MODEL-INVERSE BASED REPETITIVE CONTROL

T. Harte*, J. Hätönen*, D.H. Owens*

†Department of Automatic Control and Systems Engineering
The University of Sheffield
Mappin Street
S1 3JD Sheffield
United Kingdom
Tel: +44(0)114 222 5250
Fax: +44(0)114 222 5661
Email: J.Hatonen@sheffield.ac.uk

Keywords: Repetitive Control, Robotics, Robust Control

Abstract

This paper explores the possibility of using an inverse plant model in the repetitive control framework. When no uncertainty is present in the plant model, it is shown that the algorithm results in monotonic convergence to zero tracking error. Furthermore, the algorithm can tolerate 90 degrees of phase uncertainty in the plant model, demonstrating a good degree of robustness. Simulation results are used to demonstrate the different theoretical findings in the paper.

1 Introduction

Many signals in engineering are periodic, or at least they can be accurately approximated by a periodic signal over a large time interval. This is true, for example, of most signals associated with engines, electrical motors and generators, converters, or machines performing a task over and over again. Hence it is an important control problem to try to track a periodic signal with the output of the plant or try to reject a periodic disturbance acting on a control system.

In order to solve this problem, a relatively new research area called repetitive control has emerged in the control community. The idea is to use information from previous periods to modify the control signal so that the overall system would ‘learn’ to track perfectly a given T-periodic reference signal. The first paper that uses this ideology seems to be [5], where the authors use repetitive control to obtain a desired proton acceleration pattern in a proton synchrotron magnetic power supply.

Since then repetitive control has found its way to several practical applications, including robotics [6], motors [7], rolling processes [4] and rotating mechanisms [3].

However, most of the existing repetitive control algorithms are designed in continuous time, and they either do not give perfect tracking or they require that the original process is positive real. In order to overcome these limitations, this paper proposes a new model-inverse based repetitive algorithm for minimum-phase plants. It is shown in this paper that new algorithm results in fast convergence, and the algorithm is reasonably robust against modelling uncertainties. The algorithm is applied to an industrial-scale conveyor belt system with excellent results, further demonstrating the applicability of the algorithm to industrial control problems.

2 Problem definition

As a starting point in discrete-time Repetitive Control (RC) it is assumed that a state-space model

\[
\begin{align*}
    x(k+1) &= \Phi x(k) + \Gamma u(k) + \alpha \\
    y(k) &= Cx(k)
\end{align*}
\]

of the plant in question exists for \( k \in \ldots, -2, -1, 0, 1, 2, \ldots \).

From now on it is assumed that the plant is both controllable and observable, and that the plant is minimum-phase. Furthermore, a reference signal \( r(k) \) is given, and it is known that \( r(k) = r(k+T) \) for a given \( T \) (in other words the actual shape of \( r(k) \) is not necessarily known). The control design objective is to find a feedback controller that causes the output \( y(k) \) in the system (1) to track the reference signal as accurately as possible. More mathematically, the controller should result in

\[
\lim_{k \to \infty} e(k) = 0
\]

where \( e(k) = r(k) - y(k) \) is tracking error, when it is known that \( r(k) \) is \( T \)-periodic.

As was shown by Francis and Wonham in [2], a necessary condition for asymptotic convergence is that a controller

\[
[Mu](k) = [Ne](k)
\]
where $M$ and $N$ are suitable operators, must have an internal model of the reference signal inside the operator $M$ so that

$$[Mr](k) = 0$$ (4)

Because $r(t)$ is $T$-periodic, its internal model is $M=1-q^{-T}$, where $q^{-1}$ is the standard left-shift operator. This can be seen from the equation

$$[Mr](k) = [(1 - q^{-T})r](k) = r(k) - r(k - T) = 0$$ (5)

This leads naturally to a repetitive algorithm structure

$$u(k) = u(k - T) + [Ne](k)$$ (6)

and if $N$ is a causal LTI filter, the algorithm can be written using the $q^{-1}$-operator formalism as

$$u(k) = q^{-T}u(k) + K(q)e(k)$$ (7)

and the design consists of selecting $K(q)$. The focus of this paper concentrates on the choice

$$K(q) = \beta q^{-T}G^{-1}(q)$$ (8)

where $\beta$ is a “learning gain” and $G(q)=C(qI-\Phi)^{-1}$ is the transfer function of (1). The next section shows that this algorithm results in a fast convergence rate when no model uncertainty is present, and, quite surprisingly, that the algorithm can tolerate a reasonable degree of multiplicative uncertainty.

