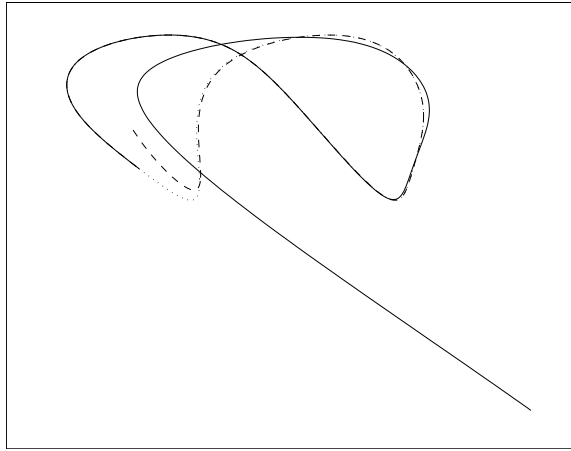


TRACKING IMPLICIT TRAJECTORIES



Neil H. Getz and Jerrold E. Marsden

Memorandum No. CPAM 629

17 February 1995

CENTER FOR PURE AND APPLIED MATHEMATICS

Department of Mathematics
University of California at Berkeley
Berkeley, CA 94720

Tracking Implicit Trajectories*

Neil H. Getz

Electrical Engineering and Computer Sciences
University of California at Berkeley
Berkeley, California 94720
`getz@eecs.berkeley.edu`

Jerrold E. Marsden

Control and Dynamical Systems
California Institute of Technology
Pasadena, California 91125
`marsden@cds.caltech.edu`

February 17, 1995

Abstract

Output tracking of implicitly defined reference trajectories is examined. A continuous-time nonlinear dynamical system is constructed that produces explicit estimates of time-varying implicit trajectories. We prove that incorporation of this “dynamic inverter” into a tracking controller provides exponential output tracking of the implicitly defined trajectory for nonlinear control systems having vector relative degree and well-behaved internal dynamics.

Key Words. dynamic, inversion, implicit, tracking, inverse kinematic problem, nonlinear control, robot control, singular perturbations

*Presented at IFAC Symposium on Nonlinear Control System Design, Tahoe City, June 26-28, 1995.

1 Introduction

In this article we will consider the problem of output tracking where the reference output which we wish to track is defined implicitly. We will rely upon a continuous time dynamical technique for inverting nonlinear maps, and will refer to this technique as *dynamic inversion* [GM94]. We will join dynamic inversion to a tracking controller in order to provide an explicit estimator for the reference output.

We will first give a brief review of the essential elements of dynamic inversion. Dynamic inversion is then incorporated into a tracking controller for tracking of the implicit reference trajectory. An example of output tracking for a simple robot arm illustrates application of the theory.

2 Dynamic Inversion

Dynamic inversion is a methodology for using continuous time dynamics to provide an estimate of time-varying roots of time dependent maps. The methodology also provides a framework in which to view and generalize certain elements of extant dynamical methods for inverting nonlinear maps using for instance gradient flows, neural networks, and the techniques of [NTV91]. In dynamic inversion one associates with a map $F(\theta, t)$ a dynamical system $\dot{\theta} = \Phi(\theta, t)$ with the crucial property that an isolated root $\theta_*(t)$ is exponentially attractive. Dynamic inversion depends intimately upon the notion of a *dynamic inverse* which we now define.

Definition 1 Let $F : \mathbb{R}^n \times \mathbb{R}_+ \rightarrow \mathbb{R}^n; (\theta, t) \mapsto F(\theta, t)$ be continuous in θ and piecewise continuous in t . Let $\theta_*(t)$ be a continuous isolated solution of $F(\theta, t) = 0$. A map $G : \mathbb{R}^n \times \mathbb{R}_+ \rightarrow \mathbb{R}^n; (w, t) \mapsto G(w, t)$ is called a **dynamic inverse** of F on the ball $\mathcal{B}_r := \{z \in \mathbb{R}^n \mid \|z\| \leq r\}$, $r > 0$, if (1) the map $G(F(\theta, t), t)$ is Lipschitz in w and piecewise continuous in t , and (2) there is a fixed real number β , with $0 < \beta < \infty$, such that

$$z^T G(F(z + \theta_*(t), t), t) \geq \beta \|z\|_2^2 \tag{1}$$

for all $z \in \mathcal{B}_r$. △

Remark 1 If $G(w, t)$ is a dynamic inverse of $F(\theta, t)$ with constant β , then for any $\mu \in \mathbb{R}_+$, $\mu G(w, t)$ is a dynamic inverse of $F(\theta, t)$ with constant $\mu\beta$.

Sufficient conditions for the existence of a dynamic inverse for all $t \in \mathbb{R}_+$ are mild as shown by the following lemma, proof of which may be found in [GM94].

Lemma 1 Let $\theta_*(t)$ be continuous isolated solution of $F(\theta, t) = 0$. Assume $F(\theta, t)$ is C^2 in θ and piecewise continuous in t . Let $D_1 F(\theta_*(t), t)$ be non-singular for all t . Let $D_1 F(\theta_*(t), t)$ and $D_1 F(\theta_*(t), t)^{-1}$ be bounded uniformly

in t . For all $z \in \mathcal{B}_r$, let $D_1^2 F(z + \theta_*(t), t)$ be bounded uniformly in t . Under these conditions, for any particular time $t_1 \geq 0$, there exists an interval $[t_0, t_2] \subset \mathbb{R}_+$ with $t_0 < t_1 < t_2$, and a ball \mathcal{B}_r such that for any particular $\theta_p \in \mathcal{B}_r$, $G(w, t) := D_1 F(\theta_p, t_1)^{-1} \cdot w$ is a dynamic inverse of $F(\theta, t)$ on \mathcal{B}_r for all $t \in [t_0, t_2]$. \square

An important special class of dynamic inverses is the class in which G is of the form $G(w, \theta, t)$, where θ is the solution of a dynamical system which estimates θ_* . Thus G inherits part of its time dependence from its θ -dependence. In this case we say that $G(w, \theta, t)$ is a **state dependent dynamic inverse** of F on \mathcal{B}_r if

$$z^T G(F(z + \theta_*(t), t), z + \theta_*(t), t) \geq \beta \|z\|_2^2 \quad (2)$$

for all $z \in \mathcal{B}_r$.

The dynamic inversion theorem below ties the dynamic inverse to dynamical estimation of a continuous isolated solution of $F(\theta, t) = 0$. This theorem is proven in [GM94].

