) - \theta_e$.

By defining $x = \dot{\theta}(t)$, $c = \eta(\theta(t) - \theta_e)$, Eq. (3) can be rewritten as $x^{\mathrm{T}}Wx/2 + c^{\mathrm{T}}x$ where superscript T denotes the transpose of a matrix/vector. W is designed as $W = I$, where $I \in R^{n\times n}$ is a unit matrix. Considering end-effector position feedback constraint $\zeta(r(t) - f(\theta))$ to Eq. (4) and $r(t) = r(0)$ for the self-motion problems, the self-motion scheme can be further reformulated as

$$\text{min.} \qquad x^{\mathrm{T}}Wx/2 + c^{\mathrm{T}}x, \tag{5}$$

$$\text{s.t.} \qquad Jx = \zeta(r(0) - f(\theta)), \tag{6}$$

where $r(0)$ denotes the end-effector position when $t = 0$; $f(\theta(t))$ denotes the actual position of the end-effector during the movement of the robot manipulator.

https://doi.org/10.1017/S0263574721000047 Published online by Cambridge University Press

2.2. Neural network solver

In order to obtain the optimal solution of self-motion scheme (5)–(6), according to the previous research [28], the following equivalent matrix equation can be obtained, that is,

$$G(t)y(t) = b(t), \tag{7}$$

where

$$G(t) = \begin{bmatrix} I & J^{\mathrm{T}}(t) \\ J(t) & 0_{m\times m} \end{bmatrix} \in R^{(n+m)\times(n+m)},\ y(t) = \begin{bmatrix} x(t) \\ \lambda(t) \end{bmatrix} \in R^{n+m},\ b(t) = \begin{bmatrix} -c(t) \\ e(t) \end{bmatrix} \in R^{n+m}$$

with $e(t) = \zeta(r(0) - f(\theta))$ and $\lambda(t) \in R^m$ is a vector of Lagrange-multiplier.

The optimal solution to self-motion scheme (5)–(6) consists of the first n elements of y. Therefore, solving the self-motion problem (5)–(6) is equivalent to solving matrix Eq. (7).

In order to obtain the optimal solution to Eq. (7), we define the following function,

$$\zeta(t) = G(t)y(t) - b(t) \in R^{n+m}. \tag{8}$$

Inspired by the design process of VG-RNN, the varying-gain recurrent neural-dynamic design formula can be written as

$$\frac{d\zeta(t)}{dt} = -\gamma \exp(t)\Phi(\zeta(t)) \tag{9}$$

where $\Phi(\cdot): R^{n+m} \to R^{n+m}$ denotes an activation-function processing-array; parameter $\gamma > 0$ is a constant scalar-valued designed parameter used to scale the convergence rate. It is worth pointing out that $\gamma\exp(t)$ will increase a lot even with a bit increase of t when $t > 0$. Besides, every scalar-valued processing-unit $\phi(\cdot)$ of array $\Phi(\cdot)$ should be a monotonically increasing odd function [28].

Substituting Eq. (8) into Eq. (9), the following VGN-SM approach can be obtained, that is,

$$G(t)\dot{y}(t) = -\dot{G}(t)y(t) - \gamma \exp(t)\Phi(G(t)y(t) - b(t)) + \dot{b}(t) \tag{10}$$

where $\dot{G}(t) = dG(t)/dt$, $\dot{y}(t) = dy(t)/dt$, $\dot{b}(t) = db(t)/dt$, $y(t) = [y_1(t), y_2(t), y_3(t) \cdots y_{n+m}(t)]^{\mathrm{T}} = [x_1(t), x_2(t), \cdots x_n(t), \lambda_1(t), \lambda_2(t) \cdots \lambda_m(t)]^{\mathrm{T}}$. According to the definition of $y(t)$, the first n elements of y are the optimal solution to (5)–(6). Equation (10) is called VGN-SM approach which is proposed in this paper to solve the self-motion problem (5)–(6). It is worth pointing out that the varying-gain recurrent neural-dynamic formula would be converted to fixed-parameter recurrent neural-dynamic formula if the time-varying parameter $\gamma\exp(t)$ is converted to γ [28], that is,

$$\frac{d\zeta(t)}{dt} = -\gamma\,\Phi(\zeta(t)). \tag{11}$$

Substituting Eq. (8) into Eq. (11), the following FPN-SM approach can be obtained,

$$G(t)\dot{y}(t) = -\dot{G}(t)y(t) - \gamma\,\Phi(G(t)y(t) - b(t)) + \dot{b}(t). \tag{12}$$

2.3. Convergence analysis

In this section, the convergence rate of VGN-SM approach is deduced. At the same time, the convergence rate of FPN-SM approach is also deduced for comparisons.

