Submitted to the 1997 International Conference on Intelligent Robots and Systems (IROS), Grenoble, France, September 8-13, 1997.

Control of Space Free-Flyers Using the Modified Transpose Jacobian Algorithm

S. Ali A. Moosavian, Evangelos Papadopoulos

Department of Mechanical Engineering & Centre for Intelligent Machines
McGill University
Montreal, PQ, Canada H3A 2A7

ABSTRACT

Transpose Jacobian control is one of the simplest algorithms used in manipulator control. However, since it is not dynamics-based, poor performance may occur in applications where high speed tracking is required. Use of high gains also worsens performance, especially in the presence of noise. In addition, lack of a systematic gain selection makes it difficult to apply. In this paper a Modified Transpose Jacobian algorithm is presented and applied to control of space free-flyers. This new algorithm employs stored data of the control command in the previous time step, resulting in improved performance. The gains of the modified algorithm do not need to be large, hence the noise rejection characteristics of the algorithm are improved. Stability analysis, based on Lyapunov's theorems, shows that both the standard and the Modified Transpose Jacobian algorithms are asymptotically stable. Simula-tions of both terrestrial and space applications show that tracking performance of this new algorithm is comparable to that of computed torque algorithms, although it does not require a priori knowledge of plant dynamics.

I. INTRODUCTION

The problem of controlling mechanical manipulators is challenging because of the strong nonlinearities and time dependencies in the equations of motion. Different algorithms have been suggested since the early research in robotic systems [1-3]. Transpose Jacobian (TJ) control is a computationally simple algorithm which has been arrived at intuitively [4]. The task error vector and its rate, both multiplied by relatively high gains, and by the Jacobian transpose matrix, result in commands that push the end-effector in a direction which tends to reduce the tracking error. In the case of using an approximate Jacobian, it has been shown that the damping matrix and the position gain matrix play an important role in the stability condition [5].

An extended Jacobian transpose control algorithm has been developed to coordinate motion control of a single manipulator mounted on a spacecraft [6]. The performance of this simple algorithm in coordinated motion control of multiple arm free-flyers has been compared to those of different model-based algorithms [7]. The comparison shows that the TJ algorithm can be efficiently employed in the control of highly nonlinear and complex systems, with many Degrees-of-Freedom (DOF). This result motivates further work on this algorithm, aiming at overcoming the lack of information about the system dynamics which causes poor performance in tracking fast trajectories.

This paper presents the Modified Transpose Jacobian (MTJ) control which yields an improved performance over the standard algorithm, by employing stored data of the previous time step control command. The MTJ algorithm is based on an approximation of feedback linearization methods, with no need to a priori knowledge of the plant dynamics terms. Its performance is comparable to that of model-based algorithms, but requires reduced computational burden. Therefore, it can be used in the control of multi-DOF space robots where computational power is limited. In the following, first the MTJ control law is derived, then the stability of this algorithm is analyzed. Simulation results are presented which compare the performance of the MTJ to that of the TJ and Model-Based algorithms in both terrestrial and space applications.

II. MTJ CONTROL LAW

Using the expressions for the kinetic and potential energy, and applying Lagrange's equations for a robotic system, the dynamics model can be obtained as follows [8]

$$\mathbf{H}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}}) = \mathbf{Q} \tag{1}$$

where $\mathbf{q}$ is the vector of generalized coordinates, $\mathbf{C}$ contains all nonlinear velocity and gravity terms, and $\mathbf{Q}$ is the vector of generalized forces. Gravity terms are practically zero in microgravity environments, and therefore can be neglected in the design of control laws for space robots. In terrestrial applications, these terms may cause static positioning errors in control, and in such case, they must be compensated separately. Therefore, in this paper, the $\mathbf{C}$ vector contains nonlinear velocity terms, only.

The output velocities $\dot{\mathbf{q}}'$ are obtained from the generalized velocities $\dot{\mathbf{q}}$ using a Jacobian matrix, $\mathbf{J}_{q'}$, as

$$\dot{\mathbf{q}}' = \mathbf{J}_{q'}(\mathbf{q})\dot{\mathbf{q}} \tag{2}$$

Assuming that $\mathbf{J}_{q'}$ is square and non-singular, Eq. (1) can be written in terms of the output variables as follows

$$\mathbf{H}'(\mathbf{q})\ddot{\mathbf{q}}' + \mathbf{C}'(\mathbf{q},\dot{\mathbf{q}}) = \mathbf{Q}' \tag{3a}$$

where

$$\tilde{\mathbf{H}} = \mathbf{J}_q^{-T}\mathbf{H}\,\mathbf{J}_q^{-1},\ \tilde{\mathbf{C}} = \mathbf{J}_q^{-T}\mathbf{C} - \tilde{\mathbf{H}}\,\dot{\mathbf{J}}_q\dot{\mathbf{q}},\ \tilde{\mathbf{Q}} = \mathbf{J}_q^{-T}\mathbf{Q} \qquad (3b)$$

