

Stable on-line neural control of systems with closed kinematic chains

M.J.Randall, A.F.T.Winfield and A.G.Pipe

Abstract: Artificial neural networks have been used extensively in control research. In industrial systems, however, it is crucial to adopt neural control structures which have a guaranteed proof of stability, especially if control system failure were to endanger life (e.g. in fast moving manipulators or transportation). In the paper, the neural control of robotic systems with closed kinematic chains is discussed and theorems guaranteeing the control stability of such systems are developed. The first class of systems have a single serial chain with a prescribed contact force when moving across a surface, i.e. the problem of hybrid position/force neural control. The second class of systems considered includes hexapod walking machines, which have a varying topology of closed kinematic chains during walking. The equations of motion can be solved by optimising contact forces according to a predefined cost function, and so the hybrid/position neural controller is extended to this class. A novel control structure which makes no initial assumptions about the system is also presented, using the concept of 'virtual neural networks': a projection of the neural controllers into the underconstrained space of the generalised co-ordinates of the equations of motion. This approach can be applied to a large number of different systems, including parallel manipulators and Stewart platforms, and it is also extended to include neural networks implemented on digital microprocessors.

1 Introduction

We are interested in the accurate control of robotic systems comprising serial linkages or tree structures. In previous work (e.g. [1–3]), researchers in our laboratory have employed artificial neural networks as part of the real time control of robotic manipulators (i.e. serial linkages). The most important aspect of that work was that the controllers had a guarantee of stability using Lyapunov techniques. Most practitioners who adopt neural networks are unable also to provide the confidence of a stability guarantee, that might make the controllers useful or attractive for industrial purposes. Following the success of that work, which resulted in up to an order of magnitude improvement in reducing tracking trajectory errors over conventional adaptive controllers, the class of systems considered was extended to include the rather more complex tree structures exemplified by a hexapod walking machine. A six-legged robot is significantly more complex to control than a manipulator for a number of reasons. First, there are many more degrees of freedom (typically 24 as opposed to 6). Secondly, during walking, the machine varies the topology of closed kinematic chains formed between the contact feet and the body. Thirdly, there are significant problems involved with navigation, finding footholds, controlling gait and maintaining balance.

These latter issues will not be considered in this paper, as they have been dealt with in [4] as well as by other authors. For walking machines, neural networks are ideal candidates for use in the control loop at the joint level because of their ability to learn the systematic dynamic uncertainties of the system and their capability to adapt quickly to the presence of obstacles or to contact force constraints during walking.

In this paper, we present four theorems concerning the joint control of systems with closed kinematic chains. The first is the problem of hybrid force position control using artificial neural networks, for a classical robot manipulator where the tool point is in contact with a surface. The second theorem extends this control technique to a typical hexapod structure. The third theorem extends the earlier work of [2] with application to the hexapod. The final theorem extends the third theorem for implementing artificial neural networks in digital microprocessors (i.e. where the time domain is discretised). Before presenting these theorems, the paper first overviews a number of key concepts required to use artificial neural networks in control theory.

2 Using neural networks in control

Artificial neural networks (or, more simply, 'neural networks') have at least four features that can usefully be exploited in adaptive control problems: (i) universal function approximation (e.g. [5]), which implies that, within a compact set, any function can be approximated within an arbitrarily small margin of error with a carefully designed network; (ii) parallel computation; (iii) distributed knowledge; and (iv) a learning ability, which implies that the neural network weights can be updated according to some

© IEE, 2000

IEE Proceedings online no. 20000759

DOI: 10.1049/ip-cta:20000759

Paper first received 7th March and in revised form 20th July 2000

The authors are with the Intelligent Autonomous Systems Engineering Laboratory, Faculty of Engineering, University of the West of England, Bristol, Coldharbour Lane, Bristol BS16 1QY, UK

IEE Proc.-Control Theory Appl., Vol. 147, No. 6, November 2000

619

rule so that the network output matches what is required within the limitations of the ‘training set’.

Neural networks have been widely used in research to solve control problems [6, 7]. Under certain conditions, it is possible to extend the stability theories that exist in traditional control theory to systems that include neural networks. This is crucially important if a neural control structure for, say, a walking machine is going to be adopted for an industrial product, so that the manufacturers have the assurance that such a system is not likely to cause damage to itself or to the environment in which it operates through controller instability. A useful distinction can be made between different neural network types that aids the process of extending stability proofs to systems with neural networks. We therefore distinguish between linear equivalent neural networks (LE) and nonlinear equivalent neural networks (a similar distinction is made in [8]). The output y of the LE class of neural network can be expressed as

$$y = \mathbf{g}^T(\mathbf{x})\mathbf{w} \quad (1)$$

where \mathbf{g} is the nonlinear mapping vector which depends on the neural network structure as well as the neural network input \mathbf{x} and has a bounded norm. A limitation of the approach is that we require \mathbf{g} to be persistently exciting, although this requirement can be removed by using the e -mod term introduced by [9] in the neural update law. \mathbf{w} is the weight vector. For multiple output neural networks, each output could be represented as an individual neural network, and so for the complete neural system, linear equivalence implies that

$$\mathbf{y} \triangleq \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} \mathbf{g}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{g}_2 & & \\ \vdots & & \ddots & \\ \mathbf{0} & & & \mathbf{g}_n \end{pmatrix}^T \begin{pmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \\ \vdots \\ \mathbf{w}_n \end{pmatrix} \triangleq \mathbf{G}^T \mathbf{w} \quad (2)$$

where \mathbf{g}_i is the nonlinear mapping vector for output i and \mathbf{w}_i is the weight vector for output i . However, in most cases, all hidden layer nonlinear mapping vectors are the same. The class of LE neural networks includes, for instance, radial basis function (RBF) neural networks [10, 11] and CMAC networks [12] but not multilayer perceptrons [13].

In the case of structures as complex as walking robots, it is generally either impractically difficult, or even impossible, to model mathematically the structured and unstructured uncertainties in the robot dynamics. There is a point here which is especially relevant when considering the use of *off-line* trained neural networks for nonlinear control. To generate an effective training set under simulation we need a complete and accurate model of the plant. However, if a complete model is available then there is almost certainly sufficient information to design an adequate conventional controller. One answer to this predicament is to gather the data for a neural network training set from the real plant, but this is often time consuming and difficult in practice. Further, it is not easy to ensure that coverage of the training set is sufficient for all on-line scenarios and therefore this approach does not lead naturally to guarantees of stability.

An alternative method to achieve adaptive neural control is to adopt an *on-line* learning régime, which removes the need for exhaustive off-line training. There are also other strengths inherent in an on-line learning approach. Amongst these is the fact that an on-line learning neural controller is able to adjust to ‘wear and tear’ and other factors which might change performance over time. In

addition the controller may be able to react usefully to situations which may not have been foreseen during simulation.

There is one big *disadvantage*, however: the run-time processing load will clearly be much greater in the on-line learning case. Nevertheless, in previous work, it has been possible to employ neural networks effectively in real-time control. In fact, in our laboratory, we have demonstrated that on-line neural control structures lead to an improvement of up to an order of magnitude in reducing tracking errors in robotic manipulators ([1], 2 ms control loop), and in walking robots they have been shown to adapt to the presence of a nonlinear elastic spring attached to the end of a leg within as few as five learning cycles ([14, 15], 10–20 ms control loops). The theorems presented in this paper underpin this use of neural control in walking machines by guaranteeing that the joint level controllers are stable under the given circumstances.

In practical terms, the controllers that follow are fairly straightforward to design and implement. They are based on PD, PID or conventional hybrid position/force control structures. The steps required for choosing appropriate P, I, D and force gains still need to be undertaken. In our controllers, their values are essentially no different from the cases where there is no neural network present. Guidelines for choosing an appropriate structure for an LE neural network are given in [10, 11]. Those papers present ‘existence proofs’ that show that universal function approximation can be achieved using neural networks: i.e. at least one neural network exists to approximate a given function. They do not provide details of how to design the neural networks in each case. However, to meet the conditions of the existence proofs, the neural networks must fulfil certain structural conditions, and so the papers make informed recommendations on neural network structures. For example, in the case of an RBF neural network, recommendations are given for the form of the basis functions and their distribution [11]. These are neural networks with a single layer of neuron weights so the structure of each does not become too complex: for instance, in the case of our hexapod walking machine, each of the 18 joint controllers includes a single layer RBF consisting of 200 neurons. The choice of spread and distribution was made by directly following the recommendations in [11], but the number of neurons was chosen by trial and error.

3 Contact force constraints for a single closed chain

The problem of adding a contact force constraint to a serial linkage is sometimes called ‘hybrid position/force’ control. It has been studied in [8] using neural networks. The problem requires a manipulator, for instance, to follow a prescribed trajectory tangential to a surface while exerting a prescribed contact force normal to the surface. The theorem in this section is original, although some of the core ideas have come from [8]. The purpose of this section is not to study the contact force problem exhaustively. Instead, it shows a proof of principle that can be extended in any number of ways, including, for instance, adding disturbance effects and unmodelled dynamics as well.

The robot dynamics with environment contacts on a prescribed surface can be described by

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \bar{\mathbf{g}}(\mathbf{q}) + \mathbf{f}_u(\mathbf{q}) = \boldsymbol{\tau} + \mathbf{J}^T \boldsymbol{\lambda} \quad (3)$$

where:

\mathbf{q} is an $n \times 1$ vector of joint displacements
 $\boldsymbol{\tau}$ is an $n \times 1$ vector of applied joint torques
 $\mathbf{H}(\mathbf{q})$ is an $n \times n$ symmetric positive definite manipulator inertia matrix
 $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}$ is an $n \times 1$ vector of centripetal and Coriolis forces
 $\bar{\mathbf{g}}(\mathbf{q})$ is an $n \times 1$ vector of gravitational effects
 $\mathbf{f}_u(\mathbf{q})$ is the remaining uncertain part of the manipulator dynamics
 $\boldsymbol{\lambda}$ is a vector of contact forces (i.e. the vector of Lagrange multipliers)
 \mathbf{J} is the Jacobian matrix associated with the contact force geometry (and a function of \mathbf{q}).

It is further assumed that the norms of $\mathbf{f}_u(\mathbf{q})$ and $\partial \mathbf{f}_u(\mathbf{q})/\partial \mathbf{q}$ are bounded.

The constraint equations reduce the number of degrees of freedom to

$$n_1 \equiv n - m \quad (4)$$

where the *reduced position variable* $\mathbf{q}_1(t) \in \mathfrak{R}^{n_1}$ describes the motion on the contact surface. According to the implicit function theorem, it is then possible to find a function $\gamma(\cdot)$ such that

$$\mathbf{q}_2 = \gamma(\mathbf{q}_1) \quad (5)$$

where $\mathbf{q}_2(t) \in \mathfrak{R}^m$ represents the dependent variables, and $\mathbf{q} = [\mathbf{q}_1^T \quad \mathbf{q}_2^T]^T$.

The robot satisfies reduced order dynamics while its motion is constrained along the surface. Therefore, define the extended Jacobian as

$$\bar{\mathbf{J}}(\mathbf{q}_1) \equiv \begin{bmatrix} \mathbf{I}_{n_1} \\ \frac{\partial \gamma}{\partial \mathbf{q}_1} \end{bmatrix} \quad (6)$$

where \mathbf{I}_{n_1} is the $n_1 \times n_1$ identity matrix. The relationships between the tangential velocity and acceleration vectors to the full joint velocity and acceleration vectors are given by:

$$\begin{aligned} \dot{\mathbf{q}} &= \bar{\mathbf{J}}(\mathbf{q}_1)\dot{\mathbf{q}}_1 \\ \ddot{\mathbf{q}} &= \bar{\mathbf{J}}(\mathbf{q}_1)\ddot{\mathbf{q}}_1 + \dot{\bar{\mathbf{J}}}(\mathbf{q}_1)\dot{\mathbf{q}}_1 \end{aligned} \quad (7)$$

By substituting the above expressions into eqn. 3 and premultiplying by the transpose of $\bar{\mathbf{J}}$, the following reduced order dynamics are given on the contact surface:

$$\bar{\mathbf{H}}\ddot{\mathbf{q}}_1 + \bar{\mathbf{C}}\dot{\mathbf{q}}_1 + \bar{\mathbf{g}}' + \bar{\mathbf{f}}_u = \bar{\mathbf{J}}^T \boldsymbol{\tau} \quad (8)$$

where $\bar{\mathbf{H}} = \bar{\mathbf{J}}^T \mathbf{H}(\mathbf{q}_1) \bar{\mathbf{J}}$, $\bar{\mathbf{C}} = \bar{\mathbf{J}}^T [\mathbf{C}(\mathbf{q}_1, \dot{\mathbf{q}}_1) \bar{\mathbf{J}} + \mathbf{H}(\mathbf{q}_1) \dot{\bar{\mathbf{J}}}]$, $\bar{\mathbf{g}}' = \bar{\mathbf{J}}^T \bar{\mathbf{g}}$ and $\bar{\mathbf{f}}_u = \bar{\mathbf{J}}^T \mathbf{f}_u$. Note that the following property has been used (which is straightforward to show):

$$\mathbf{J}(\mathbf{q}_1) \bar{\mathbf{J}}(\mathbf{q}_1) = \mathbf{0} \quad (9)$$

and that $\dot{\bar{\mathbf{H}}} - 2\dot{\bar{\mathbf{C}}}$ is skew-symmetric.

Now define the force error as follows:

$$\tilde{\boldsymbol{\lambda}} = \boldsymbol{\lambda}_d - \boldsymbol{\lambda} \quad (10)$$

where $\boldsymbol{\lambda}(t)$ is the normal force and $\boldsymbol{\lambda}_d$ is the desired force normal to the contact surface measured in a co-ordinate frame attached to the surface. In addition, the *force gain* will be defined as \mathbf{K}_f , which is a positive-definite matrix.