Lemma 1. *A smoothly time-varying strictly convex quadratic program can be described as follows:*

$$min. \qquad x^T(t)Hx(t)/2 + b^T x(t), \tag{13}$$

$$s.t. \qquad K(t)x(t) = c(t), \tag{14}$$

https://doi.org/10.1017/S0263574721000047 Published online by Cambridge University Press

Assume that $\Phi(\cdot)$ *is a monotonically increasing odd activation-function processing-array, the state variable* $y(t) = [x^T(t), \lambda^T(t)]^T$ *of VG-RNN* $(A(t)\dot{y}(t) = -\dot{A}(t)y(t) - \gamma exp(t)\Phi(A(t)y(t) - g(t)) + \dot{g}(t))$*, starting from any initial state* $y(0) = [x^T(0), \lambda^T(0)]^T \in R^{n+m}$*, globally converges to the unique theoretical solution* $y^*(t) = [x^{*T}(t), \lambda^{*T}(t)]^T \in R^{n+m}$*. In addition, the optimal solution* $x^*(t)$ *to the time-varying quadratic program (13)–(14) is the first n elements of* $y^*(t)$*.*

Proof. First, a Lyapunov-function candidate is defined as

$$v(t) = \frac{\|\zeta(t)\|_2^2}{2} = \frac{\zeta(t)^{\mathrm{T}}\zeta(t)}{2} \geq 0, \tag{15}$$

where $\zeta(t)$ denotes $\zeta(t) = A(t)y(t) - g(t)$, $\|\cdot\|_2$ denotes two norm of vector. Then the time derivative $\dot{v}(t)$ of (15) is

$$\dot{v}(t) = \frac{dv(t)}{dt} = \zeta^{\mathrm{T}}(t)\frac{d\zeta(t)}{dt}. \tag{16}$$

Second, substituting Eq. (9) into Eq. (16), we can obtain

$$\begin{aligned} \dot{v}_{\mathrm{VG}(t)} = \frac{dv(t)}{dt} = \zeta^{\mathrm{T}}(t)\frac{d\zeta(t)}{dt} &= -\gamma \exp(t)\zeta^{\mathrm{T}}(t)\Phi(\zeta(t)) \\ &= -\gamma \exp(t)\sum_{k=1}^{n+m}\zeta_k(t)\phi(\zeta_k(t)), \end{aligned} \tag{17}$$

where $\zeta_k(t)$ denotes the kth element of vector $\zeta(t)$. $\phi(t)$ denotes a processing unit of array $\Phi(\cdot)$. Since $\Phi(t)$ is a monotonically increasing odd activation-function processing-array, we can get

$$\zeta_k(t)\phi(\zeta_k(t)) \begin{cases} > 0, \text{ if } \zeta_k(t) > 0 \text{ or } \zeta_k(t) < 0, \\ = 0, \text{ if } \zeta_k(t) = 0. \end{cases} \tag{18}$$

From (18), we can see that $\dot{v}(t)$ is no-positive all the time $t \in [0, +\infty)$. According to Lyapunov theory, we can conclude that the VG-RNN is globally stable. The proof is completed. □

Lemma 2. *Consider smoothly time-varying strictly convex quadratic program problem (13)–(14), assume that VG-RNN* $(A(t)\dot{y}(t) = -\dot{A}(t)y(t) - \gamma exp(t)\Phi(A(t)y(t) - g(t)) + \dot{g}(t))$ *and FP-RNN* $(A(t)\dot{y}(t) = -\dot{A}(t)y(t) - \gamma\Phi(A(t)y(t) - g(t)) + \dot{g}(t))$ *have the same parameters and initial states, the upper bound of residual error and convergence time of VG-RNN are lower than those of FP-RNN when they are applied to solve time-varying quadratic program (13)–(14).*

Proof. First, substituting Eq. (11) into Eq. (16),

$$\begin{aligned} \dot{v}_{\mathrm{FP}(t)} = \frac{dv(t)}{dt} = \zeta^{\mathrm{T}}(t)\frac{d\zeta(t)}{dt} &= -\gamma\zeta^{\mathrm{T}}(t)\Phi(\zeta(t)) \\ &= -\gamma\sum_{k=1}^{n+m}\zeta_k(t)\phi(\zeta_k(t)), \end{aligned} \tag{19}$$

