JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 21, No. 6, November–December 1998

Engineering Notes

ENGINEERING NOTES are short manuscripts describing new developments or important results of a preliminary nature. These Notes cannot exceed 6 manuscript pages and 3 figures; a page of text may be substituted for a figure and vice versa. After informal review by the editors, they may be published wi thin a few months of the date of receipt. Style requirements are the same as for regular contributions (see inside back cover).

Synthesis of Real-Time Robust Flexible Structure Controllers

Chun-Liang Lin*
Feng Chia University,
Taichung 40724, Taiwan, Republic of China

Introduction

THE main problem in the control design for a large flexible structure is the modeling error arising from truncating the original infinite-dimensional model and from the lack of accurate knowledge of the parameters.[1] Several strategies have been reported to overcome the instability problem resulting from the modeling uncertainty.[2] Recently, a Riccati equation-based robust control law has also been proposed to treat the problem.[3] To synthesize the control law, one needs to solve Lyapunov and Riccati matrix equations. Traditionally, these equations are solved using a variety of numerical algorithms. However, carrying out these algorithms is admittedly time consuming even with high-speed computers. Thus, most algorithms are not appropriate for on-line operation. It has been pointed out that the time delay caused by solving the algebraic matrix Riccati equation on line may deteriorate the performance of an adaptive linear quadratic system significantly.[4] This Note presents a real-time solution to the robust control law proposed in Ref. 3 using recurrent neural networks. This allows an on-line implementation with faster adaptivity than the traditional sequential numeric algorithms.

Robust Control for Flexible Structures

Typical large flexible structures are usually described in the modal space by

$$\dot{x}_c = A_c x_c + B_c u, \qquad \dot{x}_r = A_r x_r + B_r u$$
$$y = C_c x_c + C_r x_r \tag{1}$$

where x_c, x_r, u, and y are, respectively, the 2n-dimensional controlled mode state, the residual mode state, the v-dimensional control vector, and the 8-dimensional output vector. The following observer-based controller is extensively adopted to control the preceding plant:

$$\dot{\hat{x}}_c = A_c\hat{x}_c + B_c u + K_c(y - C_c\hat{x}_c), \qquad u = -G_c\hat{x}_c \tag{2}$$

where G_c and K_c are, respectively, the regulator and observer gain matrices. It has been known that control for the truncated model may lead to control and observation spillover that can induce spillover instability.

Assume that the upper bound for the residual dynamics is estimated as

$$\left\| C_r(j\omega I - A_r)^{-1} B_r \right\|_\infty \le \alpha \tag{3}$$

where $\|\cdot\|_\infty$ is the H-infinity norm. A suitable robust control law with respect to the perturbation can be obtained by solving the following algebraic matrix Riccati and Lyapunov equations[3]:

$$A_c^T X_g + X_g A_c - X_g B_c B_c^T X_g < 0 \tag{4}$$

$$A_c^T X_e + X_e A_c - \beta^2 C_c^T C_c < 0 \tag{5}$$

where $\beta = 1/\alpha$. Once the matrix solutions X_e and X_g are determined, the regulator gain matrix is given by

$$G_c = B_c^T X_g \tag{6}$$

and the observer gain matrix is determined from the following matrix identity:

$$(X_e - X_g)K_c = \beta^2 C_c^T \tag{7}$$

We will assume in the following that the actual values of A_c, B_c, and C_c have been obtained from an identifier.

Recurrent Neural Networks Solving for Matrix Equations

Let the system of linear matrix equations be described as

$$G(X) = 0 \tag{8}$$

where $X \in \boldsymbol{R}^{p\times q}$ is the solution matrix, and $G \in \boldsymbol{R}^{r\times s}$ is a matrix of functions of X named as the objective function matrix. To solve the problem using recurrent neural networks, the first step is to construct an appropriate computation energy function

$$E[G(X)] = \sum_{i=1}^{r}\sum_{j=1}^{s} e_{ij}[g_{ij}(X)] \tag{9}$$

where $g_{ij}(\cdot), i = 1,\ldots,r$, and $j = 1,\ldots,s$ are the elements of $G(\cdot)$. The following functions are the common choices for $E(\cdot)$:

$$e_{ij}[g_{ij}(X)] = \tfrac{1}{2}g_{ij}^2(X), \qquad |g_{ij}(X)|, \qquad \cosh[g_{ij}(X)] \tag{10}$$

