MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

A smart actuation architecture for wireless networked control systems

Ma, Y.; Wang, Y.; Di Cairano, S.; Koike-Akino, T.; Guo, J.; Orlik, P.V.; Lu, C.

TR2018-177 December 29, 2018

Abstract

Along with the forth industry revolution, implementing industrial control systems over mainstream wireless networks such as WirelessHART, WiFi, and cellular networks becomes necessary. Well-known challenges, such as uncertain time delays and packet drops, induced by networks have been intensively investigated from various perspectives: control synthesis, network design, or control and network co-design. The status quo is that industry remains hesitant to close the loop at the control-to-actuation side due to safety concerns. This work offers an alternative perspective to address the safety concern, by exploiting the design freedom of system architecture. Specifically, we present a smart actuation architecture, which deploys (1) a remote controller, which communicates with physical plant via wireless network, accounting for optimality, adaptation, and constraints by conducting computationally expensive operations; (2) a smart actuator, which co-locates with the physical plant, executing a local control policy and accounting for system safety in the view of network imperfections. Both the remote and the local controllers run at the same time scale and cooperate through an unreliable network. We propose a policy iteration-based procedure to co-design the local and remote controllers when the latter employs the model predictive control policy. Semi-global asymptotic stability of the resulting closed-loop system can be established for certain classes of plants. Extensive simulations demonstrate the advantages of the proposed architecture and co-design procedure.

IEEE Conference on Decision and Control (CDC)

A smart actuation architecture for wireless networked control systems

Yehan Ma, Yebin Wang, Stefano Di Cairano, Toshiaki Koike-Akino, Jianlin Guo, Philip Orlik, and Chenyang Lu

***Abstract*— Along with the forth industry revolution, implementing industrial control systems over mainstream wireless networks such as WirelessHART, WiFi, and cellular networks becomes necessary. Well-known challenges, such as uncertain time delays and packet drops, induced by networks have been intensively investigated from various perspectives: control synthesis, network design, or control and network co-design. The status quo is that industry remains hesitant to close the loop at the control-to-actuation side due to safety concerns. This work offers an alternative perspective to address the safety concern, by exploiting the design freedom of system architecture. Specifically, we present a *smart actuation* architecture, which deploys (1) a *remote controller*, which communicates with physical plant via wireless network, accounting for optimality, adaptation, and constraints by conducting computationally expensive operations; (2) a *smart actuator*, which co-locates with the physical plant, executing a local control policy and accounting for system safety in the view of network imperfections. Both the remote and the local controllers run at the same time scale and cooperate through an unreliable network. We propose a policy iteration-based procedure to co-design the local and remote controllers when the latter employs the model predictive control policy. Semi-global asymptotic stability of the resulting closed-loop system can be established for certain classes of plants. Extensive simulations demonstrate the advantages of the proposed architecture and co-design procedure.**

I. Introduction

Thanks to their flexibility and low cost, the past decade has witnessed sustained interest in exploring wireless networked control systems (WNCS) and expanding their applications over industry processes, unmanned aerial vehicles, and teleoperated robots. Within the foreseeable future, the WNCS is expected to rapidly penetrate into the next generation of industrial applications, including autonomous warehouses and smart factories [1].

WNCSs face serious challenges due to the inherent indeterminism and limited throughput of wireless networks. They have spawned a variety of research directions in both network and control communities. On the network side, the adoptions of wireless sensor and actuator networks (WSANs) are accelerated by wireless standards tailored for industrial automation, such as ISA100, WirelessHART, and ZigBee. Other approaches to address indeterminism include coding [2], retransmissions and channel selection [3], [4], routing [5], and reachability-aware scheduling [6], etc. However, the unpredictable wireless conditions of an industrial WSAN mean that the reliability of the wireless network cannot be guaranteed, leading to unsafe control performance. Another important research direction is rooted in control theory to improve the systems' resiliency to network imperfections. A plethora of control designs have been performed based on the models of the original plant as well as on network parameters. To name a few, Sinopoli et al. [7] discuss Kalman filtering with intermittent measurement; Gao et al. [8] investigate robust output tracking control subject to the time delay between controllers and actuators; Wang et al. [9]–[11] model packet loss as a Bernoulli or Markov-type process and establish stochastic stability of the resultant WNCS. More recently, network and control co-design has been explored to jointly determine the control and network polices to attenuate the effects of uncertainties and limited throughput, for example, [12] on network QoS-aware adaptive control, [13] on co-design sampling periods and network throughput, [14], [15] on control and network channel allocation, [16], [17] on control and network scheduling policy, and [18] on control and network power policy, etc. Interested readers are referred to [19]–[23] for more details.

Most of the aforementioned works assume that the WNCS admits a *direct* architecture [21], where the controller and plants communicate through the wireless network. With sensing and actuation signals transmitted over an unreliable network, the key issue is how to ensure the closed-loop system safety (stability). Due to the stochastic nature of network models, most existing works either strive to obtain guaranteed stability (albeit with conservative designs) or stochastic stability, which has the drawback of being unsatisfactory to industrial practitioners. Alternatively, a *hierarchical* architecture is widely adopted in industrial WNCS, where local (lower level) controllers fulfill stabilization and tracking of local control loops, and a remote controller supervises local controllers over a unreliable network. This architecture possesses two main characteristics: the remote controller typically runs much slower than local control loops; and the remote controller provides reference inputs to local control loops for optimal process operation [21]. By decoupling the unreliable network from local control loops, stability analysis of the entire closed-loop system is relatively straightforward. In addition to its higher cost, another shortcoming of the hierarchical architecture is that the remote controller has insufficient authority to shape the transient of the closed-loop control system, and thus might cause performance loss.

