Plant Control over QoS-Enabled Packet Networks

Davide Quaglia, Riccardo Muradore and Paolo Fiorini

***Abstract*— Performance and stability in networked control systems are strongly affected by transmission delays and packet dropouts. We propose a control architecture based on the Differentiated Services technique to guarantee quality of service on the network. The crucial observation is that not all signals traveling on the network have the same importance. An adaptive packet-marking strategy has been developed to choose at run-time the transmission priority according to the importance of the data and the current network condition. System/network co-simulation applied to a bottleneck scenario validates the proposed approach and shows that better performance can be reached without increasing the bandwidth if network resources are used in a smarter way.**

I. Introduction

Networked Control Systems (NCS) are feedback control systems in which the control loop is closed through a packet-based communication network rather than by a point-to-point connection, see Figure 1. They are becoming more and more popular due to the technology improvement in wired/wireless networks and to the high flexibility they allow. This kind of systems can overcome physical barriers and be applied in remote control applications such as teleoperation and telepresence, [12]. Unfortunately the communication channel connecting plant, controller and sensors is shared and, thus, packet losses and delays may happen.

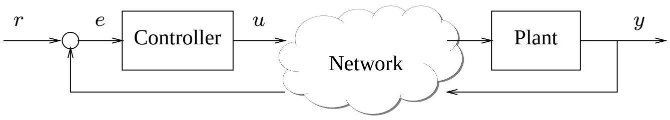

Fig. 1. Block diagram of a networked control system.

During the last decades several solutions have been proposed to guarantee the closed loop stability of the overall system and the required performance despite delays and losses. In the survey paper by [11], the authors summarized the most important results (at least till 2007). More recent contributions have been proposed for example in [22], [17], [6]. The reason for such a broad number of papers is also due to the fact that the assumptions on the network and on the plant can be very different. Besides the usual plant (controller) classification in linear/nonlinear and continuous-/discrete-time model, the assumptions on the network depend both on the level of abstraction of the communication channel (hardware levels and/or protocols) and on the statistical description of the its phenomena (constant/random delays, fixed/variable packet loss rates, etc.), [23], [10]. A traditional approach to address the time-varying network behavior consists in focusing on the worst case channel condition and choosing between two options:

- over-provisioning of the network to support periods of peak bandwidth demand with a waste of resources in less-critical periods;
- design of a robust controller for the worst case period with a loss of performance in less-critical periods.

This work proposes a novel approach in which the network is observed in its dynamic behavior and instantaneous network condition is taken into account by the control strategy. The second original contribution is the use of a transmission technique based on priorities which provides a kind of set of "virtual wires" with different levels of Quality of Service (QoS). The transmission priority is decided by the controller as part of its control strategy. The third contribution is the design of a control strategy which not only computes the command u according to the error e but also chooses the transmission priority according to the importance of the current value of command u and the current network condition in all the virtual wires. To achieve this goal the output of the controlled plant is predicted by taking into account estimated losses and delays.

This work has been made possible by the use of a simulation tool ([3]) which allows the analysis of the mutual influence between the control part and the communication part, ([16], [19], [5], [9]). The priority scheme adopted for the network is named Differentiated Services (DiffServ) and it is a standard technique to introduce Quality-of-Service guarantees in IP networks by assigning packets to either a high-QoS class, or a regular, unguaranteed class ([20]).

The paper is organized as follows. In Section II the problem is stated. In Section III the proposed approach is introduced whereas in Section IV the network scenario and all the hypothesis are described. Section V details the packet-marking architecture based on DiffServ and addresses the design of the markers. Simulation results are given in Section VI and some conclusions are drawn in Section VII.

The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007-2013 via the Projects CON4COORD, grant agreement nr. 223844, and I-SUR, grant agreement nr. 270396.

