Journal of Computer Science and Cybernetics, V.31, N.1 (2015), 71–81
DOI: 10.15625/1813-9663/31/1/5088

A COMPARISON STUDY OF SOME CONTROL METHODS FOR DELTA SPATIAL PARALLEL ROBOT

NGUYEN VAN KHANG[1,a], NGUYEN QUANG HOANG[1,b], NGUYEN DUC SANG[1], and NGUYEN DINH DUNG[2]

[1]*Department of Applied Mechanics, School of Mechanical Engineering, Hanoi University of Science and Technology*
[a]khang.nguyenvan2@hust.edu.vn; [b]hoang.nguyenquang@hust.edu.vn
[2]*Phuong Dong University*

Abstract. A comparison between three methods applied to parallel robot control namely: computed torque controller, sliding mode control and sliding mode control using neural networks is presented in this paper. The simulation results show that PD control method is only accurate when model parameters are precisely identified. In case of uncertain parameters, sliding mode and neural network sliding mode control methods are applied instead. Three controllers are implemented in Matlab for simulation. The results show that the control quality is improved by using the neural network sliding mode control method in comparison with two others.

Keywords. Delta parallel robot, computed-torque control, sliding mode control, neural network control.

1. INTRODUCTION

Today, parallel robotic manipulators are used widely in industrial applications owing to light compact structure, high stiffness and accuracy. Delta robot is one of the most successful parallel robots, with thousands of versions created around the world for several applications such as in food factories and medical field. Invented by Reymond Clavel in the early '80s, this parallel robot uses the parallelogram structure to create three translational degrees of freedom by three revolute actuators.

In most applications, the robot must move rapidly from one position to another position or follow a desired trajectory in three dimensional spaces with high precision. In order to perform this task, recently, several control methods have been investigated such as computed torque with PD controller [17], sliding mode controller [14,15,16]. The computed torque controller is easy to implement, but it cannot meet the control quality due to uncertainties in the system model and disturbances. The sliding mode controller is robust and can improve control quality. However, due to a discontinuous part, this control method can lead to chattering phenomenon, which makes difficulty to control and reduces quality control [14,15]. Another drawback of the sliding mode controller is that it requires the bounds of uncertainties and disturbances being available. The sliding mode control with online learning neural networks has been applied to serial robotic manipulators, in which the system uncertainties and disturbances are estimated by a function approximation technique [8,19-25].

In this paper, three control algorithms including inverse dynamic based, sliding mode and neural network based controllers are implemented for the delta spatial parallel robot. The simulation results show the outstanding features of the neural network based controller to the two others.

2. DYNAMIC MODEL

The equations of motion of a parallel robot can be obtained by either sub-structural method, Newton-Euler equations [1- 4] or Lagrangian multiplier equation [5,6,7]. These equations are written in matrix form as follows:

$$\begin{aligned} \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{D}\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) &= \mathbf{B}\mathbf{u} + \boldsymbol{\Phi}^T(\mathbf{q})\boldsymbol{\lambda} \\ \boldsymbol{\phi}(\mathbf{q}) &= \mathbf{0} \end{aligned} \tag{1}$$

where $\mathbf{M}(\mathbf{q}), \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}}, \mathbf{D}\dot{\mathbf{q}}, \mathbf{g}(\mathbf{q})$, and $\mathbf{u}$ are mass matrix, Coriolis and centrifugal forces, damping forces, gravitational forces, and control inputs, respectively; $\boldsymbol{\phi}(\mathbf{q})$, $\Phi(\mathbf{q}) = \partial\boldsymbol{\phi}(\mathbf{q})/\partial\mathbf{q}$, and $\boldsymbol{\lambda}$ are constrained equations, Jacobian matrix and vector of Lagrangian multipliers, respectively; $\mathbf{q} = [\boldsymbol{\theta}^T, \boldsymbol{\Psi}^T, \mathbf{x}^T]^T$ is generalized coordinate vector which includes actuated and auxiliary angles, and position of the mobile platform.

Using the method of coordinate separation and Lagrangian multipliers elimination [10-13] yields the motion equations in the form of minimum generalized coordinates as follows

$$\mathbf{M}_\theta(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \mathbf{C}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} + \mathbf{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} + \mathbf{g}_\theta(\boldsymbol{\theta}) = \mathbf{u}. \tag{2}$$

In dynamics (2) the following properties hold: $\mathbf{M}_\theta(\boldsymbol{\theta})$ - positive definite and symmetric matrix, $\mathbf{N} = [\dot{\mathbf{M}}_\theta(\boldsymbol{\theta}) - 2\mathbf{C}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})]$ - skew-symmetric matrix, and $\mathbf{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})$ - semi-positive definite matrix.

In the next section, Eq. (2) will be used for designing controller.

3. REVIEW OF THREE CONTROL METHODS FOR PARALLEL ROBOTS

The objective of the control problem is to find control forces $\mathbf{u}$ acting on the robot to drive the mobile platform to track the desired motion $\mathbf{x}_d(t)$. This means to control the actuated coordinates $\boldsymbol{\theta}(t)$ to follow its desired motion $\boldsymbol{\theta}_d(t)$ corresponding to the desired motion of mobile platform $\mathbf{x}_d(t)$. Three control algorithms are shown in this section.

