Receding Time Horizon Linear Quadratic Optimal Control for Multi-Axis Contour Tracking Motion Control[1]

Robert J. McNab
Western Digital Cooperation, San Jose, CA 95138

Tsu-Chin Tsao
Mechanical and Aerospace Engineering Department, University of California-Los Angeles, Los Angeles, CA 90095-1597

A receding time horizon linear quadratic optimal control approach is formulated for multi-axis contour tracking problem. The approach employs a performance index with fixed weights on quadratic contouring error, tracking error, and control input over a future finite horizon. The problem is then cast into a standard receding horizon LQ problem with time varying weighting matrices, which are functions of the future contour trajectory within the horizon. The formulation thus leads to a solution of time varying state feedback and finite preview gains. Stability is proven for the linear trajectory case. Experimental and simulated results for an $X-Y$ motion control problem are presented, which demonstrate the effectiveness of the control scheme and the effects of the key controller design parameters. [S0022-0434(00)01202-8]

1 Introduction

Contour tracking is an important motion control objective in such applications as multiaxis free form machining/inspection and vehicle lateral control. Contour error is the minimum distance between an output point and the desired output path. This is in contrast to tracking error which is the distance from an output point to a particular desired output point. A traditional tracking controller attempting to make an output follow some desired path would use tracking error to drive the controller. A contour tracking controller uses a combination of the tracking error and the contour error, with a stronger emphasis on the latter.

Contour errors become more significant when the dynamics or disturbances of the axes are different [1]. Traditional controllers have attempted to reduce contour error by improving tracking performance on each axis independently or by trying to match the dynamics of each axis. Cross-coupling controller [2] was developed to deal with the problem more directly. A cross-coupling controller calculates the contour error from axis error measurement and attempts to compensate for this error by adding a cross coupled feedback to the single loop axis controllers.

Kulkarni and Srinivasan [3,4] introduced an optimal cross coupling controller that used a linear quadratic regulator (LQR) scheme to correct the contour error with a proportional controller being used for tracking. A weighting matrix was placed on the contour error and the states were augmented so that the penalty function had the standard form of quadratic weighting on the augmented states. The LQR problem was solved with an algebraic Riccati equation. This solution is only optimal if the weighting matrices remain time invariant which is only the case for a straight line contour. The solution is presented in an analytical form, which requires a second-order model for each axis.

Koren and Lo [5] developed a variable gain cross-coupling controller in which the feedback gains in the contour error controller varied with the contour. A nominal proportional controller corrected tracking errors and an additional PID-controller corrected contour errors. For a particular type of contour such as linear, circular, or parabolic, an appropriate transformation from tracking error to contour error exists. This nonlinear translation varies along the contour, amounting to varying feedback gain on the tracking errors. Another approach is to perform coordinate transformation from the machine axes to the trajectory (task) coordinate and design the controllers on the directions normal and tangential to the contour [6–9].

The control approaches mentioned above are effective in reducing the contour error, however none explicitly use preview and feedback information together for motion synchronization. Tomizuka and Whitney [10] presented the finite preview LQ tracking controller, where the solution is optimal in the sense of infinite time horizon and the missing information beyond the preview window is considered as random. Deterministic extension over the signal in the preview window was also considered [11,12]. Tsao [13] considered optimal feedforward digital tracking control design as a model matching problem, where the solution is optimal for the signal induced norms. The preview length and practical constraints like maximum velocity can be conveniently included in the design by properly assigning the reference model and weighting functions. These existing optimal tracking controllers are however time invariant and do not account for time varying contour tracking.

In order to account for both contour tracking and finite preview in the same problem framework, McNab and Tsao [14] formulated the contour tracking as a receding horizon LQ problem, which uses finite preview and allows time varying weighting matrices, and presented preliminary simulation results. The selection of the weightings on tracking error, contour error, and control effort determines the controller performance and robustness. The weighting matrices in this suboptimal control problem are time varying, and correspondingly the controller gain matrices are time varying. A new feature of this controller is that it combines contour tracking and preview action in a single framework. Unlike some other contour control schemes, the proposed scheme does not need to be reformulated for different contours nor is a closed form equation describing the contour required. An arbitrary contour can be used if a number of future desired output points are given to calculate both the feedback and previewing feedforward controller gains. This paper presents the formulation, stability analysis for a limited cases, simulation, and experimental results of the above approach.

The rest of this paper is organized as follows. Section 2 formulates the contour tracking control as a receding horizon LQ problem with variable state weighting matrices. Section 3 discusses the stability conditions for the proposed scheme. Section 4 gives the simulation and experimental results and discussions on a $X-Y$ motion platform followed by conclusion in Section 5.

2 Receding Horizon LQ Formulation

Typical optimal control for tracking problems use a performance index that penalizes the tracking error and the control effort. For contour tracking control, a weight on contour error will be added. Figure 1 shows the geometric relationship between the tracking error, the contour error, and the estimated contour error. The tracking error, $e(k)$, is the vector from the actual position, $y(k)$, to the desired position, $y_d(k)$. The true contour error, $\varepsilon_{\mathrm{true}}(k)$, is the vector from the actual position to the point on the contour nearest the actual position. In other words, contour error is the distance from the output point to the desired path, while tracking error is the distance from the output point to a particular point on the path. The contour error may be small while the track-

[1]This work was supported in part by National Science Foundation under Grant DMI 9522815.

Contributed by the Dynamic Systems and Control Division for publication in the JOURNAL OF DYNAMIC SYSTEMS, MEASUREMENT, AND CONTROL. Manuscript received by the Dynamic Systems and Control Division December 15, 1998. Associate Technical Editor: T. Kurfess.

