Infinity-Norm Torque Minimization for Redundant Manipulators Using a Recurrent Neural Network

Wai Sum Tang, Jun Wang and Yangsheng Xu
Department of Mechanical and Automation Engineering
The Chinese University of Hong Kong
Shatin, New Territories, Hong Kong, China
E-mail: {wstang, jwang, ysxu}@mae.cuhk.edu.hk

Abstract

A recurrent neural network is applied for minimizing the infinity-norm of joint torques in redundant manipulators. The recurrent neural network explicitly minimizes the maximum component of joint torques in magnitude while keeping the relation between the joint torque and the end-effector acceleration satisfied. The end-effector accelerations are given to the recurrent neural network as its input, and the minimum infinity-norm joint torques is generated at the same time as its output. It is shown that the recurrent neural network is capable of effectively generating the minimum infinity-norm joint torque redundancy resolution of manipulators.

1 Introduction

In many robotic operations, it is often desirable to optimize various performance criteria while accomplishing end-effector tasks. Kinematically redundant manipulators with more degrees of freedom than essential for a given task is a promising approach to performance operations. The extra degrees of freedom in redundant manipulators provide the self-motion which can be used to avoid joint limits, obstacles, singularity and to optimize various performance criteria. The applications of redundancy resolution in manipulators were summarized by Nenchev [1]. Among those performance criteria, the optimization of joint torques is an appealing one since it is equivalent to the minimization of actuator input power which in turn makes an effective utilization of input power to the actuator.

Much effort has been devoted to the torque optimization of redundant manipulators; e.g., [2]-[6]. In these studies, the majority of the researchers [2]-[5] proposed to minimize the 2-norm or the weighted 2-norm of joint torques. The minimization of 2-norm of joint torques minimizes the sum of squares of joint torque, which does not necessarily minimize the magnitudes of individual joint torques. It is used as the optimization criterion in many robotics applications not because it is physically desirable but because it is mathematically tractable [7]. The minimization of the infinity-norm of joint torques, however, minimizes the largest component in magnitude of joint torques, which evenly distributes the joint torque loading and is consistent with the physical limits [6]. Moreover, the minimization of infinity-norm of joint torques enables a better direct monitoring and control of the magnitude of individual joint torque than that of 2-norm of joint torques. It is more desirable in applications where low individual joint torque is of primary concern.

In recent years, recurrent neural networks have been applied to robotic kinematics and dynamics control; e.g., [8]-[13]. Specifically, Ding and Chan [10], and Tang and Wang [13] presented two schemes using recurrent neural networks for optimizing the 2-norm of joint torques of redundant manipulators. These research works showed neural networks are capable of real-time torque control of redundant manipulators. Along the line of traditional approach, Shim and Yoon [6] presented a numerical iteration method modified from the algorithm proposed by Cadzow [14] for computing the minimum infinity-norm torque solution in redundant manipulators. Because the time-varying nature of torque minimization, it is highly desirable in real operation to obtain the minimum torques in hundreds of microseconds or less. The parallel solution procedures such as neural network approach are therefore more effective and efficient compared to those iteration methods implemented in digital computers. This paper presents a new approach that uses a recurrent neural network to obtain this minimum infinity-norm torque solution in real-time.

This paper has six sections. The infinity-norm minimization problem is transformed to a linear program in section 2. Section 3 describes the dynamical equations of the proposed neural network and the neural network based closed-loop manipulator control process. The global convergence of the proposed neural network is proven in Section 4. Computer simulation results are shown and discussed in Section 5. Section 6 concludes
the paper with final remarks.

2 Problem Formulation

For a redundant manipulator, the position and orientation of the end-effector in the work space is related to the joint space by a forward kinematics equation:

$$ r = f(\theta), $$

where $r \in \mathbb{R}^m$ defines the position and orientation of the end-effector in the work space, $\theta \in \mathbb{R}^n$ ($m < n$) is the joint variables, and $f(\cdot)$ is a continuous nonlinear function whose structure and parameters are known for a given manipulator.