Theorem 1 (Dynamic Inversion Theorem) *Let $\theta_*(t)$ be a continuous isolated solution of $F(\theta, t) = 0$, with $F : \mathbb{R}^n \times \mathbb{R}_+ \rightarrow \mathbb{R}^n$; $(\theta, t) \mapsto F(\theta, t)$. Assume that $G : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}_+ \rightarrow \mathbb{R}^n$; $(w, \theta, t) \mapsto G(w, \theta, t)$, is a state-dependent dynamic inverse of $F(\theta, t)$ on \mathcal{B}_r , for some finite $\beta > 0$. Let $E : \mathbb{R}^n \times \mathbb{R}_+ \rightarrow \mathbb{R}^n$; $(\theta, t) \mapsto E(\theta, t)$ be locally Lipschitz in θ and piecewise continuous in t . Assume that for some fixed $\kappa \in (0, \infty)$, $E(\theta, t)$ satisfies*

$$\left\| E(z + \theta_*(t), t) - \dot{\theta}_*(t) \right\|_2 \leq \kappa \|z\|_2 \quad (3)$$

for all $z \in \mathcal{B}_r$. Let $\theta(t)$ denote the solution to the system

$$\dot{\theta} = -\mu G(F(\theta, t), \theta, t) + E(\theta, t) \quad (4)$$

with initial condition $\theta(0)$ satisfying $\theta(0) - \theta_*(0) \in \mathcal{B}_r$. Then

$$\|\theta(t) - \theta_*(t)\|_2 \leq \|\theta(0) - \theta_*(0)\|_2 e^{-(\mu\beta - \kappa)t} \quad (5)$$

for all $t \in \mathbb{R}_+$, and in particular if $\mu > \kappa/\beta$, then $\theta(t)$ converges to $\theta_*(t)$ exponentially. \square

The map $E(\theta, t)$ in Theorem 1 is usually chosen to be a θ - and t -dependent estimator for $\dot{\theta}_*$. We will show one way of constructing such an estimator below.

Though a dynamic inverse need not be linear, a linear one is often easy to obtain as indicated by Lemma 1. Let $G(w, \theta, t) := D_1 F(\theta, t)^{-1} \cdot w$. It follows from Lemma 1 and Theorem 1 that if μ is sufficiently large, $\|\theta(0) - \theta_*(0)\|$ is sufficiently small, and $G(w, \theta(0), 0)$ is a dynamic inverse of $F(\theta, t)$ around $t = 0$, then $G(w, \theta, t)$ is a dynamic inverse of $F(\theta, t)$ for all $t > 0$. Example 1 will illustrate application of Lemma 1 and Theorem 1 to the estimation of $\theta_*(t)$.

Tracking Implicit Trajectories

Example 1 Assume that the assumptions of Lemma 1 hold. We may obtain an estimator $E(\theta, t)$ for $\dot{\theta}_*$ by differentiating $F(\theta_*(t), t) = 0$,

$$D_1 F(\theta_*(t), t) \dot{\theta}_*(t) + D_2 F(\theta_*(t), t) = 0, \quad (6)$$

solving for $\dot{\theta}_*$, and replacing θ_* by θ to get

$$E(\theta, t) := -D_1 F(\theta, t)^{-1} D_2 F(\theta, t). \quad (7)$$

Assume that r has been chosen sufficiently small, and that $D_2 F(\theta, t)$ is sufficiently bounded so that $E(\theta, t)$ satisfies (3) for all $z \in \mathcal{B}_r$. Let

$$G(w, \theta, t) := D_1 F(\theta, t)^{-1} \cdot w \quad (8)$$

and assume that r is small enough that G is a dynamic inverse of F on \mathcal{B}_r . If $(\theta(0) - \theta_*(0)) \in \mathcal{B}_r$, and μ is sufficiently large, then by Theorem 1 the approximation error $z(t) := \theta(t) - \theta_*(t)$ using (4) will decay exponentially to zero. \triangle

Example 2 shows how one may invert a time-varying matrix dynamically. See [GM95] for a more comprehensive handling of dynamic matrix inversion as well as polar decomposition.

Example 2 Consider the problem of estimating the inverse Γ_* of a time-dependent matrix $A(t)$. Assume that $A(t)$ is C^1 in t and nonsingular.

In order for Γ_* to be the inverse of $A(t)$, Γ_* must satisfy $A(t)\Gamma_* - I = 0$. Accordingly we let

$$F^M(\Gamma, t) := A(t)\Gamma - I. \quad (9)$$

We may obtain an estimator $E^M(\Gamma, t)$ for $\dot{\Gamma}_*$ by differentiating $A(t)\Gamma_* - I = 0$ with respect to t , solving for $\dot{\Gamma}_*$, and replacing all occurrences of Γ_* by Γ to get

$$E^M(\Gamma, t) = -\Gamma \dot{A}(t) \Gamma. \quad (10)$$

For a dynamic inverse consider $(D_1 F^M(\Gamma, t))^{-1} \cdot w$. Differentiating F^M with respect to Γ gives

$$D_1 F^M(\Gamma, t) = A(t) \quad (11)$$

whose inverse is Γ_* . So a choice of dynamic inverse is

$$G^M(w, \Gamma) = \Gamma \cdot w \quad (12)$$

for Γ sufficiently close to Γ_* . The dynamic inverter for this problem then takes the form

$$\begin{aligned} \dot{\Gamma} &= -\mu G^M(F^M(\Gamma, t), \Gamma) + E^M(\Gamma, t) \\ &= -\mu \Gamma (A(t)\Gamma - I) - \Gamma \dot{A}(t) \Gamma \end{aligned} \quad (13)$$

and we choose as initial conditions $\Gamma(0)$ sufficiently close to $A^{-1}(0)$. Theorem 1 guarantees that for sufficiently large μ , equation (13) will produce an estimator Γ which converges exponentially to Γ_* at a rate determined by our choice of μ . \triangle

We may now call upon insights from Example 2 in order to obtain a dynamic inverse dynamically while simultaneously using that dynamic inverse to estimate $\theta_*(t)$.

Example 3 Suppose that we wish to solve $F(\theta, t) = 0$, $\theta \in \mathbb{R}^n$. Let the assumptions of Lemma 1 hold. We wish to estimate θ_* while providing a time-dependent linear dynamic inverse of $F(\theta, t)$ based on knowledge of $D_1F(\theta, t)$. Assume that we have a representation of $D_1F(\theta, t)$ that is C^2 in θ and C^1 in t . Let Γ denote our estimator for $D_1F(\theta, t)^{-1}$. We may then estimate $\theta_*(t)$ as follows: Differentiate $F(\theta_*, t) = 0$, solve for $\dot{\theta}_*$, and substitute Γ for $D_1F(\theta_*(t), t)^{-1}$ and θ for θ_* to obtain an estimator for θ_* in terms of Γ , θ , and t ,

$$E(\Gamma, \theta, t) := -\Gamma D_2F(\theta, t). \tag{14}$$

Assume that $E(\Gamma, \theta, t)$ is C^1 in its arguments. Using $E(\Gamma, \theta, t) = [E_i(\Gamma, \theta, t)]_{i \in \underline{n}}$, by (10) we may estimate $\dot{\Gamma}_*$ with

$$E^M(\Gamma, \theta, t) := -\Gamma \left. \frac{d}{dt} D_1F(\theta, t) \right|_{\dot{\theta}=E(\Gamma, \theta, t)} \Gamma \tag{15}$$

where

$$\left. \frac{d}{dt} D_1F(\theta, t) \right|_{\dot{\theta}=E(\Gamma, \theta, t)} := \sum_{i=1}^n \frac{\partial D_1F(\theta, t)}{\partial \theta_i} E_i(\Gamma, \theta, t) + \frac{\partial D_1F(\theta, t)}{\partial t}.$$

In this case

$$F^M(\Gamma, \theta, t) := D_1F(\theta, t)\Gamma - I. \tag{16}$$

Let $G^M(w, \Gamma) := \Gamma \cdot w$ as in Example 2. Theorem 1 now tells us that we may estimate $\theta_*(t)$ with the system of coupled nonlinear differential equations

$$\begin{bmatrix} \dot{\Gamma} \\ \dot{\theta} \end{bmatrix} = -\mu \begin{bmatrix} \Gamma & 0 \\ 0 & \Gamma \end{bmatrix} \cdot \begin{bmatrix} D_1F(\theta, t)\Gamma - I \\ F(\theta, t) \end{bmatrix} + \begin{bmatrix} E^M(\Gamma, \theta, t) \\ -\Gamma D_2F(\theta, t) \end{bmatrix}$$

with guaranteed exponential convergence of (Γ, θ) to (Γ_*, θ_*) . \triangle

3 Tracking Implicit Trajectories

We now apply dynamic inversion to the problem of output tracking where the reference signal is defined implicitly. The problem we wish to solve is this: Find an input u to a control system such that for all initial states in some ball about the origin, the output y of the dynamical system converges to a desired implicitly defined output function $\theta_*(t)$. For simplicity we will assume that the nonlinear system we control has the same number of inputs as outputs.