To control such a system, a model-based, (Computed Torque), control law such as

$$\mathbf{Q} = \mathbf{J}_q^{T}\{\tilde{\mathbf{H}}\mathbf{u} + \tilde{\mathbf{C}}\} \qquad (4)$$

can be applied, where $\mathbf{u}$ is an auxiliary control signal. A usual assumption associated with this control law is that the system geometric and mass properties are known. This control law linearizes and decouples the system equations to a set of second order differential equations

$$\ddot{\mathbf{q}} = \mathbf{u} \qquad (5)$$

If $\mathbf{u}$ is computed such that

$$\mathbf{u} = \mathbf{K}_p\,\mathbf{e} + \mathbf{K}_d\,\dot{\mathbf{e}} + \ddot{\mathbf{q}}_{des} \qquad (6)$$

where $\mathbf{K}_p$, and $\mathbf{K}_d$ are positive definite gain matrices, and $\mathbf{e}$ is the tracking error defined as

$$\mathbf{e} = \mathbf{q}_{des} - \mathbf{q} \qquad (7)$$

then, the control law given by Eq. (4) guarantees asymptotic convergence of the tracking error $\mathbf{e}$. Implementing this algorithm, in addition to a priori knowledge of the system properties, requires computational power which may not be available.

If high enough gains are used, the simple Transpose Jacobian (TJ) controller can be employed [4]

$$\mathbf{Q} = \mathbf{J}_q^{T}\{\mathbf{K}_p\,\mathbf{e} + \mathbf{K}_d\,\dot{\mathbf{e}}\} \qquad (8)$$

The action of this controller can be understood by imagining generalized springs and dampers, along the variables under control, connected between the corresponding body and the desired trajectories; the stiffer the gains are, the better the tracking should be. An advantage of this algorithm is that if a physical singularity is encountered, the controller given by Eq. (8) may result in errors but will not fail computationally. Clearly, if $\mathbf{Q}$ is substituted into Eq. (3a), using Eqs. (3b) and (8), it can be seen that the error is not guaranteed to converge to zero.

To achieve both precision and simplicity, the TJ control law defined by Eq. (8) is modified, to approximate a feedback linearization solution, as

$$\mathbf{Q} = \mathbf{J}_q^{T}\{\mathbf{K}_d\,\dot{\mathbf{e}} + \mathbf{K}_p\,\mathbf{e} + \mathbf{h}(t)\} \qquad (9)$$

where $\mathbf{h}$(t) is to be determined such that an approximately linearized error dynamics is achieved. Substitution of Eq. (9) into Eq. (3a), and using Eqs. (3b) yields

$$\mathbf{K}_d\,\dot{\mathbf{e}} + \mathbf{K}_p\,\mathbf{e} = \tilde{\mathbf{H}}\ddot{\mathbf{q}} + \tilde{\mathbf{C}} - \mathbf{h}(t) \qquad (10)$$

or equivalently

$$\mathbf{K}_d\,\dot{\mathbf{e}} + \mathbf{K}_p\,\mathbf{e} = \tilde{\mathbf{Q}} - \mathbf{h}(t) \qquad (11)$$

It can be seen that if the right hand side (RHS) of Eq. (10), becomes equal to zero, then the goal of this modification is met, and the algorithm works like a model-based algorithm, although its implementation is simpler. Note that inclusion of $\ddot{\mathbf{e}}$, second derivative of error, in Eq. (9), i.e.

$$\mathbf{Q} = \mathbf{J}_q^{T}\{\ddot{\mathbf{e}} + \mathbf{K}_d\,\dot{\mathbf{e}} + \mathbf{K}_p\,\mathbf{e} + \mathbf{h}(t)\} \qquad (12a)$$

yields

$$\ddot{\mathbf{e}} + \mathbf{K}_d\,\dot{\mathbf{e}} + \mathbf{K}_p\,\mathbf{e} = \tilde{\mathbf{H}}\ddot{\mathbf{q}} + \tilde{\mathbf{C}} - \mathbf{h}(t) \qquad (12b)$$

which can result in improved performance. However, inclusion of this signal requires acceleration measurements or an estimator, and may be difficult to obtain in practice.

To make the RHS of Eq. (10) be close to zero, Eq. (11) suggests that a good approximation is to take $\mathbf{h}$(t) equal to $\tilde{\mathbf{Q}}$ at a previous small time step, $\tilde{\mathbf{Q}}\big|_{t-\Delta t}$. But, since the inclusion of this term may result in high joint torque requirements (when relatively high $\mathbf{e}$ or $\dot{\mathbf{e}}$ are imposed as disturbances), the following form is used

$$\mathbf{h}(t) = k\,\tilde{\mathbf{Q}}\big|_{t-\Delta t} \qquad (13)$$

where the *regulating factor*, k, is defined as

$$k = \begin{cases} 0 & when\ |\mathbf{e}| \geq \varepsilon\ or\ |\dot{\mathbf{e}}| \geq \dot{\varepsilon} \\ 1 & when\ |\mathbf{e}| < \varepsilon\ \&\ |\dot{\mathbf{e}}| < \dot{\varepsilon} \end{cases} \qquad (14)$$