This paper aims to reconcile the safety, optimality, and cost

This work was done while Y. Ma was an intern with Mitsubishi Electric Research Laboratories, 201 Broadway, Cambridge, MA 02139, USA. Y. Ma and C. Lu are with Department of Computer Science and Engineering, Washington University in St. Louis, St. Louis, MO 63130, USA. {yehan.ma,lu}@wustl.edu

Y. Wang, S. Di Cairano, T. Koike-Akino, J. Guo, and P. Orlik are with Mitsubishi Electric Research Laboratories, 201 Broadway, Cambridge, MA 02139, USA. {yebinwang,dicairano,kioke,guo,porlik}@merl.com

of the WNCS by exploiting the design freedom of system architectures. Main contributions of this paper are four-fold:

(i) propose a *smart actuation* architecture that combines features of direct and hierarchical architectures.
(ii) present a procedure of co-designing remote and local controllers in the case that the remote controller implements a model predictive control (MPC) policy.
(iii) establish stability of the resultant closed-loop system.
(iv) demonstrate the effectiveness of the *smart actuation* architecture in both reliable and unreliable networks through extensive simulations.

The rest of the paper is organized as follows: Section II introduces existing architectures of the WNCS and proposes the *smart actuation* architecture. Section III presents a procedure to co-design remote and local controllers and establishes the closed-loop system stability when the remote controller implements the MPC policy. Section IV provides evaluation results of the proposed architecture and co-design procedure. Conclusions and future work are stated in Section V.

II. System Architecture

This section first briefly describes two prevailing architectures, and then the *smart actuation* architecture is proposed and discussed. Interested readers are referred to [20], [21] for a comprehensive coverage on existing architectures.

A. Existing Architectures

Fig. 1 illustrates a typical schematic of *direct* architecture. Sensors transmit the measurements $y(k)$ to the remote controller via the network; and the remote controller transmits control inputs $u(k)$ to actuators the network. Although enjoying miscellaneous advantages, this architecture suffers from one notable weakness: sophisticated control and network design to ensure the closed-loop system stability.

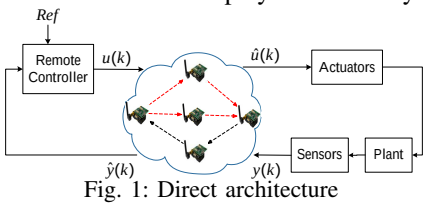


Fig. 1: Direct architecture

Fig. 2 outlines *hierarchical* architecture. The remote controller supervises local control loops by specifying their references, denoted by $y^*(k)$. The local controller generates control inputs $u(k')$ to actuators. Here we use k and k' to suggest that remote and local controllers have distinct sample rates. Since the remote controller typically runs at a much slower pace than the local one, the time scale separation principle can be applied. Thus the local controller mostly accounts for the closed-loop system stability. Thanks to its scalability and reliability [21], hierarchical architecture has been found in many industrial applications such as distributed control systems for process automation, mobile robots control systems [12], and smart power grids [24]. Compared with direct architecture, hierarchical architecture enhances reliability but increases the system cost, through such expenses as extra installation, maintenance, and cabling.

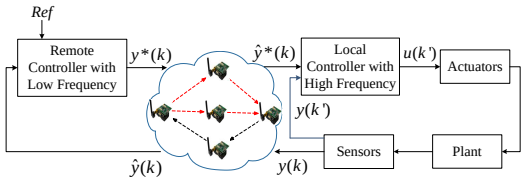


Fig. 2: Hierarchical architecture

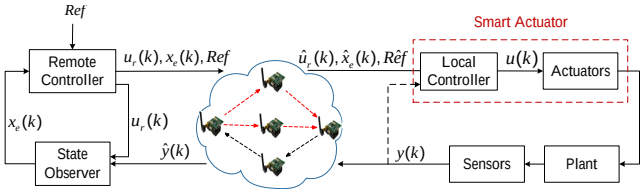


Fig. 3: Smart actuation architecture

In addition, the remote controller has insufficient authority to shape the transient of the closed-loop system.

B. Smart Actuation Architecture

Our *smart actuation* architecture is shown in Fig. 3. At time step k, the remote controller determines control input $u_r(k)$ based on the estimated state $x_e(k)$ and references, and then sends these signals to the local controller; the local controller can generate the local control input $u_l(k)$ by adopting local control law $u_l(k) = h(x(k))$. The local controller passes on either the remote control input $\hat{u}_r(k)$ from network or its own control input $u_l(k)$, depending on whether the actuation packet is delivered on time. That is,

$$u(k) = \begin{cases} \hat{u}_r(k), & \text{packet delivered,} \\ u_l(k), & \text{otherwise,} \end{cases} \tag{1}$$

where $u(k)$ is the control command to actuators. The remote controller adopts policies tackling optimality, uncertainties, and constraints, e.g. MPC, reinforcement learning, and adaptive dynamic programming. The local controller implements computationally lightweight policies for system stability.

For example, consider a nonlinear discrete-time system,

$$x(k+1) = f(x(k), u(k)),$$

where $x \in \mathbb{R}^n$ is the state and $u \in \mathbb{R}^m$ the control. The local controller implements a control policy as follows:

$$x_l(k+1) = \begin{cases} f(\hat{x}_e(k), u(k)), & \text{packet delivered,} \\ f(x_l(k), u(k)), & \text{otherwise,} \end{cases} \tag{2}$$

$$u_l(k+1) = h(x_l(k+1)).$$

If the actuation packet arrives on time, $u(k) = \hat{u}_r(k)$ according to (1), and $\hat{x}_e(k)$ is the estimated state of $x_e(k)$ received from the network. On the other hand, if the actuation packet is lost, $u(k) = u_l(k)$ according to (1), and $x_l(k)$ is the predicted state based on local policy (2) at time step $k-1$. Please note that (1) is an example of the switch rule. The switch rule should depend on remote control policy in order to establish stability. For instance, with MPC as the remote controller, a tailored switch mechanism, designed to establish stability, is described in detail in Sec. III-B.