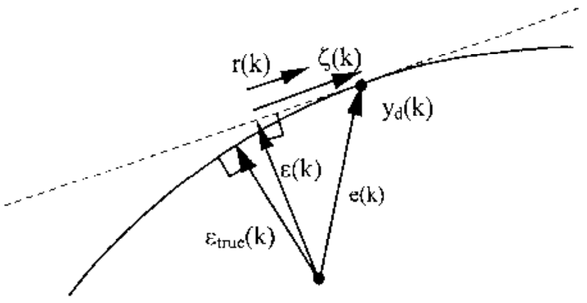

Fig. 1 Definition of tracking error and contour error

ing error is quite large, although if the tracking error is small the contour error will also be small. If the goal of the control is to reduce the contour error, this can be accomplished without insisting on a small tracking error.

The actual contour error is in general a nonlinear function of the tracking error and the desired contour, which may be inefficient to calculate in real time. A good approximation may be obtained by using a projection of the tracking error in the direction perpendicular to the contour. In Fig. 1, $r(k)$ is the unit vector tangent to the contour at the desired position. The projection of the tracking error, $e(k)$, in the direction of $r(k)$ is given by

$$\zeta(k)=(e(k)\cdot r(k))r(k). \tag{1}$$

The contour error is approximated as the difference between the tracking error and $\zeta(k)$,

$$\varepsilon(k)=e(k)-(e(k)\cdot r(k))r(k), \tag{2}$$

reformulating gives

$$\begin{aligned}\varepsilon(k)&=W(k)e(k),\\ W(k)&=I-r(k)r^T(k).\end{aligned} \tag{3}$$

Note that $W^T(k)=W(k)$ and $W^2(k)=W(k)$.

A linear quadratic penalty function that penalizes contour error, tracking error, and control effort, over the length of a preview window N_p, is given by,

$$J(k)=e^T(N_p)S(k)e(N_p)+\sum_{j=k}^{j=k+N_p-1}[e^T(k)Q_t(k)e(k)$$
$$+\varepsilon^T(k)Q_c(k)\varepsilon(k)+u^T(k)R(k)u(k)] \tag{4}$$

Substituting for $\varepsilon(k)$ yields,

$$J(k)=e^T(N_p)S(k)e(N_p)+\sum_{j=k}^{j=k+N_p-1}[e^T(k)[Q_t(k)$$
$$+W^T(k)Q_c(k)W(k)]e(k)+u^T(k)Ru(k)], \tag{5}$$

or,

$$J(k)=e^T(N_p)S(k)e(N_p)+\sum_{j=k}^{j=k+N_p-1}[e^T(k)Q(k)e(k)$$
$$+u^T(k)Ru(k)]. \tag{6}$$

To explicitly weight tracking and contour error, consider

$$J(k)=e^T(N_p)S(k)e(N_p)+\sum_{j=k}^{j=k+N_p-1}[\alpha\|e(k)\|^2+\beta\|\varepsilon(k)\|^2$$
$$+\rho\|u(k)\|^2], \tag{7}$$

then

$$Q(k)=[\alpha I+\beta W(k)],\quad 0\leqslant\alpha,\beta. \tag{8}$$

To improve contour tracking performance, contour error information has been combined with tracking error information into a linear quadratic penalty function with a time varying error weighting matrix. Several optimal control solutions to linear quadratic penalty functions are presented and discussed below.

Consider a system to be controlled given by its state space representation:

$$\begin{aligned}x(k+1)&=Ax(k)+Bu(k),\\ y(k)&=Cx(k).\end{aligned} \tag{9}$$

where $y(k)$ is a vector of the state vector $x(k)$ for which there is a desired trajectory $y_d(k)$.

The basic LQ tracking problem is the finite horizon LQ tracking problem by Anderson and Moore [15]. The penalty function,

$$J=e^T(N)Se(N)+\sum_{i=0}^{i=N-1}[e^T(i)Q(i)e(i)+u^T(i)R(i)u(i)], \tag{10}$$

is minimized over the entire time of the trajectory and its solution involves the Riccati difference equation (RDE). The controller requires information about the desired output and weighting matrices over the entire time of the problem. Often this is not available and this approach cannot be applied. The finite preview LQ tracking problem by Tomizuka and Whitney [10] is an approach allowing optimal tracking without a priori knowledge of the entire trajectory. This formulation combines finite preview information with the solution of the infinite horizon LQ regulation problem, using the algebraic Riccati equation (ARE). The infinite horizon problem is solved by assuming stochastic information beyond the preview horizon.

The receding horizon LQ tracking problem in Bitmead et al. [16] uses the same equations as the finite horizon problem except that at each time step k, the index j in Eq. (7) is set to zero and a finite time horizon problem is solved over the preview length. This finite time horizon problem uses a final time, N_p, that is much smaller than the entire time of the problem, N. This means knowledge about the desired output and the weighting matrices is only needed for the time steps included in the preview, but a new finite time horizon problem must be solved at each time step. The resulting solution for each of the finite time optimal control at the time step k is obtained by evaluating the following difference equations (Eqs. (12) and (14)) backward from the final time at $k+Np$ with the given final conditions to the current initial time at k:

$$u^{\mathrm{opt}}(k)=-[B^TH_1(k)B+R_1(k)]^{-1}B^T[H_1(k)Ax(k)+g_1(k)], \tag{11}$$

$$H_j(k)=A^T\{H_{j+1}(k)-H_{j+1}(k)B[B^TH_{j+1}(k)B$$
$$+R_j(k)]^{-1}B^TH_{j+1}(k)\}A+C^TQ_j(k)C, \tag{12}$$
$$H_{Np}(k)=C^TS(k)C,$$

$$g_j(k)=\{A-B[B^TH_{j+1}(k)B+R_j(k)]^{-1}B^TH_{j+1}(k)A\}^Tg_{j+1}(k)$$
$$-C^TQ_j(k)y_d(k+j), \tag{13}$$
$$g_{Np}(k)=-C^TS(k)y_d(k+N_p),$$

where

$$Q_j(k)=Q(j+k), \tag{14}$$

$$R_j(k)=R(j+k). \tag{15}$$

This approach is suited for the contour tracking problem in that the contour error is absorbed into the time varying state weighting matrix. In contrast, the finite preview controller [10] has fixed controller gain and hence is unable to incorporate contour tracking.