where $\zeta_k(t)$ denotes the kth element of vector $\zeta(t)$. $\phi(t)$ denotes a processing unit of array $\Phi(\cdot)$. Since $\Phi(t)$ is a monotonically increasing odd activation-function processing array, according to Lyapunov theory and Eq. (18), the FP-RNN approach is also convergent. Then, the following conclusions can be drawn by comparing the convergence rate with FP-RNN and the convergence rate of VG-RNN, that is,

$$\begin{cases} -\gamma\zeta^{\mathrm{T}}(t)\Phi(\zeta(t)) = -\gamma\exp(t)\zeta^{\mathrm{T}}(t)\Phi(\zeta(t)), \ t = 0, \\ -\gamma\zeta^{\mathrm{T}}(t)\Phi(\zeta(t)) < -\gamma\exp(t)\zeta^{\mathrm{T}}(t)\Phi(\zeta(t)), \ t > 0. \end{cases} \tag{20}$$

https://doi.org/10.1017/S0263574721000047 Published online by Cambridge University Press

Table 1. *Joint states data in experiments.*

Joint (rad)	$\boldsymbol{\theta}_{\mathbf{A}}$	$\boldsymbol{\theta}_{\mathbf{B}}$	$\boldsymbol{\theta}_{\mathbf{C}}$	$\boldsymbol{\theta}_{\mathbf{D}}$	$\boldsymbol{\theta}_{\mathbf{E}}$	$\boldsymbol{\theta}_{\mathbf{F}}$	$\boldsymbol{\theta}_{\mathbf{G}}$	$\boldsymbol{\theta}_{\mathbf{H}}$
θ_1	1.710	1.664	2.090	1.673	2.473	3.328	0.000	0.769
θ_2	2.862	3.426	3.249	1.342	1.491	1.773	0.000	0.600
θ_3	−3.162	−2.096	−1.909	−3.715	−3.020	−3.452	0.000	−0.024
θ_4	4.125	3.247	−1.005	4.187	−2.416	0.167	0.000	3.084
θ_5	−1.673	0.315	−0.505	−1.709	0.689	−0.650	0.000	−1.211
θ_6	−2.135	0.347	−0.177	1.307	1.186	0.036	0.000	−1.731

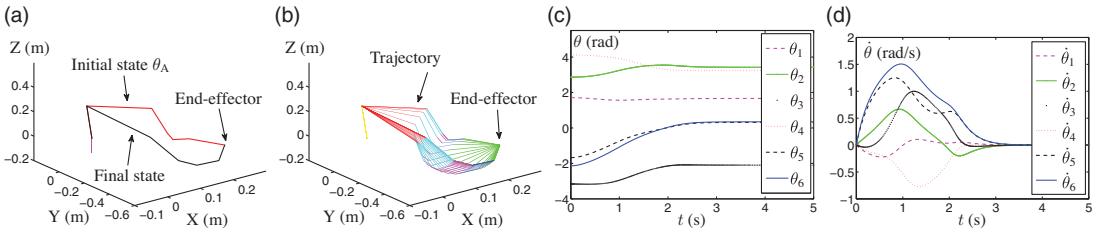


Figure 2. *Self-motion of robot manipulator from state θ_A to θ_B with $\gamma = 0.5$. (a) Initial state and final state. (b) The trajectory of redundant manipulator. (c) The joint-angle profiles. (d) The joint-velocity profiles.*

The above formula is equivalent to

$$\begin{cases} |\dot{v}_{\mathrm{VG}}| = |\dot{v}_{\mathrm{FP}}|, \ t = 0, \\ |\dot{v}_{\mathrm{VG}}| > |\dot{v}_{\mathrm{eFP}}|, \ t > 0. \end{cases} \tag{21}$$

From the above formulas, we can draw the conclusion that under the same conditions, VG-RNN approach converges faster than FP-RNN approach. The proof is completed. □

According to Lemmas 1 and 2, the proposed VGN-SM approach and FPN-SM approach are globally convergent. This means that through the VGN-SM and FPN-SM approaches, the redundant manipulator can move from any initial state $\theta(0)$ to any desired state θ_e which satisfies the self-motion condition $f(\theta(0)) = f(\theta_e)$. In addition, VGN-SM approach proposed in this paper is always stronger than FPN-SM approach no matter in accuracy or convergence rate.

3. Computer simulations

In order to demonstrate the effectiveness of the proposed VGN-SM approach, we perform the VGN-SM on a 6 DOFs robot manipulator and conduct three experiments. First, the robot manipulator is expected to move from the same initial joint state θ_{A} to different joint states θ_{B} and θ_{C}. Second, the robot manipulator is required to move from two different initial joint states, that is, θ_{D} and θ_{E} to the same expected joint state θ_{F}. Third, to illustrate the advantages of VGN-SM approach, the contrast experiments of VGN-SM approach and FPN-SM approach are carried out, in which the robot manipulator is expected to move from θ_{G} to θ_{H}. Joints of angles θ_{A}–θ_{H} used in the experiment are shown in Table 1.