Providing control input to actuators directly, the remote controller has sufficient authority to shape the closed-loop system performance. However, this treatment may lead to the dilemma encountered in the direct architecture: unsatisfactory stability and sophisticated control design. The local

controller is therefore introduced to lift this concern. This idea is consistent with results in [25], where the optimal location of controllers is investigated. It concludes that controllers should be collocated with the actuator when packets are allowed to be infinity long.

Remark 2.1: The signal flow from sensors to the local controller, represented by the dashed line in Fig. 3, is optional. With this flow, the *smart actuation* architecture is similar with the hierarchical one, except that the remote controllers of the two play different roles, and its local and remote controllers adopt the same time scale. Without this flow, it is similar to direct architecture, except that *smart actuation* architecture has local controllers in place. □

III. Smart Actuation for Stability and Performance

The closed-loop system corresponding to the *smart actuation* architecture is hybrid. Specifically it can be regarded as a switched system arbitrarily triggered by the event designating actuation packet loss. Stability analysis tools for hybrid systems can be found in [26] and references therein. This work performs stability analysis and control design based on a well-received result: if there exists a common Lyapunov function for all subsystems, then the stability of the switched system is guaranteed under arbitrary switching. It is revealed that construction of such a common Lyapunov function entails co-design of remote and local controllers. It is not trivial to perform the co-design which guarantees stability. We propose a policy evaluation-/iteration-based co-design procedure by confining the remote controller to MPC.

A. Simplification and Assumptions

We concentrate on a specific control synthesis problem by confining the remote controller to MPC. This problem is restrictive but meaningful because MPC, by taking constraints and optimality into account, is in alignment with desired features of the remote controller. As a results, MPC has been widely adopted as remote controllers of WNCSs [27]–[29]. Furthermore, because the state observer provides theoretically sound protection against loss of sensing information [7], [30], the WNCSs are more sensitive to packet loss on the actuation side [28]. stability analysis here focuses on the impact of actuation packet loss. This focus implies the following assumption.

Assumption 3.1: The closed-loop control system in the *smart actuation* architecture holds the following facts:

(i) we focus on actuation packet loss, assuming there is no packet loss at the sensor-to-control side;
(ii) delays of computations and communications within a single sampling period are ignored. Delays longer than the sampling period are regarded as packet losses.

Regarding the plant, we have the following assumptions to facilitate stability analysis and control design.

Assumption 3.2: Within a closed-loop control system, the open-loop plant features the following facts:

(i) it is stabilizable by either MPC or a local state feedback control policy $u_l(k) = h(x(k))$ in an ideal network;
(ii) its model is known, and its states are measured.

B. Nonlinear System Case

Consider a nonlinear discrete-time system,

$$x(k+1) = f(x(k)) + g(x(k))u(k),\ y(k) = x(k), \qquad (3)$$

where $x \in \mathbb{X} \subset \mathbb{R}^n$ is the state, f and g smooth vector fields, $u \in \mathbb{U} \subset \mathbb{R}^m$ the control input, and y the output. Both $\mathbb{X}$ and $\mathbb{U}$ are convex and compact, with each set containing the origin in its interior. The control objective is to steer states to the origin while minimizing a certain cost function.

Next, we illustrate co-design of the remote MPC and local control policies for system (3) to ensure that the resultant closed-loop system is semi-globally asymptotically stable.

1) Local Controller Design: Feedback stabilizing or tracking control design for a nonlinear system (3) is one of the fundamental problems in control theory. However, it is not the focus of this work. As specified in Assumption 3.2, we assume the existence of a smooth state feedback law $u_l(k) = h(x(k))$, which renders the resultant closed-loop system globally asymptotically stable; i.e., Thm. 3.3 holds.

Theorem 3.3: [31, Thm 4.2] Let $x = 0$ be an equilibrium point for the closed-loop system (3) with control $u_l(k) = h(x(k))$. There exists a continuously differentiable function $V_l : \mathbb{R}^n \to \mathbb{R}$ such that,

(i) $V_l(0) = 0$ and $V_l(x) > 0,\ \forall x \neq 0$
(ii) $\|x\| \to \infty\ \Rightarrow V_l(x)\ \to \infty$
(iii) $V_l(x(k+1)) - V_l(x(k)) < 0,\ \forall x \neq 0.$

Remark 3.4: Several control designs lead to $u_l(k) \in \mathbb{U}$, $\forall x \in \mathbb{X}$, and $u_l(k)$ renders $\mathbb{X}$, an invariant set of the control system. For simplicity, this work assumes that $u_l(k)$ always lies in $\mathbb{U}$ for all $x \in \mathbb{X}$, and makes $\mathbb{X}$ an invariant set. □

2) Remote Controller Design: Let ρ be a dummy variable. Given a constant $i \geq 0$, $\rho(i|k)$ denotes a prediction of $\rho(k+i)$, based on information available at time k; and $\rho(i|k+1)$ represents a prediction $\rho(k+i+1)$, using the information available at time $k+1$. $\rho(k) = \rho(0|k)$. Without loss of generality, for system (3) at time k, the MPC controller tries to minimize the following cost function [32],

$$V(x(k), \mathbf{u}(k)) = F(x(N|k)) + \sum_{i=0}^{N-1} l(x(i|k), u(i|k)), \qquad (4)$$

where $\mathbf{u}(k) = \mathbf{u}_r(k) = \{u_r(k), u_r(1|k), ..., u_r(N-1|k)\}$, $x(i|k)$ for $1 \leq i \leq N-1$ is the state corresponding to $\mathbf{u}_r(k)$, and N the prediction horizon. The positive definite functions $l(x,u)$ and $F(x)$ represent the stage cost and the terminal cost, respectively. At time k, the MPC controller solves an optimization problem by minimizing the cost function (4), subject to state/control constraints along the state trajectory and a terminal constraint $x(N|k) \in X_f \subset \mathbb{X}$. Assumption 3.2 implies that the optimization problem has an optimal solution $\mathbf{u}_r^*(k) = \{u_r^*(k), u_r^*(1|k), ..., u_r^*(N-1|k)\}$ at time k, and the associated cost function is given by $V_r{}^*(x(k)) = V(x(k), \mathbf{u}_r^*(k))$.

