Networked Sensing and Actuation*

Vincenzo Liberatore

July 8, 2004

Abstract

Network sensing can be integrated with actuation units to control a remote physical environment. A fundamental problem in networked control is that sensing and actuation operate in real-time and, consequently, late signals can jeopardize the stability, safety, and performance of the controlled physical environment. Therefore, a primary objective is to design end-system algorithms that make control applications more tolerant and adaptive to network vagaries and thus able to support faster dynamics for a broad class of physics. This paper proposes a set of methods collectively named contingency control that employ a version of play-back buffers to avoid negative jitter, an expiration time to curb the regular deadbeat control, a contingency control to deal with loss episodes, and an appropriately designed control gain that takes into account all of these factors. An extensive evaluation on simulations and on RTT traces show that the combined approach was able to support much faster physical dynamics across a wide range of network conditions.

1 Introduction

Network sensing can be integrated with actuation units to control a remote physical environment (Figure 1). Applications are far-reaching and include, for example, industrial automation (e.g., [18]), distributed instrumentation (e.g., [1, 9]), disaster recovery (e.g., [15]), unmanned vehicles (e.g., [12]), and home robotics (e.g., [19]). A fundamental problem in networked control is that sensing and actuation operate in real-time and, consequently, late or missing signals can jeopardize the stability, safety, and performance of the controlled physical environment. Networked control is a real-time distributed application whose effectiveness depends on its ability to tolerate losses and to adapt to delays and jitter.

The primary objective of this paper is to investigate the means to make a control application tolerant and adaptive to network vagaries. Our basic tenet is that control algorithms should combine computer communication methods (say, play-back buffers) with control-theoretical techniques (say, the choice of a feedback gain). For example, the presence of a buffer can change the value of the feedback gain and, conversely, the real-time physics can suggest a particular behavior for the play-back buffer. An integrated approach should lead to stronger results than tackling each side of the issue independently.

Networked control addresses sensing and actuation under disparate network conditions and for environments whose physics differs radically in nature and scale. As a result, distributed control applications can differ tremendously in methodology and requirements. At the same time, remote

*Division of Computer Science, Case Western Reserve University. 10900 Euclid Avenue, Cleveland, Ohio 44106-7071, USA. Fax: (216) 368 6039. E-mail: {vincenzo.liberatore, mb}@case.edu. URL: http://vincenzo.liberatore.org/NetBots/. This work has been supported in part under NSF grant CCR-0329910.

sensing and actuation present also clear similarities across a variety of environments, and so a set of general methods should be established for entire classes of applications. This paper will address networked sensing and actuation over wide-area IP networks and for the broad class of linear systems, which include any environment that can be described by linear differential or difference equations.

A physical environment is, roughly speaking, characterized by the speed of its dynamics. For example, a Tokamak reactor [23] has much faster dynamics than a manufacturing robot [18]. A faster environment is typically harder to control than a slower one. Moreover, most environments become more difficult to control as network service levels deteriorate. For example, controlled systems cannot always be made stable in the presence of packet losses [28]. As a result, network service levels impose a bound on the faster physical dynamics that can be supported in the target environment. At the same time, control applications can be rendered more tolerant and adaptive by appropriate end-host sampling and control algorithms. In summary, *the rules of the game are to design end-system algorithms that make control applications more tolerant and adaptive to network vagaries and thus able to support faster dynamics for a broad class of physics.*

The paper will focus on the concepts and explanations. A formal derivation of most results is deferred to appendices. Section 2 defines the networked control problem and gives the necessary background. Section 3 describes the wide range of sampling and control strategies discussed in this paper. Section 4 describes the evaluation methodology. Section 5 summarizes the outcome of our evaluation. Section 6 sketches an extension of the methodology to higher-dimensional linear systems. Section 7 outlines related work in the area, and Section 8 concludes the paper.

2 Networked Control

The definition of a *plant* captures the notion of a physical system where sensor data are collected and control signals are delivered with the objective of affecting the physical environment. The term “plant” originated from the industrial automation area, but it has since been applied to express a generic unit of sensing and actuation. Section 2.1 discusses an abstract plant and Section 2.2 introduces networked plants.

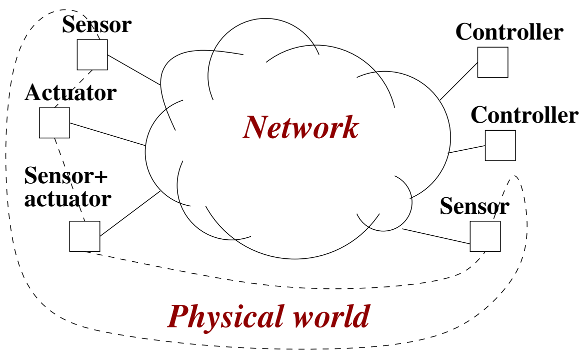

Figure 1: The *networked control* vision.

2.1 The Scalar Linear Plant

The scalar linear plant is the primary type of plant that will be considered in this paper due to its generality and relative simplicity. However, results can be generalized to higher-dimensional linear systems as sketched in Section 6. The focus on scalar linear plants is thus mostly for concreteness and ease of exposition. A scalar plant is characterized by its *state* $x(t)$, an *input* $u(t)$ (the control signal), and an *output* $y(t)$ (the sensor data), all of which are functions of the time t. In a scalar linear plant, the state, input, and output are related by the equations:

$$\begin{cases} \dot{x}(t) = ax(t) + bu(t) \ , \\ y(t) = x(t) \ , \end{cases} \tag{1}$$

where a, b are real numbers and $a > 0$. Equation (1) is generic across applications because it models any physical system characterized exactly or approximately by a differential equation.

Example. The simplest scalar linear plant is a savings account whose balance $x(t)$ increases depending on the current balance, the interest rate a, and the new deposit $bu(t)$.

The dynamics of plant (1) depend on the value of a. For example, if $u = 0$, then $x(t) = x(0)e^{at}$, which increases more rapidly for larger values of a. In general, larger values of a are thought to denote faster plant dynamics [6, 22], and a unit increase in the value of a denotes an exponentially faster plant evolution.

The control design problem is basically to achieve certain system properties by appropriately setting the input $u(t)$. For example, the design could specify a *reference output* $r(t)$ and the problem is to make $y(t)$ as close as possible to $r(t)$. In this case, a design objective could be to achieve *tracking*, which is, informally, the speed of convergence of $y(t)$ to $r(t)$.

Example. In the example of the savings account, the reference could specify a fixed balance r needed to finance retirement and the control will specify the deposits $bu(t)$.

In a *digital control system*, the plant output $y(t)$ is sampled at time $t_1, t_2, \ldots$ to generate the time series $y(t_1), y(t_2), \ldots$. *Even sampling* is the particular case in which the output is being sampled at regular intervals $t_{i+1} = t_i + T$, where T is the constant duration of a *sampling period*. The inverse of the sampling period $1/T$ is called the *sampling rate* or *sampling frequency*. Returning to the general case of arbitrary sampling, a *controller* sets the value $u(t) = u_i$ in the interval $[t_i, t_{i+1})$ as a function of $y(t_1), y(t_2), \ldots, y(t_i)$ so as to achieve, say, tracking of a reference set point r. As we shall see later on, the control is more difficult for faster plants than for slower ones. A common type of controller is the *proportional controller*, which depends only on the last sensor reading and sets $u_i = -ky(t_i)$, where $k > 0$ is called the *feedback gain*. The plant behavior becomes

$$\dot{x}(t) = ax(t) - bkx(t_i) \qquad\qquad (t_i \le t \le t_{i+1}) \ , \tag{2}$$

so that the overall system behavior depends on the value of the control gain k. The gain k can be set, for example, to achieve *deadbeat control*, which is defined as follows. First, assume that $r = 0$ (the same argument can be developed for $r \neq 0$, but with a slight notational complication). The value of k is chosen in such a way that it solves the differential equation (2) with boundary conditions: $x(t_i)$ equal to the measured plant output at time t_i and $x(t_{i+1}) = r = 0$. Figure 2 shows a solution $h(t)$ for $x(t)$ in (2) with boundaries $x(0) = -1$ and $x(1) = 0$. Since $x(t_{i+1}) = 0$, $u_{i+1} = 0$, which implies that $\dot{x}(t_{i+1}) = 0$ and so $x(t) = 0$ for all $t \ge t_{i+1}$. Therefore, at time t_{i+1}, the plant achieves the desired set point. The deadbeat controller owes its name to the fact that the system output converges to the reference with no oscillation. Deadbeat control can be illustrated

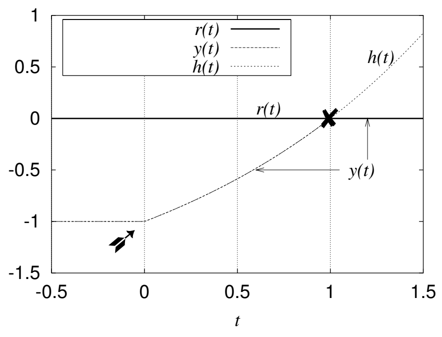

Figure 2: Plant output under a deadbeat controller.

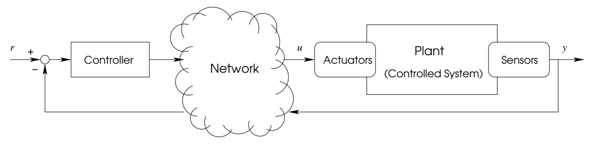

Figure 3: Network-based control.

with a whimsical analogy: a hunter (the deadbeat controller) lies in wait at the assigned location and can aim its arrow by setting the value of u. The hunter fires its arrow so that it follows the trajectory $h(t)$ until it exactly hits the flying target r at time 1.

2.2 Networked Control

A *networked control system* is a digital control system in which the sensor data $y(t_i)$ and control signals u_i ($i \geq 1$) are delivered over a network. Networked control typically involves a plant that is controlled remotely, and Figure 3 shows the corresponding information flows. Networked control is complicated by the presence of delays, jitter, and packet losses, as exemplified in Figure 4. In the first place, network latency can delay the application of a control signal at the plant site. In a networked control system, the ith sensor reading $y(t_i)$ reaches the controller after a delay, the controller computes u_i as a function of $y(t_1), y(t_2), \ldots, y(t_i)$, sends u_i back to the plant, which will apply the control u_i in the interval $[\theta_i, \theta_{i+1})$, where $\theta_i = t_i + \mathrm{RTT}_i$ and RTT_i is the value of the round-trip time at the ith step. In general, it will be convenient to subdivide the ith RTT into two terms as $\mathrm{RTT}_i = \tau + \xi_i$, where τ is a *nominal RTT* between the plant and the controller and does not depend on i, whereas ξ_i is the corresponding *jitter* and includes all time dependent variability in the end-to-end RTT. The constant value of τ is easier to deal with in that a constant τ can be used by an observer to predict the systems state forward to time $t_i + \tau$ by simulating the plant dynamics. In other words, the analysis is clearer if the RTT is broken into a constant τ, which is relatively easier to address with known control-theoretical methods, and ξ_i, which introduces novel network-related complications. The nominal RTT τ can be chosen in a variety of ways. For example, in the rest of the paper, the value of τ will be sometimes taken as the minimum RTT between controller and plant. If this is the case, then $\xi_i \geq 0$.