D. Quaglia is with the Department of Computer Science, University of Verona, Strada Le Grazie 15, Ca' Vignal 2, 37134 Verona, Italy. davide.quaglia@univr.it

R. Muradore is with the Department of Computer Science, University of Verona, Strada Le Grazie 15, Ca' Vignal 2, 37134 Verona, Italy. riccardo.muradore@univr.it

P. Fiorini is with Department of Computer Science, University of Verona, Strada Le Grazie 15, Ca' Vignal 2, 37134 Verona, Italy. paolo.fiorini@univr.it

II. Problem Statement and Related Work

Performance in networked control systems depends also on the distortion introduced in the original commands u and measurements y during the transmission process. Packets may get lost or delayed because of noise (e.g., in wireless networks) and contention among different traffic flows (e.g., the command and the measurement flows). With the term "Quality of Service"(QoS) we refer to a given level of the average and standard deviation of the end-to-end delay and of the average packet loss rate. For instance, the network intermediate systems (e.g., router, bridge, and access point) have queues to host entering packets. When the packet arrival rate is larger than the exiting rate, queue length increases leading to higher delays. When the queue is full, arriving packets are dropped. Therefore, when the packet arrival rate is non-deterministic, the level of QoS is time-varying and protocols like IP, Ethernet and WiFi do not provide QoS guarantees.

The problem has been studied for decades in the context of multimedia communications ([18]) whose features are very close to those of NCS's, i.e.:

- packets have a time reference since are generated by sequential sources;
- in case of interactive multimedia applications (e.g., telephony over IP) there is a closed loop whose delay must be kept as small as possible.

For this reason, we propose to exploit this large amount of knowledge to improve NCS performance by focusing not only on the controller design technique, as usual, but also on the design of the transmission strategy, e.g., on the design of nodes and channels, communication protocols, and service differentiation strategies. Several techniques have been proposed to guarantee a given level of QoS to multimedia information traveling on the network. They can divided into two classes. The first class improves error resilience by increasing the redundancy of the bitstream (e.g., by using error correcting codes, retransmissions, and multiple descriptions). The other class is based on a smarter use of the network resources, e.g., through traffic priority in the Differentiated Services (DiffServ) architecture, bandwidth reservation in the Integrated Services architecture, rate adaptation in TCP protocol and transmission power in wireless networks ([18]). In this work we will use the DiffServ architecture both in the controller-to-plant and plant-to-controller path since it allows to reduce both the packet loss rate and the delay of NCS communications.

Alternative solutions based on bandwidth management approaches can be found both in multimedia and control papers. In [2], the authors introduced a QoS manager to assign the "optimal" bandwidth allocation among the different real-time multimedia streams on the channel. Optimality here is related to the optimal value for quantification factor and so for the coding bit rate and distortion. In some sense this can be seen as an *one way* bandwidth allocation because the data traveling on the channel are not used in feedback way. The dynamic rate and control adaptation introduced in [24] is designed for NCS. The idea to overcome network congestion is based on increasing/decreasing the sample time of the messages (commands and measurements) sent through the channel. A system manager changes the controller parameters according to the sample time imposed by the QoS manager in order to guarantee performance and loop stability.

A completely different solution to overcome delays and packet losses is by "pure" control-based approach. The controller is designed in such a way that stability is always guarantee for whatever delay and packet loss rate. The controller design is base on the passivity theory (e.g. [25], [4]) quite popular in teleoperated system ([12], [15], [21]). Example of recent applications of passivity in NCS can be found in [13], [14]. Unfortunately guarantee stability in any channel conditions is paid in term of performance. In this paper we assume the controller as given and our goal is to show how performance can be improved by using the network in a better way. Future work will focus on integrating also the passivity-based design of the controller in such a framework.

In a DiffServ architecture, packets are assigned to one of a few classes to receive a specific forwarding behavior on nodes along their path. Assignment is performed by writing a priority value in the packet header. Priorities are used to handle packets in the queues of the intermediate systems. Ideally, each priority class is handled by a different queue. The priority mechanism is efficient if the high-priority bandwidth is a small fraction of the overall bandwidth. For this reason, a cost can be associated to each class by the network operator. Different assignment strategies have been proposed:

- **Application-based marking**: all the packets of an application flow are assigned to a given class; this approach is used to separate telephony-over-IP traffic from traditional data traffic.
- **Random marking**: a given share of the application flow is randomly chosen and marked as high-priority traffic ([8]).
- **Distortion-based marking**: the priority is based on the loss of performance (distortion) that the loss of the packet would produce at the receiver ([7]).

The goal of the paper is to solve the following problem:
Problem 1 Given a system P and a controller C connected by a Differentiated Services packet network, design markers at the controller and plant side exploiting packet priority to improve tracking performance. ■

The solution of this problem is sub-optimal because the controller is given. Nevertheless, the use of network resources in a smarter way provides insights about the level of performance improvement.