3 Convergence analysis

3.1 Nominal case

As explained above, the plant model (1) can be written using the $q^{-1}$-operator formalism as

$$y(k) = G(q)u(k)$$ (9)

Consider now the algorithm (7) when $K(q)$ is equal to (8). This results in the model-based inverse algorithm

$$u(k) = q^{-T}(u(k) + \beta G^{-1}(q)e(k))$$ (10)

Note that this algorithm is causal on the positive time-axis due to the fact that that the tracking error $e(t)$ is delayed $T$ time steps. Furthermore, the operator $K(q) = \beta q^{-T}G^{-1}(q)$ is stable along the positive time-axis, because $G(q)$ is assumed to be minimum-phase.

Multiplying (9) from the left with the plant model $G(q)$ results in the error evolution equation

$$e(k) = (1 - \beta)e(k - T)$$ (11)

This equation shows immediately, that if $0 < \beta < 2$, the algorithm results in

$$|e(k)| \leq \lambda|e(k - T)|, \quad 0 < \lambda < 1$$ (12)

demonstrating monotonic convergence between $e(k)$ and $e(k-T)$. This result also automatically implies that

$$\lim_{k \to \infty} e(k) = 0$$ (13)

Furthermore, the fastest convergence rate is obtained with $\beta = 1$, because in the case $e(k) = 0$ for $k>T-1$, i.e. the algorithm “learns” the correct input sequence in one cycle.

In the repetitive control framework, however, it is almost always the case that the plant model $G(q)$ is not known exactly, but modelling uncertainties and nonlinearities result in an uncertain model of the plant. Consequently, the next subsection establishes the degree of uncertainty that the model-inverse based algorithm can tolerate.

3.2 Multiplicative uncertainty

Consider now the case when only a nominal model $G_o(q)$ is available for the algorithm designer, and the true plant model is related to the nominal model through the equation

$$y(k) = G(q)u(k) = G_o(q)U(q)u(k)$$ (14)

where $U(q)$ is the multiplicative uncertainty of the plant model. Applying the algorithm (10) to (14) results in the error evolution equation

$$e(k) = q^{-T}(1 - \beta U(q))e(k)$$ (15)

Restrict the time axis to be positive, i.e. $k = 1,2,\ldots$. In this case (15) results in an autonomous system

$$(1-q^{-T}(1-\beta U(q)))e(k) = 0$$ (16)

with the initial conditions $e(0)=e_0, \ldots, e(T-1)=e_{T-1}$, where the initial conditions are dependent on the initial guess $u(0), \ldots, u(T-1)$. According to the Nyquist stability test (see [1]), the poles of the system (16) are inside the unit circle if the locus of

$$z^{-N}(1 - \beta U(z)) \bigg|_{|z|=1}$$ (17)
encircles the critical point (-1,0) \(N\) times, where \(N\) is the number of unstable poles of \(z^{-N}(1-\beta U(z))\). Furthermore, if the poles are inside the unit circle, this guarantees that

\[
e(k) = M\alpha^k, \ M > 0, 0 < \alpha < 1 \tag{18}
\]

and in particular, \(\lim_{k \to \infty} e(k) = 0\). Assume now that \(U(q)\) is stable. In this case \(z^{-N}(1-\beta U(z))\) does not have any unstable poles, and therefore \(N=0\), and therefore (16) is not allowed to encircle the critical point (-1,0). A sufficient condition for this is that

\[
\sup_{\omega \in [0,2\pi]} \left| z^{-N}(1-\beta U(e^{j\omega})) \right| < 1
\]

(19)