3.1. From one initial state to two expected states

In this experiment, with the proposed VGN-SM approach, the robot manipulator is expected to move from the same initial joint state θ_{A} to different joint states θ_{B} and θ_{C} and keep the end-effector motionless.

https://doi.org/10.1017/S0263574721000047 Published online by Cambridge University Press

Table 2. *Joint values and errors of robot manipulator before and after movement from joint state θ_A to θ_B and θ_C.*

	Initial state	**Expected state $\boldsymbol{\theta}_\mathbf{B}$**			**Expected state $\boldsymbol{\theta}_\mathbf{C}$**		
Joint (rad)	$\boldsymbol{\theta}_\mathbf{A}$	$\boldsymbol{\theta}_\mathbf{B}$	$\boldsymbol{\theta}(\mathbf{t=5})$	$\boldsymbol{\theta}(\mathbf{t=5})-\boldsymbol{\theta}_\mathbf{B}$	$\boldsymbol{\theta}_\mathbf{C}$	$\boldsymbol{\theta}(\mathbf{t=5})$	$\boldsymbol{\theta}(\mathbf{t=5})-\boldsymbol{\theta}_\mathbf{C}$
θ_1	1.710	1.664	1.664	5.336×10^{-5}	2.090	2.090	1.658×10^{-4}
θ_2	2.862	3.426	3.426	2.001×10^{-4}	3.250	3.250	5.504×10^{-4}
θ_3	−3.162	−2.096	−2.096	1.996×10^{-6}	−1.909	−1.909	-2.217×10^{-4}
θ_4	4.125	3.247	3.247	3.321×10^{-5}	−1.005	−1.005	-8.049×10^{-5}
θ_5	−1.673	0.315	0.315	6.072×10^{-5}	−0.505	−0.505	-6.305×10^{-5}
θ_6	−2.135	0.347	0.347	3.168×10^{-5}	−0.177	−0.177	-2.835×10^{-5}

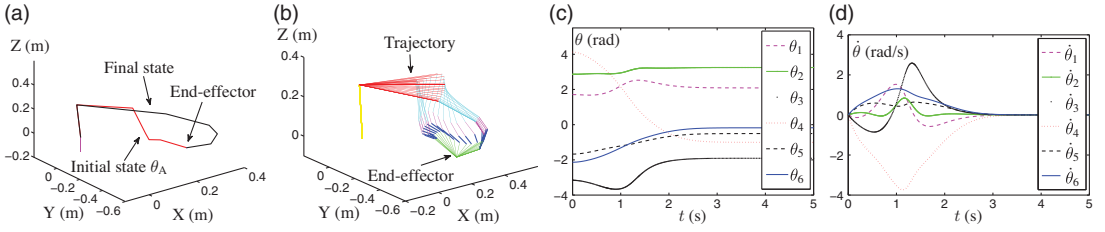


Figure 3. *Self-motion of robot manipulator from state θ_A to θ_C with $\gamma = 0.5$. (a) Initial state and final state. (b) The trajectory of redundant manipulator. (c) The joint-angle profiles. (d) The joint-velocity profiles.*

The task execution time is 5 s. The simulation results are shown in Figs. 2, 3 and Table 2. The initial and final states of the robot manipulator are shown in Figs. 2(a) and 3(a), in which the red line indicates the initial joint state of the robot manipulator, and the black line indicates the expected joint state. From Figs. 2(a) and 3(a), we can see that the end-effector remains stationary during the state transition process, and the manipulator finally moves to the expected state. That is to say, the task is completed, and the self-motion works well. Figures 2(b) and 3(b) illustrate the trajectories of the robot manipulator when it performs the self-motion task, the different colored lines indicate different joints. From the Figs. 2(b) and 3(b), we can see the robot manipulator successfully moves to the expected joint state from the initial joint state, at the same time, the end-effector is fixed. Figures 2(c)–(d) and 3(c)–(d) demonstrate the curves of joint angles profiles and joint velocity profile during the task execution process. As can be seen from Figs. 2(c)–(d) and 3(c)–(d), all the joint states converge to expected states and all of the joint velocities run back to 0. The moving time is about 3.2 s. All of these results verify that the proposed VGN-SM scheme is suitable for practical needs. Errors between the actual joint state and expected joint state are given in Table 2. Column 5 of Table 2 shows the joint drifts between the actual joint state $\theta(t=5)$ and expected joint state θ_B, in which most of joint drifts are very tiny, that is, less than 6.072×10^{-5} rad, joint drift of the third joint even achieves 1.996×10^{-6} rad. Columns 8 of Table 2 shows the joint drifts between the actual joint state $\theta(t=5)$ and expected joint state θ_C, and the joints drifts are also very tiny. These results further verify the effectiveness and accuracy of the proposed VGN-SM scheme.