Networks can suffer from packet drop-outs, so that either $y(t_i)$ or u_i can be lost. Similarly, if u_i reaches the plant too late, i.e., after u_j ($j > i$), the signal u_i is also considered as lost. Although packet losses can be reduced with techniques such as forward error correction, they cannot always

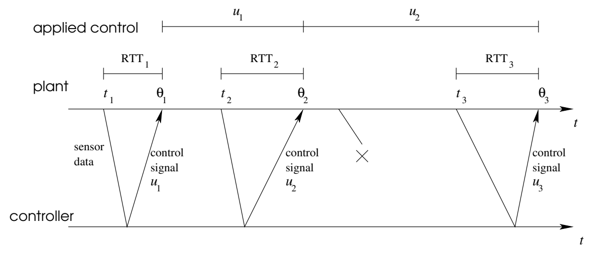

Figure 4: The timeline of network events in remote control.

be entirely eliminated. The effect of packet losses is basically to alter the plant's sampling schedule. For example, if even sampling is used and one sample $y(t_i)$ is lost, then this is equivalent to the case when no output was sampled at time t_i. In this case, the plant effectively follows the sampling schedule t'_j, where

$$t'_j = \begin{cases} t_j & \text{if } j < i \\ t_{j+1} & \text{if } j \geq i \end{cases} .$$

The sampling schedule is analogously altered if u_i is lost or reaches the plant too late. In general, packet losses can introduce uneven sampling even if the original plant's sampling was evenly spaced. In the analysis of networked control systems, the sampling period T must be replaced by a time-varying quantity $\Delta t_i = t_{i+1} - t_i$. Furthermore, the *effective sampling period* is defined as $\overline{T} = E[\Delta t_i]$ [8, 28], and the *effective sampling rate (frequency)* as $1/\overline{T}$.

On the whole, the plant samples y at time t_i $(i = 1, 2, \dots)$. The t_i sequence implicitly factors in the occurrence of packet losses and late control signals. The controller receives $y(t_i)$, computes a control u_i, and sends it to the plant. The plant receives the control at time $\theta_i = t_i + \tau + \xi_i$ and applies it continuously in the interval $[\theta_i, \theta_{i+1})$. In general, the values of t_i and ξ_i are unknown in advance since t_i depends on losses and ξ_i depends on jitter. However, it can always be assumed that $\theta_{i+1} \geq \theta_i$ because late signals are discarded.

3 Networked Controllers

3.1 Proportional Controllers

The analysis considers an initial plant state $x(0) = -1$, a reference trajectory $r(t) = 0$, and a proportional controller $u = -ky$. We will also assume that the controller is able to accurately predict the plant state $y(t_i + \tau)$ forward to time $t_i + \tau$. As a result, the control

$$u_i = -ky(t_i + \tau) \tag{3}$$

will be applied during $[\theta_i, \theta_{i+1})$. A perfect observer (prediction) is seldom available in practice, but it is necessary for methodological reasons. Specifically, if the prediction were inaccurate, it would be difficult to distinguish the effects of incorrect prediction from those due to network vagaries. Although an actual control system is also subject to state estimation errors and exogenous disturbances, this methodology allows us to focus on network-induced effects.

Equations (1), (3), and the definition of θ_i can be combined as

$$\dot{x}(t) = ax(t) - bkx(\theta_i - \xi_i) \qquad (\theta_i \leq t < \theta_{i+1}) . \tag{4}$$

Equation (4) can be integrated (Appendix A) to obtain the homogeneous recurrence

$$x_{i+1} = \alpha_i x_i + \beta_i x_{i-1} \ , \tag{5}$$

where

$$x_i = x(\theta_i - \xi_i) \ , \qquad \alpha_i = e^{a\Delta t_i} - \frac{bk}{a} e^{a\Delta s_i} + \frac{bk}{a} \ ,$$

$$\Delta s_i = \Delta t_i - \xi_i \ , \qquad \beta_i = \frac{bk}{a}\left(e^{a\Delta s_i} - e^{a\Delta t_i}\right) \ .$$

Equation (5) describes the controlled plant behavior. In particular, if $x_i = x_{i+1} = 0$, then $x_j = 0$ for all $j \geq i$. More generally, we will say that the plant state x_i *converges to the reference* r if and only if $\lim_{i\to\infty} x_i = r = 0$. If x_i is a random variable, we will say that x_i *converges to the reference* r *in the expectation* if and only if $\lim_{i\to\infty} E[x_i] = r = 0$.

The controller (3) has a parameter k (the feedback gain) that is tunable. Different choices of k will be introduced in this paper and their definition is discussed in the rest of this section.

3.2 Deterministic Control

3.2.1 Baseline: Deterministic Deadbeat Control

The ideal case assumes that jitter and losses are absent and equal-space sampling is used. Since $\xi_i = 0$, then $\beta_i = 0$. The reference $r = 0$ can be obtained by deadbeat control by setting $\alpha_i = 0$, so that $x_{i+1} = 0$. Since $\Delta t_i = T$ is constant, the equation $\alpha_i = 0$ can then be solved for the control gain k as

$$k_d = \frac{a}{b}\frac{e^{aT}}{e^{aT} - 1} \ .$$

The k_d controller is the baseline for the evaluation of the subsequent, more complex controllers.

3.2.2 Play-Back Buffers

The assumption $\xi_i = 0$ (no jitter) can be guaranteed with a play-back buffer that applies the control exactly at time $t_i + \tau$ if it arrived before that time and drops the control if it was received too late. A play-back buffer can increase the loss rate. For example, a delay spike can be transformed into a loss episode. A relaxed version of a play-back buffer is a *reverse play-back buffer*, which works exactly like a regular play-back buffer for early signals, i.e., it holds them until the play-back time $t_i + \tau$, but it does play back also late non-reordered signals as soon as they arrive. The reverse play-back method is based on the intuition that control signals are better received late than never (this intuition does not necessarily carry over to other real-time applications). Moreover, the reverse play-back scheme does not introduce additional losses, but the control signals are not necessarily applied at predictable times.

3.2.3 Alternative Deterministic Controller

A different type of controller obtains $\alpha_i = 1$ in the deterministic case by setting $k_1 = a/b < k_d$. The k_1 controller implies that $x_i = x_1$ (for all i's) so that no progress is ever made toward r. The controller k_1 seems of little use but we will show that there are circumstances in which k_1 is substantially the best possible controller.

3.3 Stochastic Deadbeat Control

Of course, packet losses cannot be predicted exactly. As a result, Δt_i is typically an unknown quantity. However, it might be possible to characterize its probability distributions and use it to establish a desirable feedback gain k. Analogously, it might be possible to characterize the probability distribution of ξ_i to establish an appropriate feedback gain and use it as an alternative to play-back buffers. The following paragraphs examine first the case of packet losses in isolation (i.e., without any jitter), then the case of jitter in isolation (i.e., assuming no packet loss), and finally the combined case of both jitter and losses.

3.3.1 Packet Losses

This section considers the case when a sample or control signal can be lost but there is no jitter. The assumption holds exactly if the network guarantees no jitter, it holds approximately if the jitter is small, and it can be enforced with a play-back buffer. If $\xi_i = 0$, then $\beta_i = 0$. The recurrence (5) becomes $x_{i+1} = \alpha_i x_i$. A stochastic loss process implies that that Δt_i is a random variable, and its moment generating function will be denoted as M_T. A probabilistic deadbeat control fires to hit the expected position of the target by setting the feedback gain k_L so that

$$E[\alpha_i] = E\left[\left(1 - \frac{bk_L}{a}\right) e^{a\Delta t_i} + \frac{bk_L}{a}\right] = 0\ ,$$

and, if $M_T(a)$ exists, this objective is accomplished by setting

$$k_L = \frac{a}{b}\left(1 + \frac{1}{E\left[e^{a\Delta t_i}\right] - 1}\right) = \frac{a}{b}\left(1 + \frac{1}{M_T(a) - 1}\right)\ .$$

Since $a > 0$, $M_T(a) > 1$, and so $k_L > 0$. Furthermore, if $M_T(a)$ is known, $E[\alpha_i] = \beta_i = 0$ implies that x_i converges to the reference in the expectation.

Example. If Δt_i is an exponential random variable with mean $\overline{T} = 1/\lambda$ and $\lambda > a$, then $k_L = \lambda/b$, that is, k_L is proportional to the effective sampling frequency. However, if $\lambda \leq a$, then $M_T(a)$ does not exists.

Example. If the sampling period is uniform and the loss probability is $p < e^{-aT}$, then Δt_i is a geometric random variable with parameter $1-p$, and so $k_L = a(1-p)e^{aT}/(b(e^{aT}-1))$. However, if $p \geq e^{-aT}$, then $M_T(a)$ does not exists.

Example. The Gilbert model [29] assumes that packet losses are induced by a two-state Markov chain. Let p ($q < e^{-aT}$) be the probability of remaining in the good (lossy) state. Then (Appendix D),

$$k_L = \frac{a}{b}\frac{(1-p)(1-q)e^{2aT} - q(1+p)e^{aT} + (1+p)}{(1-p)(1-q)e^{2aT} - pqe^{aT} + p}\ .$$

However, if $q \geq e^{-aT}$, then $M_T(a)$ does not exists.

In general, k_L can be used only if $M_T(a)$ exists. The condition that $M_T(a)$ exists imposes a constraint on the minimal performance that the network must support. For example, k_L can be used in the Gilbert model only if $q < e^{-aT}$, which in turn bounds to $1/(1-e^{-aT})$ the expected length of a run of consecutive packet losses. If the network performance violates the bound, a controller can still set $k = k_1 = a/b$, in which case $x_i = x_1$ (for all i's), so that x does not advance toward the reference but it does not move away from it either. The condition that $M_T(a)$ exist can be equivalently viewed as establishing the fastest plant that can be supported by k_L in the given network conditions. For example, the Gilbert model entails a network that can support a plant not faster than $-(\ln q)/T$. The notation $a_L^{(G)} = -(\ln q)/T$ will denote the critical plant speed assuming a Gilbert model for the loss process. Similarly, the value $a_L = -(\ln p)/T$ is the critical plant speed that is supported under the given loss rate p assuming an i.i.d. loss process.