Let $\underline{k} := \{1, 2, \dots, k\}$. Consider the following nonlinear control system:

$$\begin{cases} \dot{\xi}_j^i &= \xi_{j+1}^i, \quad i \in \underline{m}, j \in \underline{r_m} - 1 \\ \xi_r &= u \\ \dot{\eta} &= f(\xi, \eta, u, t) \\ y_i &= \xi_1^i, \quad i \in \underline{m} \end{cases} \quad (17)$$

with $\xi_j^i \in \mathbb{R}$, where $\xi_r := [\xi_1^1, \dots, \xi_1^m]^T \in \mathbb{R}^m$. Let $p := r_1 + \dots + r_m$, with $\xi \in \mathbb{R}^p$, $\eta \in \mathbb{R}^{n-p}$, input $u \in \mathbb{R}^m$, output $y \in \mathbb{R}^m$. Assume f is a smooth \mathbb{R}^p valued function of ξ , η , u , and t . Multi-input, multi-output systems having well-defined vector relative degree $[r_1, \dots, r_m]$ may be put into the form 17 through a state-dependent change of coordinates [Isi89]. We refer to the evolution of η as the *internal dynamics* of (17).

Let $y_d(t) \in \mathbb{R}^m$ satisfy $y_{di}(t) \in C^{r_i-1}$. Let

$$\|y_d\|_Y := \max_{i \in \underline{m}} \sup_{t \in \mathbb{R}_+} \{|y_{di}^{(0)}(t)|, \dots, |y_{di}^{(r_i)}(t)|\}.$$

Let \mathcal{B}_κ^Y be the open κ -ball in the $\|\cdot\|_Y$ norm. Assume that if output $y \in \mathcal{B}_\kappa^Y$, implies $\|\eta\|$ is bounded.

Let $F : \mathbb{R}^m \times \mathbb{R}_+ \rightarrow \mathbb{R}^m$ be such that $\theta_*(t) \in \mathcal{B}_\kappa^Y$ is a continuous isolated solution of $F(\theta, t) = 0$. Assume that $F(\theta, t)$ is smooth in θ and t . Let $\{\beta_i\}, i \in \{0, \dots, r-1\}$ be chosen to be the coefficients of the polynomial $s^r + \sum_{i=0}^{r-1} \beta_i s^i$ such that all roots of the polynomial have strictly negative real parts. If we had direct access to $\theta_*^{(i)}, i \in \{0, \dots, r\}$, where $\theta_*^{(k)}$ denotes the k^{th} derivative of θ_* with respect to time, and $\theta_*^{(0)} := \theta_*$, then the choice of input $u_i = \theta_*^{(r_i)} - \sum_{k=1}^{r_i} \beta_k (\xi_k^i - \theta_*^{(k-1)})$, $i \in \underline{m}$ would cause y to track θ_* with exponentially decaying error.

We will join the dynamic inverter of Example 3 to the control system (17) using singular perturbation theory to prove stability of the combination.

Let $E^0 := \theta$. In a similar manner to the manner in which $E^1(\Gamma, \theta, t)$ was obtained, we may obtain an estimator for $\theta_*^{(k)}$ for any $k \geq 1$ by the following recursive procedure: (1) Differentiate $(d^{k-1}/dt^{k-1})(F(\theta_*, t) = 0)$ with respect to t . (2) Replace Γ_* , θ_* and $\theta_*^{(k-1)}$ by their estimators Γ , θ , and $E^{k-1}(\Gamma, \theta, t)$ respectively.

Consider the dynamic inverter

$$\begin{bmatrix} \dot{\Gamma} \\ \dot{\theta} \end{bmatrix} = -\mu \tilde{G}(\tilde{F}(\Gamma, \theta, t), \Gamma) + \tilde{E}(\Gamma, \theta, t), \quad (18)$$

where

$$\tilde{F}(\Gamma, \theta, t) = \begin{bmatrix} D_1 F(\theta, t) \Gamma - I \\ F(\theta, t) \end{bmatrix}, \quad \tilde{G}(w^M, w, \Gamma) = \begin{bmatrix} \Gamma \cdot w^M \\ \Gamma \cdot w \end{bmatrix},$$

$$\tilde{E}^1(\Gamma, \theta, t) = \begin{bmatrix} E^M(\Gamma, \theta, t) \\ E^1(\Gamma, \theta, t) \end{bmatrix},$$

with $E^M(\Gamma, \theta, t)$ defined as in (15). The following theorem asserts that the concatenation of the dynamic inverter (18) with the control system (17) can be used for exponentially convergent tracking of implicit trajectories.

Theorem 2 (Implicit Tracking Theorem) *Let*

$$u_i = E_i^{r_i}(\Gamma, \theta, t) - \sum_{k=1}^{r_i} \beta_k (\xi_k^i - E_i^{k-1}(\Gamma, \theta, t)) \quad (19)$$

for $i \in \underline{m}$, where $E^0 = \theta$, and E^j , $j \in \underline{m}$ are defined as above, and where θ and Γ are the solutions to (18). If $(\theta(0) - \theta_*(0))$, $(\Gamma(0) - \Gamma_*(0))$, and $(\xi_j^i(0) - \theta_{*i}^{(j-1)}(0))$, $i \in \underline{m}$, $j \in \underline{r}_i$, are sufficiently small, then $y(t)$, the output of (17), converges exponentially to $\theta_*(t)$.

Proof. First note that if $\theta(t) \equiv \theta_*(t) \in \mathcal{B}_\kappa^Y$ then

$$u_i = \theta_*^{(r_i)} - \sum_{k=1}^{r_i-1} \beta_k (\xi_k^i - \theta_*^{(r_i)}), \quad i \in \underline{m} \quad (20)$$

and (17) has exponentially stable error tracking dynamics with $\|\eta\|$ bounded. Second, by Theorem 1, (18) has exponentially stable estimation error dynamics for ϵ sufficiently small. Third, let $\epsilon := 1/\mu$ so that the dynamic inverter becomes

$$\epsilon \begin{bmatrix} \dot{\Gamma} \\ \dot{\theta} \end{bmatrix} = -\tilde{G}(\tilde{F}(\Gamma, \theta, t), \Gamma) + \epsilon \tilde{E}(\Gamma, \theta, t). \quad (21)$$

When $\epsilon = 0$,

$$0 = -\tilde{G}(\tilde{F}(\Gamma, \theta, t), \Gamma) + 0 \quad (22)$$

which implies that $(\Gamma, \theta) = (\Gamma_*, \theta_*)$. Thus, if $\|\eta\|$ is bounded under application of (19), then by Theorem 8.3 of [Kha92] regarding exponential stability of