3.2. From different initial states to one expected state

In this experiment, the robot manipulator is expected to move to the same joint state θ_F from the different initial joint states θ_D and θ_E with VGN-SM scheme. The simulation results are shown in Figs. 4, 5, and Table 3. Figures 4(a) and 5(a) show the initial and final states. As can be seen from Figs. 4(a) and 5(a), although the robot manipulator starts from different initial states, the robot manipulator

https://doi.org/10.1017/S0263574721000047 Published online by Cambridge University Press

Table 3. *Joint values and errors of robot manipulator before and after movement from joint states θ_D and θ_E to θ_F.*

Joint	Expected state	From initial state $\boldsymbol{\theta}_\mathbf{D}$			From initial state $\boldsymbol{\theta}_\mathbf{E}$		
(rad)	$\boldsymbol{\theta}_\mathbf{F}$	$\boldsymbol{\theta}_\mathbf{D}$	$\boldsymbol{\theta}(\mathbf{t}=5)$	$\boldsymbol{\theta}(\mathbf{t}=5)-\boldsymbol{\theta}_\mathbf{F}$	$\boldsymbol{\theta}_\mathbf{E}$	$\boldsymbol{\theta}(\mathbf{t}=5)$	$\boldsymbol{\theta}(\mathbf{t}=5)-\boldsymbol{\theta}_\mathbf{F}$
θ_1	3.328	1.673	3.327	-7.280×10^{-4}	2.473	3.327	-7.602×10^{-4}
θ_2	1.773	1.342	1.773	-4.481×10^{-4}	1.491	1.772	-6.341×10^{-4}
θ_3	−3.452	−3.715	−3.452	1.910×10^{-4}	−3.020	−3.452	3.483×10^{-4}
θ_4	0.167	4.187	0.168	5.259×10^{-4}	−2.416	0.168	7.223×10^{-4}
θ_5	−0.650	−1.709	−0.650	4.325×10^{-4}	0.680	−0.649	6.244×10^{-4}
θ_6	0.036	1.307	0.036	1.824×10^{-4}	1.186	0.036	2.596×10^{-4}

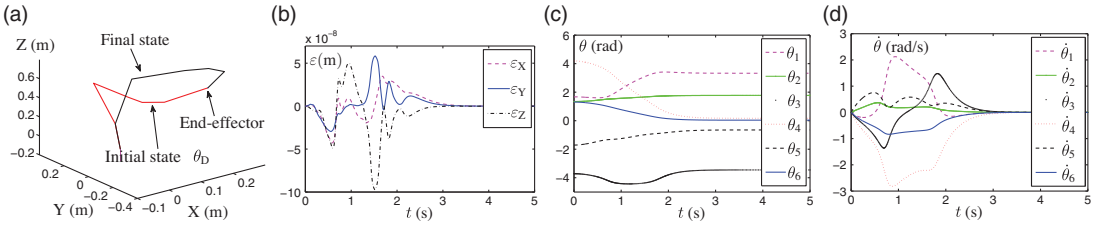


Figure 4. *Self-motion of robot manipulator from state θ_D to θ_F with $\gamma = 0.5$. (a) Initial state and final state. (b) The position error of end-effector. (c) The joint-angle profiles. (d) The joint velocity profiles.*

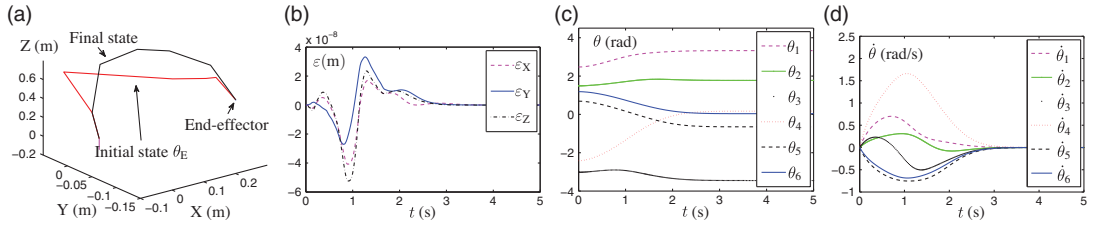


Figure 5. *Self-motion of robot manipulator from state θ_E to θ_F with $\gamma = 0.5$. (a) Initial state and final state. (b) The position error of end-effector. (c) The joint-angle profiles. (d) The joint velocity profiles.*