where ε and $\dot{\varepsilon}$ represent sensitivity thresholds. Note that factor k is initially taken equal to zero, resulting in a TJ control law at the first time step. To simplify the on-off switch for factor k, the following expression can be used

$$k = \exp(-(\frac{|\mathbf{e}|}{e_{\max}} + \frac{|\dot{\mathbf{e}}|}{\dot{e}_{\max}})) \qquad (15a)$$

where $e_{\max}$, and $\dot{e}_{\max}$ are positive real numbers which correspond to another representation of the sensitivity threshold. Note that relatively low values for sensitivity thresholds, would make the algorithm work like the standard TJ control law. In practice, $\mathbf{K}_p$ and $\mathbf{K}_d$ can be chosen as diagonal matrices, and so can be selected the regulating factor. Then, factor k in Eq. (13) should be replaced by a diagonal matrix $\mathbf{K}$ with elements defined as

$$k_{ii} = \exp(-(\frac{|\mathrm{e_i}|}{e_{\max_i}} + \frac{|\dot{\mathrm{e}}_\mathrm{i}|}{\dot{e}_{\max_i}})) \qquad (15b)$$

Including the rate of error in Eq. (15) introduces a sense of anticipation. Similarly, one can include the second rate of error, if available. However, this makes the algorithm more sensitive, and therefore sharp variations of actuator forces/torques may result.

Application of the MTJ algorithm

$$\mathbf{Q} = \mathbf{J}_q^{T}\{\mathbf{K}_d\,\dot{\mathbf{e}} + \mathbf{K}_p\,\mathbf{e} + k\,\tilde{\mathbf{Q}}\big|_{t-\Delta t}\} \qquad (16)$$

with proper selection of the sensitivity thresholds, so that the modifying term is activated (i. e. $k \approx 1.0$), and small time steps, results in the following error equation

$$k_{d_i}\dot{e}_i + k_{p_i}e_i \cong 0 \qquad (17)$$

where diagonal gain matrices, $\mathbf{K}_p$, and $\mathbf{K}_d$, have been used. Therefore, using Eq. (17), the control gains can be selected in a systematic manner, where their ratio determines error time constant, and their magnitude determines the magnitude of the control command.

Considering Eqs. (4), (8), and (16) a comparison between the algorithms, in terms of the required computational operations, is depicted in Table I. It is assumed that the inverse of the Jacobian matrix and its time derivative, which are required for implementing the MB algorithm, are available symbolically and hence computations involving these inversions are not counted. Nevertheless, the required computational effort reveals the efficiency of the TJ and the MTJ algorithms.

Table I: Required computational operations.

Algorithm	Multiplication	Additions
TJ	$3\ N^2$	$3\ N^2 - 2\ N$
MTJ	$3\ N^2 + 2$	$3\ N^2 - N + 1$
MB	$2\ N^3 + 7\ N^2$	$2\ N^3 + 5\ N^2 - 4\ N$

III. STABILITY ANALYSIS

To investigate the stability of MTJ algorithm at the origin, i.e. $\mathbf{q}_{des}, \dot{\mathbf{q}}_{des} \equiv \mathbf{0}$, the following positive definite function

$$V(\mathbf{q}, \dot{\mathbf{q}}) = \frac{1}{2}(\dot{\mathbf{q}}^T \mathbf{H}\, \dot{\mathbf{q}} + \mathbf{q}^T \mathbf{K}_p \mathbf{q}) \tag{18}$$

is introduced as a candidate for Lyapunov function, where the first term represents the kinetic energy of the manipulator and the second one denotes an artificial potential energy. Differentiation of Eq. (18) leads to

$$\dot{V}(\mathbf{q}, \dot{\mathbf{q}}) = \dot{\mathbf{q}}^T \mathbf{Q} + \dot{\mathbf{q}}^T \mathbf{K}_p \mathbf{q} \tag{19}$$

where the fact that rate of kinetic energy in a mechanical system is equal to the power provided by the external forces, has been employed. Substituting Eq. (16) into Eq. (3b) and the result into Eq. (19), for $\mathbf{q}_{des}, \dot{\mathbf{q}}_{des} \equiv \mathbf{0}$, yields

$$\dot{V} = -\dot{\mathbf{q}}^T \mathbf{K}_d\ \dot{\mathbf{q}} + \dot{\mathbf{q}}^T k\, \mathbf{Q}\Big|_{t-\Delta t} \tag{20}$$

or at $t = t_n$

$$\dot{V}_n = -\dot{\mathbf{q}}_n^T \mathbf{K}_d\ \dot{\mathbf{q}}_n + \dot{\mathbf{q}}_n^T k_n\ \mathbf{Q}_{n-1} \tag{21}$$

For guaranteed Lyapunov stability, it must be shown that $\dot{V}$ is a negative semi-definite function. To do so, a "mathematical induction" approach is followed, and therefore the proof includes two parts: n=1, and n>1.