3 Stability Analysis

Consider first a standard LQ control problem. The algebraic Riccati equation (ARE) associated with an infinite horizon LQ control problem is:

$$H=A^T\{H-HB[B^THB+R]^{-1}B^TH\}A+C^TQC \tag{16}$$

If $[A,B]$ is stabilizable, $[A,Q^{1/2}C]$ is detectable, $Q\geq 0$, and $R>0$, then there exists a unique, non-negative definite symmetric stabilizing solution H_s, i.e.,

$$A-B[B^TH_sB+R]^{-1}B^TH_s\}A \tag{17}$$

has its eigenvalues all strictly within the unit circle.

To determine the stability of the receding horizon controller, the solution (i.e., H_1) of the Riccati difference equation (RDE) in Eq. (12) must be stabilizing. Equation (12) can be reformulated in the form of the ARE for determining the stability. Define $P_j=C^TQ_jC-(H_j-H_{j+1})$ then write the RDE as

$$H_{j+1}=A^T\{H_{j+1}-H_{j+1}[B^TH_{j+1}B+R]^{-1}B^TH_{j+1}\}A+P_j. \tag{18}$$

The ARE stability criteria can be applied to the above equation, known as the fake algebraic Riccati equation (FARE) [17,18]. In other words, for the receding horizon problem, if $[A,B]$ is stabilizable, $[A,P_0^{1/2}]$ is detectable, $P_0^{1/2}\geq 0$, and $R>0$, then H_1 is stabilizing.

When the desired output trajectory is a straight line, the weighting matrix Q is time invariant. The stability for the time invariant receding horizon LQ controller has been a long standing problem attracting much research, e.g., Bitmead et al. [19,16]. One approach is to have a zero terminal state condition in the penalty function. That is equivalent to having an infinite weight for the selection of S. The drawback to this approach is a significant loss in performance due to overly emphasizing the terminal state.

A useful property of RDE for invariant weighting matrices R and Q is the monotonicity of H_j [19,20]. That is, if the nonnegative definite solution H_j of the RDE is monotonically nonincreasing (nondecreasing), i.e., $H_p\leq H_{p+1}$ ($H_p\geq H_{p+1}$), for some p, then H_p is monotonically nonincreasing (nondecreasing) for all previous times: $H_{p-m}\leq H_{p-m+1}$ ($H_{p-m}\geq H_{p-m+1}$), for all $m\geq 0$. In view of the FARE in Eq. (18), the monotonicity property suggests that if the solution of an RDE is stabilizing for some time horizon "p," then the solution for any longer horizon length with the same weighting matrices and final conditions is also stabilizing.

Utilizing the above FARE and the monotonicity property, we seek to find conditions that the resulting control will be stabilizing for any direction of straight lines with any horizon length. The rationale is to find weighting matrix that is larger than the weighting matrix corresponding to any particular angle. The ARE solution corresponding to that weighting matrix is then used as the final value for the receding horizon RDE solution. To find $Q_{\max}$ such that $Q_{\max}\geq Q(\alpha,\beta,\theta)$ for the two-axis case:

$$Q(\alpha,\beta,\theta)=\alpha I+\beta W,\quad \text{where } W=\begin{bmatrix}\cos^2\theta & \cos\theta\sin\theta\\ \cos\theta\sin\theta & \sin^2\theta\end{bmatrix}, \tag{19}$$

since $I\geq W$, $Q_{\max}$ may be selected as $(\alpha+\beta)I$.

Theorem 1. If

$[A,B]$ is stabilizable,

$[A,Q(\alpha,\beta,\theta)^{1/2}C]$ is detectable for any angle θ,

$R>0, Q(\alpha,\beta,\theta)\geq 0$,

$H_{Np}=H_S(Q_{\max})$, the ARE solution for A, B, $Q_{\max}$, and R, where $Q_{\max}\geq Q(\alpha,\beta,\theta)$ for all θ,

then, the closed-loop eigenvalues for tracking a linear trajectory of any fixed angle θ and any N_p are within the unit circle.

Proof. From the final condition above, H_{Np} is the solution of

$$H_{Np}=A^T\{H_{Np}-H_{Np}B[B^TH_{Np}B+R]^{-1}B^TH_{Np}\}A+C^TQ_{\max}C. \tag{20}$$

The first step of the Riccati difference equation is given by

$$H_{Np-1}=A^T\{H_{Np}-H_{Np}B[B^TH_{Np}B+R]^{-1}B^TH_{Np}\}A+C^TQ_{Np-1}(\alpha,\beta,\theta)C. \tag{21}$$

Simple substitution produces

$$H_{Np}-H_{Np-1}=C^TQ_{\max}C-C^TQ_{Np}(\alpha,\beta,\theta)C\geq 0. \tag{22}$$

The monotonicity of H_i is thus established for all $i\geq 0$. Furthermore the detectability of $[A,Q(\alpha,\beta,\theta)^{1/2}C]$ and $P_0\geq Q(\alpha,\beta,\theta)^{1/2}C$ implies that $[A,P_0]$ is detectable [16]. Therefore, H_1 is stabilizing in view of the FARE in Eq. (18), where $P_0\geq 0$. That is, the eigenvalues of

$$\bar{A}=A-B[B^TH_1B+R]^{-1}B^TH_1A \tag{23}$$

are all strictly within the unit circle. #

The original penalty function uses a final weight on the tracking error, $e(k+N_p)^TSe(k+N_p)$. However, using $H_{Np}=H_s(Q_{\max})$ corresponds with a final weight on a state error, $(x_d(k+N_p)-x(k+N_p))^TH_s(Q_{\max})(x_d(k+N_p)-x(k+N_p))$. $x_d(k+N_p)$ is not assumed to be known, and in general it is not available. This is not a problem in terms of feedback gains or stability, but in the original formulation, the calculation for the feedforward terms begins with the terms $C^TSy_d(k+N_p)$ replaced by $H_sx_d(k+N_p)$. Although $x_d(k+N_p)$ is unknown, it can be estimated by a state observer described in the Appendix.