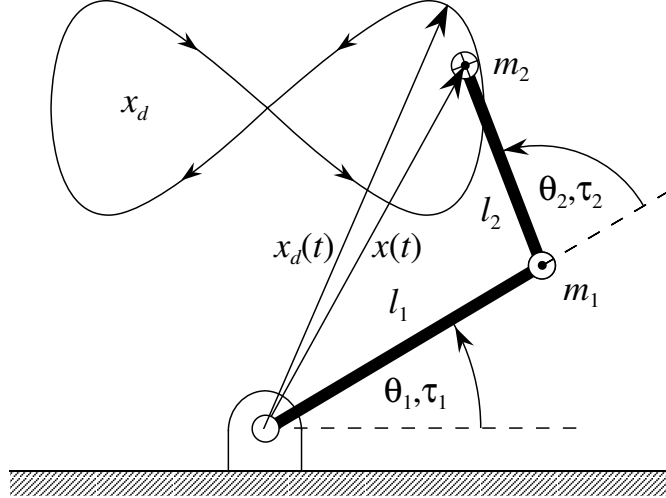


Figure 1: A two-link robot arm with joint angles $\theta = (\theta_1, \theta_2)$, joint torques $\tau = (\tau_1, \tau_2)$, end-effector position x , desired end-effector position x_d , link lengths l_1 and l_2 , and link masses m_1 and m_2 .

singularly perturbed systems, $\xi_j^i(t) \rightarrow \theta_{*i}^{(j-1)}(t)$ exponentially for all $i \in \underline{m}$, $j \in r_i$, and ϵ sufficiently small. Consequently, $y(t) \rightarrow \theta_*(t)$ exponentially.

Boundedness of $\|\eta\|$ requires only that $\|y(t) - \theta_*(t)\|$ be sufficiently small, but because the exponential stability of the tracking error is independent of η , and because the derivative estimators are C^1 in θ and Γ , we need only assure ourselves that the errors $(\theta(0) - \theta_*(0))$, $(\Gamma(0) - \Gamma_*(0))$, and $(\xi_j^i(0) - \theta_{*i}^{(j-1)}(0))$, $i \in \underline{m}$, $(j \in r_i)$, are sufficiently small. This is true by hypothesis. \square

4 A Robot Control Example

The control of robotic manipulators provides a natural setting and motivation for the tracking of implicitly defined trajectories. In this section we will apply the controller described above to the problem of output tracking for a simple model of the two-link planar robot arm of Figure 1.

The links of the robot arm are assumed rigid and of length l_1 and l_2 . The masses of each link are assumed to be point masses m_1 and m_2 located at the distal ends of link 1 and link 2 respectively. The actual and desired positions of the end-effector at time t are $x(t)$ and $x_d(t)$ respectively. We wish to make the end-effector (end of the second link) track a prescribed trajectory $x_d(t)$ in the Euclidean plane. The configuration-space of the arm is parameterized by $\theta \in \mathbb{T}^2$ where \mathbb{T}^2 is the two-torus. For our purposes we may view \mathbb{T}^2 through a single chart from \mathbb{R}^2 since neither joint of the arm will ever undergo a full circular

motion. We will assume that we may exert a control torque at each joint and will denote the vector of input torques by $\tau \in \mathbb{R}^2$. The *forward-kinematics map* $\mathcal{F} : \mathbb{R}^2 \rightarrow \mathbb{R}^2$; $\theta \mapsto \mathcal{F}(\theta)$ maps the configuration space to the Euclidean plane. Let $c_i := \cos(\theta_i)$, $c_{ij} := \cos(\theta_i + \theta_j)$, $s_i := \sin(\theta_i)$, and $s_{ij} := \sin(\theta_i + \theta_j)$ with $i, j \in \{1, 2\}$. For the two-link arm the forward-kinematics map is

$$\mathcal{F}(\theta) = \begin{bmatrix} l_1 c_1 + l_2 c_{12} \\ l_1 s_1 + l_2 s_{12} \end{bmatrix}. \quad (23)$$

The *workspace* of the robot arm is defined as $\{x \in \mathbb{R}^2 : x = \mathcal{F}(\theta), \theta \in \mathbb{T}^2\}$, the image of the configuration space through the forward-kinematics map. We choose the output of the system to be θ . We wish to determine a τ such that the end-effector position $x(t) = \mathcal{F}(\theta(t))$ converges to the desired end-effector position $x_d(t)$.

In this example we will use θ to denote the actual joint angles of the robot arm, θ_* to denote the inverse kinematic solution of $\mathcal{F}(\theta, t) = 0$, and $\hat{\theta}$ to denote the estimator for θ_* .

For each x in the interior of the workspace, there exist two configurations θ satisfying $\mathcal{F}(\theta) = x$. Letting $F(\theta, t) := \mathcal{F}(\theta) - x_d(t)$ the *inverse-kinematics problem* is to find θ_* satisfying $F(\theta, t) = 0$. For robotic manipulators this problem typically has multiple solutions. For certain configurations, kinematics may be inverted by inspection, but in general the problem is difficult and computationally expensive, making algorithms for inverse-kinematics an active area of current research. In the case of our two-link robotic arm, closed form solutions for the inverse kinematics exist (see [Cra89], p.122). For demonstration purposes we will use dynamic inversion to invert the kinematics and we will use the closed form to check our results. As long as x_d is kept away from the boundary of the workspace, the two possible inverse kinematic solutions of $F(\theta, t) = 0$ never intersect. We will choose one, by our choice of initial conditions for dynamic inversion, and track it.

The equations of motion for the two link manipulator (see [Cra89], Section 6.8) are

$$M(\theta)\ddot{\theta} + V(\theta, \dot{\theta}) + K(\theta) = \tau \quad (24)$$

where

$$\begin{aligned} M_{11}(\theta) &= l_2^2 m_2 + 2l_1 l_2 m_2 c_2 + l_1^2 (m_1 + m_2) \\ M_{12} = M_{21} &= l_2^2 m_2 + l_1 l_2 m_2 c_2 \\ M_{22} &= l_2^2 m_2, \end{aligned}$$

$$V(\theta, \dot{\theta}) = \begin{bmatrix} -m_2 l_1 l_2 s_2 \dot{\theta}_2^2 - 2m_2 l_1 l_2 s_2 \dot{\theta}_1 \dot{\theta}_2 \\ m_2 l_1 l_2 s_2 \dot{\theta}_1^2 \end{bmatrix},$$

and

$$K(\theta) = \begin{bmatrix} m_2 l_2 g c_{12} + (m_1 + m_2) l_1 g c_1 \\ m_2 l_2 g c_{12} \end{bmatrix}.$$

Tracking Implicit Trajectories

The matrix $M(\theta)$ is a positive definite symmetric mass matrix, $V(\theta, \dot{\theta})$ is the vector of centrifugal and Coriolis forces on the manipulator, and $K(\theta)$ is the gravitational force on the arm. The output of our system is x with $x = \mathcal{F}(\theta)$. Let $\theta_*(t)$ be the solution of $F(\theta, t) = 0$. If we knew $\theta_*(t)$, $\dot{\theta}_*(t)$, and $\ddot{\theta}_*(t)$ we could set

$$\tau = V(\theta, \dot{\theta}) + K(\theta) + M(\theta) \left(\ddot{\theta}_* - \beta_1(\dot{\theta} - \dot{\theta}_*) - \beta_0(\theta - \theta_*) \right) \quad (25)$$

where β_1 and β_2 are positive definite matrices in $\mathbb{R}^{2 \times 2}$ to achieve exponential convergence of θ to $\theta_*(t)$. But for generality we will assume that we don't know θ_* or its derivatives. We will use dynamic inversion to obtain them.