Assumption 3.2 indicates that there exist functions $F(\cdot), l(\cdot,\cdot), \mathscr{K}_f(\cdot)$ satisfying A1 to A4 [32, Sec. 3.3].

A1: $X_f \subset \mathbb{X}$, where X_f is closed and $0 \in X_f$.
A2: Local controller $\mathscr{K}_f(x) \in \mathbb{U},\ \forall x \in X_f$.
A3: $(f(x) + g(x)\mathscr{K}_f(x)) \in X_f,\ \forall x \in X_f$.
A4: $F(f(x) + g(x)\mathscr{K}_f(x)) - F(x) + l(x, \mathscr{K}_f(x)) \leq 0,\ \forall x \in X_f$.

To establish stability of the hybrid system resulting from the remote MPC and local controllers, it is sufficient to prove that (4) is a common Lyapunov function for subsystems associated with the local control policy or the MPC policy.

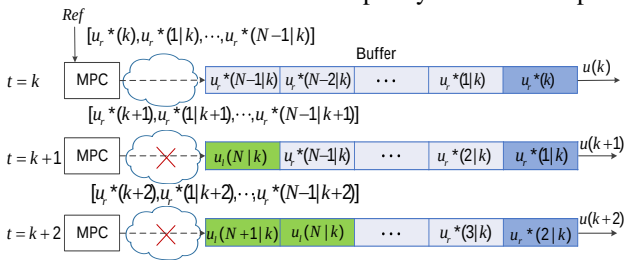


Fig. 4: Switch mechanism in the stability analysis

Remark 3.5: It is challenging to establish the stability of closed-loop system, when the smart actuator enacts the switching policy (1) and the remote controller employs MPC. Motivated by [27], [28], [33]–[35], where actuation buffer is used to address time delays and packet loss, the switch mechanism illustrated by Fig. 4 is alternatively executed by the smart actuator in the stability analysis below. Assume that the smart actuator has a buffer of size $L=N$, and the buffer stores a control sequence $\mathbf{u}(k)=\{u_r^*(k),u_r^*(1|k),\ldots,u_r^*(N-1|k)\}$. If the actuation packet is delivered at time $k+1$, the buffer is refreshed by the MPC control sequence $\mathbf{u}_r{}^*(k+1)$; otherwise, the local control policy $u_l(N|k)$ is pushed into the buffer. Consequently, the control sequence in the buffer turns into $\mathbf{u}(k)=\{u_r^*(1|k),\ldots,u_r^*(N-1|k),u_l(N|k)\}$. Previous study [27] discussed whether the last input of buffer should be set to 0 or last value when packet drops. We have developed a third option by setting it as local control input, which is favorable to stability. □

Proposition 3.6: Assume that a local control policy $u_l(k)=h(x(k))$ renders system (3) globally asymptotically stable, and that for a certain positive definition function $J(x)$, the following condition holds, for $0\le k<\infty,\forall x\in\mathbb{X},\alpha\ge 1$,

$$\alpha\cdot(l(x(k),u_l(k)))+J(x(k+1))-J(x(k))=0. \qquad (5)$$

Then the cost function (4), with $F(x)=J(x)$ and the stage cost $l(x,u)$, is a common Lyapunov function of the closed-loop system, where control input switches between the MPC policy and $u_l(k)=h(x(k))$, according to Fig. 4.

Proof: We need to show the following facts
(i) the subsystem resulting from the MPC policy has a Lyapunov function given by $V_r{}^*(x(k))$;
(ii) with $F(x)=J(x)$, (4) is a Lyapunov function of the subsystem resulting from $u_l(k)$;
(iii) in the case of switching from the MPC policy to the policy $u_l(k)$, the Lyapunov function decreases;
(iv) in the case of switching from the policy $u_l(k)$ to the MPC policy, the Lyapunov function deceases.

Proof of (i): With conditions A1–A4, the cost function (4) is a Lyapunov function for the subsystem corresponding to the MPC policy [32]. Proof is omitted.

Proof of (ii): Given $x(k)$, $u_l(k)=h(x(k))$, and system (3), we have $u_l(n|k)=u_l(k+n)$ and $x(n+1|k)=x(k+n+1)$, where $0\le n\le N$. Therefore, we have $\{u_l(k),\ldots,u_l(N+k)\}$, $\{x(1+k),\ldots,x(N+1+k)\}$, and

$$V(x(k+1))=J(x(N+1+k))+\sum_{i=1}^{N}l(x(i+k),u_l(i+k))$$

$$V(x(k+1))-V(x(k))=J(x(N+1+k))-J(x(N+k))$$
$$-l(x(k),u_l(k))+l(x(N+k),u_l(N+k)).$$

Substituting (5) into the above equation gives

$$V(x(k+1))-V(x(k))=-(\alpha-1)l(x(N+k),u_l(N+k))$$
$$-l(x(k),u_l(k)),$$

which is negative definite and implies (ii).