III. Differentiated Services in Control

A solution to guarantee QoS to the control flows without modifying the network bandwidth is to use the Differentiated Services architecture which allows to send packets using different priorities. Since the allocation of the total bandwidth among the different transmission policies is crucial

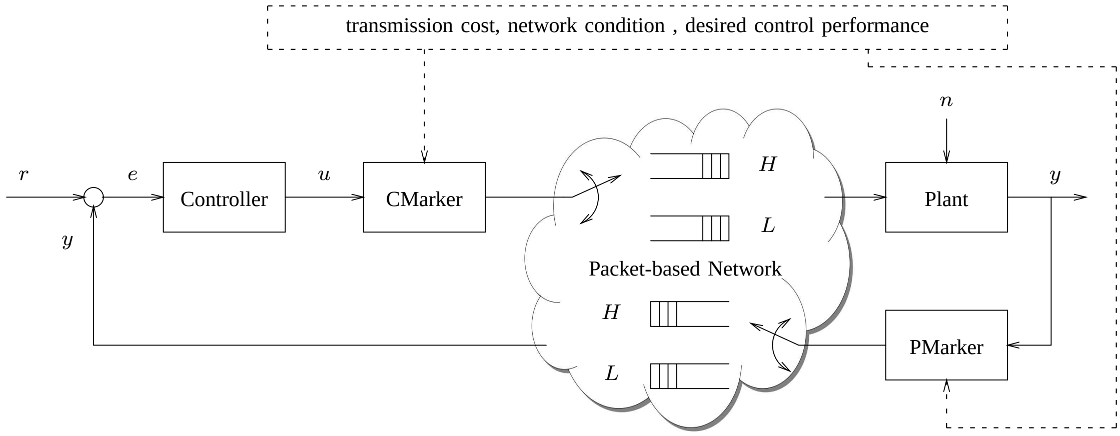

Fig. 2. Block diagram of an NCS with Differentiated Services architecture and packet markers.

for the efficiency of the network, the goal is to optimally distribute packets between the different available policies. In the present case we assume to have two policies H and L (it is straightforward to generalize this example to the case of more than two classes).

Figure 2 shows the proposed architecture. Each message, i.e., commands u and measurements y, is examined by the marker at the controller and plant side (CMarker and PMarker, respectively) and assigned to either the H or L policy depending on the current network condition, desired level of control performance, and transmission cost.

At each node of the network there are two queues for the two different forwarding priorities. Packets are picked from the H queue and, if it is empty, from the L queue. The queue corresponding to the H policy is characterized by high cost, low loss and low delays, whereas the queue corresponding to the L policy is characterized by low cost and best effort behavior.

The design of the marking strategy should handle the trade-off between the use of the expensive high-priority policy and the level of control performance. The strategy can consist either in maximizing the control performance under cost constraints or by minimizing the cost under performance constraints.

The proposed approach, called Adaptive Packet-Marking architecture (APM), aims at keeping the control performance as constant as possible despite of variations of network conditions. This task is accomplished by using efficiently the most reliable policy choice to deliver the most important packets so that the desired quality is guaranteed at the receiver.

Remark 1 When the channel is good, even the most important data units do not need to be protected against loss and delay, while more protection should be used when the channel is bad. $\Diamond$

IV. Network Model and Assumptions

Figure 3 shows an example of network scenario. Nodes are represented by circles and links by continuous arrows. The scenario represents a typical bottleneck topology in which peripheral nodes are connected through high-capacity low-delay links to a backbone link with less capacity and higher delay. Close to Node 1 and Node 2, router queues are also reported for each interface. Over this topology two end-to-end traffic flows have been defined by connecting squared boxes which represent applications. In particular, the Figure shows the packet flow between controller (Node 0) and plant (Node 3) and a concurrent flow between Node 4 and Node 5. Since the backbone capacity is shared among the different traffic flows, queue level may vary during simulation and congestion may happen.

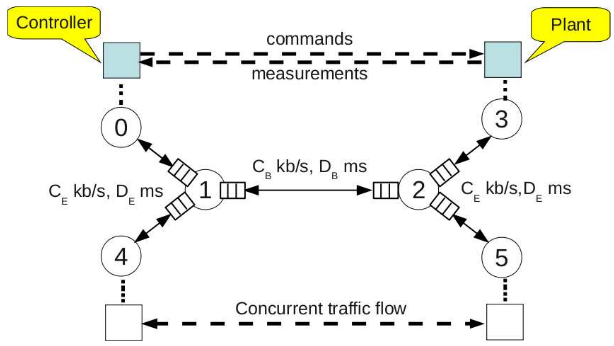

Fig. 3. Network scenario for the proposed approach.