We obtain an estimator $E^1(\Gamma, t)$ for $\dot{\theta}_*$ by differentiating $F(\theta_*, t) = 0$,

$$\frac{d}{dt} F(\theta_*, t) = D\mathcal{F}(\theta_*)\dot{\theta}_* - \dot{x}_d(t) = 0 \quad (26)$$

Consequently, $\dot{\theta}_* = D\mathcal{F}(\theta_*)^{-1}\dot{x}_d(t)$. Again for generality, rather than symbolically or numerically invert $D\mathcal{F}(\theta_*)$ we will solve $D\mathcal{F}(\theta)\Gamma - I = 0$ for Γ . Thus

$$E^1(\Gamma, t) = \Gamma\dot{x}_d(t). \quad (27)$$

We will also require an estimator for $\ddot{\theta}_*$. Note that

$$\frac{d}{dt} D\mathcal{F}(\hat{\theta}) \Big|_{\dot{\hat{\theta}}=E^1(\Gamma, t)} = \frac{\partial D\mathcal{F}(\theta)}{\partial \theta_1} E_1^1(\Gamma, t) + \frac{\partial D\mathcal{F}(\theta)}{\partial \theta_2} E_2^1(\Gamma, t). \quad (28)$$

Differentiating (26) with respect to t , solving for $\ddot{\theta}_*$, and replacing $D\mathcal{F}(\theta_*)^{-1}$ with Γ , and $\dot{\theta}_*$ with $E^1(\hat{\theta}, t)$ gives an estimator for $\ddot{\theta}_*$ in terms of Γ , and t ,

$$E^2(\Gamma, \hat{\theta}, t) := \Gamma \left(\ddot{x}_d - \frac{d}{dt} D\mathcal{F}(\hat{\theta}) \Big|_{\dot{\hat{\theta}}=E^1(\Gamma, t)} E^1(\Gamma, t) \right). \quad (29)$$

For the estimator E^M for $\dot{\Gamma}_*$ we get

$$E^M(\Gamma, \hat{\theta}, t) := -\Gamma \frac{d}{dt} D\mathcal{F}(\hat{\theta}) \Big|_{\dot{\hat{\theta}}=E^1(\Gamma, t)} \Gamma. \quad (30)$$

Assembling our results, our controller becomes

$$\begin{aligned} \dot{\Gamma} &= -\mu\Gamma(D\mathcal{F}(\hat{\theta})\Gamma - I) + E^M(\Gamma, \hat{\theta}, t) \\ \dot{\hat{\theta}} &= -\mu\Gamma(\mathcal{F}(\hat{\theta}) - x_d(t)) + E^1(\Gamma, t) \\ \tau &= M(\theta) \left(E^2(\Gamma, \hat{\theta}, t) - \beta_1(\dot{\theta} - E^1(\Gamma, t)) - \beta_0(\theta - \hat{\theta}) \right) \end{aligned} \quad (31)$$

which, by Theorem 2 give exponentially convergent tracking $\theta \rightarrow \theta_*$.

We choose $x_d(t)$ to be a time parameterized figure-eight in the workspace,

$$x_d(t) = [3.75 \cos(\pi t), 2 + 1.5 \sin(2\pi t)].$$

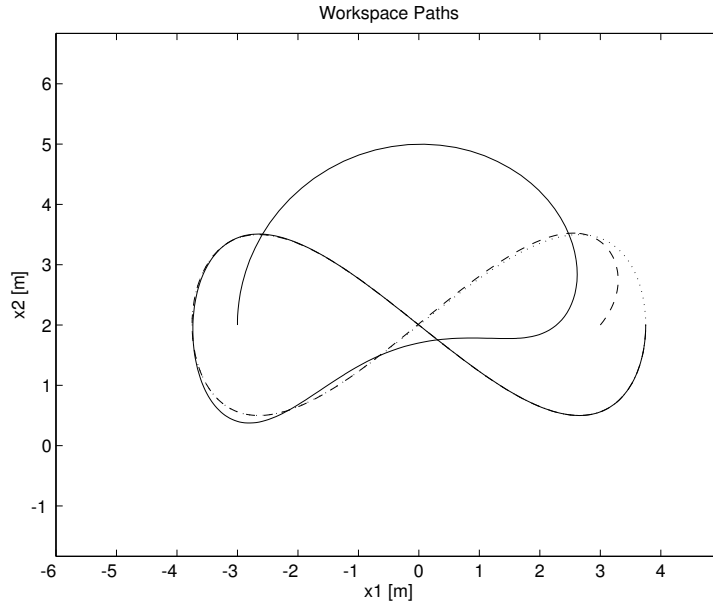


Figure 2: Workspace paths: $\mathcal{F}(\theta)$ (solid), $\mathcal{F}(\hat{\theta})$ (dashed), and $\mathcal{F}(\theta_*)$ (dotted).

Figures 2 through 4 show the results of a simulation. The integration was performed in Matlab [Mat92] using an adaptive step-size Runge-Kutta integrator. The parameters used in the simulation were $\mu = 10$, $\beta_1 = 10 I$, $\beta_0 = 100 I$, $l_1 = 3[\text{m}]$, $l_2 = 2[\text{m}]$, and $m_1 = m_2 = 1[\text{kg}]$ with $g = 9.8[\text{m/s}^2]$. The initial conditions are $\hat{\theta}(0) = [0, \pi/2]$, $\Gamma(0) = D\mathcal{F}(\hat{\theta}(0)) = [0, 1/3, -1/2, 1/3]$, $\theta(0) = [\pi, -\pi/2]$, $\dot{\theta}(0) = 0$ with all angles in radians. Figure 2 shows the resulting end-effector path (solid), desired path (dotted), and the image of $\hat{\theta}$ through \mathcal{F} in the workspace (dashed). Both the image of $\hat{\theta}$ through \mathcal{F} , and the path of the end-effector can be seen to converge to the desired path. Figure 3 shows a similar picture, but in configuration space. Again, the convergence of both $\hat{\theta}$ and θ to the inverse kinematic solution corresponding to the desired trajectory can be seen. Figure 4 shows the norm of the estimation error $\hat{\theta} - \theta_*$, (top) and the tracking error $[\theta(t), \dot{\theta}(t)] - [\theta_*(t), \dot{\theta}_*(t)]$ (bottom) graphed versus time.