Proof of (iii): The induction principle is used here. Assume that the MPC policy $\mathbf{u}_r{}^*(k)=\{u_r{}^*(k)\ldots,u_r{}^*(N-1|k)\}$ is applied at time k. The Lyapunov function is $V_r{}^*(k)$. With the packet drop at time $k+1$, the control sequence is $\mathbf{u}(k+1)=\{u_r{}^*(1|k)\ldots,u_r{}^*(N-1|k),u_l(N|k)\}$, where $u_l(N|k)$ is the local control policy. We have

$$V(k+1)-V_r{}^*(k)=J(x(N+1|k))+\sum_{i=1}^{N-1}l(x(i|k),u_r{}^*(i|k))$$
$$+l(x(N|k),u_l(N|k))-J(x(N|k))-\sum_{i=0}^{N-1}l(x(i|k),u_r{}^*(i|k)) \qquad (6)$$

where $x(N+1|k)$ is obtained from applying control $u_l(x(N|k))$ to system (3). Applying (5) to (6), one verifies $V(k+1)-V_r{}^*(k)<-l(x(k),u_r^*(k))$. By induction, one can repeat the aforementioned derivation to establish the decrease of the Lyapunov function, if the packet loss continues beyond time $k+1$. This completes the proof of (iii).

Proof of (iv): $V_r{}^*(k+1)\le V_r{}^*(k)$ because of (i). At time k, since the control sequence in the buffer is a feasible solution with a cost $V(k)$, the MPC policy $\mathbf{u}_r{}^*(k)$, obtained by solving the optimization problem, necessarily yields a cost $V_r{}^*(k)\le V(k)$. Therefore $V_r{}^*(k+1)-V(k)\le 0$. ■

Remark 3.7: Prop. 3.6 establishes that the resultant closed-loop system is semi-globally asymptotically stable over $\mathbb{X}$, by imposing a restrictive condition (5) over $\mathbb{X}$. This restriction can be lifted in many cases by relaxing Assumption 3.2 to hold over X_f. This relaxation, together with A3 for $u_l=h(x)=\mathscr{K}_f(x)$ and assuming the successful delivery of the actuation packet at $k=0$, also ensures the semi-globally asymptotic stability over $\mathbb{X}$. □

Remark 3.8: It worth noting that condition (5) is sufficient but not necessary for stability. It is for verification but not for control synthesis. Given a feedback control $u_l=h(x)$ satisfying Thm. 3.3, $l(x,u)$, and $J(x)$, it is straightforward to verify whether condition (5) holds or not. However, it is not trivial to construct the function $J(x)$ and $u_l=h(x)$ from condition (5) for a given $l(x,u)$. This is because $u_l=h(x)$ is associated with a Lyapunov function V_l, and renders V_l decay at a certain rate which is irrelevant to $l(x,u)$. □

3) Policy Evaluation-Based Co-Design Procedure: We employ the following *policy evaluation-based procedure* to bridge the gap from $u_l=h(x)$ to the construction of $J(x)$.

(i) Design a stabilizing local controller $u_{l0}=h_0(x)$.
(ii) Given $u_{l0}=h_0(x)$, $l(x,u)$, $\gamma=1$, perform 1 step of the policy evaluation as (10) to evaluate the cost $V_1(x)$ corresponding to $u_{l0}=h_0(x)$, and set $J(x)=V_1(x)$.

(iii) Solve the MPC policy by minimizing the cost function (4) with $F(x) = J(x)$.

Given a local control law $u_l = u_{l0}$, the *policy evaluation-based co-design procedure* eventually outputs a terminal cost $J(x) = V_1(x)$. We have $J(x(k+1)) - J(x) = -l(x(k), u_l(k))$ according to (10), which satisfies condition (5). This implies the stability of the control system, according to *Prop.* 3.6.

Remark 3.9: As a system of first-order nonlinear difference equations, the closed-form solution of (5) or (10) is difficult to establish. Instead, a proper approximate solution is usually of practical interest. Given $u_l = h(x)$ and parameterizations of $J(x)$, (5) or (10) is reduced to algebraic equations, and thus, the approximate solution of $J(x)$ can be readily computed. See Appendix V-A for details. □

Remark 3.10: The control policy $u_l = h(x)$ is designed to make V_l decay along the system trajectory. This implies that u_{l0} may be far from optimal w.r.t., minimizing the cost function (4). As a consequence, the *policy evaluation-based procedure* above may have difficulty in solving $J(x)$, which satisfies (5). A remedy to this issue is to introduce *policy iteration-based co-design procedure* as follows:

(i) Design a stabilizing controller $u_{l0} = h_0(x)$ and let $j = 0$.
(ii) Given $u_{lj} = h_j(x)$ and $l(x,u)$, perform the policy evaluation step (10) to evaluate the cost $V_{j+1}(x)$ corresponding to the control u_{lj}.
(iii) Given $V_{j+1}(x)$, perform policy improvement step (11).
(iv) $j = j+1$; repeat the policy evaluation and policy improvement steps until $j = M$, and then let $J(x) = V_{M+1}(x)$. M denotes the allowed number of iterations.
(v) Solve the MPC policy by minimizing the cost function (4) with $F(x) = J(x)$.

Given the stabilizing control policy u_{l0}, the control policy updated in the policy improvement step also stabilizes the system (3). Hence, the *policy iteration-based co-design procedure* eventually will produce a local control policy $u_l = u_{lM} = h_M(x)$ and the terminal cost $J(x) = V_{M+1}(x)$ in (4), which satisfy condition (5). This implies the stability of the resulting closed-loop system. □

Remark 3.11: Either policy evaluation- or iteration-based co-design procedures can be readily extended to take control constraints into account. See [36] for details. □

C. LTI System Case

Consider a linear time-invariant system

$$x(k+1) = Ax(k) + Bu(k), \quad y(k) = x(k). \tag{7}$$

1) Controller Design: For simplicity, we take $l(x,u) = x^T Qx + u^T Ru$, and $F(x) = x^T Sx$ in (4), where Q, R, S are positive definite. Rewrite the cost function as follows:

$$\begin{aligned} V(x(k)) = &\sum_{i=k}^{k+N-1} (x^T(i)Qx(i) + u^T(i)Ru(i)) \\ &+ x^T(k+N)Sx(k+N). \end{aligned} \tag{8}$$

To ensure stability of the hybrid system, we validate that subsystems corresponding to the local control policy and the MPC policy share the same Lyapunov function (8).