Differentiating (1) with respect to time gives the linear relation between the joint velocity $\dot{\theta}$ and the end-effector velocity $\dot{r}$:

$$ J(\theta)\dot{\theta} = \dot{r}, $$

where $J(\theta) \in \mathbb{R}^{m \times n}$ is the Jacobian matrix which is defined as:

$$ J(\theta) = \frac{\partial f(\theta)}{\partial \theta}. $$

Differentiating (2) with respect to time yields the relation between the joint acceleration $\ddot{\theta}$ and the end-effector acceleration $\ddot{r}$:

$$ J(\theta)\ddot{\theta} - \ddot{r} - J(\theta, \dot{\theta})\dot{\theta}, $$

where $J(\theta, \dot{\theta}) \in \mathbb{R}^{m \times n}$ is the derivative of the Jacobian matrix with respect to time. In a redundant manipulator, (2) and (1) are underdetermined since $m < n$.

In addition, the torque of robot manipulators can be expressed by using the well-known arm dynamics [2]:

$$ \tau = H(\theta)\dot{\theta} + c(\theta, \dot{\theta}) + g(\theta), $$

where $\tau \in \mathbb{R}^n$ denotes the joint torques, $\theta, \dot{\theta}, \ddot{\theta} \in \mathbb{R}^n$ are the joint variables, joint velocities and joint accelerations respectively; $H(\theta) \in \mathbb{R}^{n \times n}$ is the positive definite symmetric inertia matrix; and $c(\theta, \dot{\theta}) \in \mathbb{R}^n$ and $g(\theta) \in \mathbb{R}^n$ are components of the torque depending on Coriolis, centrifugal and gravity forces, respectively.

Equations (4) and (5) can be used to derive the relation between the joint torque and the end-effector acceleration [15]:

$$ J_{\tau} \tau = \ddot{r}, $$

where $\ddot{r} = J(\theta)H^{-1}(\dot{\theta})c(\theta, \dot{\theta}) + g(\theta) + \ddot{J}(\theta, \dot{\theta})\dot{\theta}$ and $J_{\tau} = J(\theta)H(\theta)^{-1}$.

Recall that the infinity-norm of a vector $x = [x_1, x_2, \ldots, x_n]^T \in \mathbb{R}^n$, with $T$ be the transpose operator and $| \cdot |$ denotes the absolute value of the component, is defined as:

$$ ||x||_\infty = \max_{1 \leq j \leq n} |x_j|, \quad (7) $$

$$ ||x||_\infty = \max_{1 \leq j \leq n} |e_j^T x|, \quad (8) $$

where $e_j \in \mathbb{R}^n$ is the $j$th column of the identity matrix.

The infinity-norm torque minimization of redundant manipulator is achieved by solving the following optimization problem:

$$ \text{minimize} \quad ||\tau||_\infty, $$

subject to $J_{\tau} \tau = \ddot{r}$. \quad (9)

The constrained optimization problem (9) can be transformed to a linear program. Let the objective function be

$$ v = ||\tau||_\infty = \max_{1 \leq j \leq n} |e_j^T \tau|. $$

The infinity-norm optimization problem (9) is then equivalent to:

$$ \text{minimize} \quad v, $$

subject to $|e_j^T \tau| \leq v$, \quad (11)

$$ J_{\tau} \tau = \ddot{r}. $$

The minimization of infinity norm of joint torques for redundant manipulators can thus be reformulated to the following linear program subject to inequality and equality constraints:

$$ \text{minimize} \quad v, $$

subject to $[I \quad I_n] \begin{bmatrix} \tau \\ u \end{bmatrix} \geq \begin{bmatrix} 0_n \\ 0_n \end{bmatrix}, \quad (12)$

$$ J_{\tau} \begin{bmatrix} \tau \\ u \end{bmatrix} = \ddot{r}, $$

where $I_n = (1, 1, \ldots, 1)^T \in \mathbb{R}^n$, $I \in \mathbb{R}^{n \times n}$ is the identity matrix; $0_n \in \mathbb{R}^n$ and $0_n \in \mathbb{R}^n$ are the zero vectors.