moves to the same expected state θ_F, and the end-effector remains motionless during the state transition process. These results further verify that the proposed VGN-SM scheme is effective for solving self-motion problems. Figures 4(b) and 5(b) show the position error of the end-effector. From Fig. 4(b), we can see that the maximum position error is less than 1×10^{-7} m, and in Fig. 5(b), the maximum position error is even less than 6×10^{-8} m. The most important point is that the final position error of the manipulator almost converges to 0. This validates that the proposed VGN-SM scheme has high accuracy. Figures 4(c)–(d) and 5(c)–(d) illustrate the curves of joint-angle profiles and joint-velocity profiles during the task execution process. From Figs. 4(c)–(d) and 5(c)–(d), we can find out that all the joint angles of the robot manipulator converge from initial state to the expected state, and all of the joint velocities run back to 0. It is worth pointing out that the initial velocity and the final velocity of the manipulator are both 0, which guarantees the feasibility of the proposed scheme. Errors between the actual joint state and expected joint state are given in Table 3. Similar to the previous Table 3, columns 5 of Table 3 shows the joint drifts between the actual joint state $\theta(t = 5)$ and expected joint state θ_F, columns 8 of Table 3 shows the joint drifts between the actual joint state $\theta(t = 5)$ and expected joint state θ_F. From Table 3, we can see that all the joint drifts are also very tiny. These results further verify the effectiveness and accuracy of the proposed VGN-SM scheme.

https://doi.org/10.1017/S0263574721000047 Published online by Cambridge University Press

https://doi.org/10.1017/S0263574721000047 Published online by Cambridge University Press

Table 4. *Results of comparative experiments.*

	$\gamma = 0.5$		$\gamma = 1$		$\gamma = 2$	
Indicators	**VGN-SM**	**FPN-SM**	**VGN-SM**	**FPN-SM**	**VGN-SM**	**FPN-SM**
Joint drift norm	1.222×10^{-3}	0.335	1.222×10^{-3}	0.031	1.222×10^{-3}	1.261×10^{-3}
Convergence time (s)	3.545	No convergence	2.990	No convergence	2.539	4.155
End-effector error norm (t = 5)	1.799×10^{-11}	2.271×10^{-8}	1.368×10^{-11}	8.572×10^{-9}	6.069×10^{-12}	9.934×10^{-10}

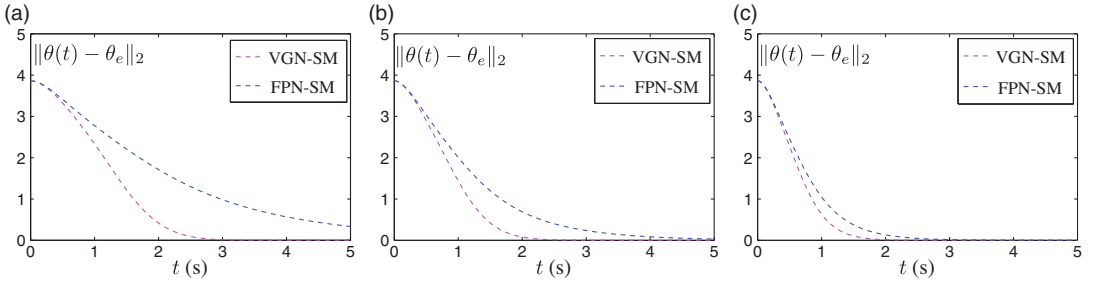


Figure 6. *The curves of $\|\theta(t)-\theta_e\|_2$ when VGN-SM approach and FPN-SM approach are used to solve self-motion problem with different parameter γ. (a) The curves of $\|\theta(t)-\theta_e\|_2$ with $\gamma=0.5$. (b) The curves of $\|\theta(t)-\theta_e\|_2$ with $\gamma=1$. (c) The curves of $\|\theta(t)-\theta_e\|_2$ with $\gamma=2$.*

3.3. Comparative experiments

According to the theoretical analysis in Section 2.3, the VGN-SM approach has faster convergence rate than FPN-SM approach, at the same time, the larger γ is set, the faster convergence rate is obtained. In this experiment, in order to verify the theoretical analysis results, we carried out a series of comparative experiments between VGN-SM approach and FPN-SM approach with different γ parameters. In the experiment, robot manipulator is required to move to θ_{H} from the initial state θ_{G}, and the experiment results are shown in Fig. 6 and Table 4.