A common thread in stochastic deadbeat control is that network vagaries should not be too large or else $M_T(a)$ does not exist and the plant would fail to converge, but they should not be too small either or else the stochastic control practically behaves like a deterministic control. In the case of packet losses, k_L differs more significantly from k_d depending on whether $M_T(a)$ differs from e^{aT}.

Example. If Δt_i is a geometric random variable with parameter $1-p$ (as above), then $M_T(a)$ differs from e^{aT} more significantly for larger values of p (Appendix C). On the other hand, $p < e^{-aT}$, so that the most informative values of p are close to e^{-aT} but slightly smaller than it.

Another strategy is to simply use the deterministic deadbeat controller k_d. The k_d controller can be intuitively justified if occasional packet losses are followed by a long period of good connectivity. The controller k_d can have the plant output diverge during poor connectivity periods, but immediately snaps back to the reference as soon as good connectivity resumes.

3.3.2 Jitter

The second controller is designed for the case when the communication is subject to jitter, but there are no packet losses. In this scenario, the interval $\Delta t_i = T$ is a constant.

Remark. The assumption holds during a *loss-free run*, i.e., a sequence of exchanges with no packet losses. Various measurements have indicated that loss-free runs are long and frequent and that, conversely, loss events are short and infrequent [14, 16, 29].

If the jitter ξ_i is a random variable with moment generating function M_J, then we can make $E[\alpha_i] = 0$ by taking

$$k_J = \frac{a}{b}\frac{e^{aT}}{e^{aT}M_J(-a) - 1}\ .$$

The original condition $k_J > 0$ holds only if $e^{aT}M_J(-a) > 1$. For example, it can be shown that $k_J > 0$ if $\xi_i \leq T$. The following propositions express a necessary and sufficient condition for convergence in the presence of jitter. Their proofs are postponed to Appendix F.

Proposition 1. *In the case when $\Delta t_i = T$ for all is, the ξ_i's are i.i.d. random variables, and the k_J controller is used, x_i converges to the reference r in the expectation if and only if*

$$M_J(-a) > \frac{e^{2aT} + 1}{e^{2aT} + e^{aT}}\ .$$

Sketch. The value of k_J makes $E[\alpha_i] = 0$, but it can also increase $E[\beta_i]$, so that convergence is ensured only when $|E[\beta_i]| < 1$. □

Corollary 2. *If $M_J(-a) > (e^{2aT} + 1)/(e^{2aT} + e^{aT})$, then $k_J > 0$.*

The controller k_J makes $E[\alpha_i] = 0$, but the gain is $k_J > k_d$, so that the value of β_i increases when compared to the baseline controller k_d. Therefore, it is unclear whether the k_J controller would actually provide better performance than k_d. The relative performance of k_J over k_d will be addressed by experiments (Section 5).

Example. Suppose that the distribution of ξ follows a gamma distribution $\Gamma(r, \lambda)$ [16] (this example is only approximate because a gamma distribution would lead to a positive probability of packet re-ordering). The significance of the parameters r and λ is as follows: $\lambda = E[\xi]/\sigma^2 = 1/(E[\xi]-m)$, where σ^2 is the variance of ξ and m is the mode of the distribution, and $r = \lambda E[\xi] = \sqrt{E[\xi]/\sigma}$ is proportional to the square-root of the inverse of the coefficient of variation. Then, $M_J(-a) = (\lambda/(\lambda+a))^r$. Therefore, the controller k_J is more significantly different from k_d when $M_J(-a)$ is small, which occurs when r is large or λ is small. On the other hand, Proposition 1 gives a bound on the fastest plant that can be tolerated under a certain amount of jitter (this can be easily seen because if a is large, then $M_J(-a) \simeq 0$ but $M_J(-a)$ should be approximately equal to 1 or more). If $aT/r \simeq 0$, then $e^{(aT)/r} \simeq aT/r$, and this bounds becomes approximately $E[\xi_i] \leq T$, i.e., the expected jitter must be less than the sampling period (or, equivalently, the sampling period must be set at design time to be longer than the expected jitter). A different case is related to the measurements that will be presented in Section 4, where λ is large. If $\lambda \gg a$, then $M_J(-a) \simeq 1 - ar/\lambda = 1 - E[\xi]a$.

The notation $a_{J,\max}$ denotes the value of a that gives rise to equality in Proposition 1 and it is an upper bound to the value of a for which there is convergence under k_J for the given jitter distribution. The notation $a_{J,\min}$ denotes the value of a that makes $M_J(-a_{J,\min}) = 0.95$. The relevance of $a_{J,\min}$ is that smaller values of $a \leq a_{J,\min}$ lead to $M_J(-a) \simeq 1$, so that $k_J \simeq k_d$ and little benefit can be derived from jitter compensation.

3.3.3 Loss and Jitter

The general case is in the presence of both loss and jitter. Let M_S be the moment generating function of Δs_i. Then,

$$k^* = \frac{a}{b}\frac{M_T(a)}{M_S(a) - 1} .$$

In general, $M_S(a)$ can be expressed as a product of M_T and M_J only if Δt_i and ξ_i are independent, a fact for which there is no empirical evidence [4]. At any rate, the combined case is subject to similar considerations as in the previous controllers, for example as pertaining to the fastest plant that can be supported by a certain network level of service.

3.3.4 Online Estimation of Moments

The stochastic controllers require the prior knowledge of at least one moment generating function. In general, the moment generating function is not known beforehand and it must be estimated online. Although the moment generating function can be in principle be estimated in many different ways, this paper adopts a TCP-like exponential moving average with weights 1/8 and 7/8. For example, M_T is estimated as

$$M_{T,i}(a) = \frac{1}{8}e^{a\Delta t_{i-1}} + \frac{7}{8}M_{T,i-1}(a) . \qquad (6)$$

3.4 Contingency Control

The contingency control is a departure from the proportional controllers above in that the controller sends to the plant *two* control signals, which will be called the *regular control* (denoted by u_i) and the *contingency control* (denoted by $u_i^{(c)}$). The plant applies the regular control when it receives it and it also applies the contingency control when it sends a new sample. More precisely, the plant

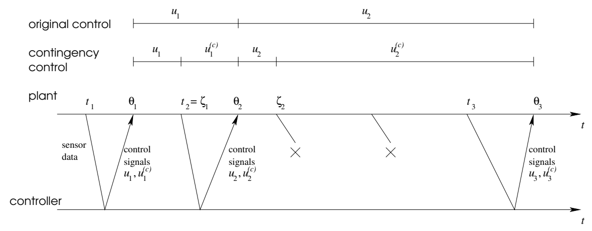

Figure 5: The timeline of network events in the remote control of a plant with contingency control.

samples the output y regularly at time $(i-1)T$ $(i = 1, 2, \dots)$, sends it to the controller, and applies the last received contingency control. If the controller receives $y((i-1)T)$, it computes a regular and a contingency control and sends them to the plant. If the plant receives these controls at time $\theta_i = t_i + \tau + \xi_i < iT$, it applies the regular control continuously until time iT, at which point it switches to the contingency control. A representative timeline is shown in Figure 5. The value t_i is defined as before, namely t_i is the ith time when the plant sent a sample that caused a control $u_i, u_i^{(c)}$ to be received. Let $\zeta_i = t_i + T$ be the time instant when the plant switches from u_i to $u_i^{(c)}$. Thus, $\theta_{i-1} \leq \zeta_{i-1} \leq t_i \leq \theta_i$.

This paper will consider the case when the controls are of the form $u_i = -k_c y(t_i + \tau)$ and $u_i^{(c)} = 0$. The main idea is to implement a form of deadbeat control by selecting a value of k_c that makes $E[x(\zeta_i)] = r$, at which point $u_i^{(c)}$ would result into $x(t) = r$ for all $t \geq \zeta_i$. The corresponding value of k_c is (Appendix G):

$$k_c = \frac{a}{b} \frac{e^{a(T-\tau)}}{e^{a(T-\tau)} M_J(-a) - 1} \ ,$$

and a generalization to higher dimensional plants is briefly sketched in Section 6. The value of k_c is formally close to k_J, and the difference is due to the fact that the regular control operates on an interval of length $T - \tau$ rather than T. Furthermore, the contingency control implicitly considers as lost any packet that arrives after the next sampling point, so that it is always the case that $\xi_i \leq T - \tau$. Since $\xi_i \leq T - \tau$, $M_J(-a)$ always exists (unlike the k_J controller) and $k_J > 0$. The main difference between the two controllers, however, is that, at time ζ_i, the contingency control $u_i^{(c)}$ will take over. As a consequence, the plant equations becomes (Appendix G):

$$x(\zeta_i) = x(t_i + \tau) \left(e^{a(T-\tau)} - \frac{bk}{a} \left(e^{a(T-\tau-\xi_i)} - 1 \right) \right) \ ,$$

which depends only on the last sample, whereas (5) depends on the last two samples. As a consequence, k_J was unable in general to guarantee convergence in the expectation even if $E[\alpha_i] = 0$ because $\beta_i \neq 0$ (Proposition 1). By contrast, contingency control always converges in the expectation for all distributions of ξ_i if an exact value of $M_J(-a)$ is known.

A fundamental property of k_c is that if $\xi_i = 0$, then $x(t) = r$ for all $t \geq \zeta_i$. Therefore, k_c achieves the reference in the absence of jitter at the first packet exchange that is not lost. On the other hand, jitter can be eliminated with a play-back buffer, especially since k_c tolerates a corresponding increase of packet drop-outs. If jitter is eliminated by a play-back buffer, then $M_J(-a) = 1$, which also eliminates the need for an online estimation of the moment generating function. Contingency and play-back can be regarded as the two sides of the same coin: contingency prevents a control from being applied for too long and (reverse) play-back prevents a control from being applied too early. The rest of the paper assumes that contingency control always adopts a form of play-back

Id	**Name**	**Start**	**End**	**Loss rate**	**min RTT** ms	**Cut-off** ms (%)
0	wopr.cwru.edu	Mon Sep 30 13:30:02	13:45:48	0%	0.235	0.5 (1.06%)
1	weatherhead.cwru.edu	Mon Sep 30 13:45:48	14:01:35	0%	0.270	0.5 (1.48%)
2	scp.grc.nasa.gov	Wed Oct 2 06:27:47	06:43:41	0.44%	60.762	70 (0.58%)
3	scp.grc.nasa.gov	Mon Sep 30 14:01:35	14:17:21	25.13%	61.026	110 (0.93%)
4	www.ri.cmu.edu	Wed Oct 2 06:43:41	06:59:32	4.38%	15.083	30 (0.91%)
5	www.comau.com	Wed Oct 2 07:39:07	07:54:54	1.22%	38.471	60 (1.22%)
6	www.icra-iros.com	Wed Oct 2 06:59:35	07:15:28	7.9%	66.503	80 (1.13%)
7	control-lab.et.tudelft.nl	Mon Sep 30 11:00:01	11:15:48	7.04%	123.439	200 (0.11%)

Table 1: Internet measurements (Year 2002).

scheme. Since k_c is resilient to both losses (as long as one control packet makes it to the other side) and jitter, uncertainties can arise only from estimation errors and exogenous disturbances.