Proposition 3.12: Consider system (7), the cost function (8), and the switching policy in Fig. 4, let S be the solution of discrete-time algebraic Riccati equation (DARE):

$$A^T SA - S + Q - A^T SB(B^T SB + R)^{-1} B^T SA = 0, \tag{9}$$

the local control policy is $u(k) = -(B^T SB + R)^{-1} B^T SAx(k)$. The resultant hybrid system is asymptotically stable.

Proof: It resembles the proof of Prop. 3.6, omitted. ■

IV. Evaluation

In this section, we present three examples to illustrate and verify the co-design procedure of local and remote controllers. Performances with different architectures are compared to corroborate the effectiveness of the smart actuation architecture. Simulations are conducted in MATLAB/Simulink, where random packet drops of both sensing and actuation sides are simulated. Sensing packet losses are handled by the Extended Kalman Filter (EKF) with intermittent observations [7]. Actuation packet losses are compensated by different structures, i.e., remote controller only, local controller only, and smart actuation (a hybrid system that adopts the switch policy (1)) architectures.

Detailed simulation results are given in Figs. 5 - **??**. Each boxplot figure shows the results of five different cases:

(i) *MPC_I*: remote MPC controller in ideal network. Since there is no packet loss, the remote controller is in effect.
(ii) *Local_I*: local controller in an ideal network.
(iii) *Hybrid*: smart actuation architecture with both local and remote controllers, but a lossy network. The hybrid system adopts the switch policy given by (1).
(iv) *Local*: local controller and a lossy network. In this case, the EKF transmits estimated states $x_e(k)$ to the local controller via the network. If the packet arrives, the local controller generates control inputs based on $x_e(k)$; otherwise, it generates control inputs based on $x_l(k)$.
(v) *MPC*: direct architecture with the buffered MPC scheme and a lossy network. The buffer size is 5.

Each boxplot is generated from 50 rounds of simulations.

A. Example 1: Linear System

Consider the load-positioning system in [37],

$$\dot{x} = A_c x + B_c u, \quad y(k) = x(k),$$

where system parameters are given in Table I and

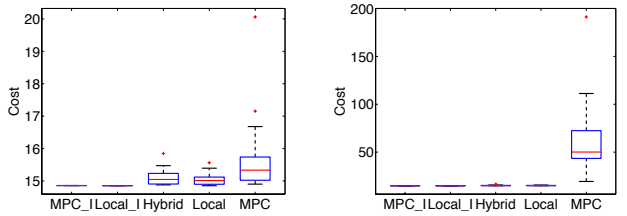
$$A_c = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & -d_L(\frac{1}{m_L} + \frac{1}{m_B}) & \frac{k_B}{m_B} & \frac{d_B}{m_B} \\ 0 & 0 & 0 & 1 \\ 0 & \frac{d_L}{m_B} & -\frac{k_B}{m_B} & -\frac{d_B}{m_B} \end{bmatrix}, \; B_c = \begin{bmatrix} 0 \\ \frac{1}{m_L} + \frac{1}{m_B} \\ 0 \\ -\frac{1}{m_B} \end{bmatrix}.$$

For simplicity, we discretize the continuous-time model using Eluer discretization, and have the discrete-time model denoted by $x(k+1) = A_d x(k) + B_d(k)$.

1) Controller Design: We would like to stabilize the load positioning system to the origin while minimizing the cost function (8) with $Q = I_4$ and $R = 1$. The controller design for this purpose is to determine the local control policy and

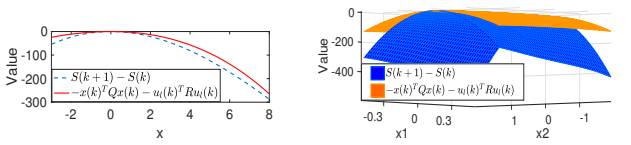
m_L	10	m_B	20	d_L	15	k_B	0.1	k_B	0.1	d_B	0.5
a_1	0.03	b_1	0.7	a	0.7	b	2	c	1.6	d	1.6

TABLE I: System parameters



(a) 80% packet loss (b) 80% packet loss, and the WNCS loses connection from 25s to 125s

Fig. 5: Costs of the linear system case



(a) first order nonlinear system (b) second order nonlinear system

Fig. 6: Policy evaluation results for nonlinear system

the matrix S. Solving the DARE, we can calculate S. And according to *Proposition* 3.12, we have the local control policy $u_l(k) = -(B_d^T S B_d + R)^{-1} B_d^T S A_d x(k)$.

2) Simulation Results: Simulation results are given in Fig. 5, where the simulation time of each round is 600s, and the sampling frequency is 6Hz. Fig. 5 (a) shows the costs when the lossy network is subject to 80% random packet loss. The cost of the *Hybrid* case is a little higher than the *Local_I* case, but lower than the *MPC* case. This is because the local control $u(k)$ is the optimal solution of the infinite time LQR problem, while the remote MPC solves it in a receding horizon manner. As a result, the *Local* case gives the lowest cost in both ideal and lossy networks. Fig. 5 (b) indicates that, when the WNCS loses network connection from 25 s to 125 s, both the *Hybrid* and *Local* cases still work properly. However, performance of the *MPC* case deteriorates drastically because the limited buffer size fails to mitigate network failure lasting for 100s.

B. Example 2: First Order Nonlinear System

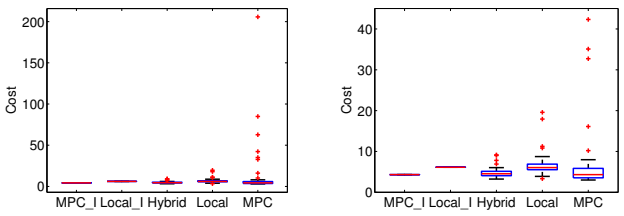
Consider the following first order nonlinear plant

$$\dot{x} = a_1 x^2 + b_1 u,$$

where model parameters are provided in Table I.