Theorem 1. On-line training of single serial chain with surface contact constraints: Consider the following robot dynamics:

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \bar{\mathbf{g}}(\mathbf{q}) + \mathbf{f}_u(\dot{\mathbf{q}}) = \boldsymbol{\tau} + \mathbf{J}^T \boldsymbol{\lambda} \quad (11)$$

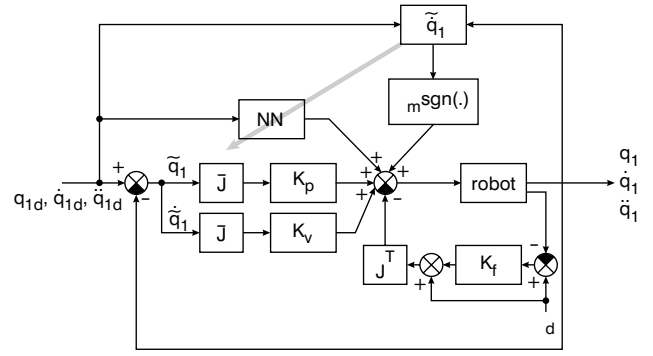


Fig. 1 Hybrid position/force control structure with neural controller

with the following control law:

$$\begin{aligned} \boldsymbol{\tau} &= \mathbf{K}_p \bar{\mathbf{J}} \tilde{\mathbf{q}}_1 + \mathbf{K}_v \bar{\mathbf{J}} \dot{\tilde{\mathbf{q}}}_1 + \mathbf{G}(\mathbf{q}_{1d}, \dot{\mathbf{q}}_{1d}, \ddot{\mathbf{q}}_{1d})^T \hat{\mathbf{w}} \\ &\quad + \varepsilon_m \text{sgn}(\dot{\tilde{\mathbf{q}}}) - \mathbf{J}^T (\boldsymbol{\lambda}_d + \mathbf{K}_f \tilde{\boldsymbol{\lambda}}) \end{aligned} \quad (12)$$

and the following neural network weight update function:

$$\dot{\hat{\mathbf{w}}} = \Gamma \mathbf{G}(\mathbf{q}_{1d}, \dot{\mathbf{q}}_{1d}, \ddot{\mathbf{q}}_{1d}) \bar{\mathbf{J}} \tilde{\mathbf{q}}_1 \quad (13)$$

where the desired independent joint values and their time derivatives are given by \mathbf{q}_{1d} , $\dot{\mathbf{q}}_{1d}$ and $\ddot{\mathbf{q}}_{1d}$, respectively. Errors in joint positions and velocities are denoted by $\tilde{\mathbf{q}}_1 \equiv \mathbf{q}_{1d} - \mathbf{q}_1$ and $\dot{\tilde{\mathbf{q}}}_1 \equiv \dot{\mathbf{q}}_{1d} - \dot{\mathbf{q}}_1$, respectively. ε_m is a constant representing the supremum of the neural network approximation error, and can be as small as possible by carefully choosing the neural network structure.

If the PD and force gains, \mathbf{K}_p , \mathbf{K}_v and \mathbf{K}_f , are positive-definite matrices and sufficiently large, then the closed-loop closed-chain serial linkage system is asymptotically stable with respect to time t , and

$$\lim_{t \rightarrow \infty} \tilde{\mathbf{q}} = \mathbf{0}, \quad \lim_{t \rightarrow \infty} \dot{\tilde{\mathbf{q}}} = \mathbf{0} \quad (14)$$

The proof is given in the Appendix (Section 12.1). The control structure is shown in Fig. 1.

4 Neural control of walking robots

Walking robots belong to a class of systems that are complex, multi-input multi-output (MIMO), crosscoupled, nonlinear, nonholonomic structures with varying topologies of closed kinematic chains. Needless to say, the control of such systems is difficult. If they are intended for industrial applications, for instance in the transportation of people or dangerous substances, it is vitally important that the joint level controllers are guaranteed to be stable. The joint level control scheme here fits into a hierarchical control architecture for autonomous walking robots that has been developed in previous work (e.g. [4]). However, the joint control scheme stands alone, and can be employed in a wider class of systems than walking robots, for instance, Stewart platforms, and parallel manipulators co-operating in automated assembly (to turn a car upside down, for example).

Various control schemes have been used in walking robots research. They range from classical controllers [16, 17], force control [18, 19], impedance control [20] and stabilising controllers [21, 22]. Relatively few researchers provide a theoretical basis for their control strategies. Notable exceptions include: [16] (monopods); [22, 23] (bipeds); [17, 18, 24, 25] (hexapods). Monopods and bipeds can be more difficult to control than hexapods

because of the demands of dynamically stable locomotion (it is always possible to keep three legs of a hexapod on the ground). However, it is more difficult to show that the controller of a hexapod is stable, because the topology is more complex and the number of degrees of freedom is typically larger. To date, no guarantee of stability exists for walking system controllers for any walking system more complex than a biped (see, for instance, [26]). Before describing the controller, it is first necessary to consider the hexapod dynamics.

5 Hexapod mechanics

The most articulate hexapod robots consist of three active degrees of freedom per leg, and 19 rigid bodies (the three links per leg, and the central body). This 'tree' structure can be considered as a system with 24 generalised co-ordinates: the 18 joint values and the six degrees of freedom associated with the central body. A diagram of the structure is shown in Fig. 2, identifying the main vectors and conventional leg labelling.

Consider the hexapod equations of motion adapted from [18]:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \bar{\mathbf{g}}(\mathbf{q}) + \mathbf{f}_u = \boldsymbol{\varphi} \quad (15)$$

where \mathbf{q} is the vector of generalised co-ordinates. \mathbf{M} is the $n \times n$ symmetric positive definite inertia tensor matrix (here $n=24$), \mathbf{C} is the $n \times n$ matrix of Coriolis and centripetal forces, the effect of gravity is $\bar{\mathbf{g}}$, \mathbf{f}_u is an $n \times 1$ vector giving the remaining uncertain part of the system dynamics (and may model the Coulomb friction, viscous friction and systematic dynamic uncertainties in the actuators or in the inertia tensor matrix). Note the following skew-symmetric property holds:

$$\dot{\mathbf{M}} = 2\mathbf{C} \quad (16)$$

$\boldsymbol{\varphi}$ is an $n \times 1$ generalised force vector given by

$$\boldsymbol{\varphi} = \mathbf{J}_F^T \cdot \mathbf{f} + \mathbf{J}_T^T \cdot \boldsymbol{\tau} \quad (17)$$

where \mathbf{f} is the $l \times 1$ vector of active contact forces, $\boldsymbol{\tau}$ is the $m \times 1$ vector of active joint torques (here $l=m=18$) and \mathbf{J}_F and \mathbf{J}_T are the Jacobians of translation and rotation, respectively, which project the torques and contact forces into the 24 degree-of-freedom generalised co-ordinate space.

The set of eqns. 15 is underspecified, representing 24 equations for 36 unknowns (in $\boldsymbol{\tau}$ and \mathbf{f}). Optimising quadratic constraints are provided by a criterion $C(\boldsymbol{\tau}, \mathbf{f})$ which is coupled to the equations of motion by a vector of Lagrangian multipliers, as expressed in the following Lagrangian function L :

$$\min_{\forall \boldsymbol{\tau}, \mathbf{f}, \boldsymbol{\lambda}} (L = C(\boldsymbol{\tau}, \mathbf{f}) + \boldsymbol{\lambda}_1^T [\boldsymbol{\varphi} - (\mathbf{M} \cdot \ddot{\mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} + \bar{\mathbf{g}} + \mathbf{f}_u)] + \boldsymbol{\lambda}_2^T [\mathbf{U} \cdot \mathbf{f}]) \quad (18)$$

where \mathbf{U} is the gait distribution matrix consisting of a '1' when a leg is in the air and a '0' when it is in contact with the ground, in order to zero force components for legs in the swing phase. $\boldsymbol{\lambda} = [\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2]^T$ is the vector of Lagrange multipliers. $\boldsymbol{\lambda}_1$ has 24 elements and $\boldsymbol{\lambda}_2$ is of size $3 \times k$ elements, where k is the number of legs not in contact with the ground. Note that these equations can be extended or reduced to apply to the whole class of systems represented by the hexapod, which includes the Stewart platform (also sometimes called a 'hexapod'), and parallel manipulators, where the object of manipulation is analogous to the walking robot's central body.

6 Hybrid position/force neural control of hexapods

It is proposed here to extend to the whole hexapod the neural network hybrid position/force control structure for a robotic manipulator similar to that described above. There are a number of important differences that need to be taken into account and a couple of assumptions need to be stated. The obvious difference is that of a significant increase in complexity. The second difference is that because the 'world frame' is not embedded in the robot, as it can be with a manipulator (the 'base' frame), there is the addi-

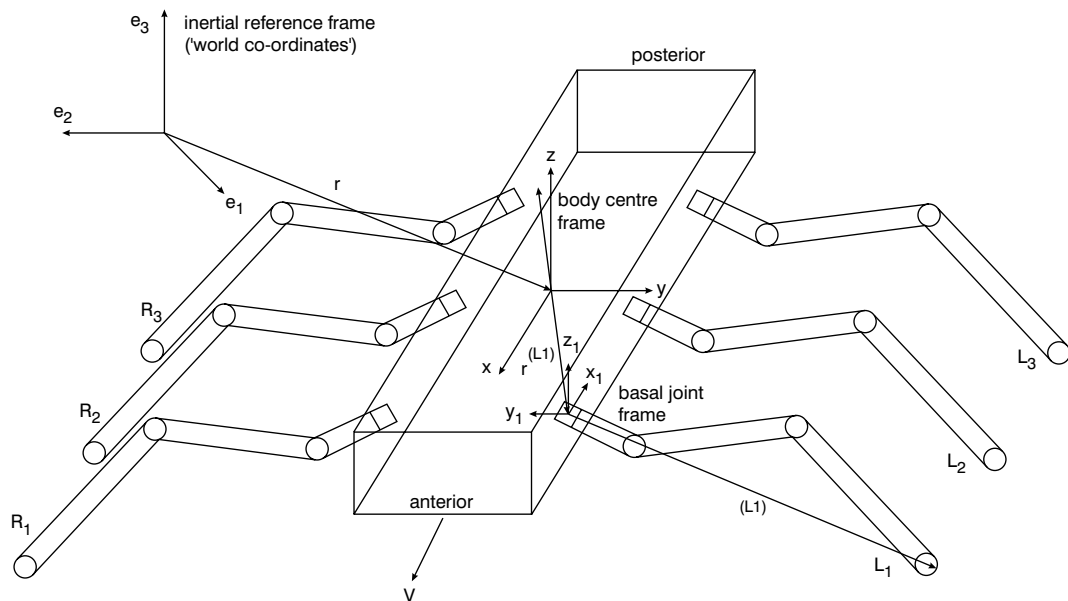


Fig. 2 Hexapod and associated vectors

\mathbf{r} is position of body centre in terms of inertial frame of 'world co-ordinates' ($\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$), $\mathbf{r}^{(j)}$ is position of basal joint of j th leg, e.g. L1, and $\mathbf{p}^{(j)}$ is endpoint vector of leg j . Also shown are angular velocity $\boldsymbol{\omega}$ and body centre velocity \mathbf{v} . Legs are labelled 'L' for left and 'R' for right, with front, middle and hind being labelled '1' to '3', respectively; hence 'L1' is front left leg

tional angular velocity Jacobian that premultiplies the torques (see eqn. 17). This makes the controller somewhat more complex.

There are three assumptions made. The first assumption concerns the nature of τ . If the equations of motion are expressed in terms of the ‘projected torques’, i.e. the projection of the 18-dimensional vector τ into the 24-dimensional space of the generalised co-ordinates such that

$$\mathbf{t} = \mathbf{J}_T^T \boldsymbol{\tau} \quad (19)$$

then it is possible to structure the dynamics in a form that compares with the neural network hybrid position/force manipulator controller presented by [8]. There are three differences: (i) that viscous friction terms have not been explicitly considered in the hexapod, whereas uncertain dynamic terms have; (ii) external forces are considered in the hexapod case—although this is not a significant difficulty; (iii) the hexapod equations are underconstrained as they stand. This point leads to the second assumption: it is possible to express the constraints represented by eqn. 18 as a constraint force vector. If the equations of motion are thus augmented, we obtain

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \bar{\mathbf{g}}(\mathbf{q}) + \mathbf{f}_u - \mathbf{J}_F^T \mathbf{f} = \mathbf{t} + \mathbf{J}^T \boldsymbol{\pi} \quad (20)$$

where \mathbf{J} is the Jacobian matrix giving the force in generalised co-ordinates representing the constraint force surface geometry due to $\boldsymbol{\pi}$, the force-torque constraints given by

$$\boldsymbol{\pi} = \mathbf{J}_F^T \boldsymbol{\pi}_F + \mathbf{J}_T^T \boldsymbol{\pi}_T \quad (21)$$

where $\boldsymbol{\pi}_F$ are the constrained forces and $\boldsymbol{\pi}_T$ are the constraints on the torques. Note that, in general, although $\boldsymbol{\pi}$ is itself a vector of Lagrangian multipliers, it is different from $\boldsymbol{\lambda}$, the Lagrangian multiplier vector in eqn. 18. Note also that, for every leg configuration and gait, there will be a different $\boldsymbol{\pi}$, which leads to a third assumption, that such a vector can be found in each of the different cases. Since it is reasonable to define a $\boldsymbol{\lambda}$ for each leg configuration, then it is reasonable to assume that a $\boldsymbol{\pi}$ can be found.

With all the above assumptions and formalism, it is now possible to follow a similar line of argument followed for the single serial case. The constraint equations reduce the number of degrees of freedom to $n_1 \equiv n - m$, where the *reduced position variable* $\mathbf{q}_1(t) \in \mathfrak{R}^{n_1}$ describes the motion of the hexapod on the substrate (contact surface). According to the implicit function theorem, it is then possible to find a function $\gamma(\cdot)$ such that $\mathbf{q}_2 = \gamma(\mathbf{q}_1)$ and

$$\mathbf{q} = [\mathbf{q}_1^T \quad \mathbf{q}_2^T]^T.$$

Note that n_1 and γ will be different for different gaits.

Define the extended Jacobian matrix $\bar{\mathbf{J}}$ and its relationship to the vectors $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ as follows:

$$\bar{\mathbf{J}}(\mathbf{q}_1) \equiv \begin{bmatrix} \mathbf{I}_{n_1} \\ \frac{\partial \gamma}{\partial \mathbf{q}_1} \end{bmatrix} \Leftrightarrow \begin{cases} \dot{\mathbf{q}} = \bar{\mathbf{J}}(\mathbf{q}_1) \dot{\mathbf{q}}_1 \\ \ddot{\mathbf{q}} = \bar{\mathbf{J}}(\mathbf{q}_1) \ddot{\mathbf{q}}_1 + \dot{\bar{\mathbf{J}}}(\mathbf{q}_1) \dot{\mathbf{q}}_1 \end{cases} \quad (22)$$

where \mathbf{I}_{n_1} is the $n_1 \times n_1$ identity matrix. These vectors will be different for different gaits and leg configurations.

