Abstract—This study considers the problem of controlling a redundant robot manipulator in the task space. We discuss the relation between the task space and joint space, and also introduce the knowledge of null space to derive the dynamic model of task and null space. Considering the model system of redundant robot manipulator that is a highly nonlinear dynamic model, with the interaction interference and uncertainty, we utilize the position and force controller design to overcome these problems by fuzzy neural network. Meanwhile, the stiffness coefficient of the environment is unknown. Therefore, the gradient descent method is used to estimate the stiffness coefficient of environment to achieve the adaptive force control. The stability analysis of the closed-loop system and the corresponding update laws are given by Lyapunov stability Theorem. Finally, we employ our proposed control scheme in the redundant robot manipulator KUKA LWR4 with 7-DOF to illustrate the performance and effectiveness.

I. INTRODUCTION

In recent years, with advances in mechanics, electronics, and computer engineering, the industrial automatic manipulators are wildly used to reduce the burden of labor. Some variable of the robot manipulators can deal with free space motion and interaction between the contact environment such as welding, polished, and cutting. That means the end effector of the robot manipulator can track the task trajectory in the free space and maintain the contact forces with the contact environment to achieve the desired force.

Many controller design methods were proposed to solve the tracking control of robot manipulator [1-20]. Literature [21] proposed a semi-global asymptotic stability analysis via Lyapunov theorem for new PID controller based on the fuzzy system for tuning PID controller gains to improve the performance. Several articles also proposed the intelligent controller. In addition, [8] presented an adaptive fuzzy logic sliding mode controller for the redundant robot manipulator. Thus, we here adopt FNN estimators to estimate the dynamic model of robot manipulator. Usually, the end effector of the robot manipulator is not only implemented the tracking trajectory but also prevent the over crash on the contact environment. In several controllers, the force control is the challenge. Therefore, the more common methods are the force/position control and the impedance control that are presented to deal with the external force control [1-3, 5, 7, 9, 11-13, 15, 22-24]. The hybrid control needs a switch strategy to transform the force and position control [1, 2, 11, 22]. In the path planning, literature [3] proposed the cubic spline to plan the trajectory, can make continuous in the junction of the different path types.

Among the various types of robot manipulators, the SCARA is wildly used in the industry. In addition, the six degree of freedom (DOF) robot manipulator, e.g., PUMA 560 can highly accuracy to implement the task trajectory, and also without dead space. Recently, non-redundant robot manipulators usually have joints of high stiffness design. It means the robot manipulator does not have enough flexibility, e.g., SCARA, PUMA 560. In order to implement more flexibility, the robot manipulator has more degree of freedom to operate in the six dimensional space for accuracy of position. The redundant robot manipulators have more DOF than general robot manipulators, therefore, redundant robot manipulator can provide additional control flexibility for complicated tasks and avoid the joint limit of a robot to implement better dynamics and kinematics. Despite the redundant robot manipulator offer some advantages and a nonlinear mapping from joint space to task space. However, this mapping is difficult to solve the problem at the joint level. That is, in order to overcome this problem, we converted the problem into velocity and acceleration level by using the pseudo-inverse of the Jacobian matrix of manipulator, and then solve the problem in the convert space. Herein, we consider the 7-DOF redundant robot manipulator KUKA LWR for simulation.

In this study, the mapping of the joint space to the task space cannot clearly to illustrate all the dynamic behavior in the redundancy case. Thus, consider the null space and the relation between the task space and joint space to derive the dynamic model of the redundant robot manipulator. However, the dynamic model of the redundant robot manipulator is nonlinear and assumed to be unknown or unavailable [3, 9, 10, 14, 25-28]. Therefore, we propose the FNN estimators to estimate the matrices of the dynamic model. In the force control scheme, the update laws of the stiffness coefficient is estimated by gradient method.