5 Conclusions

We have shown that through a well defined dynamical method we may produce explicit estimators for an implicit output reference trajectory and its time derivatives, where those estimators converge exponentially to the true values of the quantities which they estimate. Starting with a state-feedback controller

Tracking Implicit Trajectories

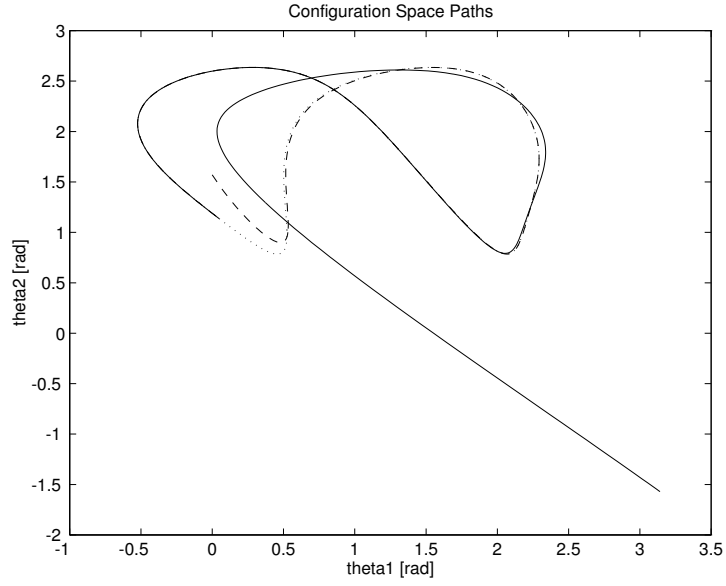


Figure 3: Configuration space paths: θ (solid), $\hat{\theta}$ (dashed), and θ_* (dotted).

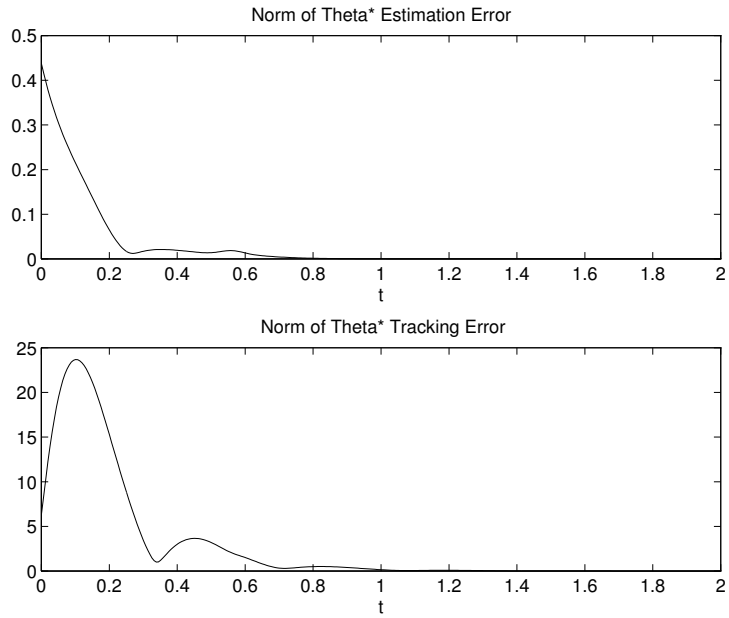


Figure 4: Error norms, $\|\hat{\theta}(t) - \theta_*(t)\|_2$ (top), and $\|(\theta(t), \dot{\theta}(t)) - (\theta_*(t), \dot{\theta}_*(t))\|_2$ (bottom).

designed for exponential tracking of explicit output reference trajectories, we replaced the explicit reference trajectory and its time derivatives by our estimators. We then proved, through an appeal to a theorem from the theory of singularly perturbed control systems, that the combination of minimum phase nonlinear plant, dynamic estimator, and controller results in exponentially convergent output tracking with well-behaved internal dynamics.

The authors are grateful to C.A. Desoer and S.M. Shahruz for their comments and advice.

References

- [Cra89] J.J. Craig. *Introduction to Robotics, Mechanics and Control*. Addison Wesley, New York, second edition, 1989.
- [GM94] N. H. Getz and J. E. Marsden. Dynamic inversion of nonlinear maps. Technical Report 621, Center for Pure and Applied Mathematics, Berkeley, California, 19 December 1994.
- [GM95] N. H. Getz and J. E. Marsden. Dynamical methods for polar decomposition and inversion of matrices. Technical Report 624, Center for Pure and Applied Mathematics, Berkeley, California, 5 January 1995.
- [Isi89] A. Isidori. *Nonlinear Control Systems, An Introduction*. Springer-Verlag, New York, second edition, 1989.
- [Kha92] H.K. Khalil. *Nonlinear Systems*. Macmillan, New York, 1992.
- [Mat92] *Matlab*. The MathWorks, Inc., Natick, Mass., 1992.
- [NTV91] S. Nicosia, A. Tornambè, and P. Valigi. A solution to the generalized problem of nonlinear map inversion. *Systems and Control Letters*, 17(5), 1991.

A Programs

In this appendix we include matlab programs used for the simulation of the implicit tracking control of a two-link robot arm. These are the programs that were used to produce the illustrations in the article.

```
% PROGRAM: runarm
%
% DESCRIPTION: This program runs a simulation of the control of
%              a two-link robot arm using dynamic inversion.
%
% NOTES: Use "plotarm" to plot results.
%
% For "Tracking Implicit Trajectories", N.H. Getz and J.E. Marsden.
%
% PROGRAM: runarm
%
% AUTHOR: Neil Getz
%
% ORGANIZATION: University of California at Berkeley
%
% DATE: 2-8-95

global r1path r2path hpath

% parameters for figure-eight
r1path = 3.75;
r2path = 1.5;
hpath = 2;

l1 = 3;    % length of first link
l2 = 2;    % length of distal link
m1 = 1;    % mass of first link
m2 = 1;    % mass of distal link
mu = 10;   % dynamic inversion gain
beta1 = 1; % control gains
beta0 = 1;
g = 9.8;   % gravitational acceleration

% Initial conditions for estimator.
thhat0 = [0; pi/2]; % Initial theta estimation
s1hat = sin(thhat0(1));
c1hat = cos(thhat0(1));
s12hat = sin(thhat0(1) + thhat0(2));
c12hat = cos(thhat0(1) + thhat0(2));
DForwhat0 = [[(-l1*s1hat-l2*s12hat), -l2*s12hat],
              [(l1*c1hat + l2*c12hat), l2*c12hat]];
```

```

gam0 = minv(DForwhat0^(-1)); % Inverse of DFor at thhat0.

% Initial conditions for plant (robot).
th0 = [pi; -pi/2]; % Starting robot configuration.
thdot0 = [0;0]; % Starting joint velocities.

q0 = [gam0; thhat0; th0; thdot0]; % Initial conditions for estimator and plant.

clear T Q t q

Q = q0';
INC = 0.1;
T0 = 0;
TF = 2;
T=T0;
N = (TF-T0)/INC; % Must make TF an integer multiple of INC.
% This loop makes it so that if you stop the simulation before
% TF is reached, you don't loose all of your data.
for(i=1:N),
    last = length(T);
    fprintf('Starting from t = %g\n',T(last));
    [t,q] = ode45('armcontrol',T(last),T(last)+INC,Q(last,:));
    lent = length(t);
    T= [T;t(2:lent)];
    Q = [Q;q(2:lent,:)];
    last = length(T);
end;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% FUNCTION: armcontrol
%
% SYNOPSIS: qdot = armcontrol(t,q)
%
% DESCRIPTION: Vector field for simulated control of a two-link planar
% robot arm with first link length l1, second link length
% l2, link 1 mass m1, link 2 mass m2 (point masses at
% distal ends). Dynamic inversion is used to invert the
% forward kinematics and incorporated into a controller
% to make the arm track a figure-eight. The argument "t"
% is the time, and "q" is thestate at time t. See below
% for the identity of the elementsof q.
%
%
% The states x, thhat1, and thhat2 are states of the
% dynamic inverter. The states th1, th2, th1dot, th2dot
% are states of the arm.
%
%