4 Experimental Methodology

The experiments fall into two broad categories, and for each category the sensitivity to all parameters was explored. For compactness, the rest of the paper will only report on the most informative cases.

Traces Networked control will be simulated on RTT traces. In the case of scalar plants, all samples and control signals are typically contained within a packet over an unreliable transport. We also assume no loss resilient coding, and so packet losses and delays are equivalent to the corresponding loss and delay in the delivery of a signal. The traces were collected with the `fping` program [7], which we modified to exclude retries and to output the RTT figures in microseconds. Each trace includes samples spaced by 200ms for a total duration of 15 minutes. The traces were collected by a Solaris workstation that was placed on a `cwru.edu` network and that pinged the geographically dispersed hosts shown in Tables 1 and 2. The workstation has a clock resolution of $3\mu s$, as measured by repeated invocations to `clock_gettime(CLOCK_HIGHRES)` [26]. A limitation of RTT traces is that ICMP packets were used [27] whereas networked control would more typically run over UDP/IP. Traces were collected originally in the year 2002 (Table 1) and the same measurements were repeated in 2004 (Table 2). It will be assumed in the simulations that the value of the minimum RTT is known to the controllers. The assumption is motivated by the stability of routes over long time-scales [20], which makes it possible to estimate intrinsic path characteristics from a few measurements at start time (e.g., [10]). At any rate, the minimum RTT was subtracted from the trace RTTs to obtain a sequence of jitter measurements ξ_i. The jitter measurements were divided into a body and a tail because a tail measurement is more likely to be treated as a packet loss and also to ensure that delay spikes would not skew the computation of various statistics, such as averages. Tables 1 and 2 show the cut-off delay between the body and tail and the percentage of jitter samples that fell in the tail. The cut-off delay was chosen so that the tail would contain approximately 1% of the measured data points [14], but was always less than the sampling interval (200ms). The body of the jitter distribution visually matched a gamma distribution, confirming the observations of [16].

Tables 4 and 5 also report the values of a_L, $a_L^{(G)}$, $a_{J,\max}$, and $a_{J,\min}$.

The two sets of measurements (Tables 1 and 2) show some significant changes in the network levels of service over time. In part, the new traces reflect the upgrades of our access network, which

Id	**Name**	**Start**	**End**	**Loss rate**	**min RTT** ms	**Cut-off** ms (%)
8	wopr.cwru.edu	Thu May 20 11:21:15	11:36:59	0%	0.233	0.6 (0.84%)
9	weatherhead.cwru.edu	Thu May 20 11:36:59	11:52:42	0%	0.282	10 (1.04%)
10	scp.grc.nasa.gov	Thu May 20 11:52:42	12:08:25	0.31%	56.783	200 (1.78%)
11	www.ri.cmu.edu	Thu May 20 12:08:25	12:24:09	0.02%	47.872	51 (0.82%)
12	www.comau.com	Thu May 20 12:24:09	12:39:52	1.71%	34.941	110 (1.16%)
13	www.icra-iros.com	Thu May 20 12:39:52	12:55:37	1.53%	58.216	110 (1.00%)
14	control-lab.et.tudelft.nl	Thu May 20 12:55:37	13:11:29	0.04%	122.908	160 (1.00%)

Table 2: Internet measurements (Year 2004).

has been significantly enhanced in terms of bandwidth and of firewall reliability. For example, the lossy scp(9/30/02) trace was due to a misconfiguration of our campus firewall that caused it to drop roughly 1/4 of the packets. This trace will still be employed in the simulations, however, because misconfigurations, as well as failures or attacks, are possible even when links are appropriately dimensioned or Quality-of-Service is supported. The loss rate decreased almost always in 20 months, it is close to 0 on all newer traces, and it is reflected by larger values of a_L and $a_L^{(G)}$. Although the minimum RTT is basically unchanged or smaller, the cut-off delays have increased, which signifies larger delays and an increased intensity of delay spikes. The body of the jitter distribution also shows more variability, as expressed by the smaller value of $a_{J,\max}$. In general, a network can trade queuing delays for losses, and these measurements suggest that, in the last 20 months and within the scope of these networks, more emphasis has been placed on reducing packet drop-outs at the expenses of higher delays and jitter.

Controllers Several types of controllers were introduced in Section 3, but some controllers are merely a step in the construction of more complex algorithms. For example, the k_1 controller serves solely to make a point if the moment generating function does not exist. The evaluation will focus on the meaningful controllers: k_d (which also serves as the comparison baseline), k_L, k_J, k_S, and k_c. The k_c controller will be considered in the presence of a reverse play-back buffer, which is considered as an integral component of the contingency mechanism. The k_d and k_L controllers will be examined both in the presence and in the absence of a reverse play-back buffer. The k_J and k_S controllers will only be considered in the absence of a play-back buffer because these controllers provide an alternative approach to jitter. When the play-back buffer is used, the play-back delay τ can be set with a variety of sophisticated methods [24] but, for simplicity, this paper uses a play-back delay equal to the cut-off value (Tables 1 and 2) or 190ms, whichever is less, so that the regular control can operate for at least a small interval. When buffers are not used, the value of τ is set to the minimum RTT recorded in a trace. In trace-based simulations, controllers use moment estimates initialized to an average collected from the same trace on which the simulations are subsequently run. Some controllers use moment estimates from Equation (6). The initial seed $M_{T,0}$ can be chosen in various ways depending on whether the controller underwent an initial training period. The choice $M_{T,0} = 1$ basically assumes no initial training, disables stochastic compensation on the first step, and then gradually reintroduces it as the moving average is updated. A second method assumes a long training period, such as for example from a continuously running plant or from a probe-based admission control mechanism [10], and specifically it derives a maximum likelihood estimate of $M_T(a)$ from the execution trace and uses it as the initial $M_{T,0}$ in Equation (6). Another complication is that several controllers have the property that if the state x reaches the reference r exactly they will not move from it for the rest of the simulation. As a result, these controllers practically use only an initial part of a trace. The effect can be avoided by introducing a disturbance on the output. Such a disturbance would not affect the plant state if it were not that the control signals depend on it. Several disturbance models are possible and our trace-based simulations add a disturbance to the output after each integration step, and the disturbance value is proportional to a Gaussian random variable with mean 0 and standard deviation D. The standard deviation D is a measure of the strength of the disturbance and our simulations use $D \propto e^{-a}$ because faster plants can in general tolerate less noise. Disturbances are introduced solely so that the simulation would use the entire trace and, as mentioned above, a thorough analysis of disturbances is beyond the scope of this paper.

Performance Metrics The simulation results can be expressed by plotting the plant output as a function of time. The simulation would be captured more succinctly by control-performance summary metrics, such as overshoot or settling times [6, 22], but these metrics are not always appropriate in the networked case. For example, the settling time is defined under the assumption that the plant output does "settle" permanently at or around the reference. In the networked setting, a plant sometimes appears to settle at the reference value, but it subsequently diverges from it due to a loss episode or a delay spike [25]. Therefore, overshoot is well-defined only for some cases, but not for others. A first metric is *overshoot*, which we define as $1+\max_i x_i$. (Our definition is consistent with that usually given for unit step response $x(0) = 0, r = 1$ [6, 22].) Another metric is the *ϵ-rise time*, which is the smallest value of t_i for which $|x_i| \leq \epsilon$. A commonly used value of ϵ is $\epsilon = 0.05$ [6, 22]. Another commonly used metric is the settling time, which however cannot be used in networked control because some network patterns can force a plant to move away from its reference even after the plant has "settled". A different type of metric is, for example, the complementary cumulative distribution function $\Pr[|y(t)| > q]$ that the system output differs from the reference by more than q. In some cases, plants will fail to converge during the course of a simulation, and we will say that a plant *crashes* if there is an i such that $|x_i| > 10^6$. In general, performance metrics above could drastically depend on the characteristics of the trace, so that it is difficult to compare, say, $\Pr[|y(t)| > q]$ across all traces for one fixed value of the plant speed a. A more meaningful comparison is to estimate the maximum value $a_{\max}$ of the plant speed a at which $|y(t) - r|$ is greater than 0.1 for less than 1% of the simulation time. The value of $a_{\max}$ intuitively expresses the order of magnitude of the maximum plant speed that can be supported by a controller under the given network conditions.

Simulations Additionally, packet losses and delays were generated from the various probabilistic models described in Section 3.3. The simulations were especially useful in confirming the importance of precisely setting the value of k and in validating the accuracy of online moment estimators. Further details are postponed to Appendix H.1.

5 Evaluation

Jitter Compensation Jitter can be compensated for either with a play-back buffer or by changing the feedback gain to k_J. A play-back buffer eliminates jitter at the potential cost of increased losses, whereas k_J is only effective for certain values of a (Tables 4 and 5). The two methods can be especially compared on the weatherhead(2004) trace because this trace has no losses, which makes it possible to study jitter in isolation, and $a_{J,\min} \ll a_{J,\max}$, which is appropriate for k_J. Experiments on the other traces confirmed that k_J is only meaningful for a range of values of a. However, the exact range boundary often differed from those in Tables 4 and 5 due to phenomena

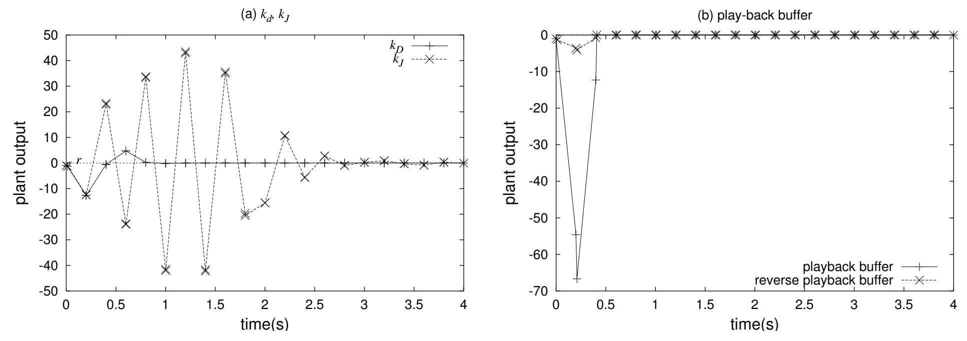

Figure 6: Plant output on the weatherhead (2004) trace for $a = 20$. The k_J controller operates in the absence of noise and uses a maximum likelihood estimate of $M_J(-a)$ collected from the trace prior to the simulation. The play-back controller is subject to disturbances ($D = e^{-a}$). Axis scales differ between charts.