The neural dynamics solving for standard linear matrix equations is proposed in Ref. 5:

$$\frac{\mathrm{d}x_{ij}}{\mathrm{d}t} = -\eta \sum_{k=1}^{r}\sum_{l=1}^{s} \frac{\partial g_{kl}[X(t)]}{\partial x_{ij}} f_{kl}\{g_{kl}[X(t)]\}$$
$$i = 1,\ldots,p, \qquad j = 1,\ldots,q \tag{11}$$

where $\eta > 0$ is the learning rate and $f_{kl}(g_{kl}) = \partial e_{kl}/\partial g_{kl}$. The mapping $f_{kl}(\cdot)$ can be interpreted as an activation function. For example, if $e_{kl}(g_{kl}) = |g_{kl}|$, then $f_{kl}(g_{kl}) = \mathrm{sgn}(g_{kl})$. Obviously, this is a hard limiting type of activation functions with the activation input g_{kl}. It is straightforward to show that the system of differential equations (11) is asymptotically stable and its steady-state matrix represents the solution matrix.[5]

Received April 4, 1997; revision received June 10, 1998; accepted for publication June 20, 1998.

*Associate Professor, Department of Automatic Control Engineering. E-mail: chunlin@fcu.edu.tw.

We may show that Eq. (11) is basically a standard result of the gradient algorithm; i.e., every element in the solution matrix is to be changed in the direction of the negative gradient of E:

$$\frac{\mathrm{d}X(t)}{\mathrm{d}t} = -\eta\frac{\partial E}{\partial X} = -\eta\begin{bmatrix} \dfrac{\partial E}{\partial x_{11}} & \cdots & \dfrac{\partial E}{\partial x_{1q}} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial E}{\partial x_{p1}} & \cdots & \dfrac{\partial E}{\partial x_{pq}} \end{bmatrix} \tag{12}$$

The architecture of the recurrent neural network solving for linear matrix equations consists of several bidirectionally connected layers of neurons. Elements of the solution matrix X are presented on the synaptic weights.

Neural Networks Solving for Robust Control Law

We now extend the result of Eq. (12) to solve the algebraic matrix Riccati equation (4) and the Lyapunov equation (5). For Eq. (4), define the objective function matrix

$$A_c^T X_g + X_g A_c - X_g B_c B_c^T X_g + Q_1 = [g_{1ij}(X_g)] \qquad i, j = 1, \ldots, 2n \tag{13}$$

where the given matrix $Q_1 = Q_1^T > 0$. A Riccati equation has many solutions. However, if the equation has a symmetric positive definite solution, then it is unique. Suppose that there is a positive definite solution to Eq. (13). To avoid the solution obtained converging to the one that is not positive definite, it is suggested to introduce an additional constraint as

$$X_g = R_1 R_1^T \tag{14}$$

where R_1 is some nonsingular matrix.[6] The corresponding objective function matrix is

$$X_g - R_1 R_1^T = [g_{2ij}(X_g, R_1)], \qquad i, j = 1, \ldots, 2n \tag{15}$$

To solve X_g, we define the computation energy

$$E_1[G(X_g, R_1)] = \frac{1}{2}\sum_{i=1}^{2n}\sum_{j=1}^{2n}\left(g_{1ij}^2(X_g) + g_{2ij}^2(X_g, R_1)\right)$$

Using Eq. (12), we can derive the following neural dynamic equations[7]:

$$\frac{\mathrm{d}X_g}{\mathrm{d}t} = -\eta_g\Big[\left(A_c - B_c B_c^T X_g\right)F_1(X_g) + F_1(X_g)\left(A_c - B_c B_c^T X_g\right)^T + F_2(X_g, R_1)\Big] \tag{16a}$$

$$\frac{\mathrm{d}R_1}{\mathrm{d}t} = \eta_{r1} F_2(X_g, R_1) R_1 \tag{16b}$$

where the learning rates $\eta_g, \eta_{r1} > 0$, and the linear activation function matrices

$$F_1(X_g) = F_1\left(A_c^T X_g + X_g A_c - X_g B_c B_c^T X_g + Q_1\right) \tag{17a}$$

$$F_2(X_g, R_1) = F_2\left(X_g - R_1 R_1^T\right) \tag{17b}$$

i.e., each element of $F_{1,2}(\cdot)$ has the unit gain. Once X_g is generated, the regulator gain is obtained as

$$G_c = B_c^T X_g \tag{18}$$

For Eq. (5), we define the objective function

$$A_c^T X_e + X_e A_c - \beta^2 C_c^T C_c + Q_2 = [g_{3ij}(X_e)] \qquad i, j = 1, \ldots, 2n \tag{19}$$