Substitute eqn. 22 into eqn. 20, premultiply by $\bar{\mathbf{J}}^T$, and use the fact that $\mathbf{J}(\mathbf{q}_1)\bar{\mathbf{J}}(\mathbf{q}_1) = 0$ (which is simple to show) to give the following contact dynamics:

$$\bar{\mathbf{M}}\ddot{\mathbf{q}}_1 + \bar{\mathbf{C}}\dot{\mathbf{q}}_1 + \bar{\mathbf{g}}' + \bar{\mathbf{f}}_u - \bar{\mathbf{f}} = \bar{\mathbf{J}}^T \mathbf{t} \quad (23)$$

where

$$\begin{aligned} \bar{\mathbf{M}} &= \bar{\mathbf{J}}^T \mathbf{M}(\mathbf{q}_1) \bar{\mathbf{J}}, \quad \bar{\mathbf{C}} = \bar{\mathbf{J}}^T [\mathbf{C}(\mathbf{q}_1, \dot{\mathbf{q}}_1) \bar{\mathbf{J}} + \mathbf{M}(\mathbf{q}_1) \dot{\bar{\mathbf{J}}}] \\ \bar{\mathbf{g}}' &= \bar{\mathbf{J}}^T \bar{\mathbf{g}}, \quad \bar{\mathbf{f}}_u = \bar{\mathbf{J}}^T \mathbf{f}_u, \quad \bar{\mathbf{f}} = \bar{\mathbf{J}}^T \mathbf{J}_F^T \mathbf{f} \end{aligned}$$

Now define the error in the generalised contact forces $\boldsymbol{\pi}$ as

$$\tilde{\boldsymbol{\pi}} = \boldsymbol{\pi}_d - \boldsymbol{\pi} = \mathbf{J}_F^T (\boldsymbol{\pi}_{Fd} - \boldsymbol{\pi}_F) + \mathbf{J}_T^T (\boldsymbol{\pi}_{Td} - \boldsymbol{\pi}_T) \quad (24)$$

where $\boldsymbol{\pi}_{Fd}$ and $\boldsymbol{\pi}_{Td}$ are the desired constraints on the forces and torques, respectively. In addition, the force gain \mathbf{K}_f acts on this error and is positive definite.

Theorem 2. On-line training of hexapod using reduced-order dynamics: Consider the hexapod dynamics of eqn. 15, together with the following control law:

$$\begin{aligned} \boldsymbol{\tau} &= \mathbf{K}_p \bar{\mathbf{J}} \tilde{\mathbf{q}}_1 + \mathbf{K}_v \dot{\bar{\mathbf{J}}} \tilde{\mathbf{q}}_1 \\ &\quad + \mathbf{G}(\mathbf{q}_{1d}, \dot{\mathbf{q}}_{1d}, \ddot{\mathbf{q}}_{1d})^T \hat{\mathbf{w}} \\ &\quad + \varepsilon_m \text{sgn}(\tilde{\mathbf{q}}_1) - (\mathbf{J}_T^T)^{-1} \mathbf{J}^T (\boldsymbol{\pi}_d + \mathbf{K}_f \tilde{\boldsymbol{\pi}}) \end{aligned} \quad (25)$$

together with the following neural network weight update law:

$$\dot{\hat{\mathbf{w}}} = \Gamma \mathbf{G}(\mathbf{q}_{1d}, \dot{\mathbf{q}}_{1d}, \ddot{\mathbf{q}}_{1d}) \mathbf{J}_T^T \dot{\bar{\mathbf{J}}} \tilde{\mathbf{q}}_1 \quad (26)$$

where $(\mathbf{J}_T^T)^{-1}$ is a right inverse of \mathbf{J}_T^T , which could be the Moore-Penrose pseudo-inverse

$$(\mathbf{J}_T^T)^+ \equiv (\mathbf{J}^T \mathbf{J})^{-1} \mathbf{J}^T$$

If the PD and force gains, \mathbf{K}_p , \mathbf{K}_v and \mathbf{K}_f , are positive-definite matrices and sufficiently large, then the closed-loop system is ‘stable in the sense of Lyapunov’ and as long as the gains are large enough such that the contact forces satisfy the following inequality:

$$\|\mathbf{f}\| < \frac{\eta_1}{\eta_2} \left[\lambda_{\min}(\mathbf{K}_v) \|\tilde{\mathbf{q}}_1\| + \lambda_{\min}(\mathbf{K}_p) \|\tilde{\mathbf{q}}_1\| \right] \quad (27)$$

where

$$\eta_1 = \sup_{\mathbf{q}_1 \in \mathfrak{R}^{n_1}} \|\bar{\mathbf{J}}\|^2 \sup_{\mathbf{q} \in \mathfrak{R}^n} \|\mathbf{J}_T\| \quad (28)$$

$n = 24, n_1$ depends on gait and

$$\eta_2 = \sup_{\mathbf{q} \in \mathfrak{R}^n} \|\mathbf{J}_F\| \sup_{\mathbf{q}_1 \in \mathfrak{R}^{n_1}} \|\bar{\mathbf{J}}\| \quad (29)$$

If, in addition, it can be shown that the following inequality holds:

$$\begin{aligned} \|\dot{\mathbf{f}}\| &< \frac{1}{\eta_4} \left[\eta_1 \lambda_{\min}(\mathbf{K}_v) \left(2 \|\tilde{\mathbf{q}}_1\| \|\dot{\tilde{\mathbf{q}}}_1\| - \frac{\eta_4}{\eta_2} \|\dot{\tilde{\mathbf{q}}}_1\| \right) \right. \\ &\quad \left. + \eta_1 \lambda_{\min}(\mathbf{K}_p) \left(2 \|\tilde{\mathbf{q}}_1\|^2 - \frac{\eta_4}{\eta_2} \|\tilde{\mathbf{q}}_1\| \right) \right. \\ &\quad \left. + \eta_3 \left(\lambda_{\min}(\mathbf{K}_v) \|\dot{\tilde{\mathbf{q}}}_1\|^2 + \lambda_{\min}(\mathbf{K}_p) \|\tilde{\mathbf{q}}_1\| \|\dot{\tilde{\mathbf{q}}}_1\| \right) \right] \end{aligned} \quad (30)$$

where

$$\begin{aligned} \eta_3 &= 2 \sup_{\mathbf{q}_1 \in \mathfrak{R}^{n_1}} \|\bar{\mathbf{J}}\| \sup_{\mathbf{q}_1, \dot{\mathbf{q}}_1 \in \mathfrak{R}^{n_1}} \|\dot{\bar{\mathbf{J}}}\| \sup_{\mathbf{q} \in \mathfrak{R}^n} \|\mathbf{J}_T\| \\ &\quad + \sup_{\mathbf{q}_1 \in \mathfrak{R}^{n_1}} \|\bar{\mathbf{J}}\| \sup_{\mathbf{q}, \dot{\mathbf{q}} \in \mathfrak{R}^n} \|\dot{\mathbf{J}}_T\| \end{aligned} \quad (31)$$

and

$$\eta_4 = \sup_{\mathbf{q}_1, \dot{\mathbf{q}}_1 \in \mathfrak{R}^{n_1}} \|\dot{\bar{\mathbf{J}}}\| \sup_{\mathbf{q} \in \mathfrak{R}^n} \|\mathbf{J}_F\| + \sup_{\mathbf{q}_1 \in \mathfrak{R}^{n_1}} \|\bar{\mathbf{J}}\| \sup_{\mathbf{q}, \dot{\mathbf{q}} \in \mathfrak{R}^n} \|\dot{\mathbf{J}}_F\| \quad (32)$$

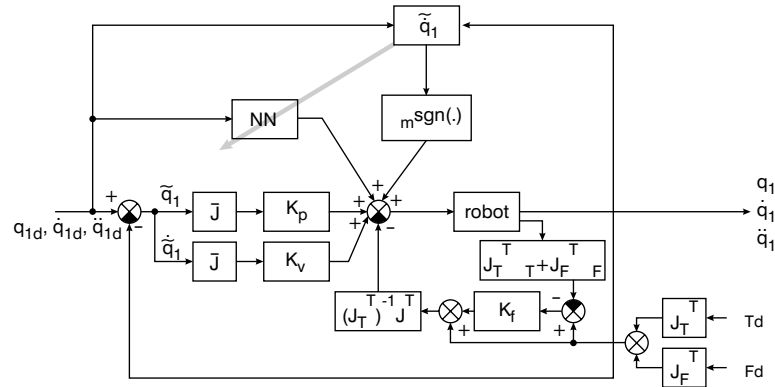


Fig. 3 Hybrid position/force controller; showing PD controller, neural network, sliding mode controller used to make neural network output robust to approximation error, and application of generalised constraint force-torques

then the closed-loop control system is asymptotically stable regardless of leg configurations or gait. The proof is given in the Appendix (Section 12.2). The control structure is shown in Fig. 3.

Some qualifying remarks need to be made concerning this theorem. eqn. 27 may seem quite a punitive restriction on possible values of the contact forces \mathbf{f} . The optimisation criterion, eqn. 18, which aims to minimise \mathbf{f} , means that this condition can only really be satisfied by high values of \mathbf{K}_v and \mathbf{K}_p . In practice choosing these gains is not too difficult, since the required values to satisfy these inequalities are typically well within valid ranges that could be chosen if the corresponding conventional hybrid position/force controller were to be used on its own without the neural network. However, the choice of contact force optimisation strategy is very important to avoid excessively high gains yet still fulfil eqn. 27.

In extending the hybrid position/force controller, the starting point was to project the torques into the space of the generalised co-ordinates and then assume that the whole system dynamics could be reduced to contact surface dynamics by augmenting the Lagrangian with multiplier constraints. An alternative approach to that taken here was described in [27]. This second, also novel, approach takes a different starting point – that of projecting the neural networks associated with the joint torques into the space of generalised co-ordinates, such that each generalised co-ordinate is ‘controlled’ by one of the neural network projections.

7 Novel neural control structure

The controller developed in this section extends the use of the neural control structure for robotic manipulators first developed by [1]. It must not automatically be assumed that the bounded-input bounded-output (BIBO) controllers such as those described by [1] are transferable to a walking machine, for two reasons. First the serial chain linkage of a robotic manipulator is fixed at one end. In the case of the walking machine, any number of legs can be on the ground; thus the system is fundamentally different structurally. For a hexapod, up to 15 closed kinematic chains can be obtained, but the exact number depends on the machine’s gait and also can vary during walking. Both the body and the ground contact force constraints are part of the control system. Thus, the contact of the foot with the ground effectively creates another ‘joint’ which does not exist in the original BIBO serial linkage. Secondly, in developing the proof of stability, the neural network

vectors describe the whole system, not just a collection of six individually controlled BIBO legs. Note that the controller presented here does not guarantee statically or dynamically stable walking gaits: that is the purpose of the higher levels in the layered control hierarchy described in [4].

By exploiting the universal approximation property of neural networks, it is possible to represent the left-hand side of eqn. 15 as follows:

$$\begin{aligned} \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \bar{\mathbf{g}}(\mathbf{q}) + \mathbf{f}_u \\ = \mathbf{G}^v(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})^T \mathbf{w}^v + \boldsymbol{\zeta}^v(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}) \end{aligned} \quad (33)$$

where \mathbf{w}^v is the unknown neural network weight vector, \mathbf{G}^v is the nonlinear mapping matrix and $\boldsymbol{\zeta}^v$ is the neural network approximation error. These neural networks will be called ‘virtual neural networks’ – hence the superscript v . Two assumptions are made:

1. The inputs to these neural networks are bounded.
2. The constraints represented by eqn. 18 give bounded values for the torques and forces. If the above assumptions are valid (and it is straightforward to show that they are), then the following theorem results.

Theorem 3. On-line learning neural control of systems with closed kinematic chains using virtual neural networks: Consider the dynamics of eqn. 15 for the hexapod exemplar with the following control law:

$$\begin{aligned} \boldsymbol{\varphi} = \mathbf{K}_p \tilde{\mathbf{q}} + \mathbf{K}_v \dot{\tilde{\mathbf{q}}} \\ + \mathbf{G}^v(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d)^T \hat{\mathbf{w}}^v + e_m^v \text{sgn}(\tilde{\mathbf{q}} + c\tilde{\mathbf{q}}) \end{aligned} \quad (34)$$

with the following update law:

$$\hat{\mathbf{w}}^v = \boldsymbol{\Gamma}^v \mathbf{G}^v(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d)(\tilde{\mathbf{q}} + c\tilde{\mathbf{q}}) - \frac{1}{2} \boldsymbol{\Gamma}^v (\dot{\boldsymbol{\Gamma}}^v)^{-1} \hat{\mathbf{w}}^v \quad (35)$$

If the PD gains \mathbf{K}_p and \mathbf{K}_v are positive definite matrices and sufficiently large, and c is a small positive constant, then the closed-loop controller is asymptotically stable, and $\lim_{t \rightarrow \infty} \tilde{\mathbf{q}} = 0$, $\lim_{t \rightarrow \infty} \dot{\tilde{\mathbf{q}}} = 0$. The proof is presented in the Appendix (Section 12.3).

A simplified form of the neural control structure is given in Fig. 4, showing the neural network (NN), the sliding mode controller, a PD controller and the desired, actual and error vectors of the generalised co-ordinates (respectively: \mathbf{q}_d , \mathbf{q} , $\tilde{\mathbf{q}} = \mathbf{q}_d - \mathbf{q}$) and their derivatives.