3.1. Computed-torque controller

The computed-torque controller has been applied to serial robotic manipulators [16, 17]. This approach can also be applied to parallel manipulators. By applying this approach, the control input is computed as following

$$\mathbf{u} = \mathbf{M}_\theta(\boldsymbol{\theta})\mathbf{a} + \mathbf{C}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} + \mathbf{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} + \mathbf{g}_\theta(\boldsymbol{\theta}) \tag{3}$$

with

$$\mathbf{a} = \ddot{\boldsymbol{\theta}}_d - \mathbf{K}_d(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{\theta}}_d) - \mathbf{K}_p(\boldsymbol{\theta} - \boldsymbol{\theta}_d), \tag{4}$$

where $\mathbf{K}_d, \mathbf{K}_p$ are positive definite matrices:

$$\mathbf{K}_p = \text{ diag}\,(k_{p1}, k_{p2}, ..., k_{pn_a}),\ k_{pi} > 0, \mathbf{K}_d = \text{ diag}\,(k_{d1}, k_{d2}, ..., k_{dn_a}),\ k_{di} > 0.$$

Substituting (3) and (4) into Eqs. (2) results in

$$\mathbf{M}_\theta(\boldsymbol{\theta})(\ddot{\tilde{\boldsymbol{\theta}}} + \mathbf{K}_d\dot{\tilde{\boldsymbol{\theta}}} + \mathbf{K}_p\tilde{\boldsymbol{\theta}}) = \mathbf{0},\ \tilde{\boldsymbol{\theta}} = \boldsymbol{\theta} - \boldsymbol{\theta}_d. \tag{5}$$

Note that $\mathbf{M}_\theta(\boldsymbol{\theta})$ is a positive definite matrix, it can be eliminated from Eq. (5):

$$\ddot{\tilde{\boldsymbol{\theta}}} + \mathbf{K}_d\dot{\tilde{\boldsymbol{\theta}}} + \mathbf{K}_p\tilde{\boldsymbol{\theta}} = \mathbf{0}. \tag{6}$$

Based on the dynamics of error shown in Eq. (6), the controller parameters are chosen as

$$k_{pi} = \omega_i^2,\ k_{di} = 2\delta_i\omega_i, \quad \text{with } 0 < \delta_i < 1,\ \omega_i > 0. \tag{7}$$

Then, the solution to Eq. (6) will converge exponentially to zero resulting in $\theta(t) \to \theta_d(t)$ and $\mathbf{x}(t) \to \mathbf{x}_d(t)$.

Remark 1. This control algorithm is simple, but it requires high accuracy in system model and parameters. Matrices and vectors $\mathbf{M}_\theta(\boldsymbol{\theta})$, $\mathbf{C}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})$, $\mathbf{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})$, and $\mathbf{g}_\theta(\boldsymbol{\theta})$ are required available and exact. However, in practice these can be obtained approximately only. In order to overcome this issue, the controller needs to be robust or adaptive. In the next subsection a sliding mode controller will be considered.

3.2. Sliding mode control

Sliding mode control is robust and insensitive to the change of system parameters and disturbances. This can be applied to a nonlinear problem. By using sliding mode control [14, 15] the system response is divided into two phases: in the first phase the state variable is forced to the sliding surface then it slides on the surface to the origin in the second one.

Here the sliding surface is chosen as the following:

$$\mathbf{s}(t) = \dot{\mathbf{e}}(t) + \Lambda\mathbf{e}(t) = \dot{\theta} - \dot{\theta}_d + \Lambda\mathbf{e}(t), \tag{8}$$

where $\mathbf{e} = \tilde{\theta} = \theta - \theta_d$ and Λ is a diagonal matrix with positive elements

$$\Lambda = \text{ diag}\,(\lambda_1, \lambda_2, ..., \lambda_{n_a}),\ \lambda_i > 0, \quad i = 1, ..., n_a.$$

To find a control law, a Lyapunov candidate function is chosen as

$$V = \frac{1}{2}\mathbf{s}^T\mathbf{M}_\theta(\theta)\mathbf{s}. \tag{9}$$

Differentiating of V with respect to time, one gets

$$\dot{V} = \mathbf{s}^T\mathbf{M}_\theta(\theta)\dot{\mathbf{s}} + \frac{1}{2}\mathbf{s}^T\dot{\mathbf{M}}_\theta(\theta)\mathbf{s} \tag{10}$$

Putting $\dot{\boldsymbol{\theta}}_r = \dot{\boldsymbol{\theta}}_d - \Lambda\mathbf{e}$ and from (8) it yields

$$\dot{\mathbf{s}} = \ddot{\mathbf{e}} + \Lambda\dot{\mathbf{e}} = \ddot{\boldsymbol{\theta}} - \ddot{\boldsymbol{\theta}}_d + \Lambda\dot{\mathbf{e}} = \ddot{\boldsymbol{\theta}} - \ddot{\boldsymbol{\theta}}_r, \quad \ddot{\boldsymbol{\theta}}_r = \ddot{\boldsymbol{\theta}}_d - \Lambda\dot{\mathbf{e}} \tag{11}$$

and

$$\dot{\boldsymbol{\theta}} = \mathbf{s} + (\dot{\boldsymbol{\theta}}_d - \Lambda\mathbf{e}) = \mathbf{s} + \dot{\boldsymbol{\theta}}_r, \tag{12}$$