In summary, if the uncertainty \(U(q)\) is stable, and it satisfies (19) for a given \(\beta\), the algorithm will converge exponentially to zero tracking error. However, condition (19) does not reveal any useful information \(U(q)\) in terms of the true plant model \(G(q)\). Note however, that

\[
(1-\beta U(e^{j\omega}))^2 = (1-\beta U(e^{j\omega}))^* (1-\beta U(e^{j\omega}))
\]

\[
= 1-\beta U(e^{j\omega}) - \beta U^*(e^{j\omega}) + \beta^2 |U(e^{j\omega})|^2
\]

(20)

which shows that for (19) to hold it is required that

\[
\text{Re}\{U(e^{j\omega})\} > 0 \text{ for } \omega \in [0,2\pi]
\]

(21)

and that \(\beta\) is sufficiently small. Note that \(\text{Re}\{U(e^{j\omega})\} > 0\) for \(\omega \in [0,2\pi]\) is equivalent to the condition that the Nyquist diagram of \(U(q)\) lies strictly in right-half plane. This, on the other hand, implies that phase of \(U(q)\) should line inside \(\pm 90^\circ\) for exponential convergence., showing a good degree robustness in the algorithm.

In summary, if the phase of the nominal plant \(G_s(q)\) lies inside a \(\pm 90^\circ\) degree tube around the phase of the true plant, and \(\beta\) is taken to be sufficiently small, the tracking error \(e(k)\) will converge exponentially to zero tracking error. Small values of \(\beta\) imply that \(u(t) = u(t-T)\), showing that an increased convergence rate (\(\beta \approx 1\)) results in decreased robustness and increased robustness (\(\beta \approx 0\)) in slow convergence rate. Consequently, the tuning of \(\beta\) plays a crucial role, but it is not yet clear how to automate this tuning process.

4 Filtering

Note that the inverse of the plant \(G^{-1}(q)\) has typically a high gain at high frequencies. Therefore one would assume that the algorithm (10) is extremely sensitive to noise \(y(t)\). In order to analyse this situation, assume that only a measured output \(y_m(t)\) is available, where the original output is corrupted through an additive noise model

\[
y_m(t) + y(t) + n(t) \tag{22}
\]

where \(n(t)\) band-limited white noise. In this case the algorithm (10) becomes

\[
u(k) = q^{-T}(u(k) + \beta G^{-1}_e(q)(r(t) - y(t) - n(t))) \tag{23}
\]

and \(y(t)\) is related to \(r(t)\) and \(n(t)\) through the equation

\[
(1-q^{-T}(1-\beta U(q)))y(t) = q^{-T}U(q)\beta(r(t) - n(t)) \tag{24}
\]

Consequently, if \(U(q) = 1\) (i.e. one has a perfect model of the plant), and the noise is uncorrelated, the algorithm does not amplify high frequency noise, but merely delays it and decreases its amplitude by \(\beta\). However, if \(U(q)\) has high amplitude at a certain frequency range (typically high frequency range), the algorithm will start to amplify the measurement noise, which will decrease the tracking performance and possibly lead to divergence.

In order to overcome this problem, consider the following “filtered version” of the original algorithm (10)

\[
u(k) = q^{-T}(F(q)u(k) + \beta G^{-1}_e(q)F(q)e(t)) \tag{25}
\]

where \(F(q)\) is causal, stable LTI system on the positive time-axis. The error evolution equation in this case becomes

\[
e(k) = q^{-T}(F(q) - \beta U(q)F(q))e(k) + (1 - F(q))r(k) \tag{26}
\]

and using a similar argument as before, it can be shown that a sufficient condition for monotonic convergence is

\[
\sup_{\omega \in [0,2\pi]} \left| F(e^{j\omega})(1-\beta U(e^{j\omega})) \right| < 1
\]

(27)

and the algorithm converges to the time sequence

\[
e(k) = (1 - F(q) - \beta F(q)U(q))^{-1}(1 - F(q))r(k) \tag{28}
\]

This equation shows that for those frequencies, where the amplitude of \(F(e^{j\omega})\) is close to unity and the phase is \(0^\circ\),
the algorithm results in perfect tracking of \( r(e^{j\omega}) \). On the other hand, if \( F(e^{j\omega}) \approx 0 \), the convergence condition (24) is met trivially. Furthermore, these frequencies are blocked before they are driven through \( G_{\alpha}(e^{j\omega}) \). Thus \( F(q) \) should be made close to unity at the frequencies where \( r(e^{j\omega}) \) has significant spectral content. Furthermore, outside this frequency range, \( F(q) \) should be close to zero in order to enhance robustness against modelling uncertainty and measurement noise.