where $Q_2 = Q_2^T > 0$ is a given matrix. The observer gain matrix is determined via

$$X_h K_c = \beta^2 C_c^T$$

where $X_h = X_e - X_g$. To ensure that the observer gain K_c is solvable, X_h is restricted to be positive definite. Because X_e and X_g are real symmetric matrices, there must exist some nonsingular matrix R_2 such that

$$X_e - X_g = R_2 R_2^T \tag{20}$$

To avoid X_h becoming singular during the transient phase, Eq. (20) will be imposed as an additional constraint. Now let

$$X_e - X_g - R_2 R_2^T = [g_{4ij}(X_e, R_2)], \qquad i, j = 1, \ldots, 2n \tag{21}$$

The computation energy is defined as

$$E_2[G(X_e, R_2)] = \frac{1}{2}\sum_{i=1}^{2n}\sum_{j=1}^{2n}\left(g_{3ij}^2(X_e) + g_{4ij}^2(X_e, R_2)\right)$$

The corresponding neural dynamics solving for X_e and R_2 can be obtained as follows:

$$\frac{\mathrm{d}X_e}{\mathrm{d}t} = -\eta_e\left[A_c F_3(X_e) + F_3(X_e)A_c^T + F_4(X_e, R_2)\right] \tag{22a}$$

$$\frac{\mathrm{d}R_2}{\mathrm{d}t} = \eta_{r2} F_4(X_e, R_2) R_2 \tag{22b}$$

where $\eta_e, \eta_{r2} > 0$ and

$$F_3(X_e) = F_3\left(A_c^T X_e + X_e A_c - \beta^2 C_c^T C_c + Q_2\right) \tag{23a}$$

$$F_4(X_e, R_2) = F_4\left(X_e - X_g - R_2 R_2^T\right) \tag{23b}$$

The matrix X_h is then determined from $X_h = X_e - X_g$.

In the preceding, $Q_{1,2}$ can be set as $\varepsilon_{1,2} I$ where $\varepsilon_{1,2} > 0$ are the tuning parameters. By changing $\varepsilon_{1,2}$, one obtains different X_e and X_g. This will assign the nominal regulator and observer poles to different locations on the complex plane.

To obtain the observer gain K_c, let

$$X_h K_c - \beta^2 C_c^T = [g_{5ij}(K_c)] \qquad i = 1, \ldots, 2n, \quad j = 1, \ldots, z \tag{24}$$

and the computation energy

$$E_3[G(K_c)] = \frac{1}{2}\sum_{i=1}^{2n}\sum_{j=1}^{z} g_{5ij}^2(K_c)$$

The neural dynamics solving for K_c is given by

$$\frac{\mathrm{d}K_c}{\mathrm{d}t} = -\eta_k X_h F_5(K_c) \tag{25}$$

where

$$F_5(K_c) = F_5\left(X_h K_c - \beta^2 C_c^T\right) \tag{26}$$

Note that we have used a recurrent neural network to find K_c instead of directly solving for the inverse matrix X_h^{-1} and executing the calculation $\beta^2 X_h^{-1} C^T = K_c$.

The architecture of the proposed networks consists of two bidirectionally connected layers. Equations (17), (23), and (26) act as the hidden layers, Eqs. (16), (22), and (25) as the output layers. Hidden layers perform a functional transformation. They calculate and propagate the objective function matrices through the activation functions $F_i(\cdot)$. The matrices Q_1, $-\beta^2 C_c^T C_c + Q_2$, and $-\beta^2 C_c^T$ act as the biasing thresholds adding to the hidden layers (17a), (23a), and (26), respectively. For the network (17), A_c and B_c are the inputs, and X_g and R_1 constitute the synaptic weights. For the network

(23), A_c and X_g are the inputs, and X_e and R_2 constitute the synaptic weights. As for the network (26), X_h acts as the unique input. All output layers perform integral transformations. There are no biases in the output layers. Solutions $x_{g_{ij}}$, r_{1ij}, x_{eij}, r_{2ij}, and k_{cij} are presented on the linking weights of the networks. During training, elements of the initial weights $R_1(0)$ and $R_2(0)$ and initial activation function matrices $F_1(\cdot) - F_5(\cdot)$ are set as small random numbers. Because Eqs. (16), (22), and (25) possess a stable feature, if the update gains are chosen sufficiently large, the desired control and observer gains will reach steady states in finite time.