Part (a). For n=1, k is equal to zero. So

$$\dot{V}_1 = -\dot{\mathbf{q}}_1^T \mathbf{K}_d\ \dot{\mathbf{q}}_1 \tag{22}$$

which vanishes when $(\mathbf{q}_1^T, \dot{\mathbf{q}}_1^T) = \mathbf{0}$, and $\dot{V}_1 \le 0.0$ for $(\mathbf{q}_1^T, \dot{\mathbf{q}}_1^T) \ne \mathbf{0}$. Therefore, $\dot{V}_1$ is negative semi-definite.

Part (b). Assuming that $\dot{V}_n$ is a negative semi-definite function, it must be shown that $\dot{V}_{n+1}$ is a negative semi-definite function. Eq. (21) can be written at $t = t_{n+1}$ as

$$\dot{V}_{n+1} = -\dot{\mathbf{q}}_{n+1}^T \mathbf{K}_d\ \dot{\mathbf{q}}_{n+1} + \dot{\mathbf{q}}_{n+1}^T k_{n+1}\ \mathbf{Q}_n \tag{23a}$$

or

$$\begin{aligned}\dot{V}_{n+1} = &-\dot{\mathbf{q}}_{n+1}^T \mathbf{K}_d\ \dot{\mathbf{q}}_{n+1} + \\ &\dot{\mathbf{q}}_{n+1}^T k_{n+1}\left(-\mathbf{K}_p\ \mathbf{q}_n - \mathbf{K}_d\ \dot{\mathbf{q}}_n + k_n \mathbf{Q}_{n-1}\right)\end{aligned} \tag{23b}$$

where

$$\dot{\mathbf{q}}_{n+1} = \dot{\mathbf{q}}_n + \Delta t_{n+1} \ddot{\mathbf{q}}_n + \mathrm{O}(\Delta t_{n+1}^2) \tag{24}$$

Note that $\dot{V}_{n+1}$ vanishes when $(\mathbf{q}_{n+1}^T, \dot{\mathbf{q}}_{n+1}^T) = \mathbf{0}$. It must then be shown that $\dot{V}_{n+1} \le 0.0$ for $(\mathbf{q}_{n+1}^T, \dot{\mathbf{q}}_{n+1}^T) \ne \mathbf{0}$. Substituting Eq. (24) into Eq. (23b), and taking Eq. (21) into consideration, yields

$$\begin{aligned}\dot{V}_{n+1} = &\ k_{n+1} \dot{V}_n - \dot{\mathbf{q}}_{n+1}^T \mathbf{K}_d \dot{\mathbf{q}}_{n+1} - k_{n+1} \dot{\mathbf{q}}_n^T \mathbf{K}_p \mathbf{q}_n - \Delta t_{n+1} \ddot{\mathbf{q}}_n^T \\ &\Big\{\mathbf{K}_p \left(k_{n+1}\mathbf{q}_n + k_{n+1} k_n \mathbf{q}_{n-1} + \cdots + k_{n+1} k_n \cdots k_2 \mathbf{q}_1\right) + \\ &\mathbf{K}_d \left(k_{n+1}\dot{\mathbf{q}}_n + k_{n+1} k_n \dot{\mathbf{q}}_{n-1} + \cdots + k_{n+1} k_n \cdots k_2 \dot{\mathbf{q}}_1\right)\Big\} \\ &+ \mathrm{O}(\Delta t_{n+1}^2)\end{aligned} \tag{25}$$

Neglecting higher order terms, noting that k_i is a positive real number, and $\dot{V}_n$ is a negative semi-definite function, Eq. (25) shows that $\dot{V}_{n+1}$ will be negative semi-definite if

$$\begin{aligned}&k_{n+1} \dot{\mathbf{q}}_n^T \mathbf{K}_p\ \mathbf{q}_n + \Delta t_{n+1} \ddot{\mathbf{q}}_n^T \\ &\Big\{\mathbf{K}_p \left(k_{n+1}\mathbf{q}_n + k_{n+1} k_n \mathbf{q}_{n-1} + \cdots + k_{n+1} k_n \cdots k_2 \mathbf{q}_1\right) + \\ &\mathbf{K}_d \left(k_{n+1}\dot{\mathbf{q}}_n + k_{n+1} k_n \dot{\mathbf{q}}_{n-1} + \cdots + k_{n+1} k_n \cdots k_2 \dot{\mathbf{q}}_1\right)\Big\} \\ &\ge -\dot{\mathbf{q}}_{n+1}^T \mathbf{K}_d\ \dot{\mathbf{q}}_{n+1} \qquad \textit{for } n \ge 1\end{aligned} \tag{26}$$

Using Eq. (3), $\ddot{\mathbf{q}}_n$ is written as

$$\ddot{\mathbf{q}}_n = \mathbf{H}_n^{-1} (\mathbf{Q}_n - \mathbf{C}_n) \tag{27}$$

which after substitution for $\mathbf{Q}_n$, results in

$$\begin{aligned}\ddot{\mathbf{q}}_n = -\mathbf{H}_n^{-1} &\Big\{\mathbf{K}_p \left(\mathbf{q}_n + k_n \mathbf{q}_{n-1} + \cdots + k_n \cdots k_2 \mathbf{q}_1\right) + \\ &\mathbf{K}_d \left(\dot{\mathbf{q}}_n + k_n \dot{\mathbf{q}}_{n-1} + \cdots + k_n \cdots k_2 \dot{\mathbf{q}}_1\right) + \mathbf{C}_n \Big\}\end{aligned} \tag{28}$$