Let τ^H, τ^L and p^H, p^L be the transmission delays and packet loss rates for the H policy and the L policy, respectively. The proposed algorithm will be based on the following network assumptions:

(A1) the network links are full duplex. This means that, for example, packets from Node 0 to Node 3 (i.e. commands) do not interfere with packets from Node 3 to Node 0 (i.e. measurements);

(A2) τ^H, τ^L, p^H, and p^L are time-varying values depending of the congestion level of the network;

(A3) due to the queueing policy, the following relationship holds

$$\tau^H \leq \tau^L, \qquad p^H \leq p^L$$

This means that the routers provide higher priority to packets labeled with H (i.e. no fair queueing policy);

(A4) the packets sent in each queue arrive in the proper order (this means that a packet can be overtaken only by packets belonging to a different queue);

(A5) the delays τ^H, τ^L and the packet loss rates p^H, p^L are in general different for the controller-to-plant path (τ^H_{CP}, τ^L_{CP}, p^H_{CP}, p^L_{CP}) and for the plant-to-controller path (τ^H_{PC}, τ^L_{PC}, p^H_{PC}, p^L_{PC})

(A6) each packet collects all the signals measured at the same sample time (i.e. all the measurements and/or all the commands);

(A7) the concurrent traffic uses both the H and L priority queues;

(A8) the delays τ^H, τ^L are smaller than the control sample time T_s.

The last assumption aims at simplifying the design of the markers and of the receivers and it will be removed in future work. In general, since there is a queue for each traffic priority, a low-priority packet with timestamp t_i can arrive later than a high-priority packet with timestamp $t_j > t_i$. The last assumption avoids this possibility and allows the receiver to assume that packets are lost after a delay larger than T_s. It is worth noting that the last assumption puts a design constraint on the size of the queues.

V. Adaptive packet-marking architecture

The problem of packet classification has two main aspects:

1) the estimation of the importance of each packet,
2) the allocation of resources (H/L policy).

The available policies have been described in the previous section, now we need to describe how to evaluate the importance of each command and measurement.

As depicted in Figure 4 the proposed architecture consists of the following elements:

- **Plant**: continuous-time system. the actuator is assumed to be an event-driven device: it applies instantaneously the command.
- **Controller**: discrete-time system with sample time T_s. The controller is a time-driven device which computes a new command at kT_s; a holder is placed in front of the controller; the error between reference and measurement is computed after each T_s interval.
- **CMarker** (**PMarker**): marker at the controller (plant)-side that decides the transmission policy for the command (measurement) packets.
- τ^H_{CP}, τ^L_{CP}, p^H_{CP}, p^L_{CP} and τ^H_{PC}, τ^L_{PC}, p^H_{PC}, p^L_{PC}: time-varying transmission delays and packet loss rates.
- **CReceiver**: receiver at the controller-side; it also computes the estimations $\mu = \{\hat{\tau}^H_{PC}, \hat{\tau}^L_{PC}, \hat{p}^H_{PC}, \hat{p}^L_{PC}\}$ to be sent to the plant-side.
- **PReceiver**: receiver at the plant-side; it also computes the estimations $\nu = \{\hat{\tau}^H_{CP}, \hat{\tau}^L_{CP}, \hat{p}^H_{CP}, \hat{p}^L_{CP}\}$ to be sent to the controller-side.

Packet loss rates are estimated by counting the lost packets over a constant number of transmitted packets evaluated through sequence numbers. Communication delays are estimated by computing a moving average on the difference between the arrival time and the timestamp within the packet payload (assuming synchronized nodes). This way of computing statistics aims at filtering out sharp variations but leads to an update delay. A further delay is introduced by the network when statistics are sent back to the source. Statistics can be even got lost during transmission.

The value of $\{p^H, \tau^H\}$ and $\{p^L, \tau^L\}$ and the corresponding estimations are time-varying quantities depending on the current channel condition. For this reason, packet classification is performed during transmission to take into account not only control and plant status but also the current channel status; therefore, the priority level $\pi_k \in \{H, L\}$ assigned to a packet at time kT_s depends on the expected performance degradation as explained in the following section.