Rewrite (12) in a standard matrix form, we have

$$ \text{minimize} \quad \begin{bmatrix} c^T \\ 0 \end{bmatrix}, $$

subject to $A_1 \begin{bmatrix} y \\ v \end{bmatrix} \geq b_1, \quad (13)$

$$ A_2 \begin{bmatrix} y \\ v \end{bmatrix} = b_2, $$

where $A_1 = \begin{bmatrix} I \quad I_n \\ -I \quad 0_n \end{bmatrix} \in \mathbb{R}^{2n \times (n+1)}$, $b_1 = \begin{bmatrix} 0_n \\ 0_n \end{bmatrix} \in \mathbb{R}^{2n}$,

$A_2 = [J_{\tau} \quad 0_n] \in \mathbb{R}^{m \times (n+1)}$, $b_2 = \ddot{r}, \quad c = (0, 0, \ldots, 0, 1)^T \in \mathbb{R}^{n+1}, \quad y = \begin{bmatrix} \tau \\ v \end{bmatrix} \in \mathbb{R}^{n+1}$.

By the dual theory [16], the dual linear program to the primal linear program defined in (13) is

$$ \text{maximize} \quad b_2^T z_2, $$

subject to $A_1^T \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + A_2^T z_2 = c, \quad (14)$

$z_1 \geq 0$, \quad $z_2$ unrestricted,
where $z_1 \in \mathbb{R}^{2n}$ and $z_2 \in \mathbb{R}^m$ are dual decision variables.

The infinity-norm joint torque minimization enables the maximum joint torque to be locally minimized over the entire trajectory. Minimizing the infinity-norm of the joint torques offers a convenient way of directly monitoring the magnitude of each individual joint torque. Assume the actuator output torque for the $i$th joint be less than $\Delta_i$, for $i = 1, 2, \ldots, n$. If we define the objective function in (9) as:

$$\text{minimize } \|\Delta - \tau\|_\infty, \quad (15)$$

all the joint torques computed by the above approach will be within their specified limits.

### 3 Network Descriptions

Since Hopfield and Tank’s seminal work [17], many neural networks have been developed for solving optimization problems. In [18], Xia and Wang presented a general methodology for designing globally convergent optimization neural networks. In this study, we will follow the proposed methodology to develop a recurrent neural network which is globally convergent to the exact optimal solution of (13) and (14) to obtain the minimum infinity-norm torque solution of redundant manipulators.

In view of the objective functions and constraints in (13) and (14), we define the energy function:

$$E(y, z_1, z_2) = \frac{1}{2} (c^T y - b_1^T z_1)^2 + \frac{1}{2} ||A_1^T y - b_2||_2^2 + \frac{1}{2} ||A_1^T z_1 + A_2^T z_2 - c||_2^2 + \frac{1}{2} (A_1 y)^T (A_1 y - |A_1 y|)$$

$$+ \frac{1}{4} z_1^T (z_1 - |z_1|). \quad (16)$$

The first term in (16) is the squared duality gap; i.e., the squared difference between the objective functions of the primal and dual linear programs defined in (13) and (14), respectively. The second and third terms are for the equality constraints of (13) and (14), respectively. The fourth and last terms are for the non-negativity constraints of (13) and (14), respectively. It can be seen that $E(y^*, z_1^*, z_2^*) = 0$ if and only if $(y^*, z_1^*, z_2^*)$ is the optimal solution of the primal and dual linear programs defined in (13) and (14), respectively. Clearly, the energy function (16) is convex and continuously differentiable.

With the energy function defined in (16), we derive the dynamical equations for the neural network solving (13) and (14) as follows:

$$C_1 \frac{dy}{dt} = -\frac{\partial E}{\partial y}, \quad C_2 \frac{dz_1}{dt} = -\frac{\partial E}{\partial z_1}, \quad C_3 \frac{dz_2}{dt} = -\frac{\partial E}{\partial z_2}, \quad (17)$$

where $C_1 \in \mathbb{R}^{(n+1)\times(n+1)}$, $C_2 \in \mathbb{R}^{2n\times2n}$ and $C_3 \in \mathbb{R}^{m\times m}$ are positive diagonal capacitive matrices which are used to scale the convergence rate of the recurrent neural network. The convergence of the recurrent neural network can be expedited by using sufficiently small value of the capacitive matrices.