The above stability condition is for any linear trajectory and preview length. For a general nonlinear trajectory, the frozen-time stability is conjectured by viewing the solution of the RDE as a form of interpolation of all the stabilizing solutions at different angle and preview length. In addition to affecting stability, the selection of H_{Np} also has a strong effect on the performance. Improved performance may be obtained with selections for H_{Np} that do not meet the sufficient stability condition in Theorem 1. This does not mean that the system is unstable since the condition of H_{Np} is not a necessary one for stability. One such selection is $H_{Np}=C^TQ_{Np}C$. This has been the choice for some GPC controller designs. Some of these designs have been shown to have stability depend on N_p. As N_p increases, H_1 approaches H_s and (H_1-H_0) approaches 0. As a result, P_0 approaches C^TQ_0C, and the system is thus stable.

4 Results on an X–Y Motion System

The theoretical developments in the previous sections are general and can be applied to any system where it is important to stay on a path. An application for these developments is a two-axis machining process. This is the application for which most of the previous developments in cross-coupling contour control were developed.

The experimental system used to obtain these results is a Bridgeport Discovery 300 vertical machining center. The position feedback sensors are encoders with resolutions of 0.8 μm/count. Control is implemented on a Spectrum digital signal processor board utilizing a Texas Instruments TMS 320C30. Due to the high sampling rate, 2 kHz, and the high model order, 8th, real time calculation of control gains is not possible for this application. All the gains are calculated before the trajectory starts. This is only possible because the entire trajectory is known. The nominal Bridgeport controller is broken at the position loop. Feed drive models for each axis are made by collecting frequency response data from velocity command to velocity measurement signals from tachometers. With the addition of an integrator to obtain position output, the resulting curve-fitted transfer functions produce the following state space matrices:

$$A=\begin{bmatrix} -0.2916 & 0.2755 & -0.8029 & 0 & 0 & 0 & 0 & 0 \\ -0.8890 & 1.6565 & -0.8540 & 0 & 0 & 0 & 0 & 0 \\ 0.0000 & 1.0343 & 0.3324 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5109 & 0.5102 & 1.1644 & 0.9037 & -0.9856 \\ 0 & 0 & 0 & -0.7566 & 0.7378 & 0.7638 & 0.9284 & -1.5212 \\ 0 & 0 & 0 & 0.0000 & 0.0344 & 0.0227 & 0.9665 & -0.2304 \\ 0 & 0 & 0 & 0.0000 & 0.0000 & 1.6801 & 2.1797 & 4.2149 \\ 0 & 0 & 0 & 0.0000 & 0.0000 & 0.0000 & 0.9843 & -0.6750 \end{bmatrix}$$

$$B=\begin{bmatrix} -0.0379 & 0 \\ 0.5879 & 0 \\ 0.0402 & 0 \\ 0 & 0.2101 \\ 0 & 0.2655 \\ 0 & 0.1732 \\ 0 & 0.6884 \\ 0 & 0.0332 \end{bmatrix}\times 10^{-5}, \quad C=\begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

The contour that is examined is a circular trajectory with a 1 mm radius that is traversed with a feed rate of 3 m/min. The circle is tracked at 8 Hz. The majority of most machining paths are smooth with large radii of curvature. This path is used as an example of a difficult contour that demonstrates a situation where contour control becomes important. The sampling rate is 2 kHz.

The parameters that are examined are α—the tracking error weight, β—the contour error weight, and N_p—the preview window length. In addition, two choices for the final error weighting matrix are examined, $H_{Np}=C^TQ_{Np}C$ and $H_{Np}=H_s(Q_{\max})$. The former penalizes the final errors in the same way that all the errors in the preview window are penalized. The latter weighs the final error more heavily in order to obtain better stability properties.

Another related controller is examined for comparison. The finite preview controller [10] is a receding horizon approach in which the weighting matrix must be time invariant and the final condition on the RDE solution is the ARE solution. This controller is similar to the one presented in this paper because it has a preview window and uses an LQ optimal approach, however it is different in that it cannot account for contour error. In fact, this controller is the same as the contour tracking controller presented in this paper for the particular case of $\beta=0$, and $H_{Np}=H_s(Q_{\max})$. In this case, the weighing matrix Q is fixed, i.e., $Q_{Np}=Q_{\max}$. Both controllers use state feedback, and so full state observers are needed for implementation.

Figure 2 shows several data points along the circular contour. As weight on the contour is added, the output moves toward the desired trajectory. Note that although this is a small radius (1 mm) curve, because of the spacing of the reference points, the linear approximation for the contour error direction is quite accurate.

Figures 3–8 are bar graphs showing tracking and contour error with $H_{Np}=C^TQ_{Np}C$ or $H_{Np}=H_s(Q_{\max})$ for different selections of