Note that there are $2n \times 2n$ dynamic equations needed to be solved in Eq. (16a) or Eq. (22a). However, nearly half the computations are redundant because $Q_{1,2}$ are constrained to be symmetric positive definite. Thus, the network for solving Eq. (16a) or Eq. (22a) can be further reduced to $2n \times (2n+1)/2$ pairs.

The overall system will reach steady state after the neural networks converge. The gain G_c reaches steady state while the network (16) converges. The gain K_c will also reach steady state after the networks (16), (22), and (25) settle. Because G_c is uniquely determined by X_g, it will converge faster than K_c. The gain K_c reaches its steady value after X_g, X_e, and R_2 converge. To ensure the steady states of X_g and X_h being positive definite, we require that R_1 and R_2 converge faster than X_g and X_h. The learning rates may be arranged in the order $\eta_{r2} \geq \eta_e \geq \eta_{r1} \geq \eta_g$. However, the optimal choice usually depends on the problem being solved. These gains may be needed to choose experimentally. The satisfactory gain η_g is, in general, selected at first; $\eta_{e,r1,2}$ are then determined. The optimal η_k is determined at last. Larger values of η result in a more rapid convergence, but it may overshoot the solution or cause numerical instability. Thus in an aim to speed up the minimum seeking, careful selection of these gains should be done.

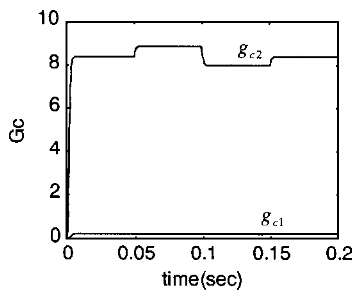

Fig. 1 Transition of the control gain.

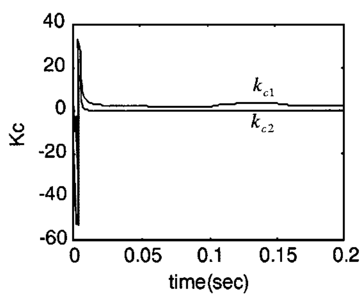

Fig. 2 Transition of the observer gain.

Simulation Result

To demonstrate our proposed result, we choose the standard beam model[8] as the control object. Let the first-order mode act as the controlled mode, the second- and third-order modes be residual modes. The nominal damping ratio ζ_n is assumed to be 0.01. For the model, the perturbation upper bound α is 0.048. We set the initial nominal position state to one and set other states to zeros. The matrices Q_1 and Q_2 are chosen as $Q_1 = 58.45I_2$ and $Q_2 = 155.85I_2$. These were found to give good performance and robust stability for the nominal controlled subsystems but are not claimed to be optimal in any sense. Learning rates $\eta_{r1} = \eta_g = 2000$, $\eta_{r2} = \eta_e = 3500$, and $\eta_k = 150$ were selected for numerical simulation.

To examine the adaptive qualities of the neural network, we let the damping ratio vary as follows: $\zeta_1 = 0.8\zeta_n$ for $0.05 \leq t < 0.1$, $\zeta_2 = 1.2\zeta_n$ for $0.1 \leq t < 0.15$, and $\zeta = \zeta_n$, otherwise. Figures 1 and 2 illustrate transients for the control gain $G_c = [g_{c1} \;\; g_{c2}]$ and observer gain $K_c = [k_{c1} \;\; k_{c2}]^T$, respectively. The gains converge, respectively, to the true solutions (within 0.01 squared error) in less than 0.01 and 0.05 s. The steady-state values (corresponding to the true solutions) are found as $G_c = [0.21 \;\; 8.40]$ and $K_c = [2.42 \;\; 0.11]^T$ for $\zeta = \zeta_n$, $G_c = [0.21 \;\; 8.83]$ and $K_c = [1.69 \;\; 0.56]^T$ for $\zeta = \zeta_1$, and $G_c = [0.21 \;\; 8.0]$ and $K_c = [3.36 \;\; 0.22]^T$ for $\zeta = \zeta_2$. The nominal steady gains assign the regulator eigenvalues to $-0.4 \pm i9.87$ and observer eigenvalues to $-0.96 \pm i9.88$. From the results we find that the neural network is capable of finding the correct solutions very quickly. Deflections for the controlled and uncontrolled beams are illustrated in Fig. 3. Because the neural dynamics respond much more quickly than the controller-plant dynamics, the transient behaviors of G_c and K_c do not affect explicitly the closed-loop response. It can also be found that the overall system is robustly stabilized.