The rest of this study is organized as follows. In Section II, the dynamic model of the redundant robot manipulator with null space in introduced. Section III introduces the adaptive force control of redundant robot manipulator via FNN scheme. In the Section VI, the simulations of the KUKA LWR the redundant robot manipulator are introduced to illustrate the effectiveness of our controller design. Finally, the conclusion is given.
II. Dynamic Model of n-link Redundant Robot Manipulator

A. Dynamic Model of n-link Redundant Robot Manipulator

The dynamic models of redundant robot manipulator are discussed. Among the robot manipulator have joints of high stiffness design. It means the robot manipulator does not have enough flexibility. Thus, we consider the redundant robot manipulator having the degree of redundancy. Since the properties of the redundancy, the mapping of the joint space to the task space cannot clearly illustrate all the dynamic behavior in the redundancy case. Therefore, we convert the problem into velocity and acceleration level by using the pseudo-inverse of the Jacobian matrix of manipulator to solve problem. In conclusion, we consider the null space and the relation between the task space and joint space to derive the dynamic model of the redundant robot manipulator.

The dynamic model of a degree of freedom (DOF) revolute joint robot manipulator is described as

\[
M(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q) + \tau_m = \tau
\]

where \(q, \dot{q}, \ddot{q} \in \mathbb{R}^n\) are the joint position, velocity, and acceleration vectors, respectively. \(M(q) \in \mathbb{R}^{n \times n}\) is the symmetric bounded positive definite inertia matrix. \(C(q, \dot{q})\dot{q} \in \mathbb{R}^n\) is the vector of centrifugal and Coriolis torque. \(g(q) \in \mathbb{R}^n\) is the vector of gravitational torque. Furthermore, \(\tau \in \mathbb{R}^n\) is the vector of control torques and \(\tau_m \in \mathbb{R}^n\) is the vector of external torques resulting from the interaction with the environment. If the manipulator is equipped with sensors in its joints or on the interaction points, external torques can be directly measured.

When we consider the motion and force control of manipulator, it is better to design control strategy in the task directly measured. In its joints or on the interaction points, external torques can be directly measured.

This means that the joint torque of null space will not clearly be described the end effector position and orientation and completely specify the configuration of the end effector. Therefore, we convert the null space motion. Therefore, the projection of the null space null space. For the general form, we can construct the null space for redundant case, the decomposed analysis shows that there is an infinity of elementary displacements without altering the configuration of the end effector. There is also affected by null space. For the general form, we can construct the relationship between the end effector forces and joint torques

\[
\tau = J^\dagger(q)F_{\text{ext}}
\]

However, this relationship is not completely for redundant case, the decomposed analysis shows that there is an infinity of elementary displacements without altering the configuration of the end effector. Thus, we define the external interaction force \(F_{\text{ext}} \in \mathbb{R}^n\). i.e.,

\[
\tau = J^\dagger(q)F_{\text{ext}} + [1 - J^\dagger(q)J(q)]\zeta
\]

where \(\zeta\) is an arbitrary generalized joint torque vector projected in the null space of \(J^\dagger\). Then, we can rewrite (1) as

\[
M(q)\ddot{q} + C(q, \dot{q})\dot{q} + g(q) + \tau_m = J^\dagger(q)F_{\text{ext}} + [1 - J^\dagger(q)J(q)]\zeta
\]

Considering the relationship between task space and null space, multiply the \(JM^{-1}\), we can obtain

\[
p_\tau + (JM^{-1}Cq - Jq) + JM^{-1}g + \tau_m = (JM^{-1}J^\dagger)F_{\text{ext}} + JM^{-1}(1 - J^\dagger J^\dagger)\zeta
\]

This means that the joint torque of null space will not generate any command acceleration, i.e.,

\[
[JM^{-1} - (JM^{-1}J^\dagger)J^\dagger]\zeta = 0
\]