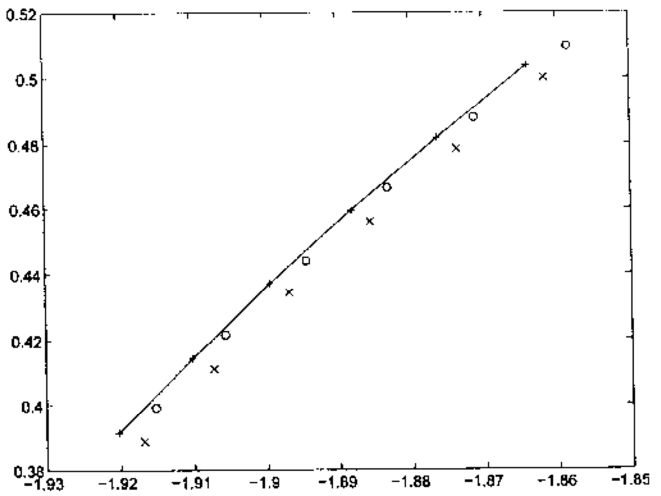

Fig. 2 Points along a circular contour, path is toward lower left

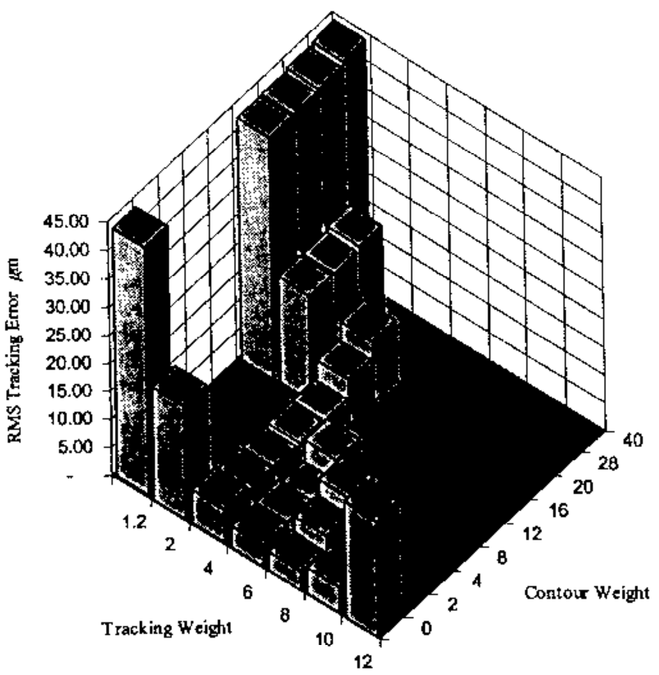

Fig. 3 Experimental RMS tracking error $H_s(Q_{Np})$

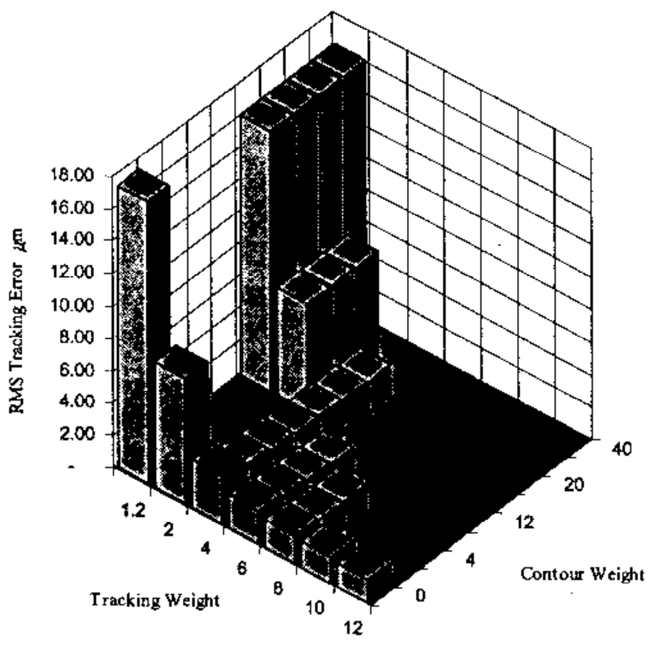

Fig. 4 Simulated RMS tracking error $H_s(Q_{Np})$

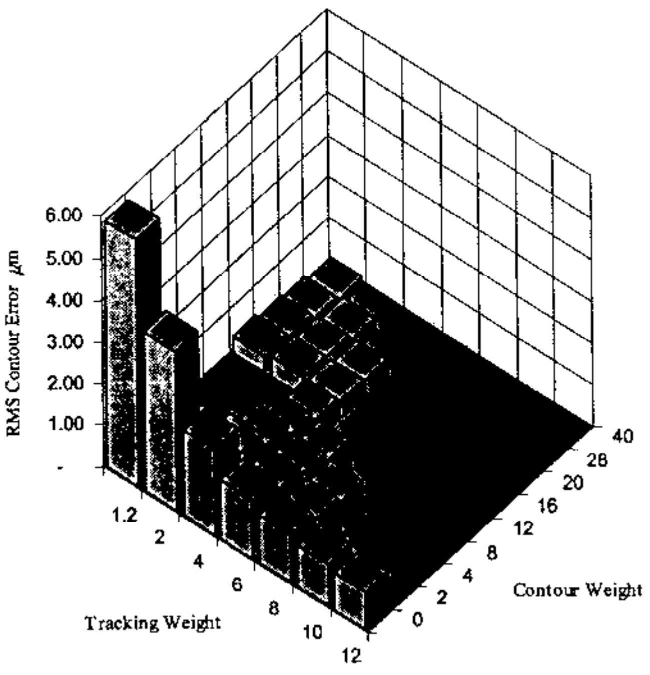

Fig. 6 Simulated RMS contour error $H_s(Q_{Np})$

tracking weight and contour weight. The preview length is 10 steps and the control effort weighting is $2e$-9. Both experimental results and simulation results are shown. The errors shown are for RMS error over one cycle of the circular trajectory and the unit is in microns.

Generally speaking, increasing the tracking weight decreases both the tracking error (Fig. 3), and the contour error (Fig. 5). Increasing the contour weight had little effect on the tracking error, but decreased the contour error.

The main difference between simulation and experimental results is that in simulation larger weights result in smaller errors as seen in Figs. 4 and 6. However, experimentally the error decreases at first and then starts to increase again when the weights become too large, Figs. 3 and 5. A larger weight results in large feedback gains and if the feedback gain is too high then unmodeled or mismodeled dynamics begin to disrupt the performance of a controller. So there is a limit to how much the errors can be penalized. The advantage of the contour tracking controller is that some of the weighting can be shifted from tracking error to contour error. This reduces the contour error at the expense of increasing tracking error.