Now, substituting Eq. (28) into (26), yields the sufficient condition for making $\dot{V}_{n+1}$ be negative semi-definite as

$$\begin{aligned}&k_{n+1} \dot{\mathbf{q}}_n^T \mathbf{K}_p \mathbf{q}_n - k_{n+1}\Delta t_{n+1} \left(\mathbf{K}_p \mathbf{q}_* + \mathbf{K}_d \dot{\mathbf{q}}_* + \mathbf{C}_n\right)^T \mathbf{H}_n^{-1} \\ &\left(\mathbf{K}_p \mathbf{q}_* + \mathbf{K}_d \dot{\mathbf{q}}_*\right) \ge -\dot{\mathbf{q}}_{n+1}^T \mathbf{K}_d\ \dot{\mathbf{q}}_{n+1} \qquad \textit{for } n \ge 1\end{aligned} \tag{29}$$

where

$$\mathbf{q}_* = \mathbf{q}_n + k_n \mathbf{q}_{n-1} + \cdots + k_n \cdots k_2 \mathbf{q}_1 \tag{30a}$$

$$\dot{\mathbf{q}}_* = \dot{\mathbf{q}}_n + k_n \dot{\mathbf{q}}_{n-1} + \cdots + k_n \cdots k_2 \dot{\mathbf{q}}_1 \tag{30b}$$

In Eq. (29), the Right-Hand-Side (RHS) is always negative (for $\dot{\mathbf{q}}_{n+1} \ne \mathbf{0}$, otherwise $\dot{V}_{n+1}$ vanishes, see Eq. (23a)), and its magnitude can be increased by selecting large damping gains, $\mathbf{K}_d$. For $k_i = 0.0$, which characterizes the standard TJ algorithm, the $\mathrm{LHS} = 0.0$ and the condition is automatically satisfied. For $k_i \ne 0.0$, if $\mathrm{LHS} \ge 0.0$ then satisfaction of the condition is guaranteed. To satisfy this condition, when $\mathrm{LHS} < 0.0$, the $|\mathrm{LHS}|$ has to be smaller than the $|\mathrm{RHS}|$. But, since Δt_i is a positive number, and $\mathbf{K}_p$ is positive definite, choosing small values for these quantities results in smaller values for the $|\mathrm{LHS}|$. So, by selecting small time steps (Δt_i), and lower gains $\mathbf{K}_p$ compared to gains $\mathbf{K}_d$, the $|\mathrm{LHS}|$ can be made smaller than the $|\mathrm{RHS}|$ in a defined region, resulting in satisfaction of the condition. So, Eq. (29) introduces a criterion for choosing gains and time steps.

Therefore, based on the above induction approach, $\dot{V}$ is a negative semi-definite function. According to Lyapunov's theorem for local stability, [9], this completes the proof for stability of the MTJ algorithm at the origin. Using Eqs. (10), (11), and (19), and diagonal gain matrices, it can be shown that $\dot{V}$ vanishes only at the origin, provided that $k\ \Delta t\ K_{p_j} / K_{d_j} < 2$. Then, it can be concluded that $\dot{V}$ is a negative definite function, and the stability is asymptotic. Then, since $V(\tilde{\mathbf{q}}, \dot{\tilde{\mathbf{q}}}) \to \infty$ as $\left\| \{\tilde{\mathbf{q}}^T, \dot{\tilde{\mathbf{q}}}^T\} \right\| \to \infty$, according to Lyapunov's theorem for global stability, [9], the algorithm is globally asymptotically stable.

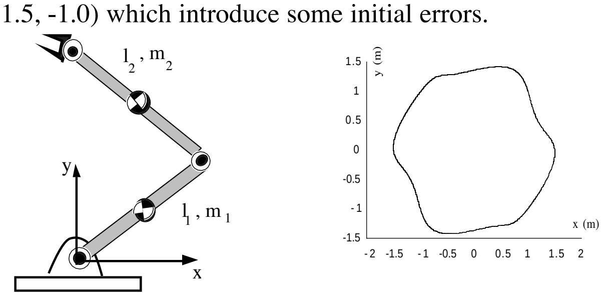
IV. SIMULATION RESULTS

In this section the performance of the MTJ control, as given by Eq. (16), is evaluated by simulation, and compared to the standard TJ, Eq. (8), and model-based algorithms, Eq. (4). First, to focus on algorithmic aspects, a simple 2-link planar manipulator is considered, see Fig. 1. Then, the new MTJ algorithm is applied to coordinated motion control of a 14-DOF space free-flyer.