Putting (11) under consideration of dynamic model (2) into (10) results in

$$\begin{aligned}\dot{V} =& \mathbf{s}^T\left[\mathbf{M}_\theta(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} - \mathbf{M}_\theta(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}}_r\right] + \frac{1}{2}\mathbf{s}^T\dot{\mathbf{M}}_\theta(\boldsymbol{\theta})\mathbf{s} \\ =& \mathbf{s}^T\left[\mathbf{u} - \mathbf{C}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} - \mathbf{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}} - \mathbf{g}_\theta(\boldsymbol{\theta}) - \mathbf{M}_\theta(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}}_r\right] + \frac{1}{2}\mathbf{s}^T\dot{\mathbf{M}}_\theta(\boldsymbol{\theta})\mathbf{s}\end{aligned} \tag{13}$$

Substituting (12) into (13) and noting the skew-symmetric property of $[\dot{\mathbf{M}}_\theta(\boldsymbol{\theta}) - 2\mathbf{C}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})]$ yields

$$\dot{V} = \mathbf{s}^T \left[\mathbf{u} - \mathbf{C}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}}_r - \mathbf{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}}_r - \mathbf{g}_\theta(\boldsymbol{\theta}) - \mathbf{M}_\theta(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}}_r\right] - \mathbf{s}^T\mathbf{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\mathbf{s} \tag{14}$$

If the system model and its parameters are known exactly, the control law will be chosen as follows

$$\mathbf{u} = \mathbf{M}_\theta(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}}_r + \mathbf{C}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}}_r + \mathbf{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}}_r + \mathbf{g}_\theta(\boldsymbol{\theta}) - \mathbf{K}_{pd}\mathbf{s}. \tag{15}$$

By choosing $\mathbf{K}_{pd}$ being positive definite matrix results in

$$\dot{V} = -\mathbf{s}^T\mathbf{K}_{pd}\mathbf{s} - \mathbf{s}^T\mathbf{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\mathbf{s} < 0, \forall \mathbf{s} \neq \mathbf{0}. \tag{16}$$

Because the exact parameters of the systems are not available, so the control law (15) is modified as

$$\mathbf{u} = \hat{\mathbf{M}}_\theta(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}}_r + \hat{\mathbf{C}}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}}_r + \hat{\mathbf{D}}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}}_r + \hat{\mathbf{g}}_\theta(\boldsymbol{\theta}) - \mathbf{K}_{pd}\mathbf{s} - \mathbf{K}_s sgn(\mathbf{s}) \tag{17}$$

in which the last term is added to ensure that $\dot{V}$ is negative. Assuming that the difference between the system and the model used for controller is defined by

$$\mathbf{h} = \Delta\mathbf{M}_\theta(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}}_r + \Delta\mathbf{C}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}}_r + \Delta\mathbf{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}}_r + \Delta\mathbf{g}_\theta(\boldsymbol{\theta}), \tag{18}$$

with

$$\Delta\mathbf{M}_\theta(\boldsymbol{\theta}) = [\mathbf{M}_\theta(\boldsymbol{\theta}) - \hat{\mathbf{M}}_\theta(\boldsymbol{\theta})], \quad \Delta\mathbf{C}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}}) = [\mathbf{C}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}}) - \hat{\mathbf{C}}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})],$$

$$\Delta\mathbf{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}}) = [\mathbf{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}}) - \hat{\mathbf{D}}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})], \quad \Delta\mathbf{g}_\theta(\boldsymbol{\theta}) = [\mathbf{g}_\theta(\boldsymbol{\theta}) - \hat{\mathbf{g}}_\theta(\boldsymbol{\theta})]$$

is bounded, means $|h_i| \leq h_{0,i}$.

The last term in (17) is chosen as

$$\mathbf{u}_{sld} = -\mathbf{K}_s sgn(\mathbf{s}), u_{sld,i} = -K_{s,i,i} sgn(s_i), K_{s,i,i} = h_{0,i} + \eta, \eta > 0.$$

With (17) and (18) Eq. (14) becomes

$$\dot{V} \leq \mathbf{s}^T\left[-\mathbf{h} - \mathbf{K}_{pd}\mathbf{s} - \mathbf{K}_s sgn(\mathbf{s})\right] = -\mathbf{s}^T\mathbf{K}_{pd}\mathbf{s} - \mathbf{s}^T\left(\mathbf{h} + \mathbf{K}_s sgn(\mathbf{s})\right). \tag{19}$$

Due to $s_i sgn(s_i) = |s_i|$, Eq. (19) becomes

$$\dot{V} \leq -\mathbf{s}^T\mathbf{K}_{pd}\mathbf{s} + \Sigma|s_i|(h_{0,i} - K_{s,ii}) \leq -\mathbf{s}^T\mathbf{K}_{pd}\mathbf{s} - \eta\Sigma|s_i| \tag{20}$$

So, the controller (17) with $\mathbf{K}_{pd} > 0$ and diagonal matrix $\mathbf{K}_s$ having $K^s_{i,i} > |h_{0i}|$ guarantees $\dot{V} < 0$, then $\mathbf{s}(t) \to \mathbf{0}$ and $\mathbf{e}(t) \to \mathbf{0}$.