A. Marker at the controller side

Let π_{k-1} be the selected policy at the previous step $t = (k-1)T_s$. In order to choose the policy π_k for the packet at time kT_s, we go through the following steps:

1) compute the estimated plant output values $\hat{y}^H_{get}(k)$ and $\hat{y}^L_{get}(k)$ in case of successful reception (by using H and L policy, respectively) and $\hat{y}_{lost}(k)$ in case of packet loss. The estimations $\hat{y}^H_{get}(k)$, $\hat{y}^L_{get}(k)$ are computed taking into account the estimated delays $\hat{\tau}^H_{CP}$ and $\hat{\tau}^L_{CP}$, respectively. For the computation of $\hat{y}_{lost}(k)$, the maximum delay is used. In the present case the maximum delay is equal to T_s due to assumption (A8).
2) compute the overall estimations $\hat{y}^H(k)$ and $\hat{y}^L(k)$ where the previous estimations are weighted with the corresponding packet loss probabilities:
$$\begin{aligned} \hat{y}^H(k) &= (1-\hat{p}^H_{CP}(k))\hat{y}^H_{get}(k) + \hat{p}^H_{CP}(k)\hat{y}_{lost}(k) \\ \hat{y}^L(k) &= (1-\hat{p}^L_{CP}(k))\hat{y}^L_{get}(k) + \hat{p}^L_{CP}(k)\hat{y}_{lost}(k) \end{aligned}$$
3) compute the displacement between the estimated plant outputs using the H and L policy
$$\hat{\varepsilon}(k) = \hat{y}^L(k) - \hat{y}^H(k) \tag{1}$$
4) compare the displacement with a user-defined threshold E to choose the policy ($\pi_k = H$ or $\pi_k = L$). A marker criterion could be
`if` $|\hat{\varepsilon}(k)| > E$
    `then` $\pi_k = H$
        $\hat{x}(k+1|\pi_k) = \hat{x}^H_{get}(k+1)$ `%next state`
    `else` $\pi_k = L$
        $\hat{x}(k+1|\pi_k) = \hat{x}^L_{get}(k+1)$ `%next state`
The use of H policy depends on the threshold E whose value is a design parameter to be set according to the working conditions (e.g., expected congestion, H cost).

It is worth highlighting again that $\hat{\tau}^H_{CP}(k)$, $\hat{\tau}^L_{CP}(k)$ and $\hat{p}^H_{CP}(k)$, $\hat{p}^L_{CP}(k)$ are estimated values and are affected by the transmission delay.

Remark 1 A similar procedure could be applied to the marker at the plant side. In this case the command should be estimated instead of the plant output for the different policies. On the other hand for the PMarker there are two important differences:

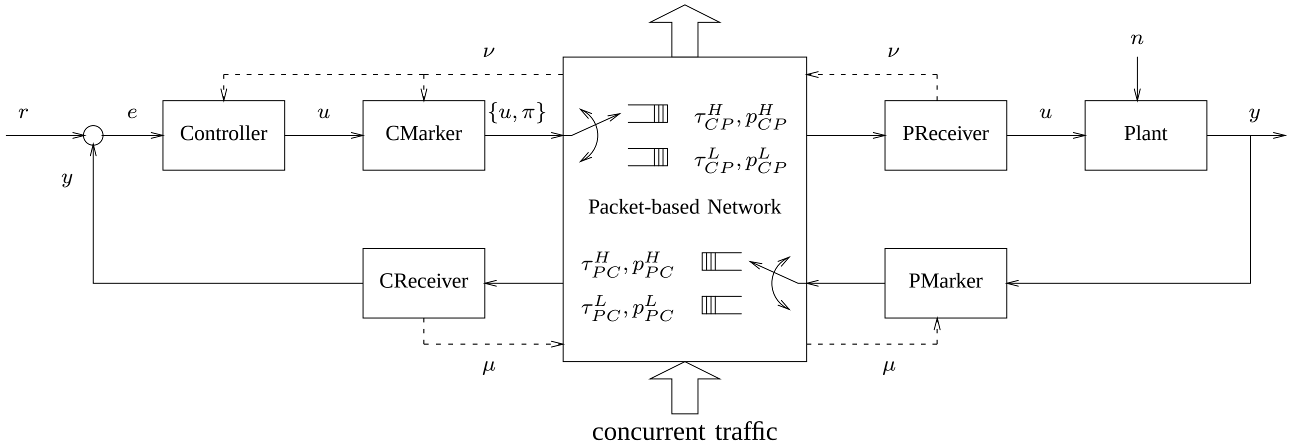

Fig. 4. Detailed block diagram of a NCS with QoS-enabled network and transmission statistics.