Example 1. Two-link planar manipulator. For the manipulator depicted in Fig. 1 the link lengths are $l_1 = l_2 = 1\ m$ and the task is tracking a trajectory defined as

$$\begin{aligned} x_{des} &= \sqrt{l_1^2 + l_2^2}\ \cos(\omega t + \pi/4) + 0.1\sin(5\omega t) \\ y_{des} &= \sqrt{l_1^2 + l_2^2}\ \sin(\omega t + \pi/4) + 0.1\sin(5\omega t) \end{aligned} \tag{31}$$

and shown in Fig. 1 (b). The mass properties of the system are $m_1 = 4.0\ kg$, $I_1 = 0.333\ kg.m^2$, $m_2 = 3.0\ kg$, and $I_2 = 0.30\ kg.m^2$. The initial conditions for joint angles and derivatives, ($q_1(0)$, $q_2(0)$, $\dot{q}_1(0)$, $\dot{q}_2(0)$), are (0.03, $\pi/2$, 1.5, -1.0) which introduce some initial errors.

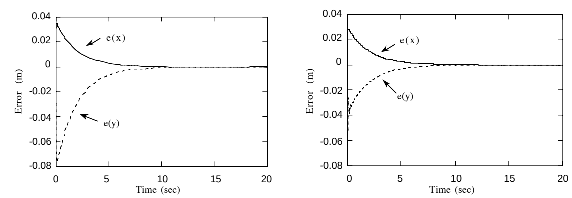


(a) (b)

Fig. 1: (a) The manipulator, (b) Desired path.

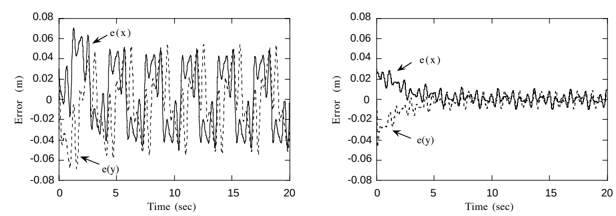
The sensitivity thresholds for the MTJ algorithm are chosen large so that $k \approx 1.0$ throughout the response. The size of the time step Δt_n is held constant, and equal to 10.0 msec. To establish a fair comparison, the gains for the algorithms under comparison are selected such that the peak of the required joint torques are approximately equal.

The performance of TJ and MTJ algorithms, in terms of the end-point error in a low-speed tracking task, is compared in Fig. 2. For the MTJ algorithm $\mathbf{K}_p$ = diag(30, 30), $\mathbf{K}_d$ = diag(60, 60), while for the TJ the gains are twice of these values. It can be seen that both algorithms yield fairly same results in a low-speed tracking task where ω in Eqs. (31) is equal to 0.05 rad/s. However, errors for the standard TJ algorithm may drastically grow from their initial values, e.g. *e(y)* in Fig. 2 (a), and also may never converge to zero. Fig. 3 shows the end-point tracking error in a high-speed tracking, where ω=2.0 rad/s. As shown in this figure, the MTJ law yields much smaller tracking errors.



(a) (b)

Fig. 2: Tracking errors for low-speed task, (a) TJ, (b) MTJ algorithm.



(a) (b)

Fig. 3: Tracking errors for high-speed task, (a) TJ, (b) MTJ algorithm.

It is expected that by selecting very high gains, the performance of the TJ algorithm can be improved. To investigate this possibility, the previous gain values for the MTJ are used, while for the TJ fairly high gains are selected, see Table II. Besides, the task speed is reduced to ω=1.0 rad/s. Here, two cases of model-based (MB) algorithms are also considered. The first is a perfect MB algorithm, while in the second one the mass properties of the dynamics model in the controller are perturbed by a 10% with respect to the true values. Note that the selected gains assure that the peaks of joint torques for all four algorithms are about the same. It is seen that the resulting tracking errors of the MTJ are still about five times smaller than the ones of the standard TJ, and even better than the ones of the perturbed MB algorithm, see Fig. 4. Note that the total energy consumption of each algorithm for performing this task, given by the time integral of $|\tau_1\ \dot{q}_1| + |\tau_2\ \dot{q}_2|$, is almost the same, i. e. (a) 153, (b) 156, (c) 153, and (d) 154 Joule.

Table II: Selected gains, Example 1.

Algorithm	$\mathbf{K}_p$	$\mathbf{K}_d$
TJ	diag(150, 150)	diag(300, 300)
MTJ	diag(30, 30)	diag(60, 60)
MB, case 1	diag(8, 8)	diag(4, 4)
MB, case 2	diag(30, 30)	diag(60, 60)

Example 2. Space free-flying robotic system. The new MTJ algorithm is now applied to the coordinated motion control of a space free-flying robotic system, and the results are compared to those of the standard TJ and model-based (MB) algorithms. The system is a 14-DOF space free-flyer as described in [10], see Fig. 5. The task is capturing a moving object based on the planned trajectories.