It is now necessary to examine the relationship between the virtual neural networks and the actual neural control

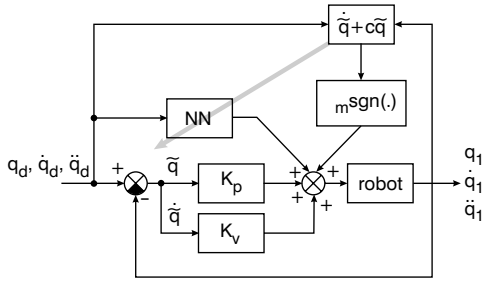


Fig. 4 Neural controller

Desired position, velocity and acceleration vectors are fed into neural network and a standard PD controller. Error between actual values and desired values is used to train network, and also forms input to sliding controller $\varepsilon_m \text{sgn}(\cdot)$.

schemes controlling each joint. The torque control law could be given by

$$\tau = \mathbf{K}'_p \tilde{\mathbf{q}}_1 + \mathbf{K}'_v \dot{\tilde{\mathbf{q}}}_1 + \mathbf{G}(\mathbf{q}_{1d}, \dot{\mathbf{q}}_{1d}, \ddot{\mathbf{q}}_{1d})^T \hat{\mathbf{w}} + \varepsilon_m \text{sgn}(\tilde{\mathbf{q}}_1 + c\tilde{\mathbf{q}}_1) \quad (36)$$

where the primes have been added to distinguish the gains from the generalised gain matrices in eqn. 23, and the torques are functions of the actuated joints \mathbf{q}_1 , which is a subset of the generalised co-ordinates \mathbf{q} . For a hexapod, this relationship is as follows:

$$\mathbf{q} = [\mathbf{r}^T \quad \mathbf{v}^T \quad \mathbf{q}_1^T]^T \quad (37)$$

where \mathbf{r} is the position of the body centre in a chosen external frame, \mathbf{v} is the vector of pitch, yaw and roll angles of the body centre, and \mathbf{q}_1 is the vector of joint angles. The relationship between the velocities can be specified as follows:

$$\dot{\mathbf{q}} = \begin{bmatrix} \frac{\partial \gamma}{\partial \mathbf{q}_1} \\ \mathbf{I}_{18} \end{bmatrix} \dot{\mathbf{q}}_1 = \mathbf{J} \dot{\mathbf{q}}_1 \Rightarrow \dot{\mathbf{q}}_1 = \mathbf{J}^{-1} \dot{\mathbf{q}} \quad (38)$$

where \mathbf{J} is a 24×18 dimensional Jacobian. According to the implicit function theorem, it is possible to find a function $\gamma(\cdot)$ such that

$$\mathbf{q}_2 = \gamma(\mathbf{q}_1) = [\mathbf{r}^T \quad \mathbf{v}^T]^T \quad (39)$$

The inverse of \mathbf{J} could be the Moore-Penrose pseudo-inverse $(\mathbf{J})^+ \equiv (\mathbf{J}^T \mathbf{J})^{-1} \mathbf{J}^T$. Note that, for certain stationary systems, \mathbf{J} could be equal to \mathbf{J}_T .

Typically a torque neural control law like eqn. 36 would have an update law equivalent to

$$\dot{\hat{\mathbf{w}}} = \Gamma \mathbf{G}(\mathbf{q}_{1d}, \dot{\mathbf{q}}_{1d}, \ddot{\mathbf{q}}_{1d}) \mathbf{J}^{-1} (\tilde{\mathbf{q}} + c\tilde{\mathbf{q}}) \quad (40)$$

By combining eqns. 39 and 36 with eqn. 17, using the assumption that the 'virtual neural networks' have an LE structure, then by choosing the simplest possible neural network structure for each of the 24 virtual neural networks (i.e. each is a single neuron $\Rightarrow \mathbf{G}^v = \mathbf{I}_{24}$), the following identities can be found relating the virtual neural network structure to the actual neural network structure. The bound on the neural network approximation error is

$$\xi^v = \mathbf{J}_T^T \xi \Rightarrow \|\xi^v\| \leq \sup_{\mathbf{q} \in \mathfrak{R}^m} \|\mathbf{J}_T\| \varepsilon_m \quad (41)$$

The virtual neural network weight update function is

$$\dot{\hat{\mathbf{w}}}^v = \mathbf{J}_T^T \mathbf{G}^T \hat{\mathbf{w}} + (\mathbf{J}_T^T \mathbf{G}^T \Gamma \mathbf{G} \mathbf{J}^{-1}) (\tilde{\mathbf{q}} + c\tilde{\mathbf{q}}) + \mathbf{J}_F^T \mathbf{f} + \mathbf{J}_F^T \dot{\mathbf{f}} \quad (42)$$

The virtual neural network learning rate and its derivative are

$$\Gamma^v = \mathbf{J}_T^T \mathbf{G}^T \Gamma \mathbf{G} \mathbf{J}^{-1}, \dot{\Gamma}^v = \dot{\mathbf{J}}_T^T \mathbf{G}^T \Gamma \mathbf{G} \mathbf{J}^{-1} + \mathbf{J}_T^T \mathbf{G}^T \Gamma \dot{\mathbf{G}} \mathbf{J}^{-1} \quad (43)$$

The generalised co-ordinate gains are

$$\mathbf{K}_p = \mathbf{J}_T^T \mathbf{K}'_p \mathbf{J}^{-1}, \mathbf{K}_v = \mathbf{J}_T^T \mathbf{K}'_v \mathbf{J}^{-1} \quad (44)$$

Since all these terms are bounded, and [1] has demonstrated the asymptotic stability of eqn. 36, then Theorem 3 establishes the stability (and some control system design heuristics) of the neural control of complex structures like a hexapod robot. The 'virtual neural networks' are therefore essentially a projection of the actual neural networks in the controlled co-ordinates space to the larger space of generalised co-ordinates of the whole system.

8 Neural control of discrete systems

The above on-line learning algorithm guarantees that, for a continuous-time system, the errors in the joint positions and velocities remain bounded, and that the closed-loop hexapod system is asymptotically stable. For implementation on a real industrial robot using affordable controllers, a continuous-time system is still unrealistic. Although analogue neural hardware exists (e.g. Intel Ni1000 and later technology), it is much more affordable to implement the neural network in software on a lower cost DSP chip or similar processor. For instance, in our prototype experimental hexapod, a Siemens C164CI microprocessor controls each leg. It is thus necessary to extend the control structures and on-line learning algorithms so that they can be implemented on digital systems. It is therefore necessary to discretise the time domain. Let us first consider the manipulator case, where the linkage is serial and only constrained at one end.

Let T be the sampling period and $\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k$ be the desired joint values at the k th sampling point, i.e.

$$\mathbf{q}_d^k = \mathbf{q}_d(kT) \quad (45)$$

$$\dot{\mathbf{q}}_d^k = \dot{\mathbf{q}}_d(kT) \quad (46)$$

$$\ddot{\mathbf{q}}_d^k = \ddot{\mathbf{q}}_d(kT) \quad (47)$$

For any time t , there always exists an integer k so that t is between the k th and the $(k+1)$ th sampling points; therefore is it assumed in the following discussion that $kT \leq t \leq (k+1)T$. The modified manipulator dynamics, based on the desired joint values, is as follows:

$$\begin{aligned} \mathbf{H}(\mathbf{q}_d) \ddot{\mathbf{q}}_d + \mathbf{C}(\mathbf{q}_d, \dot{\mathbf{q}}_d) \dot{\mathbf{q}}_d + \bar{\mathbf{g}}(\mathbf{q}_d) + \mathbf{f}_u(\mathbf{q}_d) \\ = \mathbf{H}(\mathbf{q}_d^k) \ddot{\mathbf{q}}_d^k + \mathbf{C}(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k) \dot{\mathbf{q}}_d^k + \mathbf{g}(\mathbf{q}_d^k) \\ + \mathbf{f}_u(\mathbf{q}_d^k) + \zeta(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d, \mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k) \end{aligned} \quad (48)$$

where the term on the left-hand side is the manipulator dynamics based on the desired joint values and used in the proof of the on-line learning algorithm, and the last term on the right-hand side is defined simply as

$$\begin{aligned} \zeta(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d, \mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k) \\ = (\mathbf{H}(\mathbf{q}_d) \ddot{\mathbf{q}}_d + \mathbf{C}(\mathbf{q}_d, \dot{\mathbf{q}}_d) \dot{\mathbf{q}}_d + \bar{\mathbf{g}}(\mathbf{q}_d) + \mathbf{f}_u(\mathbf{q}_d)) \\ - (\mathbf{H}(\mathbf{q}_d^k) \ddot{\mathbf{q}}_d^k + \mathbf{C}(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k) \dot{\mathbf{q}}_d^k + \mathbf{g}(\mathbf{q}_d^k) + \mathbf{f}_u(\mathbf{q}_d^k)) \end{aligned} \quad (49)$$

Since the manipulator dynamics are continuous with the desired joint values $\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d$, and these desired joint

values are normally continuous functions, then it follows that

$$\lim_{T_i \rightarrow 0} \zeta(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d, \mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k) = 0 \quad (50)$$

The first five terms on the right-hand side of eqn. 48 change only at sampling points and so they can be approximated by an LE discrete neural network. Therefore,

$$\begin{aligned} & \mathbf{H}(\mathbf{q}_d)\ddot{\mathbf{q}}_d + \mathbf{C}(\mathbf{q}_d, \dot{\mathbf{q}}_d)\dot{\mathbf{q}}_d + \bar{\mathbf{g}}(\mathbf{q}_d) + \mathbf{f}_u(\mathbf{q}_d) \\ &= \mathbf{G}(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k)^T \mathbf{w} + \boldsymbol{\zeta}(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k) \\ & \quad + \boldsymbol{\zeta}(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d, \mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k) \end{aligned} \quad (51)$$

where \mathbf{G} is the discrete neural network nonlinear mapping matrix, \mathbf{w} is the desired weight vector, $\boldsymbol{\zeta}$ is the approximation error vector, which could be as small as possible by carefully choosing the neural network structure, and $\boldsymbol{\zeta}$ is the sampling error vector, which could be as small as possible by choosing a small enough sampling period. Define ε_m such that

$$\|\boldsymbol{\zeta} + \boldsymbol{\xi}\| < \varepsilon_m \quad (52)$$

and ε_m could also be as small as possible by carefully choosing the neural network structure and by choosing a small enough sampling period.

The control structure for the dynamics eqn. 51 now becomes

$$\boldsymbol{\tau} = \mathbf{K}_p \tilde{\mathbf{q}} + \mathbf{K}_d \dot{\tilde{\mathbf{q}}} + \mathbf{G}(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k)^T \hat{\mathbf{w}} + \varepsilon_m \operatorname{sgn}(\tilde{\mathbf{q}} + c\tilde{\mathbf{q}}) \quad (53)$$

and the neural network on-line learning algorithm is

$$\dot{\hat{\mathbf{w}}} = \Gamma \mathbf{G}(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k)(\tilde{\mathbf{q}} + c\tilde{\mathbf{q}}) \quad (54)$$

or

$$\hat{\mathbf{w}}(t) = \hat{\mathbf{w}}(kT) + \Gamma \mathbf{G}(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k) \int_{kT}^t (\tilde{\mathbf{q}}(\rho) + c\tilde{\mathbf{q}}(\rho)) d\rho \quad (55)$$

Substituting eqn. 55 into eqn. 53 yields

$$\begin{aligned} \boldsymbol{\tau} &= \mathbf{K}_p \tilde{\mathbf{q}} + \mathbf{K}_d \dot{\tilde{\mathbf{q}}} + (\mathbf{G}^k)^T \hat{\mathbf{w}}^k \\ & \quad + (\mathbf{G}^k)^T \Gamma (\mathbf{G}^k)^T \int_{kT}^t (\tilde{\mathbf{q}}(\rho) + c\tilde{\mathbf{q}}(\rho)) d\rho \end{aligned} \quad (56)$$

where $\mathbf{G}^k = \mathbf{G}(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k)$ and $\hat{\mathbf{w}}^k = \hat{\mathbf{w}}(kT)$, and the integral is taken over time from the beginning of the k th sampling period.

Since the control law of eqn. 56 only uses the neural network weight values at sampling points, it is sufficient to calculate these values during on-line learning; thus eqn. 55 becomes

$$\hat{\mathbf{w}}^{k+1} = \hat{\mathbf{w}}^k + \Gamma \mathbf{G}(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k) \int_{kT}^{(k+1)T} (\tilde{\mathbf{q}}(\rho) + c\tilde{\mathbf{q}}(\rho)) d\rho \quad (57)$$

The final control law and on-line learning algorithm are summarised in the following theorem.

Theorem 4. Discrete on-line training algorithm (Theorem 7.1 of Jin [1]): Consider the manipulator dynamics of eqn. 15, together with the control law of eqn. 56 and the on-line learning algorithm of eqn. 57. If the PD gains \mathbf{K}_p and \mathbf{K}_v are positive definite matrices and sufficiently large, and c is a small enough positive constant, then the closed-loop manipulator system is asymptotically stable, and

$$\lim_{t \rightarrow \infty} \tilde{\mathbf{q}} = 0, \quad \lim_{t \rightarrow \infty} \dot{\tilde{\mathbf{q}}} = 0 \quad (58)$$

The proof is given by [1] for the on-line learning algorithm of continuous-time manipulator control. The control law only requires the neural network information (inputs, outputs and nonlinear mapping vector) at sampling points and the neural network weights change only at sampling points as well. Therefore, the neural network is discrete in the time domain and can thus be implemented easily in digital hardware.

The control structure is presented in Fig. 5. In comparison with Fig. 4, the neural network is discrete; an integral controller is added to the PD controller with a gain coefficient matrix, $\mathbf{K}_I = (\mathbf{G}^k)^T \Gamma \mathbf{G}^k$. The integration is initialised at every sampling point.

The control algorithm and update law of Theorem 4 are the discrete versions of the control algorithm in eqn. 36 and update law of eqn. 40. It is straightforward to transform the sampling error such that

$$\begin{aligned} & \mathbf{M}(\mathbf{q}_d)\ddot{\mathbf{q}}_d + \mathbf{C}(\mathbf{q}_d, \dot{\mathbf{q}}_d)\dot{\mathbf{q}}_d + \bar{\mathbf{g}}(\mathbf{q}_d) + \mathbf{f}_u(\mathbf{q}_d) \\ &= \mathbf{G}^v(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k)^T \mathbf{w}^v + \boldsymbol{\zeta}^v(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k) \\ & \quad + \boldsymbol{\zeta}(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d, \mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k) \end{aligned} \quad (59)$$

where \mathbf{G}^v is the discrete virtual neural network nonlinear mapping matrix, \mathbf{w}^v is the desired weight vector, $\boldsymbol{\zeta}^v$ is the approximation error vector, which could be as small as possible by carefully choosing the neural network structure, and $\boldsymbol{\zeta}$ is the sampling error vector, which could be as small as possible by choosing a small enough sampling period. Define ε_m^v such that

$$\|\boldsymbol{\zeta} + \boldsymbol{\zeta}^v\| < \varepsilon_m^v \quad (60)$$

and ε_m^v could also be as small as possible by carefully choosing the neural network structure and by choosing a small enough sampling period.