Figure 9 shows the effect of changing the preview window when there is only tracking error weighting and Fig. 10 shows the effect when there is contour weighting. The tracking error decreases with increasing preview length up to nine preview steps

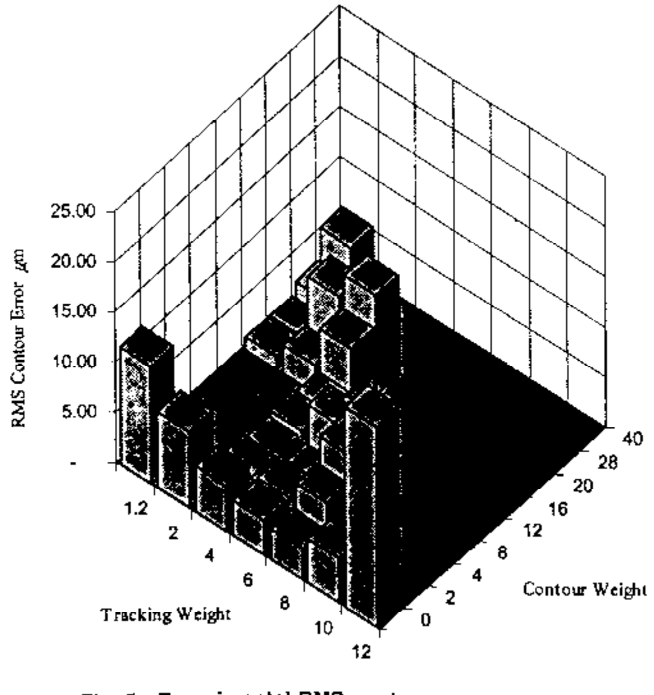

Fig. 5 Experimental RMS contour error $H_s(Q_{Np})$

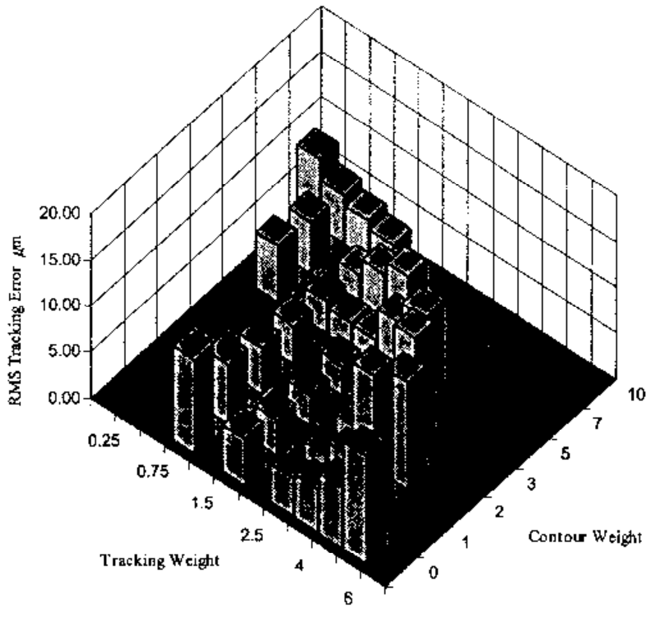

Fig. 7 Experimental RMS tracking error for $H_s(Q_{\max})$

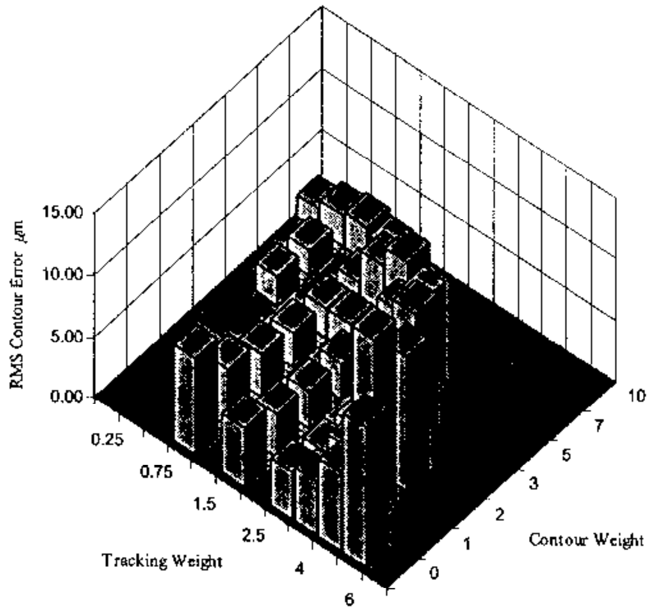

Fig. 8 Experimental RMS contour error for $H_s(Q_{\max})$

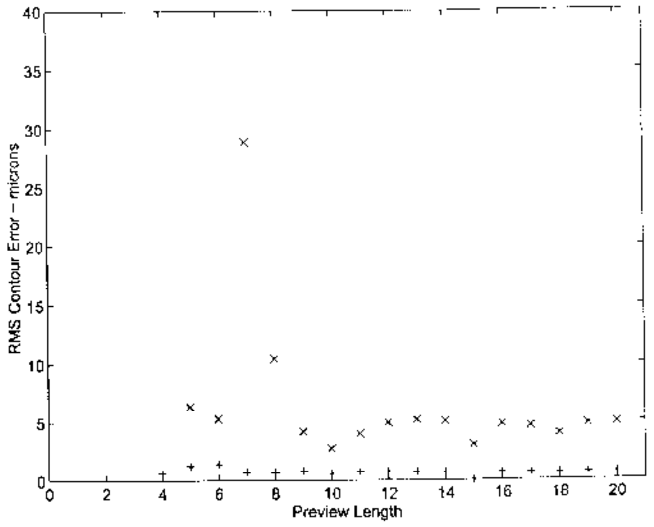

Fig. 10 Mean contour error squared versus preview length, $\alpha=0.25$, $\beta=15$ (simulated, +, and experimental, ×)

Table 1 Tracking and contouring errors for five different sets of parameters

Case	H_{Np}	α	β	RMS te μm	RMS ce μm
I	$C^TQ(N_p)C$	8.0	0.0	3.35	3.07
II	$C^TQ(N_p)C$	1.2	20.0	43.07	1.07
III	$H_s(Q_{\max})$	3.0	0.0	4.10	3.78
IV	$H_s(Q_{\max})$	0.25	15.0	6.55	1.70
Finite Preview	$H_s(Q_{Np})$	3.0	0.0	4.10	3.78

and then remains constant. The contour error is roughly the constant for most of the preview lengths. This may be somewhat dependent on the choice of error weights.