According to the above description, we can obtain the dynamic consistency matrix \(J^\dagger = JM^{-1}(JM^{-1}J^\dagger)^{-1}\). For a general solution, the joint velocities can be rewritten as

\[
\dot{q} = J^\dagger p_\tau + (1 - J^\dagger J)^\dagger \dot{q}_S
\]

where \(J^\dagger = JM^{-1}(JM^{-1}J^\dagger)^{-1}\) is the dynamically pseudo-inverse matrix, \(N^\dagger = I - J^\dagger J\), \(N^\dagger \in \mathbb{R}^{n \times n}\) is the matrix which projects \(q\) to the null space of \(J\). i.e., \(N^\dagger q = 0\), \(q_S \in \mathbb{R}^{n\text{null}}\) is the generic joint velocities vector. Based on the above (7), general solution can be obtained as

\[
q = q_\tau + q_S = J^\dagger p_\tau + (1 - J^\dagger J)q_S
\]

The particular solution \(q_\tau\) is related with the task space motion and the homogeneous solution \(q_S\) is the projected velocities vectors in the null space related with the null space motion. However, the homogeneous solution is not a minimal set to specify the null space motion. Therefore, the \(r\)-dimensional vectors are necessary to specify the null space of \(J\). \(q_\tau\) is in the row space of Jacobian matrix \(r(\tau^\dagger)\) range, and \(q_S\) is in the null space of Jacobian matrix \(N(\tau^\dagger)\) range. Since \(r(\tau^\dagger)\) and \(N(\tau)\) are Orthogonal complement space, therefore, \(q_\tau\) and \(q_S\) are Orthogonal vectors. Considering
the linear algebra theorem, we know the general solution \( \mathbf{q}^* \) is in the null space which constructs the null space of the Jacobian matrices. We define a set of smooth and linearly independent vectors \( \mathbf{Z}(\mathbf{q}) \in \mathbb{R}^{\nu \times \nu} \) satisfies \( \mathbf{J}^T \mathbf{q}^* = 0 \), therefore, the joint motion of null space will not affect the end effector motion called self-motion. \((1 - \mathbf{J}^T \mathbf{J}) \mathbf{q}_n^* = \) is the projected velocities matrices of the null space, that is, the realization of the second task can be achieved by choosing the different velocities of null space.

We can rewrite the joint velocities (2) as
\[
\dot{\mathbf{q}} = \mathbf{J}^T(\mathbf{q}) \mathbf{p} + \mathbf{J}^T(\mathbf{q}) \mathbf{p}_v + \mathbf{Z}(\mathbf{q}) \mathbf{v}.
\]

and obtain
\[
\dot{\mathbf{q}} = \mathbf{J}^T(\mathbf{q}) \mathbf{p} + \mathbf{J}^T(\mathbf{q}) \mathbf{p}_v + \mathbf{Z}(\mathbf{q}) \mathbf{v}.
\]

As above description, we can define the joint velocities and acceleration through the null space velocity in the redundant model of the task space and null space, in next section.

And then, model (1) is multiplied by \( \mathbf{J}^T(\mathbf{q}) \) and combining the task space and null space as
\[
\mathbf{J}^T \mathbf{M} \dot{\mathbf{q}} + \mathbf{J}^T \mathbf{C} \mathbf{q} + \mathbf{J}^T \mathbf{g} + \mathbf{J}^T \mathbf{\tau}_{na} = \mathbf{J}^T \mathbf{\tau}.
\]

Substituting (9) and (10) into (11), we obtain
\[
\mathbf{J}^T \mathbf{M} (\mathbf{J}^T \mathbf{p}_v + \mathbf{Z} \mathbf{v}) + \mathbf{J}^T \mathbf{C} (\mathbf{J}^T \mathbf{p}_v + \mathbf{Z} \mathbf{v}) + \mathbf{J}^T \mathbf{g} + \mathbf{J}^T \mathbf{\tau}_{na} = \mathbf{J}^T \mathbf{\tau}.
\]