It is less straightforward to transform the weight update law, eqn. 35, into the generalised co-ordinate space of the hexapod. The equivalent to eqn. 55 appears as follows:

$$\begin{aligned} \hat{\mathbf{w}}^v(t) &= \hat{\mathbf{w}}^v(kT) + \int_{kT}^t \Gamma^v(\rho) \mathbf{G}^v(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k)(\tilde{\mathbf{q}}(\rho) \\ & \quad + c\tilde{\mathbf{q}}(\rho)) d\rho - \frac{1}{2} \left[\int_{kT}^t \Gamma^v(\rho) (\dot{\Gamma}^v(\rho))^{-1} d\rho \right] \hat{\mathbf{w}}^v(t) \end{aligned} \quad (61)$$

By defining the following matrix:

$$\boldsymbol{\Omega} = \mathbf{1} + \frac{1}{2} \int_{kT}^t \Gamma^v(\rho) (\dot{\Gamma}^v(\rho))^{-1} d\rho \quad (62)$$

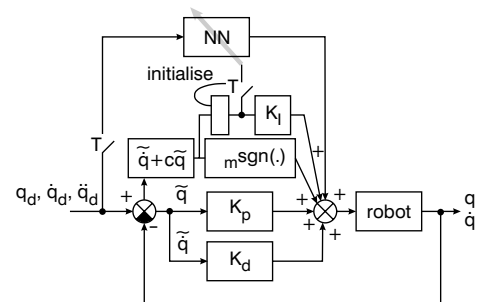


Fig. 5 Extended neural network control structure for implementation in digital hardware

it is then possible to simplify eqn. 61 to

$$\begin{aligned}\hat{\mathbf{w}}^v(t) &= \mathbf{\Omega}^{-1} \hat{\mathbf{w}}^v(kT) \\ &+ \mathbf{\Omega}^{-1} \int_{kT}^t \mathbf{\Gamma}^v(\rho) \mathbf{G}^v(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k) (\tilde{\mathbf{q}}(\rho)) \\ &+ c\tilde{\mathbf{q}}(\rho) d\rho\end{aligned}\quad (63)$$

Thus, $\mathbf{\Omega}^{-1}$ modifies the learning rate and is an effect of the projection of the neural networks into the generalised co-ordinate space. The control law, eqn. 34, therefore becomes

$$\begin{aligned}\varphi &= \mathbf{K}_p \tilde{\mathbf{q}} + \mathbf{K}_v \dot{\tilde{\mathbf{q}}} + \mathbf{G}^v(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k)^T \mathbf{\Omega}^{-1} \hat{\mathbf{w}}^v(kT) \\ &+ \mathbf{G}^v(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k)^T \mathbf{\Omega}^{-1} \int_{kT}^t \mathbf{\Gamma}^v(\rho) \mathbf{G}^v(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k) (\tilde{\mathbf{q}}(\rho)) \\ &+ c\tilde{\mathbf{q}}(\rho) d\rho + \varepsilon_m^v \operatorname{sgn}(\tilde{\mathbf{q}} + c\tilde{\mathbf{q}})\end{aligned}\quad (64)$$

Since the control law of eqn. 64 only uses the neural network weight values at sampling points, it is sufficient to calculate these values during on-line learning; thus eqn. 63 becomes

$$\begin{aligned}\hat{\mathbf{w}}^v((k+1)T) &= \mathbf{\Omega}^{-1} \hat{\mathbf{w}}^v(kT) \\ &+ \mathbf{\Omega}^{-1} \int_{kT}^t \mathbf{\Gamma}^v(\rho) \mathbf{G}^v(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k) (\tilde{\mathbf{q}}(\rho)) \\ &+ c\tilde{\mathbf{q}}(\rho) d\rho\end{aligned}\quad (65)$$

The overall control structure, as with the manipulator case (Theorem 4) is a PID controller with the integral component having a gain

$$\mathbf{K}_I = \mathbf{G}^v(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k)^T \mathbf{\Omega}^{-1} \int_{kT}^t \mathbf{\Gamma}^v(\rho) d\rho \mathbf{G}^v(\mathbf{q}_d^k, \dot{\mathbf{q}}_d^k, \ddot{\mathbf{q}}_d^k) \quad (66)$$

eqns. 59–66 make it possible to extend Theorem 3 to the discrete NN control systems for the hexapod in the same way as Theorem 4 extends the continuous time NN on-line learning algorithms presented in [1] for manipulator control. This is a very important result because it means that a carefully designed neural controller running on a digital microprocessor can be used to control almost any mechanical system with a tree structure, with a guarantee of asymptotic stability.

9 Discussion

The limitations and advantages of the new theory presented in this paper are overviewed in this section. There are a number of limitations to the control scheme given above. First, persistence of excitation has been imposed on the nonlinear mapping vectors \mathbf{g}_i . This requirement can be removed using the ε -mod technique of [9]. This involves adding the following term to the neural network update laws:

$$- \kappa \mathbf{\Gamma} \|\tilde{\mathbf{q}}\| \hat{\mathbf{w}} \quad (67)$$

where κ is a positive constant. This approach has been adopted by [8] in the neural control of manipulators.

Secondly, to demonstrate that a particular hexapod walking machine has a stable control system, a good model of the dynamics is required (eqn. 15). In reality, this involves thousands of terms, and this multiplicity of terms grows massively when calculating the conditions for stability that the gains must satisfy (see Appendix, Section 12). A sample calculation is given in [28] for hexapod legs. Even in the much simpler manipulator dynamics of [1, 8], only relatively simple simulation studies have been

achieved. However, these practical limitations do not in any way diminish the usefulness of the theoretical proofs. The basic tenet, that if the gains at the joint level controllers are sufficiently high then stability is achieved, is a helpful controller design principle. Furthermore, the torque control law does not depend on \mathbf{M} , as does, for instance, the computed torque method [29], nor in fact does it depend on the rest of the dynamics. This means that complete dynamics knowledge is not required when adopting the neural controller in practice. The great benefit of the control approach is that the neural networks ‘learn’ the dynamics. RBF neural networks have been used typically consisting of one layer with 200 neurons per joint. In practical terms, the torques are controlled using eqn. 36: the above theorems state that these controllers, proven to be stable by [1], are not made unstable by closed kinematic chains. In [28] our prototype hexapod is described in detail and experimental results demonstrating this theory are presented, showing that the legs do adapt to the contact force dynamics and, in flat surface walking, the nonoptimised neural controllers were 15% better than PD controllers. The implementation is also simpler than conventional adaptive control methods, e.g. those in [30].

The final ‘limitation’ is that all the controllers described require use of desired joint inputs rather than actual joint inputs. However, if the actual joint values can be shown to be within a compact set, or indeed can be forced to fit within a compact set with a sliding mode control, then actual joint values can be used. References [1, 8] both use a ‘reflexive stabiliser’ representation of the error as follows:

$$\dot{\mathbf{q}}_r = \dot{\mathbf{q}}_d + \mathbf{\Xi} \tilde{\mathbf{q}} \quad (68)$$

where $\mathbf{\Xi}$ is a positive definite matrix and \mathbf{q}_r is sometimes called the ‘reference joint displacement’. This has the effect of filtering the error dynamics. Reference [1] showed that the resultant neural control laws are very similar to the torque laws where desired joint inputs are used (Jin [1], Theorems 6.6 and 6.7). In the nonlinear robot functions of Theorems 1 and 2, actual joint positions and velocities have been used. It can be shown that this does not destabilise the system so long as it is required that the joint angles and velocities belong to a compact set [1]. This has been done for simplification of argument. It is a relatively straightforward extension to show that Theorems 1 and 2 satisfy these criteria by relating the gain terms to each other as $\mathbf{K}_p = \mathbf{K}_v \mathbf{\Xi}$ and, as a consequence, the fact that desired inputs were not used in these cases does not cause difficulty. In the case of these two theorems, the focus was meant to be illustrative in terms of proof of principle, and a detailed discussion of the filtered error dynamics would have been a distraction from this purpose. As to whether the actual joint values are in a compact set: for a system with closed kinematic chains this is only an issue if a particular chain is incomplete. For the hexapod, this means that problems could arise for legs in the air, since if the joint values were outside a compact set for legs in contact with the ground, then the closed chain would not be able to be completed.

There are also a number of advantages to the approach taken here. ‘Robustifying’ control terms can be added to the control laws as long as they are bounded. It is a simple matter to add conventional adaptive control terms. Some examples of this principle for standard manipulator control are given by [8]. In particular, the class of adaptive control laws described by [30] would be particularly suitable because they are in themselves asymptotically stable.

The theory presented above applies to a wide class of MIMO crosscoupled and highly nonlinear systems with varying closed kinematic chain topologies. For instance, it applies to two or more assembly line manipulators working together to turn a car body upside down. A second obvious example of such a system is a Stewart platform. It is, however, particularly interesting to consider hexapod walking systems. Within the literature there is virtually no theoretical consideration of the control of such systems. Often, control seems to be by trial and error, or various control concepts are applied without theoretical justification or demonstration of stability. To the authors' knowledge, no other work theoretically guarantees the stability of conventional closed-loop control schemes for the class of systems represented by the hexapod. Nor does any other work extend such stability proofs to neural control schemes.

The approach here does not guarantee a statically (or dynamically) stable mode of locomotion. Thus, only the bottom layer of the control architecture discussed in [4] has been guaranteed as stable. Hence, the theory presented here does not guarantee that the robot will not fall over. However, it does guarantee that if the robot should fall over, then the neural control system will not make the situation worse by becoming unstable. It also states that, if a robot is in a statically stable walk, the joint controllers *per se* will not be the cause for the robot to fall over: that would be due to an inappropriate leg configuration. However, it is conceivable that if a robot has fallen over because of an inappropriate leg configuration then delivering a particular demand trajectory could worsen the robot's predicament.

It is clear that in most realistic walking situations a robot will encounter obstacles to the path of a foot in mid-air, or not find a foothold immediately at the end of the planned swing phase trajectories. Since a leg undergoing neural control will attempt to achieve the demand trajectory, the neural controller will output higher torques in the presence of an obstacle mid-swing. As a consequence, this signal can be used as an obstacle detector. Obviously, a threshold range for the signal will be required because walking through thick grass, for instance, will cause slightly increased torques during the swing phase due to the increased resistance. In a similar fashion, if the expected contact force or torque values are not achieved at the end of the swing phase, again within a threshold, this could be used as a signal for initiating a searching algorithm to establish a stable foothold.

This theory has also introduced the novel concept of 'virtual neural networks'. On one level, they are nothing more than a projection of the control signals from one space to another (in this case from \mathfrak{R}^{18} to \mathfrak{R}^{24} for a hexapod). The technique belongs to a long tradition of mathematical tools for co-ordinate transformation, but it is a particularly powerful one, as demonstrated in the theoretical results described above. On another level, it allows for some intriguing conjecture with respect to hexapods. In insects, it has not been possible to identify the neural subsystem responsible for walking, although some individual neurones have been isolated. A stick insect has around 6000 neurones in its entire nervous system: just to control the walking, we have built a walking subsystem with 3600 artificial neurons (e.g. [15, 28]), which is a comparable number. In our structure, the neural controllers are not responsible for gait control (unlike in the insect), but it may be possible to provide this functionality in the \mathfrak{R}^{24} -space. This investigation is left for future research. Whether the inclusion of gait control in the \mathfrak{R}^{24} -space would signifi-

cantly affect the controller stability or compromise the nature of the results presented in this paper is an open question.

10 Conclusion

The neural control strategies for systems that are faced with multiple closed kinematic chains have been shown to be stable. This is, to the authors' knowledge, the only example of a control strategy with a theoretical guarantee of stability for such systems. Four theorems were presented. The first theorem demonstrated how a serial linkage with a prescribed surface contact force is stable under neural control. The second theorem extended these results to the more complex hexapod system, and showed that it could be at least 'stable in the sense of Lyapunov' under certain conditions. The third theorem extends the results presented in [1] from manipulator systems to the vastly more complex hexapod and similar structures, but did not start with any assumptions that it could use those results. To show that this system was stable, the novel concept of a 'virtual' neural network was introduced, which is essentially a projection of the neural architecture into the space of generalised co-ordinates. The fourth theorem is a restatement of Jin's Theorem 7.1 [1]. It demonstrates how discrete neural control systems for manipulators are asymptotically stable using PID controllers 'strapped around' the neural network. It was then shown how Theorem 3 of this paper can be extended to discrete systems in a manner analogous to Theorem 7.1 in [1]. Finally, a discussion of the limitations and advantages of our controllers was presented.

11 Tribute

During the final stages of preparation of this article, Mark Randall, the first author, was taken suddenly from this world.

He was a gifted researcher, and a good friend to the co-authors of this article. His research career may have been cut suddenly short, but he sowed seeds that will grow and bear fruit in our research group. His time with us was greatly valued – Tony Pipe.

12 References

- JIN, Y.: 'Intelligent neural control and its applications in robotics'. PhD thesis, University of the West of England, Bristol, 1999
- JIN, Y., PIPE, A.G., and WINFIELD, A.: 'Stable manipulator trajectory control using neural networks: Methodology, stability analysis and PUMA simulations', in OMIDVAR, O., and VAN DER SMAGT, P. (Eds.): *Progress in neural networks – neural systems for robotics* (Academic Publishing Corporation, 1997)
- CHERIAN, R.P.: 'On-line learning neuro-control of an industrial manipulator'. MSc thesis, University of the West of England, Bristol, UK, 1997
- RANDALL, M.J., and PIPE, A.G.: 'A novel soft computing architecture for the control of autonomous walking robots', *J. Soft Comput.*, (Special edition on CLAWAR Machines), 2000, 4, (3), pp. 165–185
- HORNİK, K., STINCHCOMBE, M., and WHITE, H.: 'Multilayer feedforward networks are universal approximators', *Neural Netw.*, 1989, 2, pp. 359–366
- WARWICK, K., IRWIN, G.W., and HUNT, K.J. (Eds.): 'Neural networks for control and systems' (Peter Peregrinus Ltd., 1992)
- WHITE, D.A., and SOFGE, D.A. (Eds.): 'Handbook of intelligent control—Neural, fuzzy and adaptive approaches' (Multiscience Press, 1992)
- LEWIS, F.L., JAGANNATHAN, S., and YESILDIREK, A.: 'Neural network control of robot manipulators and nonlinear systems' (Taylor and Francis, 1999)
- NARENDRA, K.S., and ANNASWAMY, A.M.: 'A new adaptive law for robust adaptation without persistent excitation', *IEEE Trans. Autom. Control*, 1987, AC-32, (2), pp. 134–145
- GIROSI, F., and POGGIO, T.: 'Networks and the best approximation property', *Cybernetics*, 1990, 63, pp. 169–176