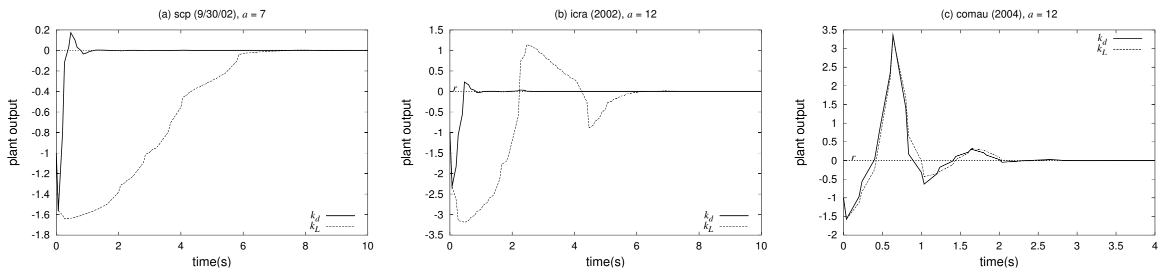

Figure 7: Plant output with the k_L and k_d controllers for $a \simeq a_L$ with no play-back buffer and no disturbances. Axis scales differs among charts.

that are not accounted for in the stochastic model, such as packet losses or the dynamic estimation of $M_J(-a)$. As for the weatherhead(2004) trace, the system output is shown in Figure 6 in the case of $a = 20 \simeq a_{J,\max} = 24.93$: the baseline controller k_d outperforms k_J both in terms of output variations and of settling times. The reason was that $k_J > k_d$, so that k_J had a larger value of $E[\beta_i]$, which in turn outweighed any advantage stemming from $E[\alpha_i] = 0$. Pronounced oscillations were also caused by the dynamic moment estimates and were less severe when a static maximum likelihood estimate was used instead. The regular play-back scheme directly snapped to the reference in most cases, but it can also introduce additional packet losses and cause x to differ from r, as demonstrated by the spike in Figure 6(b). Additionally, if the play-back delay is smaller, more packets are dropped and the plant output grew unboundedly. The reverse play-back does not introduce losses and therefore prevented significant deviations from the reference (Figure 6(b)), and substantially improved over stochastic control by making the control start time θ_i more predictable. In summary, k_J was not effective because it addresses only one term of the plant behavior equation (5). The reverse play-back mechanism was very effective in tolerating jitter without introducing losses. In the rest of the paper, all buffers will use a reverse play-back.

Loss Compensation The k_d controller is effective with a reverse play-back to tolerate jitter in the absence of packet losses, but its output varied widely in the case of heavier loss rates (e.g., spikes between $-11 \cdot 10^4$ and $5 \cdot 10^3$ in the scp(9/30/02) trace). Such large output variations would typically denote a catastrophe in the evolution of a real system. A different approach is to employ k_L to compensate for packet losses under the assumption that jitter is negligible. The traces with low loss rates have $k_L \simeq k_d$, and no significant difference was found on those traces between k_L and the baseline. In general, k_L differed more significantly from k_d on the traces with the higher loss rates. The highest loss rates were found in the scp(9/30/02) and icra(2002) traces (Figure 7). The k_d controller showed significantly better settling time and better overshoot. The reason is that k_d gets close to the reference quickly if there are no losses and little jitter and, in the absence of disturbances, it does not move from the reference for the rest of the simulation. Meanwhile, k_L is more conservative in an attempt to compensate for losses, which makes its convergence slower during periods of good connectivity. In the presence of disturbances ($D = e^{-2a}$), k_L was closer to the reference than k_d, but it still deviated significantly from it ($\pm 2 \cdot 10^5$), and so its behavior was dramatically unsatisfactory. Figure 7(c) compares k_L and k_d on the comau(2004) trace, which had the highest loss rate (1.71%) among all 2004 traces, and shows that k_L is comparable to k_d.

Loss and Jitter The controllers considered in the previous two paragraphs attempt to compensate either for jitter or for losses but not for both, and so they failed in the other case. Various approaches can address both losses and jitter. In the first place, the k_L and k_J controller can be combined into a k_S controller. However, $k_S = k_J$ if there are no losses (as in weatherhead(2004)) and $k_S \simeq k_L$ if jitter is negligible compared to losses (as in scp(9/30/2002)). Therefore, k_S inherited the same behaviors and problems as k_J and k_L in these traces. Another approach is to use k_L to compensate for losses and a reverse play-back to compensate for jitter. However, if jitter is negligible compared to losses, the controller basically reduces to pure k_L, inherited the same problems as k_L, and its settling time was actually slightly worse than the one of pure k_L.

Contingency Control Contingency control also attempts to address both jitter and losses. In particular, contingency control can improve over a play-back buffer by making the regular control end at a more predictable time. The k_c, k_L, and k_L (buffer) controllers had the best performance on the scp(9/30/02) trace according to a variety of metrics for $a = 7$ and $D = e^{-2a}$, and Figure 8 gives a snapshot of the corresponding output during an interval that contains the largest spike incurred by k_c. In this snapshot, the contingency control gave rise to only one spike of smaller amplitude, and it recovered more quickly from that spike. Figure 9 gives the complementary cumulative

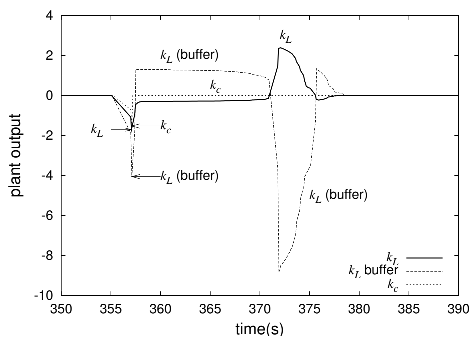

Figure 8: Snapshot of plant output with the k_c and k_L controllers with play-back buffer, $a = 7$, and $D = e^{-2a}$.

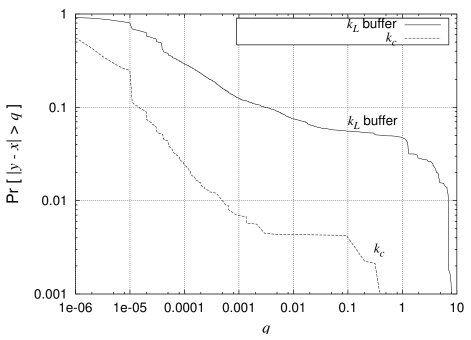

Figure 9: Complementary cumulative probability distribution of $|y(t) - r|$ in the scp(9/30/02) trace with $a = 7$ for the k_c and k_L controllers during the largest k_c spike after the first minute of the simulation.

probability distribution of $|y(t) - r|$ for the two controllers that had the best performance on this trace. In particular, the figure shows that contingency can improve performance significantly over a play-back buffer scheme on a lossy trace. The moment estimate was close to 1 throughout the simulation, and so there was no significant difference if the $M_J(a)$ term is ignored (i.e., set to $M_J(a) = 1$).

Trace Summary The simulations are summarized across all traces in Figure 10, which reports the computed value of $a_{\max}$ with a precision of ± 0.1. The contingency control almost always achieved the largest $a_{\max}$, and it was predictable with $a_{\max} \simeq 150$ across all traces. Reverse play-back especially helped on the traces that are dominated by jitter but have no losses. The contingency control improved over reverse play-back on the traces with higher loss rates. The contingency is effective if the $M_J(a)$ term is set to 1 throughout the simulation. Conversely, it did suffer if $M_J(a)$ is estimated dynamically because a moving average estimate was sensitive to the recent past, which was not always a good predictor of the immediate future.

Conclusions The experiments highlighted two factors. First, a reverse play-back typically has better performance than a pure play-back scheme because, intuitively, a late control signal is better than none. Second, k_c works better if the $M_J(a)$ term is ignored (i.e., $M_J(a) = 1$) in that $M_J(a) \simeq 1$ for most of the simulations and it is otherwise difficult to obtain a good predictive estimate of $M_J(a)$.

The main conclusion is that contingency control outperformed stochastic network compensation. An intuitive explanation can be expressed in the hunter analogy. Contingency control is analogous to a hunter that sees two flying targets (a timely signal and a packet loss) and shoots at each one of them. The stochastic control tries to catch two birds with one stone by aiming to the middle, and by doing so in fact misses both.

6 Multi-Dimensional Plants

The methodology of contingency control can be extended to multi-dimensional linear plants. The state vector evolves according to $\mathbf{x}_{i+1} = A\mathbf{x}_i + B\mathbf{u}_i$, where $\mathbf{x}$ is an n-dimensional state vector and A is an $n \times n$ matrix. Under certain controllability and observability assumptions, the state $\mathbf{x}$ can be set to 0 by a sequence of at most n control actions. The regular control contains the first of

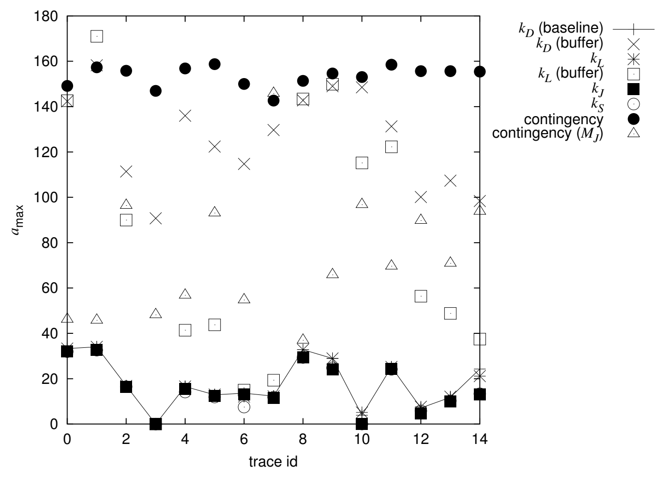

Figure 10: The $a_{\max}$ value expresses the order of magnitude of the plant speed that can be supported under various traces ($D = e^{-2a}$). The line joins the points corresponding to the baseline for better visibility.

these actions, and the contingency the remaining ones. When the contingency control is triggered, it would apply the remaining steps in this sequence of controls followed by $\mathbf{u} = 0$ until connectivity resumes. The intuition is that, in the absence of further control, the plant would continue the application of the last received plan, which would bring the plant output to the desired reference.