```

Tracking Implicit Trajectories

```
% AUTHOR: Neil Getz
%
% ORGANIZATION: University of California at Berkeley
%
% DATE: 2-8-95
%
% For "Tracking Implicit Trajectories", N.H. Getz and J.E. Marsden.

function qdot = armcontrol(t,q)

global r1path r2path hpath

l1 = 3; % Length of first link.
l2 = 2; % Length of second link.
m1 = 1; % Mass at distal end of first link
m2 = 1; % Mass at distal end of second link
mu = 10; % Dynamic inversion gain.
beta1 = 10; % Control gain.
beta0 = 100; % Control gain.
k = 9.8; % Gravitational acceleration

% Desired End-Effector Trajectory
[xd,ddtxd,ddtddtxd] = figeight(t,r1path,r2path,hpath);
xd = xd';
ddtxd = ddtxd';
ddtddtxd = ddtddtxd';

gam = q(1:4); % gamma: States to estimate dynamic inverse
thhat1 = q(5); % thhat: States to estimate inverse kinematic soln.
thhat2 = q(6);
th1 = q(7); % th: States of robot arm.
th2 = q(8);
th1dot = q(9);
th2dot = q(10);

% Abbreviations for sines and cosines.
c1hat = cos(thhat1);
c2hat = cos(thhat2);
c12hat = cos(thhat1+thhat2);
s1hat = sin(thhat1);
s2hat = sin(thhat2);
s12hat = sin(thhat1+thhat2);
%
c1 = cos(th1);
c2 = cos(th2);
c12 = cos(th1+th2);
```

```
s1 = sin(th1);
s2 = sin(th2);
s12 = sin(th1+th2);

% Mass matrix
M11 = l2^2*m2 + 2*l1*l2*m2*c2+l1^2*(m1 + m2);
M12 = l2^2*m2 + l1*l2*m2*c2;
M22 = l2^2*m2;
M = [M11, M12; M12, M22];

% Coriolis and Centrifugal Forces
V = [-m2*l1*l2*s2*th2dot^2 - 2*m2*l1*l2*s2*th1dot*th2dot;
      m2*l1*l2*s2*th1dot^2];

% Gravitational Forces
K = [(m2*l2*k*c12 + (m1+m2)*l1*k*c1); m2*l2*k*c12];

% Forward Kinematics map on thhat.
Forwhat = [l1*c1hat+l2*c12hat; l1*s1hat+l2*s12hat];

% Its differential.
DForwhat = [[(-l1*s1hat-l2*s12hat), -l2*s12hat],
             [(l1*c1hat + l2*c12hat), l2*c12hat]];

% Partial of its differential w.r.t. th1.
dDForwd1hat = [[(-l1*c1hat-l2*c12hat), -l2*c12hat],
               [-l1*s1hat-l2*s12hat, -l2*s12hat]];

% Partial of its differential w.r.t. th2.
dDForwd2hat = [[-l2*c12hat, -l2*c12hat],
               [-l2*s12hat, -l2*s12hat]];

% Estimator for d/dt thhat.
E1 = m(gam)*ddtXd;

F = Forwhat-xd;

G = m(gam);

% d/dt D(Forw) with d/dt th -> E1
C = dDForwd1hat*E1(1) + dDForwd2hat*E1(2);

% d/dt m(x).
EM = minv(-m(gam)*C*m(gam));

FM = minv(m(gam)*DForwhat-eye(2));
```

Tracking Implicit Trajectories

```
% Dynamic inverse for dynamic inverse definition.
GM = [m(gam)',zeros(2,2); zeros(2,2), m(gam)'];

% Estimator for d2/dt2 thhat.
E2 = m(gam)*(ddtddtxd - C*E1);

% CONTROLLER
% Dynamic inverter.
ddtgam = -mu*GM*FM + EM;
ddtthhat = -mu*G*F + E1;
%
% Feedback torque.
tau = V + K + M*( ...
    E2 - beta1*([th1dot;th2dot]-E1) ...
    - beta0*([th1;th2]-[thhat1;thhat2])
);

% Arm dynamics.
ddtth = [th1dot; th2dot];
ddtthdot = M\(-V-K+tau);
qdot = [ddtgam; ddtthhat; ddtth; ddtthdot];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% FUNCTION: figeight
%
% SYNOPSIS: [xd,ddtxd,ddtddtxd] = figeight(t,r1,r2,h)
%
% DESCRIPTION: Generates a position, velocity, and acceleration of a
%               time-parameterized figure-eight in the euclidean plane.
%               Each output item is a 1 by 2 vector. xd is the x,y
%               position, ddtxd is the corresponding velocity, and
%               ddtddtxd is the corresponding acceleration. The
%               arguments are
%
%               t, the time
%               r1, half of the width of the figure-eight
%               r2, half of the height of the figure-eight
%               h, the y-coordinate of the center of the figure
%                   eight. The x-coordinate of the center is 0.
%
%               The period of the figure-eight is two seconds.
%
% For "Tracking Implicit Trajectories", N.H. Getz and J.E. Marsden.
%
% AUTHOR: Neil Getz
%
```

```
% ORGANIZATION: University of California at Berkeley
%
% DATE: 2-8-95

function [xd,ddttd,ddtddttd] = figeight(t,r1,r2,h)
xd = [(r1*cos(pi*t)), (h + r2*sin(2*pi*t))];
ddttd = [(-r1*pi*sin(pi*t)), (r2*2*pi*cos(2*pi*t))];
ddtddttd = [(-r1*pi*pi*cos(pi*t)), (-r2*2*pi*2*pi*sin(2*pi*t))];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% FUNCTION: m
%
% SYNOPSIS: Convert a vector to a matrix.
%

function M = m(x)
M = [x(1),x(2);x(3),x(4)];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% FUNCTION: m
%
% SYNOPSIS: Convert a matrix to a vector.
%

function x = minv(M)
x = [M(1,1);M(1,2);M(2,1);M(2,2)];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Program: plotarm
%
% DESCRIPTION: Plots various useful signals from the simulation of the
%               implicit tracking controller for the two-link robot
%               arm.
%
% For "Tracking Implicit Trajectories", N.H. Getz and J.E. Marsden.
%
% AUTHOR: Neil Getz
%
% ORGANIZATION: University of California at Berkeley
%
% DATE: 2-8-95

global r1path r2path hpath
```