That is, system (1) can be transferred to the task space and null space as
\[
\begin{bmatrix} \mathbf{A} & 0 \\ 0 & \mathbf{A} \end{bmatrix} \begin{bmatrix} \dot{\mathbf{p}} \\ \dot{\mathbf{q}} \end{bmatrix} + \begin{bmatrix} \mathbf{B}_n & \mathbf{B}_v \\ \mathbf{B}_n & \mathbf{B}_v \end{bmatrix} \begin{bmatrix} \mathbf{p} \\ \mathbf{v} \end{bmatrix} + \begin{bmatrix} \mathbf{G}_n \\ \mathbf{G}_v \end{bmatrix} \begin{bmatrix} \mathbf{e} \\ \mathbf{v} \end{bmatrix} = \begin{bmatrix} \mathbf{F}_n \\ \mathbf{F}_v \end{bmatrix}.
\]

where \( \mathbf{A} = \mathbf{A}_n \mathbf{R}_n \), \( \mathbf{B}_n = \mathbf{A}_n \mathbf{R}_n + \mathbf{B}_v \mathbf{R}_n \), \( \mathbf{G}_n = \mathbf{J}^T \mathbf{g} \) are the system parameters of task space model. \( \mathbf{B}_v = \mathbf{B}_v \mathbf{R}_n \), \( \mathbf{G}_v = \mathbf{Z} \mathbf{g} \) are the system parameters of the null space model.

In this study, we assume the task space and null space dynamic models of the redundant robot manipulator are complicated and unknown [3, 9, 10, 14, 25-27, 32], the actual parameters also have estimated error at the measure process. In order to solve these problems, we utilize the fuzzy neural networks (FNN) estimators to treat them. That is, this study utilizes the FNN method to estimate the dynamic model of the redundant robot manipulator and design the adaptive force controller, and then, the end effector can achieve our desired trajectory in task space and tracks the desired force for contact space even if the dynamic model is uncertain and in unknown environment.

B. Fuzzy Neural Networks

FNN is an implementation of fuzzy logic system with trainable parameters which is similar to an artificial neural network. It can acquire the linguistic information from human experts as well as adapt itself by using training data to achieve better performance [1, 2, 33]. According to the above-mentioned description, we can estimate the unknown dynamic model of the redundant robot manipulator by FNN. As we consider the inference engine rules of this single-output, the \( j \)th fuzzy rules for estimators is described as

\[
\text{IF } x_i \text{ is } A_{ij} \text{ and } \ldots \text{ and } x_n \text{ is } A_{nj}, \text{ THEN } y \text{ is } \omega_j
\]

where \( x_i \) is the input linguistic variable, \( A_{ij} \) is the fuzzy term represented by Gaussian function, and \( \omega_j \) is the consequent part for output \( y \).

Herein, the construction of each layer is given as follows. The first layer is the input layer that only transmits input variable to the next layer. The second layer acts as a fuzzification operation, which consists of Gaussian membership of the input layer:
\[
O_j^{(1)}(t) = \text{exp} \left[ \frac{- (x_i(t) - m_j)^2}{\sigma_j^2} \right]
\]

where \( m_j \) and \( \sigma_j \) are the center and width of the Gaussian functions. The third layer is a rule layer which consists of a table of products of the membership layer, and the \( t \)-norm product is adopted, that is:
\[
O_j^{(2)}(t) = \prod_{j=1}^{R} O_j^{(1)}(t)
\]

The final layer is defuzzification and the output is
\[
y = O^{(2)}(t) = w^T \psi_j = \sum_{j=1}^{R} \sum_{j=1}^{R} O_j^{(2)}(t)
\]

where \( w = [w_1, w_2, \ldots, w_R]^T \) is the weighting vector and \( R \) is the number of rules.

III. ADAPTIVE FORCE CONTROLLER DESIGN

This section introduces about the adaptive force controller design by the fuzzy neural network (FNN). There is three steps as follow. First, we utilize FNN to estimate the uncertainties of the robot dynamic model. Second, we design the controller to track the desired trajectory. Third, considering the end effector contact the unknown environment, we design the adaptive force control.