Noting that, the controller (17) has a non-continuous part $\mathbf{K}_s sgn(\mathbf{s})$. It causes unwanted fluctuations with high frequencies around the sliding surface, which is called "chattering" phenomenon. In order to reduce chattering, the discontinuous function $sgn(\cdot)$ will be replaced by a smooth function of $\arctan(cs), c \gg 1$.

3.3. Sliding mode control with RBF neural networks

In the previous section, the uncertainty $\mathbf{h}$ defined by (18) is suppressed by the discontinuous part $\mathbf{u}_{sld} = -\mathbf{K}_s sgn(\mathbf{s})$. The diagonal elements of the matrix $\mathbf{K}_s$ are chosen depending on the maximum value of each element h_i. In this section, it is assumed that the function $\mathbf{h}$ defined in (18) can be approximated by a function depending on generalization error $\mathbf{h}(\mathbf{s})$. The unknown function $\mathbf{h}(\mathrm{s})$ is the main reason to reduce the control quality. If this effect is compensated, the control accuracy can then be improved. According to Stone-Weierstrass theorem [18] an appropriate *Artificial Neural Network* (ANN) can be chosen with a limited number of neurons that can approximate an unknown nonlinear function with a given accuracy. For approximating function $\mathbf{h}(\mathbf{s})$ the following simple structure ANN is selected

$$\mathbf{h}(\mathbf{s}) = \mathbf{W}\boldsymbol{\sigma} + \boldsymbol{\varepsilon} = \hat{\mathbf{h}}(\mathbf{s}) + \boldsymbol{\varepsilon} \tag{21}$$

where $\mathbf{W}$ is the weight matrix, $\boldsymbol{\sigma} = \boldsymbol{\sigma}(\mathbf{s}) = [\sigma_1, \sigma_1, ..., \sigma_n]^T$, with $\sigma_i(\mathbf{s})$ being the Radial Basis Function (RBF), where the Gaussian RBF function is

$$\sigma_i = \exp\left(-(s_i - c_i)^2/\lambda_i^2\right), \tag{22}$$

where c_i is the center and λ_i the deviation parameter that are freely chosen.
In Eq. (21) the output of an ANN

$$\hat{\mathbf{h}}(\mathbf{s}) = [\hat{h}_1, \hat{h}_2, ..., \hat{h}_n]^T = \mathbf{W}\boldsymbol{\sigma} \tag{23}$$

is the approximation of $\mathbf{h}(\mathbf{s})$, $\boldsymbol{\varepsilon}$ is the approximation error. With $\| \mathbf{h}(\mathbf{s}) \| \leq h_0$, there is a bound for $\boldsymbol{\varepsilon}$ as $\| \boldsymbol{\varepsilon} \| \leq \varepsilon_0$.
Denoting $\mathbf{w}_i$ the i-th column vector of matrix $\mathbf{W}$ yields

$$\hat{\mathbf{h}}(\mathbf{s}) = [\hat{h}_1, \hat{h}_2, ..., \hat{h}_n]^T = \mathbf{W}\boldsymbol{\sigma} = \sum_{i=1}^{n} \sigma_i \mathbf{w}_i \tag{24}$$

and

$$\hat{h}_i = \sum_{j=1}^{n} w_{ji}\sigma_j, i = 1, ..., n \tag{25}$$

where the weights w_{ji} will be updated for the approximation of neural network.
The chosen ANN is a RBF neural network having one hidden layer as shown in Figure 1. This structure has been proved to satisfy the Stone-Weierstrass theorem [18]. The control problem is now to find the control input $\mathbf{u}$ and the learning algorithm of w_{ji} of neural network according to Eq. (25) so $\mathbf{s}(t) \to \mathbf{0}$ and the position error $\mathbf{e}$ will converge to zero, that guarantees $\boldsymbol{\theta}(t) \to \boldsymbol{\theta}_d(t)$. For that the following theorem [23, 25] is available.

Theorem 1. *The trajectory $\boldsymbol{\theta}(t)$ of dynamic system defined by Eq. (2) with RBF neural network according to Eqs. (24) and (22) and sliding surface in Eq. (8) will track the desired trajectory $\boldsymbol{\theta}_d(t)$ with error $\mathbf{e} = \boldsymbol{\theta}(t) - \boldsymbol{\theta}_d(t) \to \mathbf{0}$ if the control input $\boldsymbol{u}$ and the learning algorithm $\dot{\mathbf{w}}_i$ are applied as follows:*

$$\mathbf{u} = \hat{\mathbf{M}}_\theta(\boldsymbol{\theta})\ddot{\theta}_r + \hat{\mathbf{C}}_\theta(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}}_r + \hat{\mathbf{D}}_\theta(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}}_r + \hat{\mathbf{g}}_\theta(\boldsymbol{\theta}) - \mathbf{K}_{pd}\mathbf{s} - \gamma\frac{\mathbf{s}}{\| \mathbf{s} \|} + (1+\eta)\mathbf{W}\boldsymbol{\sigma} \tag{26}$$

$$\dot{\mathbf{w}}_i = -\eta\sigma_i\mathbf{s}, \tag{27}$$

where matrix $\mathbf{K}_{pd}$ is a freely chosen symmetric positive definite matrix, and $\eta > 0$, $\gamma > 0$.