For any column vector $\xi$, $\|\xi\|_\infty = 2(\xi)^{-}$, where $(\xi)^{-} = (\xi_1^{-}, \xi_2^{-}, \ldots, \xi_n^{-})^T$ and $\xi_i^{-} = \min\{0, \xi_i\}$, the dynamical equations defined in (17) can be thus specifically expressed as:

$$C_1 \frac{dy}{dt} = -[c(c^T y - b_1^T z_2) + A_1^T (A_1 y)^- + A_2^T (A_2 y - b_2)], \quad (18)$$

$$C_2 \frac{dz_1}{dt} = -[(z_1)^- + A_1 (A_1^T z_1 + A_2^T z_2 - c)], \quad (19)$$

$$C_3 \frac{dz_2}{dt} = -[-b_2(c^T y - b_1^T z_2) + A_2 (A_1^T z_1 + A_2^T z_2 - c)]. \quad (20)$$

Figure 1 shows the architecture of the recurrent neural network. In order to ensure a good position tracking accuracy, the neural network is cooperated with the resolved acceleration control strategy [19]. Figure 2 delineates the block diagram for the neural network based resolved acceleration control process of manipulators. In this closed-loop control algorithm, the desired trajectory and the feedback law are expressed in terms of task variables:

$$\ddot{\tau} = \ddot{r}_d + K_v (\ddot{r}_d - \ddot{r}) + K_p (r_d - r), \quad (21)$$

where $r_d$, $\dot{r}_d$ and $\ddot{r}_d$ are the desired end-effector positions, velocities and accelerations, respectively, and $r$, $\dot{r}$ and $\ddot{r}$ are the actual end-effector positions, velocities and accelerations, respectively, and $K_p$ and $K_v$ are feedback gain matrices. In this context, these accelerations of the end-effector $\ddot{r}$ are fed into the neural network, and the neural network instantly generates the command signal $y$ which contains the minimum infinity-norm joint torque vector at the same time to drive the robot.

Let the position errors of the end-effector be $e = r_d - r$, then (21) can be reduced to:

$$\ddot{e} + K_v \dot{e} + K_p e = 0. \quad (22)$$

If the values of $K_v$ and $K_p$ are chosen such that the characteristic roots of (22) have negative real parts, the actual end-effector position $r$ approaches to the desired end-effector position $r_d$ asymptotically. Hence, a good tracking accuracy is achieved.
4 Stability Analysis

In this section, we prove the recurrent neural network is globally convergent to the exact optimal solution. The dynamical equations defined in (17) of the recurrent neural network can be rewritten as:

$$C \frac{du}{dt} = -\nabla E(u),$$

(23)

where $u = (y^T, z^T)^T$ and $z = (z_1^T, z_2^T)^T$; $C = \text{diag}(C_1, C_2, C_3)$ and $\nabla E(u)$ is the gradient of the energy function $E(u)$ defined in (16).

**Lemma 1**: Suppose that $f : D \subset \mathbb{R}^n \rightarrow \mathbb{R}^1$ is differentiable on a convex set $D_0 \subset D$. Then $f$ is convex on $D_0$ if and only if

$$(z - y)^T \nabla f(y) \leq f(z) - f(y), \quad \forall y, z \in D_0$$

(24)

where $\nabla f(y)$ is the gradient of $f(y)$.

**Proof**: See [16].

**Lemma 2**: The optimal solutions to the linear programs (13) and (14) are $y^*, z^*$, respectively, if and only if $E(u^*) = 0$. And

$$(u^* - u)^T \nabla E(u) \leq -E(u),$$

(25)

where $u^* = (y^{*T}, z^{*T})^T$ and $u = (y^T, z^T)^T$.

**Proof**: From the definition of the energy function (16), it can easily see that $E(u^*) = 0$ if and only if $u^*$ is the optimal solution of (13) and (14). Also, since $E(u) \geq 0$, continuously differentiable for all $u$ and convex, from Lemma 1 we have the conclusion of Lemma 2.

**Theorem**: The neural network defined in (23) is globally stable and convergent to the exact optimal solutions of the linear programs (13) and (14).

**Proof**: Without loss of generality, let $C_1, C_2$ and $C_3$ be identity matrices. Consider the Lyapunov function:

$$V(u) = \frac{1}{2} (u^* - u)^T (u^* - u),$$

(26)

where $u^* = (y^{*T}, z^{*T})^T$ and $y^*, z^*$ are the optimal solutions to the linear programs (13) and (14), respectively. Differentiating (26) with respective to time yields:

$$\frac{dV}{dt} = \frac{dV}{du} \left( \frac{du}{dt} \right) = (u^* - u)^T \frac{du}{dt}.$$  

(27)