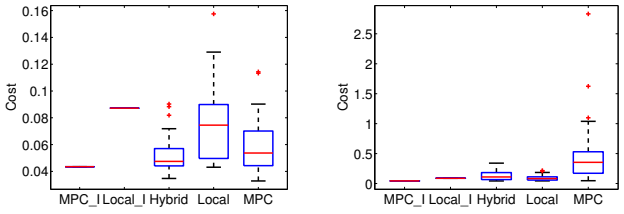
1) Controller Design: We would like to regulate the state to the origin while minimizing the cost (4), where $l(x,u) = x^T Q x + u^T R u$, terminal cost $F(x) = S(x)$. The local control law is $u_l = \frac{1}{b_1}(-x - a_1 x^2)$. To design terminal cost $S(x)$ for the remote MPC policy, we take $Q = 1, R = 1, \alpha = 1.2$, and choose $S(x) = x^T W x$, with W being a positive definite matrix. Based on the policy evaluation-based co-design method given in Sec. III-B.3, we obtain $S(x) = 10.16x^2$. As long as $S(x(k+1)) - S(x(k)) \le x^T(k)Qx(k) + {u_l}^T(k)Ru_l(k)$ holds for the entire feasible sets of state and control inputs, (5) is satisfied. The results of policy evaluation are given by Fig. 6 (a), where the curve of $S(x(k+1)) - S(x(k))$ is always below that of $x^T(k)Qx(k) + {u_l}^T(k)Ru_l(k)$ in the feasible set of x.

2) Simulation Results: Fig. 7 presents the costs of 5 different cases for the first order nonlinear system. Simulation



(a) Costs when 20% packet loss (b) zoom-in of (a)

Fig. 7: Costs of the 1st order nonlinear system under 20% loss



(a) 30% packet loss (b) 70% packet loss

Fig. 8: Costs of 2-state nonlinear system

time of each round is 60s, and the sampling frequency is 3Hz. The *MPC_I* case outperforms the *Local_I* since the former takes optimality into account by solving a finite-time optimal control problem versus the latter merely solves the stabilizing problem. When under 20% packet loss, the *Hybrid* case outperforms the *Local* and *MPC* cases in the sense that the cost distribution of the *Hybrid* has a lower mean value than the *Local* case, and has a smaller variance than the *MPC*.

C. Example 3: Second Order Nonlinear System

Consider the following second order nonlinear plant

$$\dot{x}_1 = a x_1^2 - b x_1^3 + c x_2, \ \dot{x}_2 = u,$$

where system parameters are provided in Table I.

1) Controllers Design: Again, we would like to regulate the state to the origin while minimizing the cost, which has same format with the cost function in Sec. IV-B and $Q = 3I_2, R = 1$. The local control law is given by:

$$z = x_2 + \frac{a}{c}x_1^2 + \frac{d}{c}x_1, u_l = -z - x_1 - (\frac{2a}{c}x_1 + \frac{d}{c})(-bx_1^3 - dx_1 + z).$$

Following the policy evaluation-based design method in Sec. III-B.3, we choose $\alpha = 3$, and get $S(x) = 17.39x_1^2 + 23.16x_2^2$. Fig. 6 (b) presents the results of policy evaluation. One can verify that in the feasible set of x, $S(x(k+1)) - S(x(k)) \le x^T(k)Qx(k) + {u_l}^T(k)Ru_l(k)$. Then (5) is satisfied.

2) Simulation Results: Fig. 8 (a) and (b) show the costs of five different cases for the second order nonlinear system under 0%, 30%, and 70% of packet loss. Simulation time of each round is 100s, and the sampling frequency is 10Hz. Again, with the ideal network, the cost of the *MPC* case is smaller than the *Local* case. When the network experiences 30% packet loss (Fig. 8 (a)), the *Local* case has a higher cost than both the *Hybrid* and the *MPC* cases. More interestingly, the *Hybrid* case outperforms the *MPC* case in the sense that the mean value and variance of the cost distribution for the former is smaller. When the network suffers from more severe packet loss (Fig. 8 (b)), both the *Hybrid* and *Local*

cases yield lower costs than the *MPC* case because (1) the linearized MPC prediction is not accurate away from k; (2) the buffer size is not enough under severe packet loss.

V. Conclusions and Future Work

We proposed a *smart actuation* architecture, which combines features of direct and hierarchical architectures: a remote controller accounts for optimality, adaptation, and constraints by conducting computationally expensive operations; a smart actuator executes a local control policy and accounts for system safety in the view of network imperfections. Stabilities for linear and nonlinear plant cases can be guaranteed by a policy iteration-based co-design procedure when the remote controller employs the MPC policy. Simulation results show that the smart actuation architecture works well in both reliable and unreliable networks. Many interesting issues remain open, for instance, the case of stability analysis for output feedback, if the smart actuation architecture can outperform other cases when network time-delay is considered, what if the plant model is partially known, and how to extend this result to large scale systems.

Appendix

A. Policy Iteration Algorithm

This content is included for self-completeness. Interested readers are referred to [38]. The notion of goal-directed optimal behavior is captured by defining a cost function

$$V(x(k)) = \sum_{i=k}^{\infty} \gamma^{i-k} l(x(i), h(x(i))),$$

with $0 < \gamma \leq 1$ a discount factor. Function $l(x(i), h(x(i)))$ is stage cost, which is a measure of the one-step cost of control.