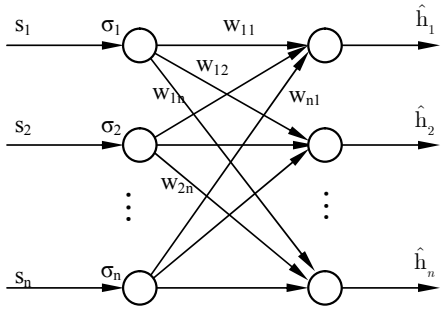


Figure 1: RBF Neural Network used to approximation uncertainties of the robot

The theorem can be proved by Lyapunov direct method guaranteeing the asymptotic stability,

$$V = \frac{1}{2}\mathbf{s}^T\mathbf{M}_\theta(\theta)\mathbf{s} + \frac{1}{2}\sum_{i=1}^{n}\mathbf{w}_i^T\mathbf{w}_i. \tag{28}$$

The mass matrix $\mathbf{M}_\theta(\boldsymbol{\theta})$ is symmetric positive definite, so $V(t) > 0$ is for all $(\mathbf{s}^T, \mathbf{w}_1^T, ..., \mathbf{w}_{n_a}^T)^T \neq \mathbf{0}$ and $V(t) = 0$ if and only if $(\mathbf{s}^T, \mathbf{w}_1^T, ..., \mathbf{w}_{n_a}^T)^T = \mathbf{0}$. Taking the derivation of function $V(t)$ with respect to time yields

$$V = \mathbf{s}^T\mathbf{M}_\theta(\boldsymbol{\theta})\dot{\mathbf{s}} + \frac{1}{2}\mathbf{s}^T\dot{\mathbf{M}}_\theta(\boldsymbol{\theta})\mathbf{s} + \sum_{i=1}^{n}\mathbf{w}_i^T\dot{\mathbf{w}}_i \tag{29}$$

Using Eqs. (2), (11) and (12) in Eq. (29) results in

$$\dot{V} = \mathbf{s}^T\left[\mathbf{u} - \mathbf{C}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}}_r - \mathbf{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}}_r - \mathbf{g}_\theta(\boldsymbol{\theta}) - \mathrm{M}_\theta(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}}_r\right] - \mathbf{s}^T\mathrm{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\mathbf{s} + \sum_{i=1}^{n_a}\mathbf{w}_i^T\dot{\mathbf{w}}_i \tag{30}$$

Noting that in Eq. (30) the skew-symmetric property of $[\dot{\mathbf{M}}_\theta(\boldsymbol{\theta}) - 2\mathbf{C}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})]$ has been used. Substituting Eq. (26) and Eq. into Eq. (30) resulting in the following

$$\dot{V} = \mathbf{s}^T\left[-\mathbf{h} - \mathbf{K}_{pd}\mathrm{s} - \gamma\frac{\mathbf{s}}{\parallel \mathbf{s} \parallel} + (1+\eta)\mathbf{W}\sigma\right] - \mathbf{s}^T\mathbf{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\mathbf{s} + \sum_{i=1}^{n_a}\mathbf{w}_i^T\dot{\mathbf{w}}_i \tag{31}$$

By using the approximation (21) it yields

$$\begin{aligned}\dot{V} &= \mathbf{s}^T\left[-\mathbf{W}\sigma - \boldsymbol{\varepsilon} - \mathbf{K}_{pd}\mathbf{s} - \gamma\tfrac{\mathbf{s}}{\|\mathbf{s}\|} + (1+\eta)\mathbf{W}\sigma\right] - \mathbf{s}^T\mathbf{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\mathbf{s} + \sum_{i=1}^{n_a}\mathbf{w}_i^T\dot{\mathbf{w}}_i \\ &= -\mathbf{s}^T\mathbf{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\mathbf{s} - \mathbf{s}^T\mathbf{K}_{pd}\mathbf{s} - \gamma\mathbf{s}^T\tfrac{\mathbf{s}}{\|\mathbf{s}\|} + \mathbf{s}^T\boldsymbol{\varepsilon} - \mathbf{s}^T\eta\mathbf{W}\sigma + \sum_{i=1}^{n_a}\mathbf{w}_i^T\dot{\mathbf{w}}_i\end{aligned} \tag{32}$$

Note that with the adaptation law defined by (27), the last term of Eq. (32) can be rewritten in the form

$$\sum_{i=1}^{n}\mathbf{w}_i^T\dot{\mathbf{w}}_i = -\eta\sum_{i=1}^{n}\mathbf{w}_i^T\mathbf{s}\boldsymbol{\sigma}_i = -\eta\mathbf{s}^T\mathbf{W}\boldsymbol{\sigma}. \tag{33}$$