Figures 7 and 8 show tracking error and contour error with the more conservative final error weight of $H_{N_p}=H_s(Q_{\max})$. The trends are the same as for when $H_{N_p}=C^TQ_{N_p}C$, but the performance is not quite as good. Table 1 summarizes the best results obtained with the two different final error weighting schemes. Performance is slightly improved by using a final condition that has no stability assurances. Regardless of which final weighting matrix is used, the use of a penalty against contour errors has the desired result of reducing them. To obtain best performance in terms of contour error, the penalty against the tracking error must be reduced, increasing the tracking error.

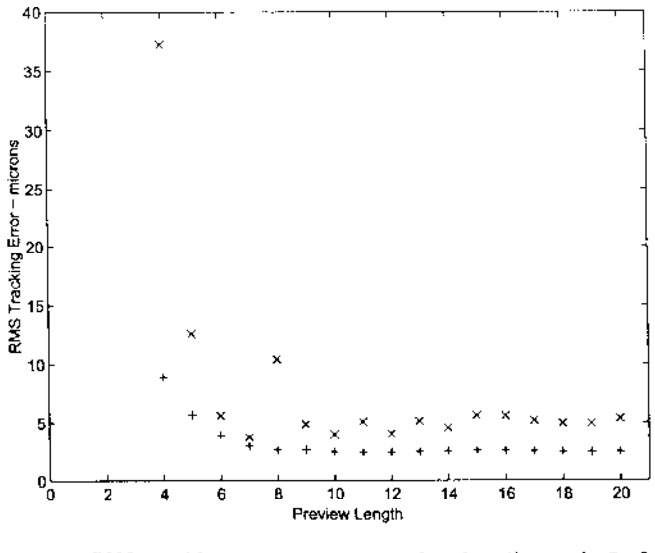

Fig. 9 RMS tracking error versus preview length, $\alpha=4$, $\beta=0$ (simulated, +, and experimental, ×)

5 Conclusion

The receding horizon LQ controller has been applied to improve contouring accuracy by formulating a penalty function that includes contour error, tracking error, and control effort over a finite future time window. Simulation and experimental results show that this controller improves contour accuracy at the expense of tracking error as intended. The design parameters needed to adjust performance are limited to weights on control effort, tracking error, and contour error, preview length, and choice of final error weighting. The design is effective for arbitrary contours.

Two limitations of this scheme are the computational complexity and the lack of stability proof for general nonlinear contour. At each time step, a number of recursive matrix computation is required over the preview length. This should not pose a significant limitation with the increasing real-time digital processing speed. The second suggests that common stability guidelines used in receding horizon optimal, such as the relation between the preview length and the system's rise time, may be applicable to the present problem. While the proposed approach has been tested to render stable response for various nonlinear trajectories in simulation and experiment, whether the stability conditions in Theorem 1 for any linear trajectory can be extended and proven for nonlinear trajectories remain unsolved.

Appendix

An estimate for the desired state vector can be obtained if the desired trajectory is assumed to continue with constant velocity beyond the preview window. First define

$$\overline{\Delta y_d}=y_d(k+N_p)-y_d(k+N_p-1), \text{ and so}$$

$$y_d(k+N_p+j+1)-y_d(k+N_p+j)=\overline{\Delta y_d}, \quad \text{for } j \geqslant 0.$$

Because the system has a single free integrator, a constant velocity is generated by a constant control signal, $\bar{u}$. It is computationally simpler to consider one axis at a time. Consider a third order system, although the procedure is similar for higher order systems. From the system equations, one can obtain;

$$y_d(k+N_p)=Cx_d(k+N_p)$$

$$y_d(k+N_p+1)=C(Ax_d(k+N_p)+B\bar{u})$$

$$y_d(k+N_p+2)=CA^2x_d(k)+(CAB+CB)\bar{u}$$

$$y_d(k+N_p+3)=CA^3x_d(k)+(CA^2B+CAB+CB)\bar{u}$$

which can be written

$$\begin{bmatrix} y_d(k+N_p) \\ y_d(k+N_p)+\overline{\Delta y_d} \\ y_d(k+N_p)+2\overline{\Delta y_d} \\ y_d(k+N_p)+3\overline{\Delta y_d} \end{bmatrix}=\begin{bmatrix} C & 0 \\ CA & CB \\ CA^2 & C(A+I)B \\ CA^3 & C(A^2+A+I)B \end{bmatrix}\begin{bmatrix} x_d(k+N_p) \\ \bar{u} \end{bmatrix}.$$

If the matrix is nonsingular, then a solution can be obtained for x_d. This is likely if the system is observable because the lower left portion will have full rank and the upper right element is 0, but this must be checked for any particular system.