Conclusion

This Note presents an on-line solution of a Riccati equation-based robust control law for flexible structures. The nature of parallel and distributed processing renders the proposed neural networks possessing the computational advantages over traditional sequential algorithms in real-time applications. In addition, as with most recurrent neural networks, they can be realized with very large-scale integration technology. Although the approach presented only treats linear time-invariant plants, it possesses the potential to investigate adaptive control problems or control problems for time-varying systems.

Acknowledgment

This research was sponsored by the National Science Council, Taiwan, Republic of China, under Grant NSC 88-2213-E-035-036.

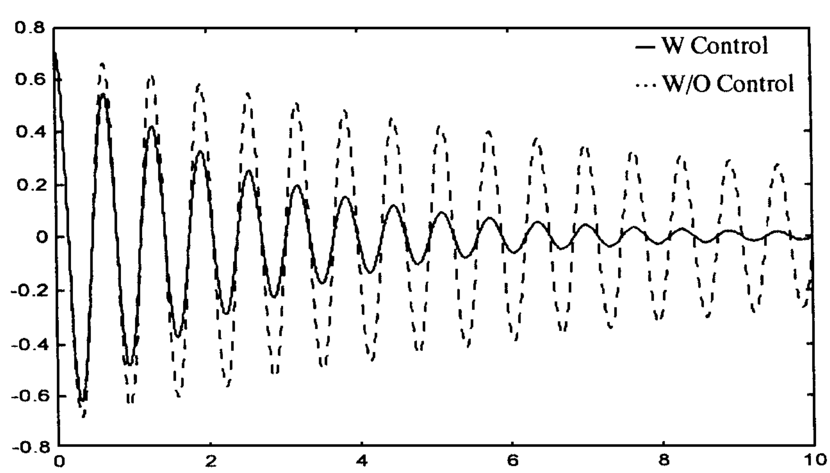

Fig. 3 Responses of controlled and uncontrolled beam deflections.

References

[1]Joshi, S. M., *Control of Large Flexible Space Structures*, Springer-Verlag, Berlin, 1989, pp. 24–26.

[2]Lin, C. L., "Robust Control of Flexible Structures Using Residual Mode Filters," *Journal of Guidance, Control, and Dynamics*, Vol. 16, No. 5, 1993, pp. 973–977.

[3]Lin, C. L., and Chen, B. S., "Robust Observer-Based Control of Large Flexible Structures," *Journal of Dynamic Systems, Measurement, and Control*, Vol. 116, Dec. 1994, pp. 713–722.

[4]Sun, J., and Ioannou, P., "Robust Adaptive LQ Control Schemes," *IEEE Transaction on Automatic Control*, Vol. AC-37, No. 1, 1992, pp. 100–106.

[5]Wang, J., "Recurrent Neural Networks for Solving Linear Matrix Equations," *Computers and Mathematics with Applications*, Vol. 26, No. 9, 1993, pp. 23–34.

[6]Leon, S. J., *Linear Algebra with Applications*, Macmillan, New York, 1994, pp. 347–355.

[7]Lin, C. L., "Neural Net-Based Adaptive Linear Quadratic Control," *Proceedings of the 1997 IEEE International Symposium on Intelligent Control*, Istanbul, Turkey, 1997, pp. 187–192.

[8]Czajkowski, E. A., and Preumont, A., "Spillover Stabilization and Decentralized Modal Control of Large Space Structures," AIAA Paper 87-0903, 1987.

Proportional Navigation Through Predictive Control

S. E. Talole*

Institute of Armament Technology, Pune 411 025, India

and

Ravi N. Banavar†

Indian Institute of Technology, Bombay 400 076, India

Introduction

A NEW formulation of the proportional navigation guidance (PNG) law using recently developed continuous-time predictive control approach is proposed. With certain assumptions, this guidance law exhibits similarity with some of the well-known versions of PNG laws. Simulations are carried out to assess the performance of this guidance law in comparison with conventional PNG against maneuvering targets. The results show that the performance of the present formulation is quite superior as compared to that of PNG.

Continuous-Time Predictive Control

The continuous-time predictive control approach offers an alternative strategy for designing controllers for nonlinear systems. In this approach, first introduced by Lu,[1] the state or output response of the nonlinear dynamic system is predicted by appropriate expansions and a quadratic cost function based on the predicted errors in the actual response, and the desired response and current control expenditure is minimized pointwise to obtain an optimal, nonlinear feedback control law. The performance index was modified by Singh et al.[2] for output tracking problems of input–output feedback linearizable systems, wherein the cost function is based on a function consisting of predicted errors, as well as their derivatives and integral. A brief outline of this approach is presented here for the sake of completeness. The reader is referred to Refs. 1 and 2 for more complete development.