Finally, inserting Eq. (33) into Eq. (32) results in

$$\dot{V} = -\mathbf{s}^T\mathbf{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\mathbf{s} - \mathbf{s}^T\mathbf{K}_{pd}\mathbf{s} - \gamma\mathbf{s}^T\frac{\mathbf{s}}{\parallel \mathbf{s} \parallel} + \mathbf{s}^T\boldsymbol{\varepsilon}. \tag{34}$$

By choosing $\gamma = \delta + \varepsilon_0$ with $\delta > 0$, it yields

$$\dot{V} = -\mathbf{s}^T\mathbf{D}_\theta(\boldsymbol{\theta},\dot{\boldsymbol{\theta}})\mathbf{s} - \mathbf{s}^T\mathbf{K}_{pd}\mathbf{s} - \delta \parallel \mathbf{s} \parallel -(\varepsilon_0 \parallel \mathbf{s} \parallel + \mathbf{s}^T\boldsymbol{\varepsilon}) \tag{35}$$

Given $\parallel \boldsymbol{\varepsilon} \parallel \leq \varepsilon_0$ so $\dot{V}(t) < 0$ for all $\mathbf{s} \neq \mathbf{0}$ and $\dot{V}(t) = 0$ if and only if $\mathbf{s} = \mathbf{0}$. According to the Lyapunov direct stability theorem, $\mathbf{s} \to \mathbf{0}$ is available, and therefore, from Eq. (8) $\mathbf{e} = \boldsymbol{\theta}(t) - \boldsymbol{\theta}_d(t) \to 0$ is derived. So the theorem as well as the stability of the overall sliding mode control system using neural network has been proved.

4. NUMERICAL SIMULATION

In this section, some simulations in universal software Matlab are implemented to compare the results of three presented algorithms. For simulation, the numerical methods are applied to obtain the actuated coordinates [9]. The model of the Delta robot is given in Fig. 2 and the robot parameters are as follows:

$L_1 = 0.3$; $L_2 = 0.8$; $R = 0.266$; $r = 0.04$;[m],
$\alpha_A = 0; \alpha_B = 2\pi/3; \alpha_C = 4\pi/3$,[rad],
$m_1 = 0.416$; $m_2 = 2 \times 0.195$; $m_P = 0.3$, [kg],
$\mathbf{I}_1 = 0.0125\,\mathrm{diag}\,(0,1,1)$[kg.m^2],
$\mathbf{I}_2 = 0.0208\,\mathrm{diag}\,(0,1,1)$[kg.m^2].

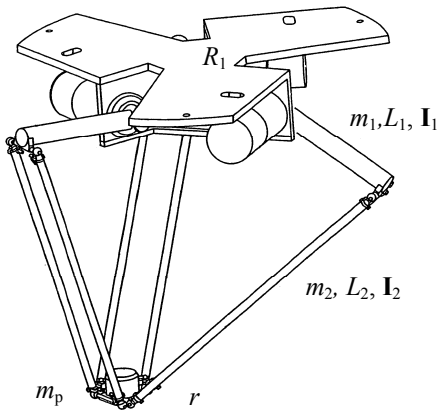


Figure 2: Delta robot model.

It is assumed that the disturbances are summarized to the actuating coordinates as

$$\mathbf{d}_\theta = [\ \sin(20t) \quad \cos(20t) \quad \sin(20t)\ ]^T.$$

The parameters of three controllers are chosen as:

1. Computed-torque with PD controller

$$\mathbf{K}_p = 100\,\mathrm{diag}\,(1,1,1); \mathbf{K}_d = 50\,\mathrm{diag}\,(1,1,1);$$

2. Sliding mode controller

$$\Lambda = diag(20, 20, 20);\ \mathbf{K}_{pd} = \text{diag}\,(10, 10, 10);\ \mathbf{K}_s = \text{diag}\,(30, 30, 30);$$

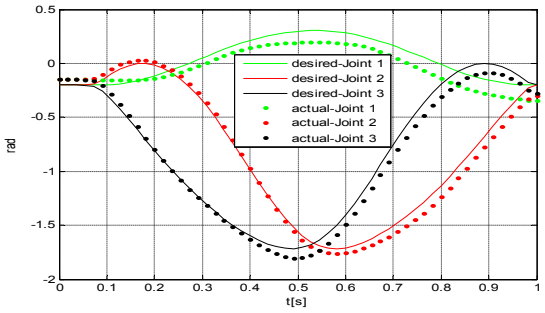
3. Neural network based sliding mode controller

$$\mathbf{K}_{pd} = 10\,\text{diag}\,(1, 1, 1); \Lambda = \text{diag}\,(20, 20, 20);$$
$$\eta = 1; \gamma = 20;$$
$$\lambda_1 = 1; \lambda_2 = 2; \lambda_3 = 3;$$
$$c_1 = 0.01; c_2 = 0.02; c_3 = 0.03.$$