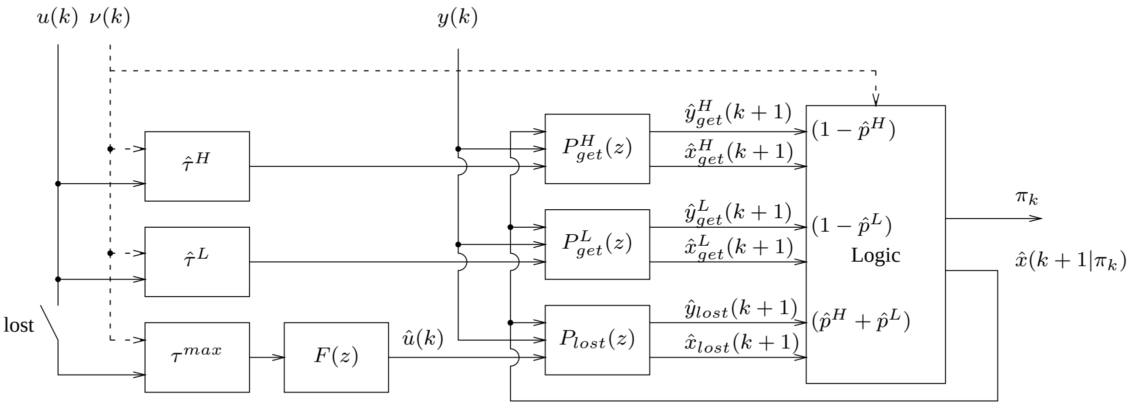

Fig. 5. Block diagram of the marker at the controller side.

1) since the controller is a discrete time system running at T_s and the maximum delay is exactly equal to T_s, there is not difference to send the packet using the H or L queue. The inequalities $\tau^H \leq \tau^L \leq T_s$ implies that even though y^H and y^L may arrive at different time, we have $y^H(k) = y^L(k)$ at $t = kT_s$. This remark does not hold when the difference between $\hat{\tau}^L_{PC}$ and $\hat{\tau}^H_{PC}$ is higher than one sample time or when the controller is an event-driven device;
2) since the reference r at time k is not know at the plant side, it is not possible to estimate $u(k)$. The only thing we can do is to estimate the contribution to $u(k)$ due to the feedback, i.e. $C(z)y(k)$.

The conclusion is that in our set up it is possible to write the PMarker as a threshold comparing the measurements between two consecutive sample time. $\diamondsuit$

B. Estimators in the LTI case

If the plant is a linear time invariant (LTI) continuous-time plant it is easy to provide explicit expressions for $\hat{y}^H_{get}(k)$, $\hat{y}^L_{get}(k)$ and $\hat{y}_{lost}(k)$. Let's assume that the plant has the following state space representation

$$\left\{ \begin{array}{rcl} \dot{x}(t) & = & Ax(t) + B\bar{u}(t) \\ y(t) & = & Cx(t) \end{array} \right. \tag{2}$$

where the input $u(t)$ is delayed by τ, $\bar{u}(t) = u(t-\tau)$. Since $u(t)$ is a step-wise signal, we have

$$\bar{u}(k) = \left\{ \begin{array}{ll} u(k-1), & t \in [t_k, t_k+\tau) \\ u(k), & t \in [t_k+\tau, t_{k+1}). \end{array} \right. \tag{3}$$

The discrete-time equivalent system of the LTI continuous-time system (2) sampled at T_s takes the form ([1])

$$\left\{ \begin{array}{rcl} x(k+1) & = & \Phi_A x(k) + \Gamma_1(\tau)u(k-1) + \Gamma_0(\tau)u(k) \\ y(k) & = & Cx(k) \end{array} \right.$$

where $\Phi_A = e^{AT_s}$ and

$$\Gamma_0(\tau) = \int_0^{T_s-\tau} e^{At}dtB, \qquad \Gamma_1(\tau) = e^{A(T_s-\tau)}\int_0^{\tau} e^{At}dtB.$$

Defining $\hat{x}(k) := \hat{x}(k|k-1)$ and $\hat{y}(k) := \hat{y}(k|k-1)$, and let G be the output injection matrix (ex. Kalman/Luenberger gain), the predictors for the plant output are