- 11 SANNER, R.M., and SLOTINE, J.J.E.: 'Gaussian networks for direct adaptive control', *IEEE Trans. Neural Netw.*, 1992, 3, pp. 837–863
- 12 ALBUS, J.S.: 'A new approach to manipulator control: The cerebellar model articulation controller (CMAC)', *Trans. ASME, J. Dyn. Syst., Meas. Control*, 1975, 97, pp. 220–227
- 13 RUMELHART, D.E., and MCCLELLAND, J.L. (Eds.): 'Parallel distributed processing, 1, (MIT Press, 1988)
- 14 RANDALL, M.J., PIPE, A.G., and WINFIELD, A.F.T.: 'Stable on-line neural control of hexapod joint trajectories'. Proceedings IEEE Control Applications, 1998, Trieste
- 15 RANDALL, M.J., PIPE, A.G., and WINFIELD, A.F.T.: 'Blueprint for autonomy in CLAWAR machines'. Proceedings of International Conference of *Climbing and walking robots* 1998, 26–28 November 1998, Brussels, pp. 131–136
- 16 RAIBERT, M.H.: 'Legged robots that balance' (MIT Press, Cambridge, MA, 1986)
- 17 SONG, S.M., and WALDRON, K.J.: 'Machines that walk: The adaptive suspension vehicle' (MIT Press, Cambridge, MA, 1989)
- 18 WEIDEMANN, H.-J.: 'Dynamik und Regelung von Sechsheinigen Robotern und Natürlichen Hexapoden' (VDI-Verlag, Düsseldorf, 1993)
- 19 SCHMUCKER, U., SCHEIDER, A., IHME, T., DEVJANIN, E., and SAVITSKY, K.: 'Force control in locomotion of legged vehicle and body movement for mounting operations'. Proceedings of 9th World Congress on *Theory of machines and mechanisms*, 1995, Milan, pp. 2363–2367
- 20 HARDARSON, F., ERIKSSON, B., RIDDERSTRÖM, C., WADDEN, T., and WIKANDER, J.: 'Experiments with impedance control of a single compliant robot leg', in VIRK, G.S., RANDALL, M.J., and HOWARD, D. (Eds.): 'Climbing and walking robots (CLAWAR'99)' (Professional Engineering Publishing, Bury St. Edmunds, 1999), pp. 319–331
- 21 FUKUDA, T., KOMATA, Y., and ARAKAWA, T.: 'Stabilization control of biped locomotion robot based learning with GAs having self-adaptive mutation and recurrent neural networks'. Proceedings 1997 IEEE International Conference on *Robotics and automation*, 1995, Albuquerque, pp. 217–223
- 22 MEDRANO-CERDA, G.A., and ELDUKHRI, E.E.: 'Biped robot locomotion in the sagittal plane', *Trans. Inst. Meas. Control*, 1997, 19, (1), pp. 38–49
- 23 BLAJER, W., and SCHIEHLEN, W.: 'Walking without impacts as a motion/force control problem', *Trans. ASME, J. Dyn. Syst. Meas. Control*, 1992, 114, (4), pp. 660–665
- 24 GARDNER, J.F.: 'Force distribution and trajectory control for closed kinematic chains with applications to walking machines'. PhD thesis, Ohio State University, 1987
- 25 DEVJANIN, E., GURFINKEL, V., GURFINKEL, E., EFREMOV, V., LENSKY, A., SCHNEIDER, A., and SHTILMAN, L.: 'Walking robot with supervisory control', *Mech. Mach. Theory*, 1981, 16, pp. 31–36
- 26 GRIZZLE, J.W., ABBA, G., and PLESTAN, F.: 'Proving asymptotic stability of a walking cycle for a five DOF biped robot model', in VIRK, G.S., RANDALL, M.J., and HOWARD, D. (Eds.): 'Climbing and walking robots (CLAWAR'99)' (Professional Engineering Publishing, Bury St. Edmunds, 1999), pp. 69–81
- 27 RANDALL, M.J., PIPE, A.G., WINFIELD, A.F.T., and JIN, Y.: 'Adaptive neural control of walking robots with guaranteed stability', in VIRK, G.S., RANDALL, M.J., and HOWARD, D. (Eds.): 'Climbing and walking robots (CLAWAR'99)' (Professional Engineering Publishing, Bury St. Edmunds, 1999), pp. 111–121
- 28 RANDALL, M.J.: 'Adaptive neural control of walking robots' (Professional Engineering Publishing, Bury St. Edmunds, 2000), to be published
- 29 PAUL, R.P.: 'Modeling, trajectory calculation and servoing of a computer controlled arm'. Technical Report AIM-177, Stanford University Artificial Intelligence Lab., 1972
- 30 BAYARD, D.S., and WEN, J.T.: 'New class of control laws for robot manipulators. Part 2: Adaptive case', *Int. J. Control*, 1988, 47, (5), pp. 1387–1406
- 31 WEN, J.T., and BAYARD, D.S.: 'New class of control laws for robot manipulators. Part 1: Non-adaptive case', *Int. J. Control*, 1988, 47, (5), pp. 1361–1385

13 Appendices

13.1 Proof of Theorem 1

The control law is

$$\begin{aligned} \tau &= \mathbf{K}_p \tilde{\mathbf{q}}_1 + \mathbf{K}_v \dot{\tilde{\mathbf{q}}}_1 \\ &+ \mathbf{G}(\mathbf{q}_{1d}, \dot{\mathbf{q}}_{1d}, \ddot{\mathbf{q}}_{1d})^T \tilde{\mathbf{w}} \\ &+ \varepsilon_m \operatorname{sgn}(c\tilde{\mathbf{q}}) - \mathbf{J}^T(\lambda_d + \mathbf{K}_f \tilde{\lambda}) \end{aligned} \quad (69)$$

First define the nonlinear robot function as

$$\begin{aligned} \mathbf{h}(\tilde{\mathbf{q}}_1, \dot{\tilde{\mathbf{q}}}_1, \mathbf{q}_{1d}, \dot{\mathbf{q}}_{1d}, \ddot{\mathbf{q}}_{1d}) \\ = \mathbf{H}(\mathbf{q}_1) \tilde{\mathbf{J}}(\mathbf{q}_1) \ddot{\mathbf{q}}_{1d} + [\mathbf{C}(\mathbf{q}_1, \dot{\mathbf{q}}_1) \tilde{\mathbf{J}}(\mathbf{q}_1) \\ + \mathbf{H}(\mathbf{q}_1) \dot{\tilde{\mathbf{J}}}(\mathbf{q}_1)] \dot{\mathbf{q}}_{1d} + \tilde{\mathbf{g}}(\mathbf{q}_1) + \mathbf{f}_u(\mathbf{q}_1) \end{aligned} \quad (70)$$

Because of the universal approximation function, the left-hand side of eqn. 70 can be represented with a neural network such that

$$\begin{aligned} \mathbf{h}(\tilde{\mathbf{q}}_1, \dot{\tilde{\mathbf{q}}}_1, \mathbf{q}_{1d}, \dot{\mathbf{q}}_{1d}, \ddot{\mathbf{q}}_{1d}) \\ = \mathbf{G}(\tilde{\mathbf{q}}_1, \dot{\tilde{\mathbf{q}}}_1, \mathbf{q}_{1d}, \dot{\mathbf{q}}_{1d}, \ddot{\mathbf{q}}_{1d})^T \mathbf{w} \\ + \xi(\tilde{\mathbf{q}}_1, \dot{\tilde{\mathbf{q}}}_1, \mathbf{q}_{1d}, \dot{\mathbf{q}}_{1d}, \ddot{\mathbf{q}}_{1d}) \end{aligned} \quad (71)$$

where the neural network approximation error can be made as small as possible by carefully choosing the neural network structure such that

$$\|\xi\| \leq \varepsilon_m \quad (72)$$

Now substitute the error terms into eqn. 8 to give the error dynamics

$$\ddot{\mathbf{H}}\tilde{\mathbf{q}}_1 = -\ddot{\mathbf{C}}\tilde{\mathbf{q}}_1 + \ddot{\mathbf{J}}^T \mathbf{h} - \ddot{\mathbf{J}}^T \tau \quad (73)$$

Substitute the control law into eqn. 73, then use eqns. 9 and 71 to obtain the following expression for the error dynamics:

$$\begin{aligned} \ddot{\mathbf{H}}\tilde{\mathbf{q}}_1 &= -\ddot{\mathbf{J}}^T \mathbf{K}_v \tilde{\mathbf{J}}\tilde{\mathbf{q}}_1 - \ddot{\mathbf{J}}^T \mathbf{K}_p \tilde{\mathbf{J}}\tilde{\mathbf{q}}_1 - \ddot{\mathbf{C}}\tilde{\mathbf{q}}_1 \\ &+ \ddot{\mathbf{J}}^T \mathbf{G}(\tilde{\mathbf{q}}_1, \dot{\tilde{\mathbf{q}}}_1, \tilde{\mathbf{q}}_1)^T \tilde{\mathbf{w}} + \ddot{\mathbf{J}}^T \xi - \ddot{\mathbf{J}}^T \varepsilon_m \operatorname{sgn}(\tilde{\mathbf{q}}_1) \end{aligned} \quad (74)$$

Consider the following Lyapunov candidate:

$$V(\mathbf{q}_1, \dot{\mathbf{q}}_1) = \frac{1}{2} \tilde{\mathbf{q}}_1^T \ddot{\mathbf{H}}\tilde{\mathbf{q}}_1 + \frac{1}{2} \tilde{\mathbf{w}}^T \Gamma^{-1} \tilde{\mathbf{w}} \quad (75)$$

where all terms are as previously defined. Taking the derivative along the solution (and exploiting skew-symmetry) yields the following Lyapunov derivative:

$$\begin{aligned} \dot{V}(\mathbf{q}_1, \dot{\mathbf{q}}_1) &= \dot{\tilde{\mathbf{q}}}_1^T \ddot{\mathbf{H}}\tilde{\mathbf{q}}_1 + \frac{1}{2} \dot{\tilde{\mathbf{q}}}_1^T \dot{\tilde{\mathbf{H}}}\tilde{\mathbf{q}}_1 + \tilde{\mathbf{w}}^T \Gamma^{-1} \dot{\tilde{\mathbf{w}}} \\ &= \dot{\tilde{\mathbf{q}}}_1^T [-\ddot{\mathbf{J}}^T \mathbf{K}_v \tilde{\mathbf{J}}\tilde{\mathbf{q}}_1 - \ddot{\mathbf{J}}^T \mathbf{K}_p \tilde{\mathbf{J}}\tilde{\mathbf{q}}_1 \\ &+ \ddot{\mathbf{J}}^T \mathbf{G}(\tilde{\mathbf{q}}_1, \dot{\tilde{\mathbf{q}}}_1, \tilde{\mathbf{q}}_1)^T \tilde{\mathbf{w}} + \ddot{\mathbf{J}}^T \xi - \ddot{\mathbf{J}}^T \varepsilon_m \operatorname{sgn}(\tilde{\mathbf{q}}_1)] \\ &+ \dot{\tilde{\mathbf{w}}}^T \Gamma^{-1} \tilde{\mathbf{w}} \end{aligned} \quad (76)$$

Substituting the update law, it can be seen that this function can be overbounded by

$$\begin{aligned} \dot{V} &\leq -(\lambda_{\min}(\mathbf{K}_v) \|\ddot{\mathbf{J}}\|^2) \|\tilde{\mathbf{q}}_1\|^2 \\ &- (\lambda_{\min}(\mathbf{K}_p) \|\ddot{\mathbf{J}}\|^2) \|\tilde{\mathbf{q}}_1\| \|\dot{\tilde{\mathbf{q}}}_1\| \\ &- (\varepsilon_m - \|\xi\|) \|\ddot{\mathbf{J}}\| \|\tilde{\mathbf{q}}_1\| \end{aligned} \quad (77)$$

The second derivative is given by

$$\begin{aligned} \ddot{V}(\mathbf{q}_1, \dot{\mathbf{q}}_1) &= -2\dot{\tilde{\mathbf{q}}}_1^T \ddot{\mathbf{J}}^T \mathbf{K}_v \tilde{\mathbf{J}}\tilde{\mathbf{q}}_1 - 2\dot{\tilde{\mathbf{q}}}_1^T \ddot{\mathbf{J}}^T \mathbf{K}_p \tilde{\mathbf{J}}\tilde{\mathbf{q}}_1 \\ &- 2\dot{\tilde{\mathbf{q}}}_1^T \ddot{\mathbf{J}}^T \mathbf{K}_v \dot{\tilde{\mathbf{J}}}\tilde{\mathbf{q}}_1 - 2\dot{\tilde{\mathbf{q}}}_1^T \ddot{\mathbf{J}}^T \mathbf{K}_p \dot{\tilde{\mathbf{J}}}\tilde{\mathbf{q}}_1 \\ &- \ddot{\mathbf{J}}^T [\varepsilon_m \operatorname{sgn}(\tilde{\mathbf{q}}_1) - \xi] \end{aligned} \quad (78)$$