References

[1] Poo, N., and Bollinger, J. G., 1972, "Dynamic Errors in Type 1 Contouring Systems," IEEE Trans. Ind. Appl., **IA-8**, No. 4, pp. 477–484, July.
[2] Koren, Y., 1980, "Cross-Coupled Biaxial Computer Control for Manufacturing Systems," ASME J. Dyn. Syst., Meas., Control, **102**, pp. 265–272.
[3] Kulkarni, P. K., and Srinivasan, K., 1989, "Optimal Contouring Control of Multi-Axial Feed Drive Servomechanisms," ASME J. Eng. Ind., **111**, pp. 140–148.
[4] Srinivasan, K., and Kulkarni, P. K., 1990, "Cross-Coupled Control of Biaxial Feed Drive Servomechanisms," ASME J. Dyn. Syst., Meas., Control, **112**, pp. 225–232.
[5] Koren, Y., and Lo, C. C., 1991, "Variable-Gain Cross-Coupling Controller for Contouring," Ann. CIRP, **40**, pp. 371–374.
[6] Chiu, G. T-C., and Tomizuka, M., 1995, "Contouring Control of Machine Tool Feed Drive Systems: a Task Coordinate Frame Approach," Dynamic System and Control, the *Proceedings of the 1995 ASME International Mechanical Engineering Congress and Exposition*, San Francisco, CA, DSC Vol. 57-2, pp. 503–510.
[7] Chiu, T-C., and Tomizuka, M., 1998, "Coordinated Position Control of Multi-Axis Mechanical Systems," ASME J. Dyn. Syst., Meas., Control, **110**, pp. 389–393.
[8] Yen, J.-Y., Ho, H.-C., and Lu, S.-S., 1998, "Decoupled Path-Following ontrol Algorithm Based Upon the Decomposed Trajectory Error," *Proceedings of the 37th IEEE -Conference on Decision and Control*, pp. 3189–3194, Tampa, FL.
[9] Lo, C.-C., and Chung, C. Y., 1999, "Tangential-Contouring Controller for Biaxial Motion Control," ASME J. Dyn. Syst., Meas., Control, **121**, pp. 126–129.
[10] Tomizuka, M., and Whitney, D. E., 1975, "Optimal Discrete Finite Preview Problems (Why and How Is Future Information Important?)," ASME J. Dyn. Syst., Meas., Control, **97**, pp. 319–325.
[11] Tomizuka, M., and Rothenthal, D. E., 1979, "On the Optimal Digital State Vector Feedback Controller with Integral And Preview Actions," ASME J. Dyn. Syst., Meas., Control, **101**, No. 2, pp. 172–178
[12] Tomizuka, M., Dornfeld, D., Bian, X.-Q., and Cai, H.-C., 1980, "Application of Microcomputers to Automated Weld Quality Control," ASME J. Dyn. Syst., Meas., Control, **102**, No. 2, pp. 62–68.
[13] Tsao, T.-C., 1994, "Optimal Feed-Forward Digital Tracking Controller Design," ASME J. Dyn. Syst., Meas., Control, **116**, pp. 583–592.
[14] McNab, R. J., and Tsao, T.-C., 1994, "Multi-Axis Contour Tracking: A Receding Horizon Linear Quadratic Optimal Control Approach," *ASME International Mechanical Engineering Congress and Exposition*, Chicago, IL, DSC-Vol. 55-2, pp. 895–902.
[15] Anderson, B. D. O., and Moore, J. B., 1971, *Linear Optimal Control*, Prentice Hall, Englewood Cliffs, NJ.
[16] Bitmead, R. R., Gevers, M., and Wertz, V., 1990, *Adaptive Optimal Control: The Thinking Man's GPC*, Prentice Hall, Englewood Cliffs, NJ.
[17] Poubelle, M. A., Petersen, I. R., Gevers, M., and Bitmead, R. R., 1986, "A Miscellany of Results on an Equation of Count J. F. Riccati," IEEE Trans. Autom. Control., **AC-31**, pp. 651–654.
[18] Poubelle, M. A., Bitmead, R. R., and Gevers, M., 1988, "Fake Algebraic Riccati Techniques and Stability," IEEE Trans. Autom. Control., **AC-33**, pp. 379–381.
[19] Bitmead, R. R., Gevers, M., Petersen, I. R., and Kaye, R. J., 1985, "Monotonicity and Stabilizability Properties of Solutions of the Riccati Difference Equation: Propositions, Lemmas, Theorems, Fallacious Conjectures and Counterexamples," Syst. Control Lett., **5**, pp. 309–315.
[20] de Souza, C. E., 1989, "Monotonicity and Stabilizability Results for the Solution of the Riccati Difference Equation," *Proc. Workshop on the Riccati Equation in Control, Systems and Signals*, S. Bittanti, ed., Como, Italy, pp. 38–41.

Modeling of Flexible Robot-Payload Systems Through Component Synthesis

T. Zhou, J. W. Zu, and A. A. Goldenberg
Department of Mechanical and Industrial Engineering,
5 King's College Road, University of Toronto,
Toronto, Ontario, Canada M5S 3G8

In this paper, a new modeling method is developed for analyzing the dynamic behavior of a system consisting of a rigid robotic manipulator and a flexible sheet metal payload. The component mode synthesis method is applied to reduce the degrees of freedom of the payload and to model the interfaces between the robot gripper and the payload. Using nonlinear compatibility functions, the method is modified to synthesize the dynamics of the entire robot-payload system. Exact models are developed capable of describing both large and small rigid-body motions. A modular form is derived and the coupling dynamics is formulated in a computationally efficient manner. Numerical examples are presented to demonstrate the effectiveness of the modeling method.
[S0022-0434(00)01102-3]

1 Introduction

Recently, there has been a growing interest in manipulator systems handling flexible payloads due to their potential applications in industry [1–3]. Most of the current work treats the payload as a flexible beam; whereas in reality, the payload can have an irregular shape and the interface between the manipulator and the payload is complex. A generic system tackled in this paper is shown in Fig. 1, in which the robot and its gripper are considered rigid while the payload exhibits significant flexibility. The gripper can be reconfigured to accommodate the varying sizes and shapes of different payloads.

A major performance concern for such a robot-payload system is the vibration of the payload due to its flexibility and acceleration while in motion. The vibrational behavior is caused by dynamic coupling effects between rigid-body motions and elastic deflections. Generally, the nonlinear coupling dynamics is configuration dependent and computationally expensive (e.g. Cuadrado et al. [4]). A similar system has been modeled in a previous study [5]. The coupling effects were neglected for practicality reasons, and hence the vibration problem could not be considered. However, for the performance and accuracy requirements desired for high-speed flexible systems, the vibration is significant and can no longer be ignored. The coupling effects are vital for vibration concerns and dynamic formulas with both accuracy and simplicity are essential in practice. Yet, the dynamics of the tackled system has not been exactly modeled by any other researchers in a computationally efficient manner.

Thus, the objective of this work is to obtain an *accurate* and

Contributed by the Dynamic Systems and Control Division of THE AMERICAN SOCIETY OF MECHANICAL ENGINEERS. Manuscript received by the Dynamic Systems and Control Division January 12, 1998. Associate Technical Editor: N. Olgac.