A. Adaptive Tracking Control

The knowledge of the dynamic model matrices \( \mathbf{A}_n, \mathbf{B}_n, \mathbf{B}_v, \mathbf{G}_n, \) and \( \mathbf{G}_v \) are not exactly known, we here estimate them by the FNNs. At first, we define \( \mathbf{e} = x - x_d \) and \( s = \dot{x} + k \mathbf{e} \) where \( k \) is the \( \mathbb{R}^{\nu \times \nu} \) positive definite diagonal matrix, then, we consider the dynamic model (13), we have
\[
\begin{bmatrix} \mathbf{A}_n & 0 \\ 0 & \mathbf{A}_n \end{bmatrix} \begin{bmatrix} \dot{\mathbf{p}} \\ \dot{\mathbf{q}} \end{bmatrix} + \begin{bmatrix} \mathbf{B}_n & \mathbf{B}_v \\ \mathbf{B}_n & \mathbf{B}_v \end{bmatrix} \begin{bmatrix} \mathbf{p} \\ \mathbf{v} \end{bmatrix} + \begin{bmatrix} \mathbf{G}_n \\ \mathbf{G}_v \end{bmatrix} \begin{bmatrix} \mathbf{e} \\ \mathbf{v} \end{bmatrix} = \begin{bmatrix} \mathbf{F}_n \\ \mathbf{F}_v \end{bmatrix}.
\]

Defining \( \mathbf{\hat{A}}_n = \hat{\mathbf{w}}_n \mathbf{\psi}_A, \mathbf{\hat{B}}_n = \hat{\mathbf{w}}_n \mathbf{\psi}_B, \mathbf{\hat{B}}_v = \hat{\mathbf{w}}_n \mathbf{\psi}_B, \mathbf{\hat{G}}_n = \hat{\mathbf{w}}_n \mathbf{\psi}_G, \mathbf{\hat{G}}_v = \hat{\mathbf{w}}_n \mathbf{\psi}_G, \) where \( \mathbf{\hat{A}}_n, \mathbf{\hat{B}}_n, \mathbf{\hat{B}}_v, \mathbf{\hat{G}}_n, \) and \( \mathbf{\hat{G}}_v \) are the estimation of \( \mathbf{A}_n, \mathbf{B}_n, \mathbf{B}_v, \mathbf{G}_n, \) and \( \mathbf{G}_v \), respectively, \( \mathbf{\hat{W}}_n \) are the weighting vectors of the FNN, and \( \mathbf{\psi} \) are the firing strength of the corresponding fuzzy rules.

We introduce the adaptive controller as follows. \[ \text{Theorem 1: Consider the redundant robot manipulator (1), the adaptive tracking controller can be designed as} \]
\[
\begin{bmatrix} \mathbf{F}_n \\ \mathbf{F}_n \end{bmatrix} = \mathbf{\hat{A}}_n (\hat{x}_n - k \mathbf{e}) + \mathbf{\hat{B}}_n (\hat{x}_n - k \mathbf{e}) + \mathbf{\hat{G}}_n - k_n \mathbf{v} + \mathbf{\hat{F}}_{\text{robust}}
\]

\[
\begin{bmatrix} \mathbf{F}_n \\ \mathbf{F}_n \end{bmatrix} = \mathbf{\hat{B}}_n (\hat{x}_n - k \mathbf{e}) + \mathbf{\hat{G}}_n - k_n \mathbf{v} + \mathbf{\hat{F}}_{\text{robust}}
\]
\[ F_{\text{error}} = -\delta \left( \text{sgn}(s)(\dot{x}_d - \dot{K}_e) - \delta \text{sgn}(s)(\dot{x}_d - \dot{K}_e) - \delta \text{sgn}(s) \right) \]

\[ F_{\text{error}} = -\delta \left( \text{sgn}(s)(\dot{x}_d - \dot{K}_e) - \delta \text{sgn}(s) \right) \]

and the update laws of the FNNs can be chosen as

\[ \dot{w}_A = -Q_A \psi_A (\dot{x}_e - \dot{K}_e) s^T \]
\[ \dot{w}_B = -Q_B \psi_B (\dot{x}_e - \dot{K}_e) s^T \]
\[ \dot{w}_C = -Q_C \psi_C (\dot{x}_e - \dot{K}_e) s^T \]

where \( Q_A, Q_B, Q_C \), and \( Q_C \) are diagonal positive definite matrices, \( \gamma \) is positive adaptive parameter rate and \( s = \dot{e} + k \dot{e} \). The asymptotically convergence of tracking error is certified by the Lyapunov stability theorem, i.e., the state \( x \) will follow the desired trajectory \( x_d \) when \( t \) approaches infinity, where \( K_0 \) and \( K_0 \) are positive definite constant matrices.