So if $\lambda_{\min}(\mathbf{K}_v) > 0$ and $\lambda_{\min}(\mathbf{K}_p) > 0$, then $V > 0$, $\dot{V} \leq 0$ and $\ddot{V} \leq 0$. This gives us 'stability in the sense of Lyapunov', so that $\tilde{\mathbf{q}}$ and $\tilde{\mathbf{w}}$ and hence $\hat{\mathbf{w}}$ are bounded; thus

$$\int_0^\infty -\dot{V} dt < \infty \quad (79)$$

It remains to be shown that the force error is bounded, before asymptotic stability can be shown. Reference [8] demonstrates the boundedness of the force tracking error as follows. It can be shown that all the terms on the right-hand side of eqn. 74 are bounded. Since $\ddot{\mathbf{H}}$ is invertible, then this establishes that $\ddot{\tilde{\mathbf{q}}}$ is bounded. Now, substitute the

control law, eqn. 69, into the error dynamics, eqn. 73, to obtain

$$\begin{aligned} \mathbf{H}\ddot{\tilde{\mathbf{q}}} &= -(\mathbf{C}\bar{\mathbf{J}} + \mathbf{H}\dot{\bar{\mathbf{J}}})\dot{\tilde{\mathbf{q}}} + \mathbf{h} - \mathbf{J}^T\lambda - \mathbf{G}^T\hat{\mathbf{w}} - \mathbf{K}_v\bar{\mathbf{J}}\dot{\tilde{\mathbf{q}}} \\ &\quad + \mathbf{J}^T[\lambda_d + \mathbf{K}_f\tilde{\lambda}] - \mathbf{K}_p\bar{\mathbf{J}}\dot{\tilde{\mathbf{q}}} - \varepsilon_m \operatorname{sgn}(\dot{\tilde{\mathbf{q}}}) \\ &= -\mathbf{K}_v\bar{\mathbf{J}}\dot{\tilde{\mathbf{q}}} - (\mathbf{C}\bar{\mathbf{J}} + \mathbf{H}\dot{\bar{\mathbf{J}}})\dot{\tilde{\mathbf{q}}} \\ &\quad + \mathbf{G}^T\hat{\mathbf{w}} + \mathbf{J}^T[\mathbf{I} + \mathbf{K}_f]\tilde{\lambda} - \mathbf{K}_p\bar{\mathbf{J}}\dot{\tilde{\mathbf{q}}} - \varepsilon_m \operatorname{sgn}(\dot{\tilde{\mathbf{q}}}) \end{aligned} \quad (80)$$

$$\begin{aligned} \mathbf{J}^T[\mathbf{I} + \mathbf{K}_f]\tilde{\lambda} &= \mathbf{H}\ddot{\tilde{\mathbf{q}}} + \mathbf{K}_v\bar{\mathbf{J}}\dot{\tilde{\mathbf{q}}} \\ &\quad + (\mathbf{C}\bar{\mathbf{J}} + \mathbf{H}\dot{\bar{\mathbf{J}}})\dot{\tilde{\mathbf{q}}} - \mathbf{G}^T\hat{\mathbf{w}} + \mathbf{K}_p\bar{\mathbf{J}}\dot{\tilde{\mathbf{q}}} + \varepsilon_m \operatorname{sgn}(\dot{\tilde{\mathbf{q}}}) \\ &\equiv \mathbf{b}(\tilde{\mathbf{q}}, \dot{\tilde{\mathbf{q}}}, \mathbf{q}_{1d}, \dot{\mathbf{q}}_{1d}, \ddot{\mathbf{q}}_{1d}, \hat{\mathbf{w}}) \end{aligned} \quad (81)$$

where all the quantities on the right-hand side are bounded. Therefore, premultiply both sides by \mathbf{J} to obtain

$$\begin{aligned} \mathbf{J}\mathbf{J}^T[\mathbf{I} + \mathbf{K}_f]\tilde{\lambda} &= \mathbf{J}\mathbf{b}(\tilde{\mathbf{q}}, \dot{\tilde{\mathbf{q}}}, \mathbf{q}_{1d}, \dot{\mathbf{q}}_{1d}, \ddot{\mathbf{q}}_{1d}, \hat{\mathbf{w}}) \end{aligned} \quad (82)$$

$$\tilde{\lambda} = [\mathbf{I} + \mathbf{K}_f]^{-1}(\mathbf{J}\mathbf{J}^T)^{-1}\mathbf{J}\mathbf{b}(\tilde{\mathbf{q}}, \dot{\tilde{\mathbf{q}}}, \mathbf{q}_{1d}, \dot{\mathbf{q}}_{1d}, \ddot{\mathbf{q}}_{1d}, \hat{\mathbf{w}})$$

where we rely on the fact that $\mathbf{J}\mathbf{J}^T$ is nonsingular. eqn. 82 shows that the force tracking error $\tilde{\lambda}(t)$ is bounded and can be made as small as desired by increasing the force tracking gain \mathbf{K}_f .

Thus, the boundedness of all the signals in the control law verifies the boundedness of $\tilde{\mathbf{q}}$, and hence \dot{V} , and thus the uniform continuity of \dot{V} . This allows us to invoke Barbalat's Lemma in connection with eqn. 79, to conclude that \dot{V} goes to zero with t , and hence that $\tilde{\mathbf{q}}(t)$ vanishes.

13.2 Proof of Theorem 2

Using the universal approximation feature of neural networks the following can be defined:

$$\begin{aligned} \mathbf{M}(\mathbf{q}_1)\bar{\mathbf{J}}(\mathbf{q}_1)\ddot{\mathbf{q}}_{1d} + [\mathbf{C}(\mathbf{q}_1, \dot{\mathbf{q}}_1)\bar{\mathbf{J}}(\mathbf{q}_1) + \mathbf{M}(\mathbf{q}_1)\dot{\bar{\mathbf{J}}}(\mathbf{q}_1)]\dot{\mathbf{q}}_{1d} \\ + \bar{\mathbf{g}}(\mathbf{q}_1) + \bar{\mathbf{f}}_u(\mathbf{q}_1, \dot{\mathbf{q}}_1) \\ = \mathbf{J}_T^T[\mathbf{G}(\tilde{\mathbf{q}}_1, \dot{\tilde{\mathbf{q}}}_1, \mathbf{q}_{1d}, \dot{\mathbf{q}}_{1d}, \ddot{\mathbf{q}}_{1d})^T \mathbf{w} \\ + \xi(\tilde{\mathbf{q}}_1, \dot{\tilde{\mathbf{q}}}_1, \mathbf{q}_{1d}, \dot{\mathbf{q}}_{1d}, \ddot{\mathbf{q}}_{1d})] \end{aligned} \quad (83)$$

where the neural network approximation error vector ξ can be made as small as possible by carefully choosing the neural network structure such that

$$\|\xi\| \leq \varepsilon_m \quad (84)$$

Following a series of substitutions, it is possible to obtain the following error dynamics:

$$\begin{aligned} \bar{\mathbf{M}}\ddot{\tilde{\mathbf{q}}}_1 &= -\bar{\mathbf{J}}^T\mathbf{J}_T^T\mathbf{K}_v\bar{\mathbf{J}}\dot{\tilde{\mathbf{q}}}_1 - \bar{\mathbf{J}}^T\mathbf{J}_T^T\mathbf{K}_p\bar{\mathbf{J}}\dot{\tilde{\mathbf{q}}}_1 - \bar{\mathbf{C}}\dot{\tilde{\mathbf{q}}}_1 \\ &\quad + \bar{\mathbf{J}}^T\mathbf{J}_T^T\mathbf{G}(\tilde{\mathbf{q}}_1, \dot{\tilde{\mathbf{q}}}_1, \mathbf{q}_{1d}, \dot{\mathbf{q}}_{1d})^T\tilde{\mathbf{w}} - \bar{\mathbf{J}}^T\mathbf{J}_T^T\varepsilon_m \operatorname{sgn}(\dot{\tilde{\mathbf{q}}}_1) \\ &\quad + \bar{\mathbf{J}}^T\xi + \bar{\mathbf{J}}^T\mathbf{J}_F^T\mathbf{f} \end{aligned} \quad (85)$$

Consider the following Lyapunov candidate:

$$V(\mathbf{q}_1, \dot{\mathbf{q}}_1) = \frac{1}{2}\tilde{\mathbf{q}}_1^T\bar{\mathbf{M}}\dot{\tilde{\mathbf{q}}}_1 + \frac{1}{2}\tilde{\mathbf{w}}^T\Gamma^{-1}\tilde{\mathbf{w}} \quad (86)$$

where all terms are as previously defined. Taking the derivative along the solution yields the following Lyapunov derivative:

$$\begin{aligned} \dot{V}(\mathbf{q}_1, \dot{\mathbf{q}}_1) &= \dot{\tilde{\mathbf{q}}}_1^T\bar{\mathbf{M}}\dot{\tilde{\mathbf{q}}}_1 + \frac{1}{2}\dot{\tilde{\mathbf{q}}}_1^T\dot{\bar{\mathbf{M}}}\dot{\tilde{\mathbf{q}}}_1 + \dot{\tilde{\mathbf{w}}}^T\Gamma^{-1}\tilde{\mathbf{w}} \\ &= \dot{\tilde{\mathbf{q}}}_1^T[-\bar{\mathbf{J}}^T\mathbf{J}_T^T\mathbf{K}_v\bar{\mathbf{J}}\dot{\tilde{\mathbf{q}}}_1 - \bar{\mathbf{J}}^T\mathbf{J}_T^T\mathbf{K}_p\bar{\mathbf{J}}\dot{\tilde{\mathbf{q}}}_1 \\ &\quad + \bar{\mathbf{J}}^T\mathbf{J}_T^T\mathbf{G}(\tilde{\mathbf{q}}_1, \dot{\tilde{\mathbf{q}}}_1, \mathbf{q}_{1d}, \dot{\mathbf{q}}_{1d})^T\tilde{\mathbf{w}} - \bar{\mathbf{J}}^T\mathbf{J}_T^T\varepsilon_m \operatorname{sgn}(\dot{\tilde{\mathbf{q}}}_1) \\ &\quad + \bar{\mathbf{J}}^T\xi + \bar{\mathbf{J}}^T\mathbf{J}_F^T\mathbf{f}] + \dot{\tilde{\mathbf{w}}}^T\Gamma^{-1}\tilde{\mathbf{w}} \end{aligned} \quad (87)$$

Define

$$\eta_5 = \sup_{\mathbf{q}_1 \in \mathcal{D}^m} \|\bar{\mathbf{J}}\| \sup_{\mathbf{q} \in \mathcal{D}^n} \|\mathbf{J}_T\|$$

then substitute the expressions and the update law, eqn. 13; then it can be seen that the Lyapunov derivative can be overbounded such that

$$\begin{aligned} \dot{V} &\leq -\eta_1\lambda_{\min}(\mathbf{K}_v)\|\dot{\tilde{\mathbf{q}}}_1\|^2 - \|\dot{\tilde{\mathbf{q}}}_1\|(\eta_5(\varepsilon_m - \|\xi\|) - \eta_2) \\ &\quad - \eta_1\lambda_{\min}(\mathbf{K}_p)\|\dot{\tilde{\mathbf{q}}}_1\|\|\tilde{\mathbf{q}}_1\| \end{aligned} \quad (88)$$

or

$$\begin{aligned} \dot{V} &\leq -\eta_1\lambda_{\min}(\mathbf{K}_v)\|\dot{\tilde{\mathbf{q}}}_1\|^2 + \eta_2\|\mathbf{f}\|\|\tilde{\mathbf{q}}_1\| \\ &\quad - \eta_1\lambda_{\min}(\mathbf{K}_p)\|\dot{\tilde{\mathbf{q}}}_1\|\|\tilde{\mathbf{q}}_1\| \end{aligned} \quad (89)$$

since the error terms $\varepsilon_m - \|\xi\| \mapsto 0$ from above and $\eta_5 > 0$. For $V \leq 0$, the following conditions must be applied (note that $\eta_1 > 0$):

$$\lambda_{\min}(\mathbf{K}_v) > 0 \quad \text{and} \quad \lambda_{\min}(\mathbf{K}_p) > 0 \quad (90)$$

hence the condition eqn. 27.

By considering the constants, eqns. 31 and 32, it can be shown that the second derivative of the Lyapunov function can be overbounded such that, after removing the error limit,

$$\begin{aligned} \ddot{V} &\leq -2\lambda_{\min}(\mathbf{K}_v)\eta_1\|\dot{\tilde{\mathbf{q}}}_1\|\|\ddot{\tilde{\mathbf{q}}}_1\| - \lambda_{\min}(\mathbf{K}_p)\eta_3 + \eta_4\|\mathbf{f}\| + \eta_2\|\dot{\mathbf{f}}\| \\ &\quad - (2\lambda_{\min}(\mathbf{K}_p)\eta_1 + \lambda_{\min}(\mathbf{K}_v)\eta_3)\|\dot{\tilde{\mathbf{q}}}_1\|^2 \end{aligned} \quad (91)$$

which leads to eqn. 30. If \mathbf{f} can be shown to be bounded, then satisfying eqn. 27 leads to a system that is 'stable in the sense of Lyapunov'. If, in addition, eqn. 30 holds, then the system is asymptotically stable because all the terms would be bounded and both the first and second derivatives of V are nonpositive.

It remains to determine whether the force error is bounded before asymptotic stability can be shown. The argument presented for this is along the same lines as that given for Theorem 1. All the terms on the right-hand side of eqn. 34 are bounded. Since $\bar{\mathbf{M}}$ is invertible, then this establishes that $\ddot{\tilde{\mathbf{q}}}$ is bounded. Now, substitute the control law eqn. 25 into the error dynamics, eqn. 34, to obtain

$$\begin{aligned} \mathbf{M}\ddot{\tilde{\mathbf{q}}} &= -(\mathbf{C}\bar{\mathbf{J}} + \mathbf{M}\dot{\bar{\mathbf{J}}})\dot{\tilde{\mathbf{q}}} + \mathbf{h} - \mathbf{J}^T\boldsymbol{\pi} - \mathbf{G}^T\hat{\mathbf{w}} - \mathbf{K}_v\bar{\mathbf{J}}\dot{\tilde{\mathbf{q}}} \\ &\quad + \mathbf{J}^T[\boldsymbol{\pi}_d + \mathbf{K}_f\tilde{\boldsymbol{\pi}}] - \mathbf{K}_p\bar{\mathbf{J}}\dot{\tilde{\mathbf{q}}} - \varepsilon_m \operatorname{sgn}(\dot{\tilde{\mathbf{q}}}) \\ &= -\mathbf{K}_v\bar{\mathbf{J}}\dot{\tilde{\mathbf{q}}} - (\mathbf{C}\bar{\mathbf{J}} + \mathbf{M}\dot{\bar{\mathbf{J}}})\dot{\tilde{\mathbf{q}}} + \mathbf{G}^T\hat{\mathbf{w}} \\ &\quad + \mathbf{J}^T[\mathbf{I} + \mathbf{K}_f]\tilde{\boldsymbol{\pi}} - \mathbf{K}_p\bar{\mathbf{J}}\dot{\tilde{\mathbf{q}}} - \varepsilon_m \operatorname{sgn}(\dot{\tilde{\mathbf{q}}}) \end{aligned} \quad (92)$$

Rearranging gives

$$\begin{aligned} \mathbf{J}^T[\mathbf{I} + \mathbf{K}_f]\tilde{\boldsymbol{\pi}} &= \mathbf{M}\tilde{\mathbf{J}}\tilde{\mathbf{q}} + \mathbf{K}_v\tilde{\mathbf{J}}\dot{\tilde{\mathbf{q}}} + (\mathbf{C}\tilde{\mathbf{J}} + \mathbf{M}\dot{\tilde{\mathbf{J}}})\tilde{\mathbf{q}} - \mathbf{G}^T\tilde{\mathbf{w}} \\ &+ \mathbf{K}_p\tilde{\mathbf{J}}\tilde{\mathbf{q}} + \varepsilon_m \text{sgn}(\tilde{\mathbf{q}}) \\ &\equiv \mathbf{b}(\tilde{\mathbf{q}}, \tilde{\mathbf{q}}_1, \mathbf{q}_{1d}, \dot{\tilde{\mathbf{q}}}_1, \ddot{\tilde{\mathbf{q}}}_1, \tilde{\mathbf{w}}) \end{aligned} \quad (93)$$

where all the quantities on the right-hand side are bounded. Therefore, premultiply both sides by \mathbf{J} to obtain

$$\begin{aligned} \mathbf{J}\mathbf{J}^T[\mathbf{I} + \mathbf{K}_f]\tilde{\boldsymbol{\pi}} &= \mathbf{J}\mathbf{b}(\tilde{\mathbf{q}}, \tilde{\mathbf{q}}_1, \mathbf{q}_{1d}, \dot{\tilde{\mathbf{q}}}_1, \ddot{\tilde{\mathbf{q}}}_1, \tilde{\mathbf{w}}) \\ \tilde{\boldsymbol{\pi}} &= [\mathbf{I} + \mathbf{K}_f]^{-1}(\mathbf{J}\mathbf{J}^T)^{-1}\mathbf{J}\mathbf{b}(\tilde{\mathbf{q}}, \tilde{\mathbf{q}}_1, \mathbf{q}_{1d}, \dot{\tilde{\mathbf{q}}}_1, \ddot{\tilde{\mathbf{q}}}_1, \tilde{\mathbf{w}}) \end{aligned} \quad (94)$$

where we rely on the fact that $\mathbf{J}\mathbf{J}^T$ is nonsingular. eqn. 94 shows that the force tracking error $\tilde{\boldsymbol{\pi}}(t)$ is bounded and can be made as small as desired by increasing the force tracking gain \mathbf{K}_f .