Tracking Implicit Trajectories

```
r1 = 3;
r2 = 1.5;
h = 2;

l1 = 3;
l2 = 2;
m1 = 1;
m2 = 1;
mu = 10;
beta1 = 1;
beta0 = 1;
g = 9.8;

GAM1 = Q(:,1);
GAM2 = Q(:,2);
GAM3 = Q(:,3);
GAM4 = Q(:,4);
THHAT1 = Q(:,5);
THHAT2 = Q(:,6);
TH1 = Q(:,7);
TH2 = Q(:,8);
TH1DOT = Q(:,9);
TH2DOT = Q(:,10);

% Desired End-Effector Trajectory
[XD,DDTXD,DDTDDTXD] = figeight(T,r1path,r2path,hpath);

C1 = cos(TH1);
C2 = cos(TH2);
C12 = cos(TH1+TH2);
S1 = sin(TH1);
S2 = sin(TH2);
S12 = sin(TH1+TH2);

C1HAT = cos(THHAT1);
C2HAT = cos(THHAT2);
C12HAT = cos(THHAT1+THHAT2);
S1HAT = sin(THHAT1);
S2HAT = sin(THHAT2);
S12HAT = sin(THHAT1+THHAT2);

EndEf = [l1*C1+l2*C12, l1*S1+l2*S12];
FORWHAT = [l1*C1HAT+l2*C12HAT, l1*S1HAT+l2*S12HAT];
DFORWHAT = [(-l1*S1HAT-l2*S12HAT), -l2*S12HAT, ...
            (l1*C1HAT + l2*C12HAT), l2*C12HAT ];
```

```

DDFORWD1HAT = [(-11*C1HAT-12*C12HAT), -12*C12HAT , ...
               -11*S1HAT-12*S12HAT, -12*S12HAT ];
DDFORWD2HAT = [-12*C1HAT, -12*C1HAT, -12*S1HAT, -12*S1HAT];

ActualTh = zeros(length(T),2);
for i = 1:length(T),
    ActualTh(i,:) = actualtheta(T(i));
end;
THACT1 = ActualTh(:,1);
THACT2 = ActualTh(:,2);
C1ACT = cos(THACT1);
C2ACT = cos(THACT2);
C12ACT = cos(THACT1+THACT2);
S1ACT = sin(THACT1);
S2ACT = sin(THACT2);
S12ACT = sin(THACT1+THACT2);
DFORWACT = [(-11*S1ACT-12*S12ACT), -12*S12ACT, ...
            (11*C1ACT + 12*C12ACT), 12*C12ACT ];
DDFORWD1ACT = [(-11*C1ACT-12*C12ACT), -12*C12ACT , ...
               -11*S1ACT-12*S12ACT, -12*S12ACT ];
DDFORWD2ACT = [-12*C1ACT, -12*C1ACT, -12*S1ACT, -12*S1ACT];
THDOTACT = zeros(length(T),2);
THDOTDOTACT = zeros(length(T),2);
% CACT is d/dt DF(THACT).
CACT = zeros(length(T),4);
for i = 1:length(T),
    THDOTACT(i,:) = ( m(DFORWACT(i,:))\DDTDXD(i,:) )';
    CACT(i,:) = minv( ...
    m(DDFORWD1ACT(i,:))*THDOTACT(i,1) ...
    + m(DDFORWD2ACT(i,:))*THDOTACT(i,2) ...
    )';
    THDOTDOTACT(i,:) = ( ...
    m(DFORWACT(i,:))\ ( DDTDDTDXD(i,:) - m(CACT(i,:))*THDOTACT(i,:) ) ...
    )';
end;

figure(1); % WorkPaths.eps
plot(EndEf(:,1),EndEf(:,2),'-',XD(:,1),XD(:,2),':', ...
     FORWHAT(:,1),FORWHAT(:,2),'--');
title('Workspace Paths');
xlabel('x1 [m]'); ylabel('x2 [m]');
axis('equal');
print -deps WorkPaths.eps

figure(2); % ConfigPaths.eps
plot(TH1,TH2,'-',ActualTh(:,1),ActualTh(:,2),':',THHAT1,THHAT2,'--');
xlabel('theta1 [rad]'); ylabel('theta2 [rad]');

```

Tracking Implicit Trajectories

```
title('Configuration Space Paths');
print -deps ConfigPaths.eps

figure(3); % XVsT.eps
subplot(2,1,1),
plot(T,EndEf(:,1),'-',T,XD(:,1),':', T,FORWHAT(:,1),'--');
xlabel('t'); ylabel('x1');
title('Workspace Trajectories');
subplot(2,1,2),
plot(T,EndEf(:,2),'-',T,XD(:,2),':', T,FORWHAT(:,2),'--');
xlabel('t'); ylabel('x2');
print -deps XVsT.eps

figure(4); % ThetaVsT.eps
subplot(2,1,1),
plot(T,TH1,'-',T,ActualTh(:,1),':',T,THHAT1,'--');
xlabel('t'); ylabel('theta1');
title('Configuration Space Trajectories');
subplot(2,1,2),
plot(T,TH2,'-',T,ActualTh(:,2),':', T,THHAT2,'--');
xlabel('t'), ylabel('theta2');
print -deps ThetaVsT.eps

figure(5); % EstErrEnergy.eps
EstNorm = zeros(length(T),1);
TrackNorm = zeros(length(T),1);
EstNorm = zeros(length(T),1);
for i = 1:length(T),
    % Norm of estimation error.
    EstNorm(i) = norm([THHAT1(i),THHAT2(i)]-ActualTh(i,:));
    s1act = sin(ActualTh(i,1));
    s12act = sin(ActualTh(i,1) + ActualTh(i,2));
    c1act = cos(ActualTh(i,1));
    c12act = cos(ActualTh(i,1)+ActualTh(i,2));
    DForwhat = [(-l1*s1act-l2*s12act), -l2*s12act;
                (l1*c1act + l2*c12act), l2*c12act];
    ActualThDot = DForwhat\DDTXD(i,:);
    TrackNorm(i) = norm( ...
        [TH1(i),TH2(i),TH1DOT(i),TH2DOT(i)]-[ActualTh(i,:), ...
        ActualThDot]);
end;
subplot(2,1,1)
plot( T, EstNorm,'-');
xlabel('t');
title('Norm of Theta* Estimation Error');
subplot(2,1,2)
plot(T, TrackNorm,'-');
```

```

xlabel('t');
title('Norm of Theta* Tracking Error');
print -deps ErrEnergy.eps

E1 = zeros(length(T),2);
E2 = zeros(length(T),2);
CHAT = zeros(length(T),4);
for i = 1:length(T),
    E1(i,:) = (m(Q(i,:))*DDTXD(i,:))';
    CHAT(i,:) = minv( ...
m(DDFORWD1HAT(i,:))*E1(i,1) + m(DDFORWD2HAT(i,:))*E1(i,2) ...
)');
    E2(i,:) = (...
m(Q(i,:))*(...
DDTDDTXD(i,:)'- m(CHAT(i,:))*E1(i,:)' ...
) ...
)');
end;

figure(6),
subplot(2,1,1),
plot(T, E1(:,1), 'r-', T, THDOTACT(:,1), 'y:');
xlabel('t'); ylabel('E1(1) (-) and d/dt theta*(1) (:)'');
title('E1(1) (-), d/dt theta*(1) (:)'');
subplot(2,1,2),
plot(T, E1(:,2), '- ', T, THDOTACT(:,2), ':');
xlabel('t'); title('E1(2) (-) and d/dt theta*(2) (:)'');

figure(7),
subplot(2,1,1),
plot(T, E2(:,1), '- ', T, THDOTDOTACT(:,1), ':');
xlabel('t');
title('E2(1) and d^2 theta_*(1) / dt^2 ');
subplot(2,1,2),
plot(T, E2(:,2), 'c-', T, THDOTDOTACT(:,2), 'b:');
xlabel('t');
title('E2(2) and d^2 theta_*(2) / dt^2');

figure(8)
subplot(2,1,1),
plot(T(1:length(T)-1), E1(1:(length(T)-1),1), '- ', ...
T(1:length(T)-1), diff(THHAT1)./diff(T), ':');
xlabel('t'); ylabel('E1(1)');
subplot(2,1,2),
plot(T(1:length(T)-1), E1(1:(length(T)-1),2), '- ', ...
T(1:length(T)-1), diff(THHAT2)./diff(T), ':');
title('Checking E1 (-) as Estimator of d/dt thetahat (:)'');

```

Tracking Implicit Trajectories

```
xlabel('t'); ylabel('E1(2)');
```