**Proof:** Omitted here due to the limitation of writing space.

**B. Adaptive Force Control**

According to the above description, We know the inertia and damping parameters only have influence on the transient response \([1, 2]\), therefore we can obtain the steady-state response relationship

\[ F_e = K_e (x - x_e) \]

where \( K_e \) is the stiffness coefficient of contact environment, \( x_e \) is the position of environment. For the desired force, we consider the Hooke's law, then, the relation between environment and end-effector position can be written as

\[ F_e = K_e (x_e - x_e) \]

where \( x_e \) is the ideal trajectory for the desired force.

By connecting (22) and (23), we can design our control goal \( F_g = F_e \). But the knowledge of stiffness coefficient \( K_e \) is not exactly known \([1, 2]\), \( x_e \) cannot be obtained. Hence, we should estimate the stiffness coefficient \( \hat{K}_e \) to reach the desired trajectory and \( F_g \) can achieve the force control goal. We propose the adaptive force control to estimate the stiffness coefficient \( K_e \) by gradient descent method. The schematic description shows in Fig. 1.

According to the above-mentioned description, the update law of stiffness coefficient by gradient descent method is explained below. First we design the error of the force

\[ e_{F_{\text{error}}} = F_g - F_e \]

Consider the contact direction is only one direction. \( \hat{K}_e \) is the scalar element of \( K_e \), and \( e_{F_{\text{error}}} = f_j - f_{\text{error}} \). We assume the objective function

\[ V_{F_{\text{error}}} = \frac{1}{2} e_{F_{\text{error}}} e_{F_{\text{error}}} \]

The gradient of (25) can be written as

\[ \frac{\partial V_{F_{\text{error}}}}{\partial K_e} = e_{F_{\text{error}}} e_{F_{\text{error}}} \]

By the gradient descent method, the update of stiffness coefficient \( K_e \) is

\[ \Delta K_e = -\eta e_{F_{\text{error}}} e_{F_{\text{error}}} \]

where \( \eta \) is the learning rate. The learning rate will concern the convergence reaction, which means the small value of \( \eta \) causes slower convergence and the bigger one causes the unstable result.

**Theorem 2:** Consider the adaptive force control, to guarantee the update of stiffness coefficient \( K_e \), we can choose the learning rate \( \eta \)

\[ 0 \leq \frac{\eta}{2} \left( \frac{\partial e_{F_{\text{error}}}}{\partial K_e} \right)^2 \leq 1 \]

where \( \eta \) is time-variant learning rate.

**Proof:** Omitted due to the limitation of writing space.

**IV. SIMULATION RESULTS**

Herein, we consider the redundant robot manipulator LWR (Light Weight Robot) made by KUKA company, as shown in Fig. 2. The KUKA LWR has 7-link robot manipulator, i.e. it is redundant. The Lagrangian dynamic formulation is used here \([3, 15, 34, 35]\). In this section, we utilize our proposed control scheme in the redundant robot manipulator KUKA LWR lightweight arm, which is composed of seven independent revolute joints. The parameters of the hardware are the same as literature \([34]\).

Most of familiar robot manipulators are non-redundancy. That means those robot manipulators track the same trajectory only the one solution to solve the problem. For redundant case, \( n > 6 \), that is the extent of the manipulator redundancy is given by \( r = n - m \), which defines the redundancy degree of the manipulator. When the end effector of the redundant robot manipulator implements the task, the second joint will maintain the same joint position as the original.
The initial desired trajectory is \( \mathbf{q}(0) = [10 20 30 10 50 60 70] \times \pi/180 \), \( \mathbf{x}(0) = [-0.0498 0.0688 0.7916 2.5670 -2.7584 -0.392] \). Consider the tracking control problem with initial error with initial condition as \( \mathbf{q}(0) = [10 20 30 10 50 60 70] \times \pi/180 \), \( \mathbf{x}(0) = [-0.0474 0.0801 0.7988 2.8033 -2.7479 -0.5963] \). The initial weighting vector of FNN are randomly chosen in [0 1]. The fuzzy rule numbers are all chosen nine.