Thus, the boundedness of all the signals in the control law verifies the boundedness of $\tilde{\mathbf{q}}$, and hence \dot{V} assuming that the conditions eqns. 27 and 30 are met. If so, the uniform continuity of \dot{V} is established. This allows us to invoke Barbalat's Lemma in connection with

$$\int_0^\infty -\dot{V}dt < \infty \quad (95)$$

to conclude that V goes to zero with t , and hence that $\tilde{\mathbf{q}}(t)$ vanishes.

13.3 Proof of Theorem 3

Consider the following nonlinear robot function and its neural network approximation:

$$\begin{aligned} \mathbf{M}(\mathbf{q}_d)\ddot{\mathbf{q}}_d + \mathbf{C}(\mathbf{q}_d, \dot{\mathbf{q}}_d)\dot{\mathbf{q}}_d + \mathbf{g}(\mathbf{q}_d) + \mathbf{f}_u(\mathbf{q}_d) \\ = \mathbf{G}^v(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d)^T \mathbf{w}^v + \boldsymbol{\xi}^v(\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d) \end{aligned} \quad (96)$$

and the following Lyapunov function:

$$\begin{aligned} V(\tilde{\mathbf{q}}, \tilde{\mathbf{q}}) &= \frac{1}{2}\tilde{\mathbf{q}}^T \mathbf{M}(\mathbf{q})\tilde{\mathbf{q}} + \frac{1}{2}\tilde{\mathbf{q}}^T \mathbf{K}_p \tilde{\mathbf{q}} + c\tilde{\mathbf{q}}^T \mathbf{M}(\mathbf{q})\dot{\tilde{\mathbf{q}}} \\ &+ \frac{1}{2}c\tilde{\mathbf{q}}^T \mathbf{K}_v \tilde{\mathbf{q}} + \frac{1}{2}\tilde{\mathbf{w}}^{vT} (\boldsymbol{\Gamma}^v)^{-1} \tilde{\mathbf{w}}^v \end{aligned} \quad (97)$$

where c is a positive constant. The derivative is

$$\begin{aligned} \dot{V} &= \tilde{\mathbf{q}}^T \dot{\mathbf{M}}\dot{\tilde{\mathbf{q}}} + \frac{1}{2}\tilde{\mathbf{q}}^T \dot{\mathbf{M}}\dot{\tilde{\mathbf{q}}} + \tilde{\mathbf{q}}^T \mathbf{K}_p \dot{\tilde{\mathbf{q}}} + c\tilde{\mathbf{q}}^T \mathbf{M}(\mathbf{q})\ddot{\tilde{\mathbf{q}}} \\ &+ c\tilde{\mathbf{q}}^T \dot{\mathbf{M}}(\mathbf{q})\dot{\tilde{\mathbf{q}}} + c\tilde{\mathbf{q}}^T \mathbf{M}(\mathbf{q})\ddot{\tilde{\mathbf{q}}} + c\tilde{\mathbf{q}}^T \mathbf{K}_v \dot{\tilde{\mathbf{q}}} \\ &+ \tilde{\mathbf{w}}^{vT} (\boldsymbol{\Gamma}^v)^{-1} \dot{\tilde{\mathbf{w}}}^v + \frac{1}{2}\tilde{\mathbf{w}}^{vT} (\dot{\boldsymbol{\Gamma}}^v)^{-1} \tilde{\mathbf{w}}^v \\ &+ \frac{1}{2}c\tilde{\mathbf{q}}^T \dot{\mathbf{K}}_v \tilde{\mathbf{q}} + \frac{1}{2}\tilde{\mathbf{q}}^T \dot{\mathbf{K}}_p \tilde{\mathbf{q}} \end{aligned} \quad (98)$$

Note that, unlike in previous derivations, it cannot be assumed that $\boldsymbol{\Gamma}^v$ or the gains consist of constant terms. This is because the projection of torques and forces into the space of generalised co-ordinates is likely to cause these three terms to be dependent on \mathbf{q} and therefore t .

After some work, and following substitution of the neural update law, and using some results from [31], it is

possible to show that the Lyapunov derivative can be overbounded such that

$$\begin{aligned} \dot{V} &\leq -\left(\lambda_{\min}(\mathbf{K}_v) - c\mu - \frac{\eta_{10}}{2}\right)\|\tilde{\mathbf{q}}\|^2 \\ &- \left(c\lambda_{\min}(\mathbf{K}_p) - \eta_{13}\right)\|\tilde{\mathbf{q}}\|^2 + c\|\tilde{\mathbf{q}}\|\left(3\eta_{10}\|\tilde{\mathbf{q}}\| + \frac{\eta_9}{2}\|\tilde{\mathbf{q}}\|^2\right) \\ &- \left(\varepsilon_m^v - \|\boldsymbol{\xi}^v\|\right)\left(\|\tilde{\mathbf{q}}\| + c\|\tilde{\mathbf{q}}\|\right) \\ &+ \|\tilde{\mathbf{q}}\|\left(\|\tilde{\mathbf{q}}\| + c\|\tilde{\mathbf{q}}\|\right)\left(\eta_8 + \eta_{12}\eta_7 + \frac{3}{2}\eta_6^2\eta_{11}\right)\|\tilde{\mathbf{q}}\| \end{aligned} \quad (99)$$

where

$$\begin{aligned} \mu &= \sup_{\mathbf{q} \in \mathcal{N}^n} \|\mathbf{M}(\mathbf{q})\|, \quad \eta_6 = \sup_{t \geq 0} \|\dot{\mathbf{q}}_d(t)\|, \quad \eta_7 = \sup_{t \geq 0} \|\ddot{\mathbf{q}}_d(t)\| \\ \eta_8 &= \sup_{\mathbf{q} \in \mathcal{N}^n} \left\| \frac{\partial \mathbf{g}(\mathbf{q})}{\partial \mathbf{q}} \right\| + \sup_{\mathbf{q} \in \mathcal{N}^n} \left\| \frac{\partial \mathbf{f}_u(\mathbf{q})}{\partial \mathbf{q}} \right\| \\ \eta_9 &= \sum_{i=1}^n \sup_{\mathbf{q} \in \mathcal{N}^n} \|\mathbf{M}_i(\mathbf{q})\|, \quad \mathbf{M}_i(\mathbf{q}) = \frac{\partial \mathbf{M}(\mathbf{q})}{\partial q_i}, \\ \eta_{10} &= \sup_{t \geq 0} \|\dot{\mathbf{q}}_d(t)\| \eta_9 \\ \eta_{11} &= \sum_{i=1}^n \sum_{j=1}^n \sup_{\mathbf{q} \in \mathcal{N}^n} \left\| \frac{\partial \text{Col}_j[\mathbf{M}_i(\mathbf{q})]}{\partial q} \right\| \end{aligned}$$

where $\text{Col}_j[\mathbf{M}]$ is the j th column of matrix \mathbf{M} ,

$$\begin{aligned} \eta_{12} &= \sum_{j=1}^n \sup_{\mathbf{q} \in \mathcal{N}^n} \left\| \frac{\partial \text{Col}_j[\mathbf{M}(\mathbf{q})]}{\partial \mathbf{q}} \right\| \\ \eta_{13} &= \frac{1}{2c} \lambda_{\min}(\dot{\mathbf{K}}_v) + \frac{1}{2c} \lambda_{\min}(\dot{\mathbf{K}}_p) \end{aligned}$$

Also, define $\eta_{14} = \frac{1}{2}(3c\eta_{10} + \eta_8 + \eta_{12}\eta_7 + \frac{3}{2}\eta_6^2\eta_{11})$.

By completing the square for the cross term, the Lyapunov derivative becomes

$$\begin{aligned} \dot{V} &\leq -\alpha_1 \|\tilde{\mathbf{q}}\|^2 - \alpha_2 \|\dot{\tilde{\mathbf{q}}}\|^2 + \gamma_{12} \|\tilde{\mathbf{q}}\| \|\dot{\tilde{\mathbf{q}}}\|^2 \\ &- \left(\varepsilon_m - \|\boldsymbol{\xi}\|\right)\left(\|\tilde{\mathbf{q}}\| + c\|\tilde{\mathbf{q}}\|\right) \end{aligned} \quad (100)$$

or

$$\dot{V} \leq -\alpha_1 \|\tilde{\mathbf{q}}\|^2 - \alpha_2 \|\dot{\tilde{\mathbf{q}}}\|^2 + \gamma_{12} \|\tilde{\mathbf{q}}\| \|\dot{\tilde{\mathbf{q}}}\|^2 \quad (101)$$

where

$$\begin{aligned} \alpha_1 &= c\left(\lambda_{\min}(\mathbf{K}_p) - (\eta_8 + \eta_{12}\eta_7 + \frac{3}{2}\eta_6^2\eta_{11} + \eta_{13}) - \rho^2\eta_{14}\right) > 0 \\ \alpha_2 &= \left(\lambda_{\min}(\mathbf{K}_v) - \frac{1}{2}\eta_{10} - c\mu - \frac{\eta_{14}}{\rho^2}\right) > 0 \\ \gamma_{12} &= \frac{1}{2}c\eta_{15} \end{aligned}$$

and ρ^2 is arbitrary, chosen such that $\alpha_1 > 0$. Also, choose

$$\lambda_{\min}(\mathbf{K}_p) > (\eta_8 + \eta_{12}\eta_7 + \frac{3}{2}\eta_6^2\eta_{11} + \eta_{13})$$

and choose $\lambda_{\min}(\mathbf{K}_v)$ large enough such that $\alpha_2 - \gamma_{12}[V_0/\xi_1]^2 > 0$, where

$$\begin{aligned} V_0 &= \left(\frac{1}{2}\tilde{\mathbf{q}}^T \mathbf{K}_p \tilde{\mathbf{q}} + \frac{1}{2}c\tilde{\mathbf{q}}^T \mathbf{K}_v \tilde{\mathbf{q}} + c\tilde{\mathbf{q}}^T \mathbf{M}(\mathbf{q})\dot{\tilde{\mathbf{q}}} \right. \\ &\left. + \frac{1}{2}\tilde{\mathbf{q}}^T \mathbf{M}(\mathbf{q})\dot{\tilde{\mathbf{q}}} + \tilde{\mathbf{w}}^T \boldsymbol{\Gamma}^{-1} \tilde{\mathbf{w}} \right) \Big|_{t=0} \end{aligned}$$

and

$$\xi_1 = \inf_{\|\tilde{\mathbf{q}}\|=1} \left[\frac{1}{2} \tilde{\mathbf{q}}^T \mathbf{K}_p \tilde{\mathbf{q}} + \frac{1}{2} c \tilde{\mathbf{q}}^T \mathbf{K}_v \tilde{\mathbf{q}} - \frac{1}{2} c \mu l^2 \right] > 0$$

for some constant l which satisfies the following condition:

$$\xi_2 = \inf_{\mathbf{q}} \left[\inf_{\|\tilde{\mathbf{q}}\|=1} \left[\frac{1}{2} \tilde{\mathbf{q}}^T \mathbf{M}(\mathbf{q}) \tilde{\mathbf{q}} - \frac{1}{2} \left(\frac{c\mu}{l^2} \right) \right] \right] > 0$$

All the above conditions result in the following. From Lemma 2.1 in [31], $\exists \lambda_2 > 0$ and $\exists \lambda > 0$ such that the crossterms can be eliminated and the Lyapunov derivative becomes

$$\dot{V} \leq -\alpha_1 \|\tilde{\mathbf{q}}\|^2 - \lambda_2 \|\dot{\tilde{\mathbf{q}}}\|^2 \leq -\lambda V \quad (102)$$

Therefore, the closed-loop system is exponentially stable, and it establishes that \mathbf{q} , $\dot{\mathbf{q}}$, $\hat{\mathbf{w}}$ are bounded $\Rightarrow \boldsymbol{\tau}$ is bounded $\Rightarrow \tilde{\mathbf{q}}$ is bounded.

To prove that $(\tilde{\mathbf{q}}, \dot{\tilde{\mathbf{q}}})$ is asymptotically stable, we build another Lyapunov candidate (cf. [1]):

$$V_1 = V - \int_0^t \left(\dot{V}(\rho) + \alpha_1 \tilde{\mathbf{q}}(\rho)^T \tilde{\mathbf{q}}(\rho) + \lambda_2 \dot{\tilde{\mathbf{q}}}(\rho)^T \dot{\tilde{\mathbf{q}}}(\rho) \right) d\rho \quad (103)$$

The first and second derivatives of this Lyapunov candidate are

$$\dot{V}_1 = -\alpha_1 \tilde{\mathbf{q}}^T \tilde{\mathbf{q}} - \lambda_2 \dot{\tilde{\mathbf{q}}}^T \dot{\tilde{\mathbf{q}}}, \quad \ddot{V}_1 = -2\alpha_1 \dot{\tilde{\mathbf{q}}}^T \tilde{\mathbf{q}} - 2\lambda_2 \dot{\tilde{\mathbf{q}}}^T \ddot{\tilde{\mathbf{q}}} \quad (104)$$

\dot{V}_1 is uniformly continuous since \dot{V}_1 is bounded and $\int_0^t \dot{V}_1(\rho) d\rho = V_1(t) - V_1(0) < \infty$. Thus, by applying Barbalat's Lemma, we get $\lim_{t \rightarrow \infty} \dot{V}_1 = 0$ or, equivalently, $\lim_{t \rightarrow \infty} \tilde{\mathbf{q}} = 0$, $\lim_{t \rightarrow \infty} \dot{\tilde{\mathbf{q}}} = 0$.