From Lemma 2, we have

$$\frac{dV}{dt} = (u^* - u)^T \nabla E(u) \leq -E(u) \leq 0,$$

(28)

since $E(u) \geq 0$. Hence, the neural network defined in (23) is Lyapunov stable [20]. By LaSalle’s invariance principle [20], we know all trajectories $u(t)$ will converge to the largest invariant set in the set $S$:

$$S = \{ u \in \mathbb{R}^{n+m+1} \mid \dot{V} = 0 \},$$

From (27), $\dot{V} = 0$ implies $\dot{u} = 0$. Thus, the set $S$ is equal to the set of equilibrium points of system (23):

$$\{ u \in \mathbb{R}^{n+m+1} \mid V = 0 \} = \{ u \in \mathbb{R}^{n+m+1} \mid \dot{u} = 0 \}.$$

The neural network defined in (23) is therefore convergent to its equilibrium points.

From Lemma 2, $E(u^*) = 0$ if and only if $\nabla E(u^*) = 0$. Therefore, $u^*$ makes $\dot{u} = 0$. The equilibrium points of the system (23) are thus the optimal solutions to the linear programs (13) and (14).

Since $E(u)$ is continuously differentiable for all $u$ and convex, the local minimum is equivalent to the global minimum. The neural network defined in (23) is therefore globally stable and convergent to the optimal solutions of the linear programs (13) and (14). The proof is complete.

5 Simulation Results

Computer simulations are conducted to demonstrate the effectiveness of the proposed neural network approach to obtain the minimum infinity-norm torque
solution of redundant manipulators. The simulations are based on a three-link unit length and unit mass planar rotary manipulator as shown in Fig. 3. The simulation motions of the end-effector are straight-line Cartesian paths starting and ending with zero velocity and bang-bang type acceleration, with equal and constant deceleration and acceleration in z and y directions with magnitude of 0.5 m/s². The duration of motion is 2 seconds. The initial states of the manipulator are \( \theta(0) = [\frac{\pi}{6}, \frac{\pi}{6}, -\frac{\pi}{3}]^T \) rad and \( \dot{\theta} = [0, 0, 0]^T \) rad/s.

In this simulation, the differential equations (18), (19) and (20) are solved by using the fourth-order Runge-Kutta method implemented in MATLAB. The capacitive matrices in (18), (19) and (20) are set to be \( C_1 = 2 \times 10^{-6} I \), \( C_2 = 2 \times 10^{-6} I \) and \( C_3 = 2 \times 10^{-6} I \).

Figure 4 shows the joint torque transient behaviors against motion time computed by neural networks. The three subplots illustrate the joint torque profiles of each joint computed by the 2-norm minimization and the infinity-norm minimization criteria, respectively. The minimum 2-norm solutions are computed by the neural network presented in [13]. The simulation results show that joint 2 requires the maximum driving torque among the joints over the whole trajectory with 2-norm minimization. It is clear that the driving torque for joint 2 computed by infinity-norm minimization is lower than that computed by 2-norm minimization over the entire trajectory. It is also noted that the driving joint torques for joint 1 and joint 2 computed by the infinity-norm minimization criterion are almost the same from \( t = 0 \) s to \( t = 1 \) s in terms of magnitude. While from \( t = 1 \) s to \( t = 2 \) s, the driving joint torques for joint 1, joint 2 and joint 3 are virtually the same in terms of magnitude, and thus the joint loading is evenly shared by the joints during this period. Figure 5 shows the infinity-norm of the joint torque vectors computed by different optimization criteria.

Figure 6 depicts the transient behaviors of the energy function which is shown to converge to zero in less than 4 ms and being zero thereafter. Therefore, the proposed neural network is capable of effectively computing the minimum infinity-norm torque solution for redundant manipulators in real time.

6 Concluding Remarks

This paper presents a recurrent neural network for solving the minimum infinity-norm torque solution of redundant manipulators in real time. The infinity-norm
torque minimization is an attractive alternative in redundant manipulators when low individual joint torque is of primary concern. Unlike the numerical iteration methods, with the parallel distributed nature of neural computation, the proposed neural network approach provides efficient, real-time control for redundant manipulators which is desirable for time varying robotics operations. In comparison to the feedforward supervised neural networks, the approach eliminates the need for training, and is guaranteed asymptotically convergent to the exact optimal solution. With dedicated hardware, the present neural network approach can be realized for real-time torque control of redundant manipulators.

**REFERENCES**