7 Related Work

Networked control is a relatively new field but already has a wealth of results [13] and is related to the area of real-time embedded distributed systems (e.g., [17]). The networked control model in Section 2 follows the original assumptions in which sensors are time-driven whereas controllers and actuators are event-driven [21]. The paper has focused on proportional controllers but, naturally, many other types of controllers exist [6, 22]. Co-simulation for co-design is a novel approach in which the design of the control system and the design of the network are tackled together, and therefore should be analyzed (and simulated) together [5]. A taxonomy of real-time applications introduces the concepts of tolerance and adaptiveness. Moreover, other real-time applications, such as VoIP or video streaming, also use end-point algorithms to achieve higher levels of tolerance or adaptiveness or both [24]. Since contingency control sets an expiration date to the regular control, it is also suitable for adoption in event-driven sampling, where a sensor sends a sample only when the output has changed by at least a certain threshold value [2, 11]. Event-driven sampling is particularly suitable for energy savings in wireless networks. However, event-driven sampling creates an ambiguity for the more traditional controllers, which cannot ascertain whether a sample was dropped or simply not sent. The ambiguity is alleviated in the case of contingency control due to the presence of a fail-safe action $u_i^{(c)}$. Previous work on optimal loss compensation mostly addresses i.i.d. losses [3], which are not relevant in a Gilbert model.

8 Conclusions

In this paper, we have formalized the problem of making networked control tolerant and adaptive to network vagaries. In addition to the specific results, a main contribution of this paper is to formulate networked control in a way that is cleaner and more suitable for further Networks research. The paper also introduced novel performance metrics that are natural and appropriate for networked control. The paper described several types of controllers. Stochastic controllers attempted to overcome network vagaries purely with control-theoretical arguments. At the other extreme, play-back buffers did not alter the controller as such, but rather introduced a mechanism to eliminate jitter. An integrated approach is to employ a set of methods collectively called contingency control: a reverse play-back buffer avoids negative jitter, an expiration time curbs the regular aggressive control, a contingency control deals with loss episodes, and an appropriately designed controller k_c takes into account all of these factors. An extensive evaluation on simulations and RTT traces showed that the combined approach was able to support much faster physical dynamics ($a_{\max} \simeq 150$) across a wide range of network conditions.

Acknowledgments

Work supported in part under NSF CCR-0329910. We thank Michael S. Branicky for many useful discussions.

References

[1] A. Al-Hammouri, A. Covitch, D. Rosas, M. Kose, W. S. Newman, and V. Liberatore. Compliant control and software agents for internet robotics. In *WORDS*, 2003.

[2] Karl Johan Åström and Bo Bernhardsson. Comparison of periodic and event based sampling for first-order stochastic systems. In *Preprints 14th World Congress of IFAC*, volume J, pages 301–306, Beijing, P.R. China, July 1999.

[3] Babak Azimi-Sadjadi. Stability of networked control systems in the presence of packet losses. In *CDC*, 2003.

[4] Lawrence S. Brakmo, Sean W. O'Malley, and Larry L. Peterson. TCP Vegas: new techniques for congestion detection and avoidance. In *SIGCOMM*, pages 24–35, 1994.

[5] M. S. Branicky, V. Liberatore, and S. Phillips. Co-simulation for co-design of networked control systems. In *American Control Conference*, 2003.

[6] Richard C. Dorf and Robert H. Bishop. *Modern Control Systems*. Prentice-Hall, Upper Saddle River, NJ, 2001.

[7] Thomas Dzubin. fping - a program to ping hosts in parallel. `http://www.fping.com`.

[8] Justin R. Hartman. Networked control systems co-simulation for co-design: Theory and experiments. Master's thesis, Case Western Reserve University, 2004.

[9] K. Jeffay, T. Hudson, and M. Parris. Beyond audio and video: Multimedia networking support for distributed, immersive virtual environments. In *Proceedings of the 27th EUROMICRO Conference*, pages 300–307, September 2001.

[10] Gunnar Karlsson, Henrik Lundqvist, and Igacio Más Ivar. Single-service quality differentiation. In *IWQoS*, pages 265–272, 2004.

[11] M. Lemkin et al. Velocity estimation from widely spaced encoder pulses. In *American Control Conference*, June 1995.

[12] V. Liberatore et al. IP communication and distributed agents for unmanned autonomous vehicles. In *AIAA-UAV*, 2003.

[13] Vincenzo Liberatore et al. Networked Control Systems Repository. `http://home.cwru.edu/ncs/`.

[14] D. Loguinov and H. Radha. Measurement study of low-bitrate Internet video streaming. In *IMW*, November 2001.

[15] L. Matthies et al. A portable, autonomous, urban reconnaissance robot. In *The 6th International Conference on Intelligent Autonomous Systems*, July 2000.

[16] A. Mukherjee. On the dynamics and significance of low frequency components of Internet load. *Internetworking: Research and Experience*, 5:163–205, December 1994.

[17] D. J. Musliner et al. CIRCA: A cooperative intelligent real-time control architecture. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(6), 1993.

[18] W. S. Newman et al. Design lessons for building agile manufacturing systems. *IEEE Trans. Robotics and Automation*, 16(3):228–238, June 2000.

[19] Marco L. Ngai, Vincenzo Liberatore, and Wyatt S. Newman. An experiment in remote robotics. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2190–2195, 2002.

[20] Kathleen Nichols, Van Jacobson, and Kedarnath Poduri. A per-domain behavior for circuit emulation in IP networks. *CCR*, April 2004.

[21] J. Nilsson. *Real-Time Control Systems with Delays.* PhD thesis, Lund Institute of Technology, 1998.

[22] Katsuhiko Ogata. *Modern Control Engineering.* Prentice-Hall, Upper Saddle River, NJ, third edition, 1997.

[23] B. Penaflor. Current status of DIII-D realtime digital plasma control. *IEEE Transactions on Nuclear Science*, 47(2):201–204, 2000.

[24] Larry L. Peterson and Bruce S. Davie. *Computer Networks.* Morgan Kaufmann, 2000.

[25] B. P. Robinson and V. Liberatore. On the impact of bursty cross-traffic on distributed real-time process control. In *Workshop on Factory Communication Systems (WFCS)*, 2004.

[26] W. Richard Stevens. *TCP/IP Illustrated*, volume 1. Addison-Wesley.

[27] Amgad Zeitoun, Zhiheng Wang, and Sugih Jamin. RTTometer: Measuring path minimum RTT with confidence. In *IEEE Workshop on IP Operations and Management (IPOM 2003)*, October 2003.

[28] Wei Zhang. *Stability Analysis of Networked Control Systems*. PhD thesis, Case Western Reserve, August 2001.

[29] Yin Zhang, Nick Duffield, Vern Paxson, and Scott Shenker. On the constancy of Internet path properties. In *IMW*, 2001.

A Plant Dynamics

We integrate the differential equation (4) with a boundary condition on $x(\theta_i)$ to obtain

$$x(t) = \left(x(\theta_i) - \frac{bkx(\theta_i - \xi_i)}{a}\right) e^{a(t-\theta_i)} + \frac{bkx(\theta_i - \xi_i)}{a} \, . \tag{7}$$

In particular,

$$\begin{aligned} x(\theta_{i+1} - \xi_{i+1}) &= \left(x(\theta_i) - \frac{bkx(\theta_i - \xi_i)}{a}\right) e^{a(\theta_{i+1}-\theta_i-\xi_{i+1})} + \frac{bkx(\theta_i - \xi_i)}{a} \\ x(\theta_{i+1}) &= \left(x(\theta_i) - \frac{bkx(\theta_i - \xi_i)}{a}\right) e^{a(\theta_{i+1}-\theta_i)} + \frac{bkx(\theta_i - \xi_i)}{a} \, . \end{aligned}$$

Define $\hat{x}_i = x(\theta_i)$. Observe that

$$\hat{x}_{i+1} = e^{a\xi_{i+1}} x_{i+1} + \frac{bkx_i}{a}\left(1 - e^{a\xi_{i+1}}\right) \, .$$

Hence, if $x_i = x_{i+1} = 0$, then also $\hat{x}_{i+1} = 0$, and, by Equation (7), $x(t) = 0$ for all $t \geq \theta_{i+1}$. Observe that $\Delta t_i = t_{i+1} - t_i = \theta_{i+1} - \theta_i - \xi_{i+1} + \xi_i$ and $\Delta s_i = \theta_{i+1} - \theta_i - \xi_{i+1}$. We now have

$$\begin{aligned} x_{i+1} &= \left(e^{a\xi_i} x_i + \left(1 - e^{a\xi_i}\right)\frac{bkx_{i-1}}{a} - \frac{bkx_i}{a}\right) e^{a(\theta_{i+1}-\theta_i-\xi_{i+1})} + \frac{bkx_i}{a} \\ &= \frac{bk(x_{i-1} - x_i)}{a} e^{a\Delta s_i} + \left(x_i - \frac{bkx_{i-1}}{a}\right) e^{a\Delta t_i} + \frac{bkx_i}{a} \\ &= \left(e^{a\Delta t_i} - \frac{bk}{a} e^{a\Delta s_i} + \frac{bk}{a}\right) x_i + \frac{bk}{a}\left(e^{a\Delta s_i} - e^{a\Delta t_i}\right) x_{i-1} \\ &= \alpha_i x_i + \beta_i x_{i-1} \, , \end{aligned}$$

which is the homogeneous recurrence (5).