The update laws gain matrices are selected as:
\[
\mathbf{Q}_i^k = \text{diag}(q_i, ..., q_i), \quad q_i = 10^{-6}, \quad i = 1, ..., 9
\]
\[
\mathbf{Q}_i^1 = \text{diag}(q_i, ..., q_i), \quad q_i = 10^{-3}, \quad i = 1, ..., 9
\]
\[
\mathbf{Q}_i^2 = \text{diag}(q_i, ..., q_i), \quad q_i = 10^{-4}, \quad i = 1, ..., 9
\]
\[
\mathbf{Q}_i^3 = \text{diag}(q_i, ..., q_i), \quad q_i = 10^{-4}, \quad i = 1, ..., 9
\]
\[
\mathbf{Q}_i^4 = \text{diag}(q_i, ..., q_i), \quad q_i = 10^{-4}, \quad i = 1, ..., 9
\]

The environment can be modeled as linear spring \( \mathbf{F}_{\text{env}} = \mathbf{K}_e (\mathbf{x} - \mathbf{x}_e) \) which is the force acting on the environment.

The desired force is selected as 5 N. The control gain matrices and the environment stiffness are selected as \( \mathbf{k} = \text{diag}(25, 25, 45, 30, 140, 40) \) and \( \mathbf{k}_e = \text{diag}(80, 80, 80, 10^{-5}, 10^{-4}, 10^{-4}) \). The matrices of the target impedance is selected as \( \mathbf{K}_e = 25 \), \( \mathbf{K}_x = 300(\text{N/m}) \). \( \mathbf{K}_e = 1.5 \times 10^4 \).

Figure 3 shows the simulation of robot manipulator motion results. Since the system having small initial tracking error, Fig. 4 shows a damping during the few time and then achieve the tracking control. Figure 5 shows the tracking position and the tracking error of position, we can see the performance and effectiveness of our control scheme. In the first second, since the deviation of the initial condition, there is an error and convergence quickly, and also, the end effector contact the environment in the third second, therefore it causes an error and damping. Figure 6 shows the joint torque of the robot manipulator, since the trajectory is planned by cubic spline to lead the position, velocity and acceleration continuous and also contact the environment in the third second, thus, it will causes the more torque, we can see there is a damping and more torque in the third second. Figure 7 shows the force of end effector and we can see the performance of the adaptive force control. Figure 8 shows the tracking performance of the stiffness coefficient \( \mathbf{K}_e \).

The results of Fig. 5 illustrate that tracking of estimate stiffness has a good performance to track the desired estimate stiffness. Figure 6 shows the force control behavior which has a good performance. Figure 6 shows the velocity of the joint, we can see the velocity of the second joint is maintained in the zero to make the second joint constant. Figure 7 shows the force tracking performance, we can saw that the end effector contacts the environment at the third second, the force can track to the desired force. Figure 8 shows the tracking performance of the stiffness coefficient \( \mathbf{K}_e \) by choosing the suitable parameter.
This study has introduced an adaptive force controller design for a redundant robot manipulator in the task space. We discuss the relation between the task space and joint space, and also introduce the knowledge of null space to overcome these problems by fuzzy neural network. The stiffness coefficient of the environment is unknown. Therefore, the gradient descent method is used to estimate the stiffness coefficient of environment to achieve the adaptive force control. The stability analysis of the closed-loop system and the corresponding update laws are given by Lyapunov stability theorem. Finally, we employ our proposed control scheme in the redundant robot manipulator KUKA LWR4 with 7–DOF to illustrate the performance and effectiveness.

REFERENCES