- H policy chosen at time $t = kT_s$ ($\pi_k = H$, $\tau_k = \hat{\tau}_k^H$)

$$\begin{cases} \hat{x}^H_{get}(k) &= \hat{x}(k|\pi_{k-1}) \text{ initialization} \\ \hat{x}^H_{get}(k+1) &= \Phi_A \hat{x}^H_{get}(k) + \Gamma_1(\hat{\tau}_k^H)u(k-1) + \\ & +\Gamma_0(\hat{\tau}_k^H)u(k) + G(y(k) - C\hat{x}^H_{get}(k)) \\ \hat{y}^H_{get}(k) &= C\hat{x}^H_{get}(k) \end{cases}$$

- L policy chosen at time $t = kT_s$ ($\pi_k = L$, $\tau_k = \hat{\tau}_k^L$)

$$\begin{cases} \hat{x}^L_{get}(k) &= \hat{x}(k|\pi_{k-1}) \text{ initialization} \\ \hat{x}^L_{get}(k+1) &= \Phi_A \hat{x}^L_{get}(k) + \Gamma_1(\hat{\tau}_k^L)u(k-1) + \\ & +\Gamma_0(\hat{\tau}_k^L)u(k) + G(y(k) - C\hat{x}^L_{get}(k)) \\ \hat{y}^L_{get}(k) &= C\hat{x}^L_{get}(k) \end{cases}$$

- packet get lost (i.e. delay $\tau_k = T_s$)

$$\begin{cases} \hat{x}_{lost}(k) &= \hat{x}(k|\pi_{k-1}) \text{ initialization} \\ \hat{x}_{lost}(k+1) &= \Phi_A \hat{x}_{lost}(k) + \Gamma_1(T_s)u(k-1) + \\ & +G(y(k) - C\hat{x}_{lost}(k)) \\ \hat{y}_{lost}(k) &= C\hat{x}_{lost}(k) \end{cases}$$

  where we assume to hold the previous command whenever the new command does not arrive (i.e. $\hat{u}(k) = u(k-1)$, $\Gamma_0(T_s) = 0$)

Since the measurements are sampled at T_s, the controller is a discrete-time system and the delay is smaller than T_s, then it is right to use $y(k)$ in the estimator equations. If $y(k)$ does not arrive, we have to use the last received measurement, $\hat{y}(k) = y(k-1)$, or to go open loop.

The estimators are needed not only to cope with model uncertainty and unknown initial conditions as usual, but also to compensate for the discrepancy between the true transmission delays and the estimated ones, and the packets that get lost without any acknowledgment.

C. Quality of Service

As stated in Section II, the DiffServ is only a way to guarantee the QoS. Even if in this work we do not go into detail about this important aspect, it is worth giving at least an idea about a possible approach to solve this problem. In multimedia, the QoS is usually measured in terms of video and audio quality and/or in terms of number of lost packets. In NCS, a possible QoS measure could be the standard deviation of the tracking error computed on a moving windows, i.e.

$$\Lambda_{[k-N,k]} = \frac{1}{N}\sum_{i=1}^{N} \|e(k-N+i) - \bar{e}_{[k,k+N]}\|_2^2$$

where

$$\begin{aligned} e(k) &= r(k) - y(k) \\ \bar{e}_{[k-N,k]} &= \frac{1}{N}\sum_{i=1}^{N} e(k-N+i). \end{aligned}$$

Since the goal is to keep $\Lambda_{[k-N,k]}$ almost constant in spite of network congestion and variations on the reference signal, it is possible to relate the threshold E used within the adaptive marking algorithm (see Section V-A) with the current value of the error variance. In order to avoid unstable behavior, the choice of N and how often the threshold E has to be changed need to be accurately tuned during the design phase.

VI. Experimental Validation

The proposed packet marking strategy has been tested through the simulation of the scenario depicted in Figure 4; the co-simulation technique described in [3] has been used to model controller and plant in Matlab/Simulink and the channel with a network simulator. For the plant, a normalized model of an electrical DC motor has been considered as follows:

$$G(s) = \frac{b}{s(1+as)}, \qquad a = 3,\, b = 4. \tag{4}$$

Its discrete-time representation at T_s can be easily found in [1] and so all the estimators (i.e. L matrix) in Section V-B can be designed by pole allocation. For simplicity's sake the controller consists of a static gain $C(z) = k$ since the focus of the work is to show that transmission policies can improve control performance (i.e., reduce the tracking error $e(k) = r(k) - y(k)$) when the network is congested. The sampling interval T_s has been set to 50 ms. With reference to the network scenario of Figure 3, a concurrent ON/OFF traffic is sent over the bottleneck with the L policy to cause congestion in some given time intervals. The bottleneck capacity is 512 kb/s, the minimum transmission delay is 1.5 ms, the NCS command/measurement bitrate is 10.24 kb/s (for each direction) and the bitrate of the concurrent traffic is 507 kb/s when active.