B Deterministic Deadbeat Control

An approach similar to the baseline can be used also when $\beta_i \neq 0$ is a constant. In this case, the jitter ξ_i is also constant, and we let $\xi_i = \xi$ for all i's. The intuitive interpretation is that either the constant RTT τ was estimated with an error or, as in [21, 28], the controller was designed for local operation but is being used even when delays are present in the feedback loop. In this case, the condition $\alpha_i = 0$ leads to

$$k'_d = \frac{a}{b} \frac{e^{aT}}{e^{a(T-\xi)} - 1} \, .$$

Proposition 3. *In the case of no losses, $\xi > 0$ and $k = k'_d$, the plant state converges to the reference $r = 0$ if and only if*

$$\xi < T - \frac{1}{a} \ln \frac{1 + e^{2aT}}{1 + e^{aT}} \, .$$

Proof. By design, k'_d implies $\alpha_i = 0$, so that $x_{i+1} = \beta x_{i-1}$. The characteristic equation of this recurrence is $\rho^2 = \beta$, and so the amplitude of x_i decreases with time if and only if $|\beta| < 1$. If $\xi > 0$, then

$$|\beta| = \frac{bk'_d}{a}\left(e^{aT} - e^{a(T-\xi)}\right) = \frac{e^{aT}}{e^{a(T-\xi)} - 1} e^{aT}\left(1 - e^{-a\xi}\right) = \frac{e^{2aT}}{e^{a(T-\xi)} - 1}\left(1 - e^{-a\xi}\right) \, .$$

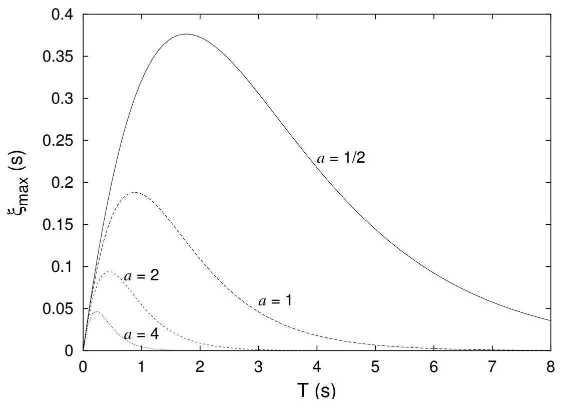

Figure 11: The bound $\xi_{\max}$ on the jitter ξ as a function of the sampling period T (Proposition 3).

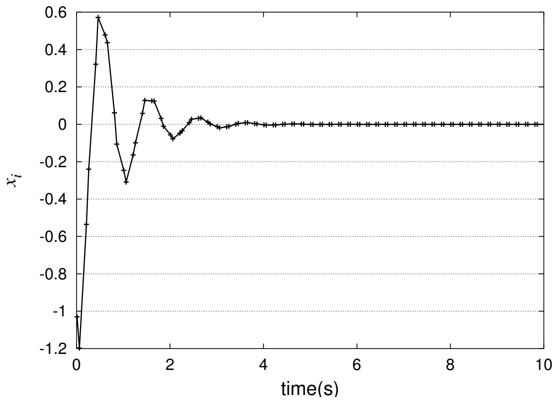

Figure 12: A case with jitter $\xi \neq 0$ that leads to a damped oscillation of the plant output.

Therefore, $|\beta| < 1$ occurs if and only if $e^{2aT}\left(1 - e^{-a\xi}\right) < e^{a(T-\xi)} - 1$, or $e^{-a\xi} > (1 + e^{2aT})/(e^{aT} + e^{2aT})$. The claim follows by taking the logarithm of both sides. $\square$

Figure 11 shows the upper bound on ξ as a function of T for various values of a. In general, as the product aT increases, $\ln\left(1 + e^{2aT}\right) / \left(1 + e^{aT}\right) \sim aT$, and so little jitter can be tolerated with faster plants or slow sampling rates.

Corollary 4. *If $\xi > 0$ (undercompensation of delays), an oscillation is present in the plant output.*

Proof. If $\xi > 0$, then $\beta < 0$. The corollary follows from the characteristics equation derived in the proof of the previous result. $\square$

Figure 12 shows the case $T = 0.2$, $\xi = 0.05$, $k'_d \simeq 6.88$, which leads to a damped oscillation in the plant output.

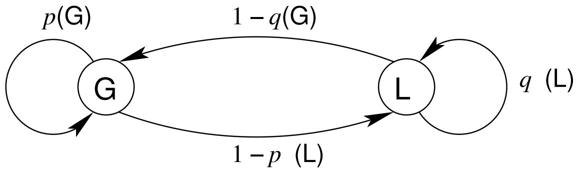

Figure 13: The Gilbert model for packet losses.

C I.i.d. Losses

If the sampling period is uniform and of unit length and the loss probability is $p < e^{-a}$, then Δt_i is a geometric random variable with parameter $1-p$. Then,

$$M_T(a) = \frac{(1-p)e^a}{1-pe^a} \ ,$$

and the difference

$$M_T(a) - e^a = pe^a \frac{e^a - 1}{1 - pe^a} > 0$$

is maximized by taking the derivative with respect to p. However, the derivative is $e^a(e^a-1)/(1-pe^a)^2$, which is always positive, and thus p should be taken as large as possible.

D Gilbert Model

The Gilbert model [29] (Figure 13) is a two-state Markov chain. The two states are labeled G (the "good" state) and L (the "lossy" state). Let p be the probability of remaining in the G state and q the probability of remaining in the L state. The chain generates a G or an L upon each state transition, where the jth symbol determines whether the control packet at step j was received at the plant (G) or not (L). For example, the string $GGLLG\ldots$ denotes losses at step 3 and 4, but good reception during the other steps. Assume even sampling and assume that time is normalized so that the sampling period is 1. Then, Δt_i is the length of the ith maximal subsequence of the form L^*G. For example, the string $GGLLG\ldots$ gives $\Delta t_i = \Delta t_2 = 1$ and $\Delta t_3 = 3$. At the end of each subsequence L^*G, the Markov chain is in the G state. Therefore, at the beginning of the next subsequence, the state is G with probability p and L with probability $1-p$. If the initial state is G, then $\Delta t_i = 1$. If the initial state is L, then $\Delta t_i = 1 + G$, where G is a geometric random variable with mean $1/(1-q)$. Therefore,

$$\overline{T} = E[\Delta t_i] = p + (1-p)\left(1 + \frac{1}{1-q}\right) = \frac{2-p-q}{1-q} \ ,$$

and, if $q < e^{-a}$

$$M_T(a) = E[e^{a\Delta t_i}] = p + (1-p)E[e^{a(G+1)}] = p + (1-p)e^a E[e^{aG}] = p + (1-p)(1-q)\frac{e^{2a}}{1-qe^a} \ .$$

Therefore,

$$\begin{aligned} k_L &= \frac{a}{b}\left(1+\frac{1}{M_T(a)-1}\right) \\ &= \frac{a}{b}\left(1+\frac{1-qe^a}{p-pqe^a+(1-p)(1-q)e^{2a}}\right) \\ &= \frac{a}{b}\frac{(1-p)(1-q)e^{2a}-q(1+p)e^a+(1+p)}{(1-p)(1-q)e^{2a}-pqe^a+p} \; . \end{aligned}$$

E Effective Sampling Rate

The gain k_L is a function of $M_T(a)$, which in turn can be expressed as a McLaurin series of its derivatives $M^{(n)}(0) = E[(\Delta t_i)^n]$ calculated at the origin. Thus, k_L accounts for all of the moments of the distribution of Δt_i. However, if $M_T(a) \simeq M_T(0) + aM_T'(0) = 1 + a\overline{T}$, then

$$k_L \simeq k_L' = \frac{1}{b}\left(a+\frac{1}{\overline{T}}\right) \; .$$

The error in the approximation of $M_T(a)$ is $a^2M''(\hat{a})/2 = a^2E[(\Delta t_i)^2 e^{\hat{a}\Delta t_i}]/2 \geq 0$ for some $0 < \hat{a} < a$, and so $k_L' \geq k_L$. The control k_L' has the advantage of depending only on the effective sampling rate $1/\overline{T}$ rather than on the entire distribution of Δt_i. However, k_L' can only be applied if the error is small. Furthermore, k_L' can take a value $k_L' > k_1$ even when k_L does not exist. For example, in the case of an exponential distribution, $k_L' = (a+\lambda)/b$ even when $\lambda < a$. The effective sampling period $\overline{T}$ was the only factor considered in previous work [8, 28], with conclusions that were often difficult to explain analytically.

F Jitter

Proof of Proposition 1. We first claim that α_i is independent of x_i. The quantity α_i is a function of constant terms and of the random variable ξ_i, which represents the jitter or the time difference between $t_i + \tau$ and θ_i. Meanwhile, x_i is the state at time $t_i + \tau$ and thus it is not affected by a future jitter ξ_i. By the same token, β_i is independent of x_{i-1}. Therefore, $E[x_{i+1}] = E[\alpha_i]E[x_i] + E[\beta_i]E[x_{i-1}] = E[\beta_i]E[x_{i-1}]$. Moreover, since the ξ_i's are i.i.d random variables, so are the β_i's, and so

$$\lim_{i\to\infty} x_{2i+1} = \lim_{i\to\infty} x_1 E\left[\prod_{j=1}^{i}\beta_{2j}\right] = \lim_{i\to\infty} x_1 \prod_{j=1}^{i} E[\beta_{2j}]$$

converges to $r = 0$ if and only if $|E[\beta_{2j}]| < 1$. Since the β_i's are identically distributed, the same condition also guarantees that $\lim_{i\to\infty} x_{2i} = 0$. We now have that

$$\begin{aligned} E[\beta_i] &= E\left[\frac{bk_J}{a}\left(e^{a(T-\xi)}-e^{aT}\right)\right] \\ &= \frac{e^{aT}}{e^{aT}M_J(-a)-1}e^{aT}E[e^{-a\xi_i}-1] \\ &= \frac{e^{2aT}}{e^{aT}M_J(-a)-1}(M_J(-a)-1) \; . \end{aligned}$$

Therefore, $E[\beta_i] > -1$ if and only if $M_J(-a) > (e^{2aT}+1)/(e^{2aT}+e^{aT})$, which proves the claim. $\square$

Proof of Corollary 2. It can be easily verified that $e^{aT}M_J(-a) - 1 > 0$ when $M_J(-a) > (e^{2aT} + 1)/(e^{2aT} + e^{aT})$. □

This section gives a sufficient condition for convergence in the presence of jitter.

Lemma 5. *If* $|\beta_i| \leq \beta < 1$, *then* $\lim_{i\to\infty} E[x_i] = 0$.

Proof. We have that $|x_{i+1}| = |\beta_i||x_{i-1}| \leq \beta|x_{i-1}|$, so that $E[|x_{i+1}|] \leq \beta E[|x_{i-1}|]$. Therefore, $\lim_{i\to\infty} E[x_i] = 0$ if $\beta < 1$. □

In turn, such value of k_T and the condition that $|\beta| < 1$ will effectively put a bound on ξ_i and T.

Proposition 6. *If there is a* $\beta < 1$ *with*

$$0 \leq \xi_i \leq -\frac{1}{a}\ln\left(1 - \beta\frac{e^{aT}M_J(-a) - 1}{e^{2aT}}\right) \qquad (i \geq 1)\ ,$$

then x_i *converges to the reference* r *in the expectation.*

Proof. The condition on ξ_i ensures that $-1 \leq \beta \leq \beta_i \leq 0$, and the proposition follows from the previous result. □

The proposition states that larger jitter ξ_i can be tolerated with shorter values of T.

G Contingency Control

It remains to establish the value of k, for which we first need to determine the plant dynamics.