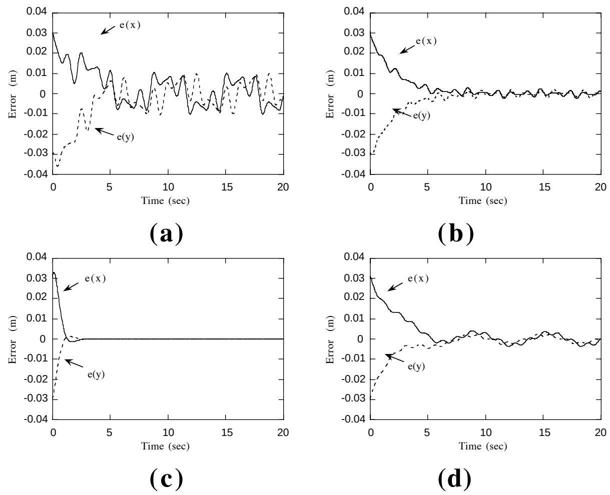


Fig. 4: Tracking errors, (a) TJ, (b) MTJ, (c) MB, case 1, (d) MB, case 2.

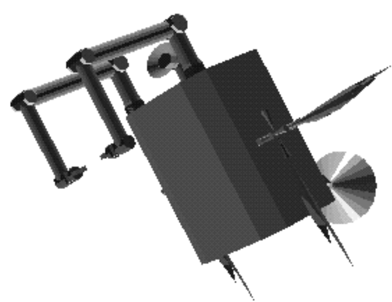


Fig. 5: The space free-flyer, Example 2.

To include the effects of model uncertainties in the MB law, the mass properties of the model used in the control algorithm are perturbed with respect to the true parameters by 5%. Table III shows the gains used for alternative controllers. The size of the time step, Δt_n for the MTJ implementation, is held constant and equal to 10.0 msec. The sensitivity thresholds for the MTJ controllers, to be substituted into Eq. (15b), are $\mathbf{e}_{max}$ = (1e-2,1e-2,1e-2,1e-2,1e-2,1e-2,1e-1,...,1e-1)^T, and $\dot{\mathbf{e}}_{max}$ = (1e-1,1e-1,1e-1,1e-1,1e-1,1e-1,1.0,... , 1.0)^T.

Figure 6 shows tracking errors for the first end-effector which can be considered as typical errors in controlling variables. For the TJ algorithm, these errors are much higher (almost 50 times higher than those of the MTJ), especially at the beginning when the system is accelerating. As discussed before, this is because the TJ algorithm is does not include information about the dynamics of the system. However, it is seen that the error for the MTJ algorithm remains very small, throughout the maneuver. Note that when the object enters the manipulator fixed-base workspace, and the manipulators start moving ($t \approx 58$ sec), tracking errors appear due to the dynamic coupling and also due to the transition phase from joint-space to task-space control. These errors eventually vanish, in all three algorithms.

Table III: Selected gains, Example 2.

Algorithm	$\mathbf{K}_p$	$\mathbf{K}_d$
TJ	diag(300,300,300, 200,...,200,100, 100)	diag(600,600,600, 400,...,400,200, 200)
MB, MTJ	diag(150,...,150, 50,50)	diag(300,...,300, 100,100)

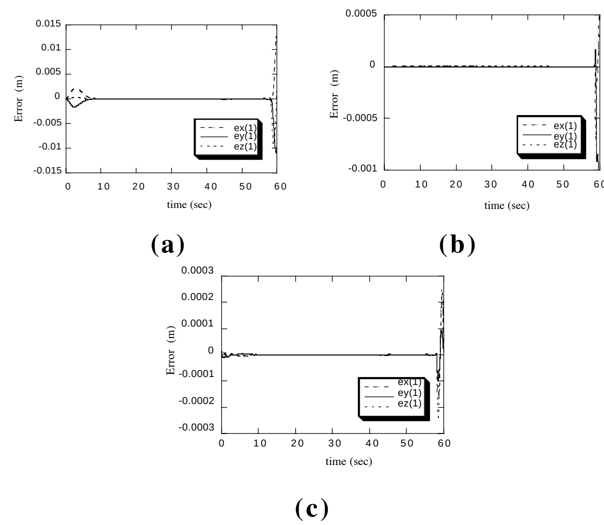
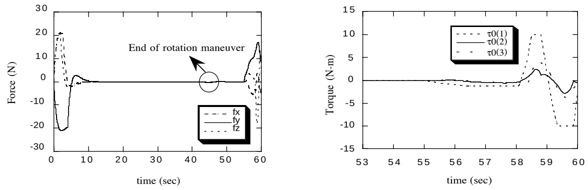


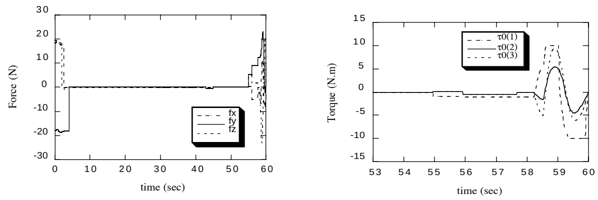
Fig. 6: Tracking position errors for the first end-effector, (a) TJ, (b) MB, (c) MTJ.