Figure 6 shows the curves of $\|\theta(t)-\theta(e)\|_2$. From Fig. 6, we can see that as γ increases, convergence time of VGN-SM and FPN-SM would decrease. When γ is small, the FPN-SM scheme does not converge during the task time. The most important thing is that the convergence rate of VGN-SM is always greater than FPN-SM when using the same parameter. This is in line with theoretical results in Section 2.3. The convergence time shown in the fourth row of Table 4 also proves the above conclusion. Evidently, the higher convergence time means faster joint velocity, which is limited by the physical limits of the robot manipulator. Therefore, γ should be as small as possible under the condition of satisfying the precision requirement. Row 5 of Table 4 shows the norm of end-effector position errors when $t=5$. We can find out that the error decreases as γ increases and the error is very small, almost negligible, that is, less than 1.799×10^{-11}. At the same time, we can find that the error of VGN-SM is at least two orders of magnitude smaller than that of FPN-SM. The experiments fully verify that the convergence rate and accuracy of VGN-SM are better than those of FPN-SM.

4. Conclusion

In this paper, a varying-gain neural self-motion (VGN-SM) approach has been proposed, developed, and investigated to solve self-motion problems of redundant robot manipulators. A convergence theorem of the VGN-SM for solving self-motion has been given and proved in detail. Three task execution experiment results have been carried out and the experiments have verified the effectiveness and accuracy of the proposed VGN-SM scheme. Specifically, the joint drifts synthesized by the VGN-SM are very tiny, even achieve 1.996×10^{-6} rad. The end-effector position error even achieves 6.069×10^{-12}. For comparisons, the traditional fixed-parameter neural self-motion (FPN-SM) approach has also been presented. A series of comparative experiments between the VGN-SM and FPN-SM approaches performed on a redundant robot manipulator have verified that the proposed VGN-SM approach is more efficient and accurate for solving the self-motion problem. In this paper, we only use the minimum velocity norm indicator as the optimization target. In the future research, we will consider using multiple indicators (i.e., minimum velocity norm and infinity-norm velocity minimization) as the optimization target.

https://doi.org/10.1017/S0263574721000047 Published online by Cambridge University Press

Acknowledgments

This work was supported in part by the National Natural Science Foundation under Grants 61603142 and 61633010, the Guangdong Foundation for Distinguished Young Scholars under Grant 2017A030306009, the Guangdong Special Support Program under Grant 2017TQ04X475, the Science and Technology Program of Guangzhou under Grant 201707010225, the Fundamental Research Funds for Central Universities under Grant x2zdD2182410, the Key Science Research Program of Shaanxi Province under Grant 20JS022, and the Scientific Research Starting Foundation of South China University of Technology.