Consider an interval $[\zeta_{i-1}, \theta_i)$. Since $u_i^{(c)} = 0$ is applied continuously in this interval, the plant is described by the equation $\dot{x}(t) = ax(t)$, which is integrated to give $x(t) = x(t_i + \tau)e^{a(t-t_i-\tau)}$ for all $t \in [\zeta_i, \theta_i)$. In particular, $x(\theta_i) = x(t_i + \tau)e^{a\xi_i}$. In the interval $[\theta_i, \zeta_i)$, the plant is described by the equation $\dot{x}(t) = ax(t) - bkx(t_i + \tau)$, which integrates to

$$x(t) = \left(x(\theta_i) - \frac{bkx(t_i+\tau)}{a}\right)e^{a(t-\theta_i)} + \frac{bkx(t_i+\tau)}{a} = x(t_i+\tau)\left(e^{a(t+\xi_i-\theta_i)} - \frac{bk}{a}\left(e^{a(t-\theta_i)} - 1\right)\right)\ .$$

Therefore,

$$x(\zeta_i) = x(t_i+\tau)\left(e^{a(T-\tau)} - \frac{bk}{a}\left(e^{a(T-\tau-\xi_i)} - 1\right)\right)\ ,$$

and

$$E[x(\zeta_i)] = x(t_i+\tau)\left(e^{a(T-\tau)} - \frac{bk}{a}\left(e^{a(T-\tau)}M_J(-a) - 1\right)\right)\ ,$$

which can be set to 0 by choosing k equal to

$$k_c = \frac{a}{b}\frac{e^{a(T-\tau)}}{e^{a(T-\tau)}M_J(-a) - 1}\ .$$

Suppose that $\xi_i = 0$ and that $x(t_i + \tau)$ is estimated with an error, due either to disturbances or to an imprecise system model. Then, the control u_i becomes $u_i = -k_c(1+\epsilon)x(t_i+\tau)$ for some error term ϵ. The controller k_c makes $|x(\zeta_i)| < x(t_i + \tau)$ if and only if $-e^{a(T-\tau)} < \epsilon < e^{a(T-\tau)}$, which puts a bound on the combination of error, plant speed, and sampling period.

Model	Parameter	Values
Plant	a	3
	b	1
	$\overline{T}$	200ms
Network	τ	10ms
i.i.d. loss	p	$0.01, 0.02, 0.05$ $0.1, 0.15, 0.2, 0.5, 0.6$
Exponential loss	λ	5
Gilbert loss	p	$0.99, 0.98, 0.95,$ $0.9, 0.8, 0.5, 0.4$
	q	$0, 0.1, 0.2, 0.5$
Gamma jitter	λ	200, 500, 1000
	r	1, 2, 3

Table 3: Parameters of systems and networks in simulation based on probabilistic network models.

H Methodology

H.1 Simulations

In the first set of simulations, packet losses and delays were generated from various probabilistic models. For example, one such model would assume that there is no jitter and packet losses are generated with a Gilbert model, as in an example in Section 3.3.1. In general, all probabilitstic models in Section 3 were simulated (Table 3). Model-based simulations can be used to assess the accuracy of the dynamic estimators defined in Section 3.3.4. The estimators should induce a plant output that is close to the output obtained with a controller that uses the exact value of the moment generating function for the same probabilistic model. The evaluation of the estimators starts with an initial unit value for the moments, so that the stochastic compensation is initially disabled. Thus, the simulation tested implicitly the simultaneous convergence of the plant and of the moment estimator.

H.2 Bounds on a

The distribution parameters λ and r are given in Table 4 and 5 and were estimated as $\lambda = \mu/\sigma_e^2$, and $r = \lambda\mu$, where μ is the average observed jitter and σ_e^2 is the sample variance of the jitter.

The bounds on a are predicted from the probabilistic models after they were fit to the traces and, as such, the bounds could differ from those obtained in a complete simulation. In particular, M_J is the moment generating function of the gamma distribution that was fit to the body of the jitter distribution.

I Probabilistic Evaluation

Figure 14 is an example of certain effects that are especially visible with high loss rates. The figure shows the plant output for $p = q = 0.5$ (i.e., an overall drop rate of 50%). The value of q is close to the critical threshold $q < e^{-aT} \simeq 0.55$, which is in the range where k_L and k_d differ more significantly. In the first place, the estimator is substantially worse than the exact $M_T(a)$, which follows the general trend for these large drop rate, as discussed above. The figure also compares the k_L and k_d controllers. The k_L controller has lower overshoot but larger rise time than k_d. The

Name	a_L	$a_L^{(G)}$	λ	r	$a_{J,\min}$	$a_{J,\max}$
wopr.cwru.edu	∞	∞	70557.3	2.5	1460	33.6514
weatherhead.cwru.edu	∞	∞	52324.9	1.83	1485	33.7086
scp.grc.nasa.gov (Oct 2)	27.15	14.98	1575.3	3.00	27.14	16.9387
scp.grc.nasa.gov (Sep 30)	6.9	6.49	198.62	1.03	10.11	13.0512
www.ri.cmu.edu	15.65	12.55	1114.7	1.74	33.25	17.7509
www.comau.com	22.05	14.54	329.15	1.44	11.94	13.6788
www.icra-iros.com	12.7	6.44	945	1.28	38.5	18.3321
control-lab.et.tudelft.nl	13.25	12.7	309.71	1.28	12.71	13.9109

Table 4: Statistical and control-theoretical characteristics of the collected traces (Year 2002).

Name	a_L	$a_L^{(G)}$	λ	r	$a_{J,\min}$	$a_{J,\max}$
wopr.cwru.edu	∞	∞	27952.4	1.05	1550	33.3984
weatherhead.cwru.edu	∞	∞	356.7	0.10	1.81	24.9346
scp.grc.nasa.gov	28.88	13.20	46.5	1.39	3.44	6.1942
www.ri.cmu.edu	42.59	∞	2652.0	0.73	101.3	24.8884
www.comau.com	20.34	21.72	75.9	1.10	4.41	9.0900
www.icra-iros.com	20.90	∞	59.4	0.25	0.78	14.1837
control-lab.et.tudelft.nl	39.12	∞	91.9	0.06	0.30	21.7209

Table 5: Statistical and control-theoretical characteristics of the collected traces (Year 2004).

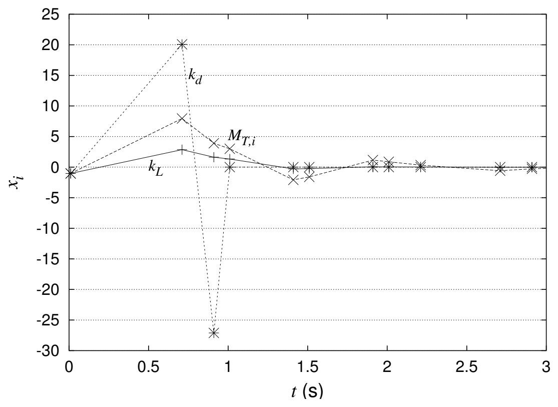

Figure 14: Plant output for a network model with no jitter and losses generated by a Gilbert model with $p = q = 0.5$.

reason is that the k_d controller is designed for the ideal case of no losses and it severly suffers during periods of poor connectivity, so that it swings widely above and below the reference. However, the controller immediately snaps back to the reference as soon as connectivity resumes. By contrast, the k_L controller attempts to keep a behavior that balances between the periods of poor and good connectivity. The relative behavior of k_d and k_L basically held for all values of p and q, even though the stochastic nature of the network model occasionally resulted in anomalous outcomes with low probability. At any rate, the behavior of k_d and k_L justifies intuitively contingency control, in that contingency attempts to combine the good features of the former during good connectivity periods and of the latter during bad connectivity. Finally, the estimator $M_{T,i}$ had lower overshoot than k_d even though estimators were not particularly accurate at these high drop rates.

J Evaluation

J.1 Probabilistic Network Models

The first set of simulations involves the generation of jitter and packet losses according to the various probabilistic models. The simulation confirmed the importance of setting the correct value for the gain k: for example, Figure 15 shows that a 10% loss rate with $k = k_L$ has comparable performance to the case when no packet is lost but k is not exactly equal to k_d. The dynamic estimation of $M_T(a)$ produced results comparable to those obtained with a prior knowledge of $M_T(a)$. For example, dynamic estimation increased the overshoot by less than 1% in a simulation with a 20% i.i.d. loss probability. The plant output deteriorated more significantly only for higher loss rates. Similarly, the dynamic estimation of $M_J(-a)$ led to overshoot within 1% of the exact value, and the estimate of $M_S(a)$ gave rise to behaviors analogous to those caused by a dynamic $M_T(a)$. Further detailed are given in Appendix I.

J.2 Contingency Control

In summary, the scp(9/30/02) trace with $a = 7$ and $D = e^{-2a}$ led to:

- The baseline k_d swang between $\pm 9 \cdot 10^5$.

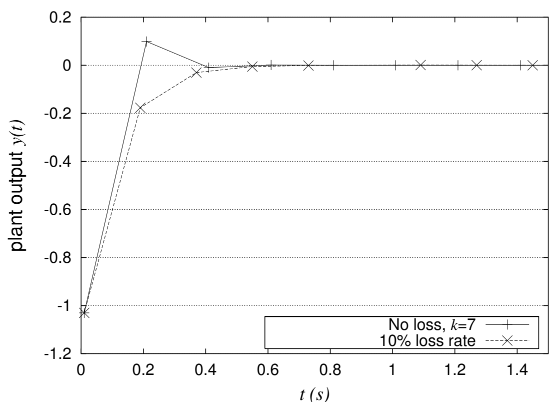

Figure 15: Convergence with no jitter for the case of no losses ($k = 7 \neq k_d = 6.47$) and of a constant loss probabilty of 10% ($k = k_L$).

- The contingency controller had 4 short spikes through a 15 minute simulation and the spike intensity ranged as $-1.6 \leq y \leq 0.7$. Its estimate of $M_J(-a)$ was close to 1 and no significant differences were introduced if the term was set to 1 throughout the simulation.
- The k_d controller had more spikes of large intensity (from $-11 \cdot 10^4$ to $5 \cdot 10^3$) even with the addition of a reverse play-back.
- The k_L controller (no play-back buffer) had significant spikes ($\pm 2 \cdot 10^5$), its rise time was 30 times longer than k_c (6 seconds versus 0.2 seconds), and it typically took longer to recover from a spike, as shown for example in Figure 8.
- The k_L controller improves with the addition of a play-back buffer: it had a few spikes of intensity between -8 and 5. However, its behavior during these spikes resembled that of k_L with no buffer (Figure 8), and its rise time was longer than k_L (7 seconds).