Figure 7 shows the applied control forces/torques on the spacecraft. Comparison of the spacecraft thruster forces, shows that the peak of the required forces is about the same for the TJ and MB algorithms, while in the case of MTJ it reaches the actuator saturation limits. The profiles of thruster forces, in most parts of the maneuver, is staircase for the MB while for the TJ algorithm, it is a smooth approximation of those profiles. For the MTJ algorithms, the profile is similar to the one of the TJ, at the beginning, and to that of the MB, at the end. This means that the value of the regulating factor which corresponds to the position error of spacecraft center of mass, is close to zero at the beginning, and almost equal to one at the end.

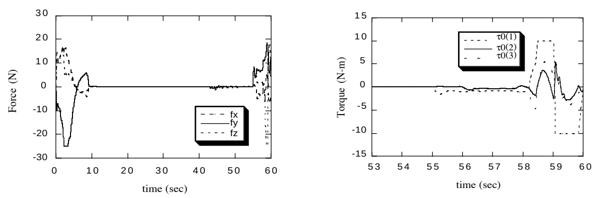
As shown in Figure 7, in all three algorithms the applied torques on the spacecraft, result in reaching actuator saturation limits of the first torque component, in attempting to compensate for the disturbances caused by manipulator motions (starting at $t \approx 58$ sec). The applied actuator joint torques for the manipulators, not shown here, have a profile similar to those applied on the spacecraft.



(a)



(b)



(c)

Fig. 7: Thruster forces (left) and applied torques on the spacecraft (right), (a) TJ, (b) MB, (c) MTJ.

In practice, noise will corrupt any available feedback. Therefore, one should examine the noise rejection capabilities of all algorithms, and especially of those that rely on high gains. It was shown by simulation, see [10], that the required torques for the MTJ algorithm are almost as smooth as for a perfect MB control. On the contrary, for the TJ algorithm higher gains are required for better tracking, which lead to poor noise rejection characteristics. The substantially reduced computational requirements compared to the MB, and the good tracking and noise rejection performance characteristics in comparison with the TJ, suggest that the MTJ algorithm is a good candidate in the control of multiple manipulator space free-flying robotic systems.

V. CONCLUSIONS

This paper presented the Modified Transpose Jacobian (MTJ) algorithm and its application to coordinated motion control of space free-flyers. The MTJ approximates a feedback linearization solution with no need of a priori knowledge of the plant dynamics. Stability analysis, based on Lyapunov theorems, shows that both the standard and the MTJ algorithms are globally asymptotically stable. It was shown by simulation that the performance of the MTJ controller in both terrestrial and space applications is comparable to that of Model-Based algorithms, with the advantage that less computational power is needed. Unlike the standard Transpose Jacobian, this algorithm works well in high speed tasks without requiring the use of high gains. The substantially reduced computational requirements, and the tracking and noise rejection performance characteristics suggest that the MTJ algorithm is a promising alternative in applications where model-based controllers can not be used due to computational limitations.

VI. ACKNOWLEDGMENTS

The support of this work by the Natural Sciences and Engineering Council of Canada (NSERC) is acknowledged. We would also like to acknowledge support of the first author from the Iran Ministry of Higher Education.

REFERENCES

[1] Dubowsky, S. and Des Forges, D. T., “The Application of Model-Referenced Adaptive Control to Robotic Manipulators,” *ASME Journal of Dynamic Systems, Measurement & Control*, Vol. 101, 1979.

[2] Youcef-Toumi, K. and Ito, O., “Controller Design for Systems with Unknown Dynamics,” *Proc. of American Cont. Conf.*, Minneapolis, USA, 1987.

[3] Umetani, Y. and Yoshida, K., “Resolved Motion Control of Space Manipulators with Generalized Jacobian Matrix,” *IEEE Trans. on Robotics and Automation*, Vol. 5, No. 3, 1989, pp. 303-314.

[4] Craig, J., *Introduction to Robotics, Mechanics and Control*, Addison Wesley, Reading, MA, 1989.

[5] Miyazaki, F., Masutani, Y., and Arimoto, S., “Sensor feedback using approximate Jacobian,” *Proc. USA-Japan Symp. on Flexible Automation - Crossing Bridges*, Advances in Flexible Automation & Robotics, Minneapolis, MN, USA, 1988.

[6] Papadopoulos, E. and Dubowsky, S., “Coordinated Manipulator/Spacecraft Motion Control for Space Robotic Systems,” *Proc. IEEE Int. Conf. Robotics & Automation*, Sacramento, CA, USA, Apr. 1991.

[7] Papadopoulos, E. and Moosavian, S. Ali A., “Dynamics & Control of Multi-arm Space Robots During Chase & Capture Operations,” *Proc. of the Int. Conf. on Intelligent Robots and Systems*, Munich, Germany, September 12-16, 1994.

[8] Papadopoulos, E. and Moosavian, S. Ali A., “Dynamics & Control of Space Free-Flyers with Multiple Arms,” *Journal of Advanced Robotics*, Vol. 9, No. 6, 1995, pp. 603-624.

[9] Slotine, J.-J. E. and Li, W., "Applied Nonlinear Control," Prentice Hall, 1991, Pages 62-65.

[10] Moosavian, S. Ali A., "Dynamics and Control of Free-Flying Manipulators Capturing Space Objects," *Ph.D. thesis*, McGill University, Montreal, Canada, June 1996.