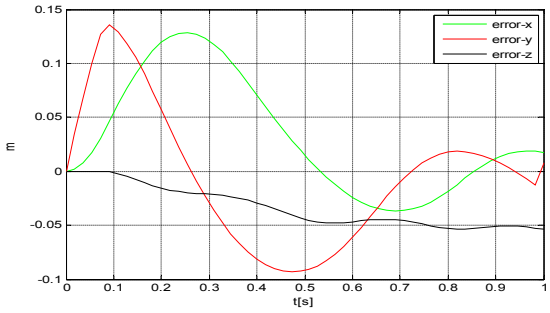
In all simulations, the mobile platform will be forced to track the given trajectory defined by

$$x_P = 0.3(1 - \cos 2\pi t)\ [m],$$
$$y_P = 0.3 \sin 2\pi t\ [m],$$
$$z_P = 0.7\ [m].$$

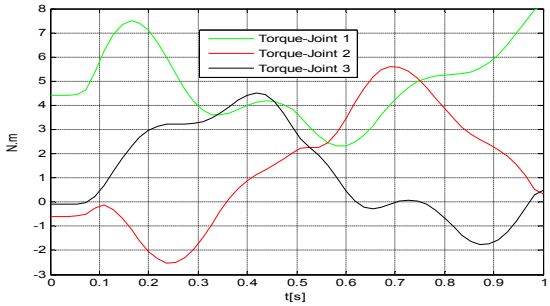
In simulations, the parameters of the controllers are chosen being about 30% different to those of the robot. Simulation results are shown in Figures 3, 4 and 5.



3a) Time history of joint variables



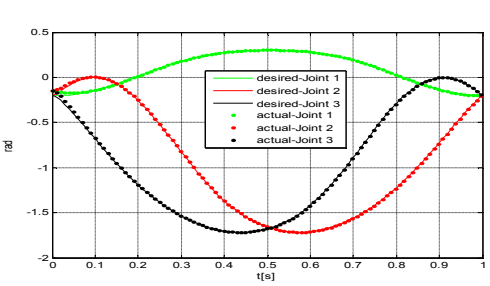
3b) Position error of mobile platform



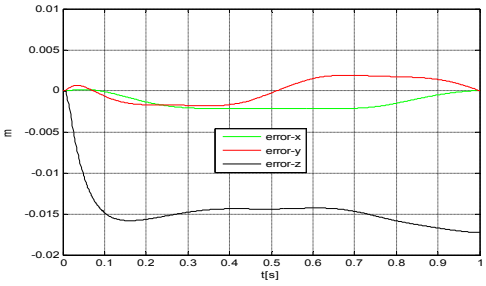
3c) Joint torques

Figure 3: Simulation results by computed-torque + PD controller

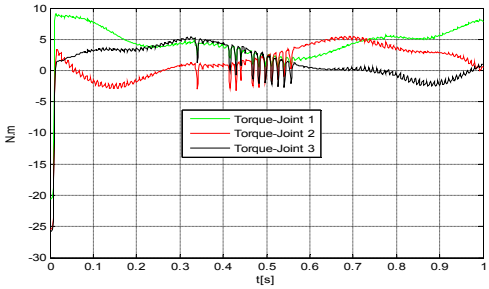
The simulation results show that with large model errors and disturbances, the system responses to controllers are different. The control inputs by computed-torque control method are smooth curves (Fig. 3c), but its tracking errors are greater than those by two other controllers. From Figs. 4c and 5c it is clear that the control inputs of sliding mode with neural networks are better than those of sliding mode.



4a) Time history of joint variables

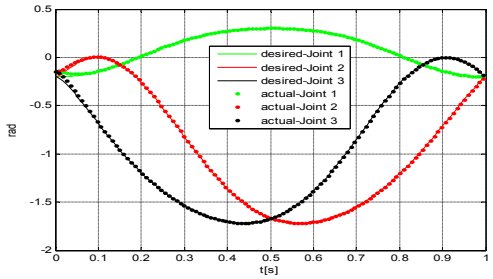


4b) Position error of mobile platform

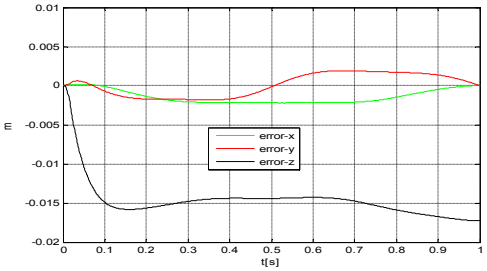


4c) Joint torques

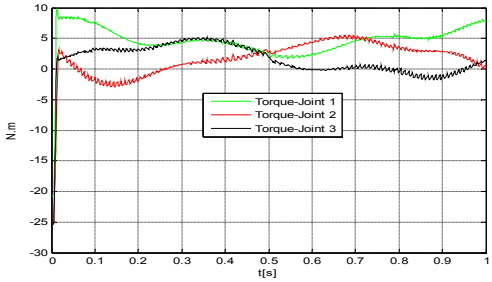
Figure 4: Simulation results by sliding mode controller



5a) Time history of joint variables



5b) Position error of mobile platform



5c) Joint torques

Figure 5: Simulation results by neural network-sliding mode controller

5. CONCLUSION

This paper presents three control algorithms applying to Delta spatial parallel robot. Firstly, the dynamic model of the closed loop system is transformed to the minimal driven coordinates by using constrained equation. Then, the controllers are designed based on this model. The computed torque with PD controller is simple for implementation, but it can not compensate for the system uncertainties and disturbances. Meanwhile, sliding mode controller and sliding mode controller using neural networks are robust against the system uncertainties and disturbances. According to the simulation results, sliding mode controller and neural network sliding mode controller compensate for uncertainties, and reduce tracking errors.