5 Simulation examples

Consider the plant model

\[
G(s) = \frac{s + 1}{s^2 + 5s + 6}
\]

(29)

which is sample with \( T_s=0.001 \) s using zero-order hold. The system is supposed to track a sinusoidal reference

\[
r(t) = \sin\left(\frac{2\pi}{6}t\right)
\]

(30)

Figure 1 shows the tracking error when the algorithm (10) is run with \( \beta=0.5 \) on the plant (29).

![Figure 1. Tracking error in the nominal case](image1)

The algorithm “learns” to track the reference signal in 3 cycles or so, demonstrating a very fast convergence rate.

In order to test how the algorithm (10) can cope with measurement noise, \( y(t) \) is corrupted with additive noise that has significant spectral content on the frequency range above 40 Hz. A “typical” noise vector is shown in Figure 2.

![Figure 2. A typical noise vector](image2)

Figure 3 shows how the algorithm performs without any filtering for different values of \( \beta \). In this simulation it is assumed that \( U(q)=1 \).

![Figure 3. \( y(t) \) for different values of \( \beta \)](image3)

The results clearly support the theoretical findings so far, i.e. a decrease in \( \beta \) will decrease the convergence speed but increase the robustness of the algorithm against measurement noise in the limit.

In order to test the idea of using filtering to mitigate the effects of measurement noise, a 5th order Butterworth filter was design with a cut-off frequency of 5 Hz. The cut-off frequency was selected so that the measurement noise would be blocked, but at the same time near perfect tracking would be obtained in the frequency range of \( r(t) \). In this simulation \( \beta \) is taken to be 0.5. Figure 4 shows the result without any filtering, and it is clear that the tracking error has decreased the tracking capability. Figure 5, on the other hand, shows \( y(t) \) when the Butterworth filter is being used. The tracking in terms of amplitude is near perfect, but there is a slight phase difference between the output and reference. This is due to the fact that the Butterworth filter will introduces phase lag at every singe frequency, and therefore perfect tracking in terms of phase cannot be achieved.

![Figure 4. Tracking error without filtering](image4)

![Figure 5. Tracking error with Butterworth filter](image5)
As a final example, consider the case where the true plant model is given by

$$G(s) = \frac{s + 1}{s^2 + 5s + 6} \frac{10}{s + 10}$$

(31)

but the nominal model used in the design of the algorithm is taken to be

$$G_o(s) = \frac{s + 1}{s^2 + 5s + 6}$$

(32)

where the “high-frequency pole” is not properly modelled. It can be shown numerically that the resulting multiplicative uncertainty $U(q)$ is positive real, and therefore the algorithm should converge. Figure 6 shows the results for different values of $\beta$, as expected, the algorithm converges.

6 Conclusions

This paper has investigated the possibility of using a model-inverse based algorithm in the repetitive control framework. When the model is a perfect replicate of the true plant, it has been shown that the algorithm results in monotonic convergence to zero tracking error. In the case of uncertainty in the model, it has been proven that if the phase of the model lies inside a $\pm 90^\circ$ tube around the phase of the true plant, the algorithm converges exponentially to zero tracking error.

Future work consists of applying non-causal filters on the “windowed” tracking error $[e(k), e(k-1), \ldots, e(k-T)]$ in order to enhance the robustness properties of the algorithm at high frequencies. Another line of future work is to adaptively change the learning gain $\beta$ so that a better balance between convergence rate and robustness would be achieved.

Acknowledgements

J. Häätönen and T. Harte are supported by the EPSRC contract No GR/R74239/10.

References