Consider a single input/single output nonlinear system

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}) + g(\boldsymbol{x})u, \qquad y = c(\boldsymbol{x}) \tag{1}$$

Received March 27, 1998; revision received July 13, 1998; accepted for publication July 14, 1998.

*Scientist, Faculty of Guided Missiles, Girinagar.

†Associate Professor, Systems and Control Engineering.

assumed to be input–output feedback linearizable,[3] where $\boldsymbol{x} \in R^n$ is the state vector, y and u are the scalar output and input, $f(\boldsymbol{x})$ and $c(\boldsymbol{x})$ are continuously differentiable nonlinear functions, and $g(\boldsymbol{x})$ is a continuous function of $\boldsymbol{x}(t)$.

Suppose the desired output response of the system is given by a reference trajectory $y^*(t)$. The desired trajectory is such that it satisfies the system (1). Let the relative degree of y be γ. Then differentiating y by γ times yields

$$y^{(\gamma)} = a(\boldsymbol{x}) + \mathrm{D}(\boldsymbol{x})u \tag{2}$$

where $y^{(\gamma)}$ is γth derivative of y with respect to time. Now define a function s as

$$s \triangleq e^{(\gamma-1)} + k_{\gamma-1}e^{(\gamma-2)} + \cdots + k_1 e + k_0 x_s, \qquad \dot{x}_s = e \tag{3}$$

where $e(t) \triangleq y(t) - y^*(t)$ is the output tracking error and $k_i > 0$ are the gains to be chosen by the designer such that the polynomial

$$\lambda^\gamma + k_{\gamma-1}\lambda^{(\gamma-1)} + \cdots + k_1\lambda + k_0 \tag{4}$$

is Hurwitz. The function s is a linear combination of the tracking error e, its derivatives, and includes an integral term as well. Differentiating Eq. (3) gives

$$\dot{s} = y^{(\gamma)} - y^{*(\gamma)} + z \tag{5}$$

where $z \triangleq (k_{\gamma-1}e^{(\gamma-1)} + \cdots + k_1\dot{e} + k_0 e)$. To obtain the predictive controller, consider a performance index as

$$J(u) = \tfrac{1}{2}\{qs^2(t+h) + ru^2(t)\} \tag{6}$$

where $q > 0$ and $r \geq 0$ are weightings to be chosen, $h > 0$ is the prediction horizon, and $s(t+h)$ is the predicted value of s at the instant $(t+h)$. The predicted value of $s(t)$ at time $(t+h)$ is obtained by expanding $s(t+h)$ in a first-order Taylor series, and using Eqs. (2), (3), and (5) yields

$$s(t+h) \approx s(t) + h\left(a + z - y^{*(\gamma)} + \mathrm{D}u\right) \tag{7}$$

By applying the necessary condition for minimization of J with respect to u, i.e., $\partial J/\partial u = 0$, a closed-form solution for the control command $u(t)$ is obtained as

$$u = -\frac{h\,\mathrm{D}q}{(r + h^2\mathrm{D}^2 q)}\left[s + h\left(a + z - y^{*(\gamma)}\right)\right] \tag{8}$$

To obtain satisfactory tracking performance, it is necessary to choose proper values for the gains k_i, the weightings q and r, and the prediction horizon h. In Ref. 2, it has been shown that controller (8) with $r = 0$ achieves asymptotic tracking of the desired output trajectory and also offers robustness in presence of parametric uncertainty.

Formulation of Guidance Law

Consider a two-dimensional engagement geometry as shown in Fig. 1, where the missile and target are treated as point masses.

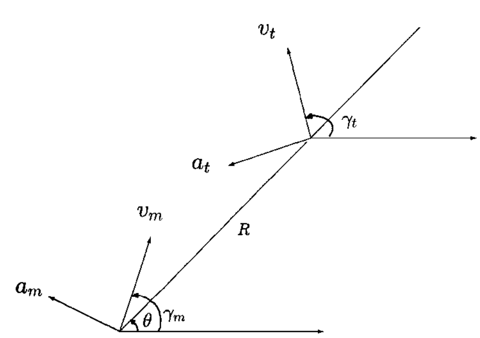

Fig. 1 Two-dimensional engagement geometry.