1) Initialization step: select any admissible (i.e, stabilizing) control policy $h_0(x(k))$ as initialization.
2) Policy evaluation step: for $0 \leq j < \infty$,
$$V_{j+1}(x(k)) = l(x(k), h_j(x(k))) + \gamma V_{j+1}(x(k+1)). \quad (10)$$
3) Policy improvement step: for $0 \leq j < \infty$,
$$h_{j+1}(x(k)) = \arg\min_{h(.)} (l(x(k), h_j(x(k))) + \gamma V_{j+1}(x(k+1))). \quad (11)$$

Policy evaluation (10) involves solving an infinite dimension problem. A commonly-used treatment resorts to approximating the value function and control policy [39], for instance, $V(x) = W^T \Phi(x), h(x) = \Gamma^T \Psi(x)$, where W and Γ are the approximation coefficient vectors, and Φ, Ψ are the basis vectors given by $\Phi(x) = [\phi_1(x)\ \phi_2(x)\ ...\phi_N(x)]^T, \Psi(x) = [\psi_1(x)\ \psi_2(x)\ ...\psi_q(x)]^T$. Then the policy evaluation formula is given by $W^T(\Phi(x(k)) - \Phi(x(k+1))) = l(x(k), h(x(k)))$.

References

[1] X. Li et al., "A review of industrial wireless networks in the context of industry 4.0," *Wireless Networks*, 2017.
[2] C. Wu et al., "Coded packets over lossy links: A redundancy-based mechanism for reliable and fast data collection in sensor networks," *Computer Networks*, 2014.
[3] D. Gunatilaka et al., "Impacts of channel selection on industrial wireless sensor-actuator networks," *INFOCOM*, 2017.
[4] A. R. Mesquita et al., "Redundant data transmission in control/estimation over lossy networks," *Automatica*, 2012.
[5] M. Sha et al., "Empirical study and enhancements of industrial wireless sensor-actuator network protocols," *IEEE IoT Journal*, 2017.
[6] F. Dobslaw et al., "End-to-end reliability-aware scheduling for wireless sensor networks," *TII*, 2016.
[7] B. Sinopoli et al., "Kalman filtering with intermittent observations," *TAC*, 2004.
[8] H. Gao et al., "Network based h-infinity output tracking control," *TAC*, 2008.
[9] Z. Wang et al., "Robust h_∞ control for networked systems with random packet losses," *IEEE SMC, Part B (Cybernetics)*, 2007.
[10] H. Li et al., "Control of nonlinear networked systems with packet dropouts: interval type-2 fuzzy model-based approach," *IEEE Transactions on Cybernetics*, 2015.
[11] J. Wu et al., "Design of networked control systems with packet dropouts," *TAC*, 2007.
[12] Y. Tipsuwan et al., "Gain adaptation of networked mobile robot to compensate QoS deterioration," in *IECON*, 2002.
[13] S. H. Hong, "Scheduling algorithm of data sampling times in the integrated communication and control systems," *TCST*, 1995.
[14] K. Gatsis et al., "Control-aware random access communication," in *ICCPS*, 2016.
[15] E. G. W. Peters et al., "Controller and scheduler codesign for feedback control over IEEE 802.15.4 networks," *TCST*, 2016.
[16] Y. Ma et al., "Holistic cyber-physical management for dependable wireless control systems," *TCPS*, 2018.
[17] M. S. Branicky et al., "A scheduling and feedback co-design for networked control systems," in *CDC*, 2002.
[18] K. Gatsis et al., "Optimal power management in wireless control systems," *TAC*, 2014.
[19] C. Lu et al., "Real-time wireless sensor-actuator networks for industrial cyber-physical systems," *Proceedings of the IEEE*, 2016.
[20] J. P. Hespanha et al., "A survey of recent results in networked control systems," *Proceedings of the IEEE*, 2007.
[21] Y. Tipsuwan et al., "Control methodologies in networked control systems," *Control Eng Pract*, 2003.
[22] L. Schenato et al., "Foundations of control and estimation over lossy networks," *Proceedings of the IEEE*, 2007.
[23] K.-D. Kim et al., "Cyber-physical systems: a perspective at the centennial," *Proceedings of the IEEE*, 2012.
[24] B. Fateh et al., "Wireless network design for transmission line monitoring in smart grid," *IEEE Trans. Smart Grid*, 2013.
[25] C. L. Robinson et al., "Optimizing controller location in networked control systems with packet drops," *JSAC*, 2008.
[26] H. Lin et al., "Stability and stabilizability of switched linear systems: a survey of recent results," *TAC*, 2009.
[27] M. Lješnjanin et al., "Packetized MPC with dynamic scheduling constraints and bounded packet dropouts," *Automatica*, 2014.
[28] B. Li et al., "Wireless routing and control: a cyber-physical case study," in *ICCPS*, 2016.
[29] D. M. de la Peña et al., "Lyapunov-based model predictive control of nonlinear systems subject to data losses," *TAC*, 2008.
[30] X. Liu et al., "Kalman filtering with partial observation losses," in *CDC*, 2004.
[31] H. K. Khalil, *Nonlinear Systems*, 3rd ed. Englewood Cliffs, NJ: Prentice-Hall, 2002.
[32] D. Q. Mayne et al., "Constrained model predictive control: Stability and optimality," *Automatica*, 2000.
[33] A. Bemporad, "Predictive control of teleoperated constrained systems with unbounded communication delays," in *CDC*, 1998.
[34] J. Fischer et al., "Optimal sequence-based lqg control over tcp-like networks subject to random transmission delays and packet losses," in *ACC*, 2013.
[35] D. E. Quevedo et al., "On stochastic stability of packetized predictive control of non-linear systems over erasure channels," *IFAC Proceedings*, 2010.
[36] M. Abu-Khalaf et al., "Nearly optimal control laws for nonlinear systems with saturating actuators using a neural network HJB approach," *Automatica*, 2005.
[37] V. Shilpiekandula et al., "Load positioning in the presence of base vibrations," in *ACC*, 2012.
[38] F. L. Lewis et al., "Reinforcement learning and adaptive dynamic programming for feedback control," *IEEE Circuits Syst. Mag*, 2009.
[39] Y. Jiang et al., "Optimal codesign of nonlinear control systems based on a modified policy iteration method," *TNNLS*, 2015.