ACKNOWLEDGMENT

This paper has been completed with the financial support by the Vietnam National Foundation for Science and Technology Development (NAFOSTED).

REFERENCES

[1] W. Schiehlen and P. Eberhard, *Technische Dynamik* (3. Auflage). Vieweg+Teubner, Wiesbaden, 2012.

[2] A. A. Shabana, *Dynamics of Multibody Systems* (3.Edition). Cambridge University Press, New York, 2005.

[3] F. Amirouche, *Fundamentals of Multibody Dynamics*. Birkhäuser, Boston, 2004.

[4] J. G. Jalon and E. Bayo, *Kinematic and Dynamic Simulation of Multibody Systems – The Real-Time Challenge*. Springer-Verlag, New York, 1994.

[5] Tsai Lung-Wen, *Robot Analysis - The Mechanics of Serial and Parallel Manipulators*. John Wiley & Sons, New York, 1999.

[6] Nguyen Van Khang, "Consistent definition of partial derivatives of matrix functions in dynamics of mechanical systems". *Mechanism and Machine Theory* vol. 45, pp. 981-988, 2010.

[7] Nguyen Van Khang, "Kronecker product and a new matrix form of Lagrangian equations with multipliers for constrained multibody systems". *Mechanics Research Communications,* 38, pp. 294-299, 2011.

[8] Nguyen Van Khang and Luong Anh Tuan, "On the sliding mode control of redundant parallel robots using neural networks", in *Proc. of the 3th IFToMM International Symposium on Robotics and Mechatronics*, Singapore, Research Publishing, 2013, pp. 168-177.

[9] Nguyen Van Khang, Nguyen Phong Dien, and Luong Anh Tuan, "A comparative study on the computational efficiency of some numerical methods for solving the inverse kinematics of redundant robots", in *Proc. of the 3th IFToMM International Symposium on Robotics and Mechatronics*, Singapore, Research Publishing, 2013, pp. 513-522.

[10] R. A. Wehage and E. J. Haug, "Generalized coordinate partitioning for dimension reduction in analysis of constrained dynamic systems". Trans. *ASME, Journal of Mechanical Design,* vol. 104, pp. 247-255, 1982.

[11] W. Blajer, W. Schiehlen, and W. Schirm, "A projective criterion to the coordinate partitioning method for multibody dynamics". *Archive of Applied Mechanics,* vol. 64, no. 2, pp. 86-98, 1994.

[12] T. Geike and J. McPhee, "Inverse dynamic analysis of parallel manipulators with full mobility". *Mechanism and Machine Theory,* vol. 38, pp. 549-562, 2003.

[13] H. Abdellatif and B. Heimann, "Computational efficient inverse dynamics of 6-DOF fully parallel manipulators by using the Lagrangian formalism," *Mechanism and Machine Theory,* vol. 44, pp. 192–207, 2009.

[14] J. E. Slotine and W. Li, "*Applied Nonlinear Control*". Prentice Hall, New York, 1991.

[15] V. I. Utkin, "*Sliding Modes in Control and Optimization*," Springer-Verlag, Berlin, 1992.

[16] L. Sciavicco and B. Siciliano, *Modelling and Control of Robot Manipulators* (2. Edition). Springer-Verlag, London, 2000.

[17] R. Kelly, V. Santibanez, A. Loria, *Control of Robot Manipulators in Joint Space*. Springer-Verlag, London, 2005.

[18] N. E. Cotter, "The Stone-Weierstrass theory and its applications to neural networks". *IEEE Transactions on Neural Network* 1, pp. 290-295, 1990.

[19] K. Jezernik, M. Rodic, R. Safaric, B. Curk, "Neural network sliding mode robot control". *Robotica* 15, pp. 23-30, 1997.

[20] F. Sun, Z. Sun, and P.Y. Woo, "Neural network-based adaptive controller design of robotic manipulators with an observer". *IEEE Transactions on Neural Networks* 12 (1), pp. 54-67, 2001.

[21] O. Barambones and V. Etxebarria, "Robust neural control for robotic manipulators". *Automatica* 38, pp. 235-242, 2002.

[22] R.-J. Wai, "Tracking control based on neural network strategy for robot manipulator". *Neurocomputing* 51, pp. 425-445, 2003.

[23] P.T. Cat and N. T. Hiep, "Robust PID sliding mode control of robot manipulators with online learning neural networks", in *Proc. of the European Control Conference*, Budapest, Hungary, 2009, pp. 2187-2192.

[24] Zhao-Hui Jiang, "Trajectory control of robot manipulators using a neural network controller". In "*Robot Manipulators: Trend and Development*" (Editors by A. Jimener and B. Hadithi), InTech, pp. 361-375, 2010.

[25] P. T. Cat, *Some modern control methods for industrial robots* (in Vietnamese). Thai Nguyen University, 2009.

Received on October 16 - 2014
Revised on March 10 - 2015