References

[1] A. Mohammadi, E. Rezapour, M. Maggiore and K. Y. Pettersen, "Maneuvering control of planar snake robots using virtual holonomic constraints," *IEEE Trans. Cont. Syst. Technol.* **24**(3), 884–899 (2016).
[2] Z. Zhang, Z. Yan and H. Yang, "A varying parameter recurrent neural network for solving nonrepetitive motion problems of redundant robot manipulators," *IEEE Trans. Cont. Syst. Technol.* **27**(6), 2680–2687 (2018).
[3] K. Li and Y. Zhang, "State adjustment of redundant robot manipulator based on quadratic programming," *Robotica* **30**(3), 477–489 (2012).
[4] G. S. Chyan and S. Ponnambalam, "Obstacle avoidance vontrol of redundant robots using variants of particle swarm optimization," *Robot. Comput.-Integr. Manufact.* **28**(2), 147–153 (2012).
[5] M. Jin, C. Zhou and Y. Liu, "Hybrid Impedance Control of 7-DOF Redundant Manipulator with Dual Compliant Surface," *IEEE International Conference on Mechatronics and Automation* (2015) pp. 1424–1429.
[6] D. Guo and Y. Zhang, "Acceleration-level inequality-based MAN scheme for obstacle avoidance of redundant robot manipulators," *IEEE Trans. Ind. Electron.* **61**(12), 6903–6914 (2014).
[7] E. Zergeroglu, D. D. Dawson, I. W. Walker and P. Setlur, "Nonlinear Tracking Control of Kinematically Redundant Robot Manipulators," *Proceedings of the 2000 American Control Conference, ACC* (2004) pp. 129–132.
[8] W. Xu, J. Zhang, B. Liang and B. Li, "Singularity snalysis and avoidance for robot manipulators with non-spherical wrists," *IEEE Trans. Ind. Electron.* **63**(1), 277–290 (2016).
[9] X. Li and C. C. Cheah, "Global task-space adaptive control of robot," *Automatica* **49**(1), 58–69 (2013).
[10] Z. Zhang, L. Zheng, J. Yu and Y. Li, "Three recurrent neural networks and three numerical methods for solving a repetitive motion planning scheme of redundant," *IEEE/ASME Trans. Mechatron.* **22**(3), 1423–1434 (2017).
[11] J. Hollerbach and K. Suh, "Redundancy resolution of manipulators through torque optimization," *IEEE J. Robot. Automat.* **3**(4), 308–316 (1987).
[12] K. Suh and J. Hollerbach, "Local Versus Global Torque Optimization of Redundant Manipulators," *IEEE International Conference on Robotics and Automation* (1987) pp. 619–624.
[13] 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 Cybernet.* **34**(5), 2126–2132 (2004).
[14] Y. Zhang, "On the LVI-based Primal-Dual Neural Network for Solving Online Linear and Quadratic Programming Problems," *Proceedings of the American Control Conference* (2005) pp. 1351–1356.
[15] J. P. Puga and L. E. Chiang, "Optimal trajectory planning for a redundant mobile manipulator with non-holonomic constraints performing push-Pull tasks," *Robotica* **26**(3), 385–394 (2008).
[16] O. Khatib and A. Bowling, "Optimization of the Inertial and Acceleration Characteristics of Manipulators," *Proceedings of IEEE International Conference on Robotics and Automation* (1996) pp. 2883–2889.
[17] C. A. Klein and C. H. Huang, "Review of pseudoinverse control for use with kinematically redundant manipulators," *IEEE Trans. Syst. Man Cybernet.* **SMC-13**(2), 245–250 (1983).
[18] R. Hussain, A. Qureshi, R. A. Mughal, R. Ijaz, N. Rashid, M. I. Tiwana and J. Iqbal "Inverse Kinematics Control of Redundant Planar Manipulator with Joint Constraints Using Numerical Method," *International Conference on Control, Automation and Systems* (2015) pp. 806–810.
[19] C. Qiu, Q. Cao and S. Miao, "An on-line task modification method for singularity avoidance of robot manipulators," *Robotica* **27**(4), 539–546 (2009).
[20] J. Wang, Q. Hu and D. Jiang, "A lagrangian network for kinematic control of redundant manipulators," *IEEE Trans. Neural Netw.* **10**(5), 1123–1132 (1999).
[21] 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).
[22] H. Ding and J. Wang, "Recurrent neural networks for minimum infinity-norm kinematic control of redundant manipulators," *IEEE Trans. Syst. Man Cybernet. A Syst. Hum.* **29**(3), 269–276 (1999).
[23] J. W. Burdick, "On the Tnverse Kinematics of Redundant Manipulators: Characterization of the Self-Motion Manifolds," *Proceedings of the 4th International Conference on Advanced Robotics Columbus* (1989) pp. 25–34.

https://doi.org/10.1017/S0263574721000047 Published online by Cambridge University Press

[24] J. A. Euler, R. V. Dubey and S. M. Babcock, "Self motion determination based on actuator velocity bounds for redundant manipulators," *J. Robot. Syst.* **6**(4), 417–425 (1989).
[25] N. Rahmanian-Shahri and I. Troch, "Collision-avoidance for redundant robots through control of the self-motion of the manipulator," *J. Intell. Robot. Syst.* **16**, 123–149 (1996).
[26] Y. Zhang, N. Tan, Z. Tan and J. Chen, "Quadratic-optimization based self-motion planning with no target-configuration assigned and verified via PA10 simulations," *J. Jinan Univ.* **31**(3), 229–234 (2010).
[27] Y. Zhang, H. Zhu, Z. Tan and B. Cai, "Self-Motion Planning of Redundant Robot Manipulators Based on Quadratic Program and Shown via PA10 Example," *International Symposium on Systems and Control in Aerospace and Astronautics* (2008).
[28] Z. Zhang, Y. Lu and L. Zheng, "A new varying-parameter convergent-differential neural-network for solving time-varying convex QP problem constrained by linear-equality," *IEEE Trans. Automat. Cont.* **63**(12), 4110–4125 (2018).

Cite this article: P. Zhang, X. Ren and Z. Zhang (2021). "An efficient self-motion scheme for redundant robot manipulators: a varying-gain neural self-motion approach", Robotica **39**, 1897–1908. https://doi.org/10.1017/S0263574721000047

https://doi.org/10.1017/S0263574721000047 Published online by Cambridge University Press