Figure 6 shows the simulation results obtained by using the proposed marking strategy for the application traffic. From top to bottom, the Figure reports the time series of the reference r and the measurements y, the estimated delay from controller to plant $\hat{\tau}_{CP}$, the displacement $\hat{\varepsilon}(k)$ in (1), the output $\pi_k \in \{H, L\}$ of the marker at the controller side, and the behavior of the ON/OFF concurrent traffic. It is worth highlighting that the "trapezoidal shape" of the delay is due to the congestion mechanism producing delays and losses as explained in Section II. The value of the displacement (and the corresponding transmission policy) depends both on the importance of the command u (i.e., when the error $e = r - y$ is large) and on the congestion level of the network. For instance, the displacement exceeds the threshold ($E = 10^{-5}$ in this case) both after time 5 s when the reference signal r has been changed and after time 6.5 s when packet loss rate and delay are large (even though the error is small). If the error is negligible, as at time 18-20 s, control packets are still sent as low-priority traffic even if the network is heavily congested. This property avoids to waste resources without improving control performance significantly and to worsen the congestion.

Table I compares the proposed marking strategy with simpler approaches, i.e., sending all the application traffic either with the L policy or with the H policy. The former approach represents a traditional un-guaranteed scenario while the latter allows to assess the upper bound on control performance for the given assumptions since application packets never get lost and they are only affected by propagation delay. The Table reports the standard deviation of the tracking error, the use of the H policy for the application traffic, and the packet

Marker strategy	std tracking error	H policy share on application traffic	packet loss rate on concurrent traffic
always L	0.1045	0%	0.4%
APM	0.1028	25%	0.5%
always H	0.0991	100%	1%

TABLE I

COMPARISON AMONGST DIFFERENT MARKING STRATEGIES.

loss rate experienced by the concurrent traffic. By modifying the threshold E within the APM algorithm, it is possible to trade off between performance and cost (i.e., usage of H policy).

Since in the proposed example the simulation horizon is small and the maximum delay is smaller than one sample time T_s, the difference in terms of standard deviation is really small. Nevertheless, for scenarios with larger time horizon and more congested network the improvement will be much more remarkable, i.e. the gain in performance worth the cost of the H policy.

In the proposed example by using the H policy for only the 25% percent of the time it is possible to reduce the standard deviation of the tracking error of about the 63% (SBAGLIATO 0.6%) with respect to the pure L policy. The values in the last column in Table I represent the effect of the marking strategy of the application under design on the concurrent traffic. If all the packets of the control application are sent as high-priority data, the concurrent traffic could be damaged severely. APM strategy can trade-off between control performance and fair use of network resources since the H policy is used only for *important data*. If all the users of the network adopts the APM strategy, the improvement of transmission quality will be global since it is reasonable to assume that *important data* coming from different processes are usually uncorrelated in time.

Table II reports the standard deviation of the tracking error and the use of H policy for three different threshold values: $E = 10^{-6}, 10^{-5}, 10^{-4}$. As expected the lower the value of E the lower the tracking std is and the higher the percentage of H policy is, i.e.

$$\downarrow E \quad \Rightarrow \quad \downarrow \mathrm{Var}\{e\}, \qquad \uparrow \%H. \tag{5}$$

A QoS manager should be designed (not done here) on top of the marker for choosing in real time (but at a smaller rate than the control/marker loop rate) the optimal value for E. Such supervisor module should relate the threshold with the current estimate of the error variance in order to maintain the error as much constant as possible.

VII. CONCLUSION

In this paper a first step toward the Communication and Control Co-design (C^3 design) has been proposed by using the Differentiated Services technique to assure Quality of Service. Two markers at the plant- and controller-side have been designed to choose the communication policy for commands and measurements in order to obtain better performance without increasing the bandwidth. The crucial observation is that not all signals traveling on the network have the same importance and so the network resources can be used in a smarter way. Next steps will be to provide more solid theoretical background to the proposed architecture, to extend the method to more than two policies and to develop a full C^3-design where the control strategy takes into account both control and communication issues.

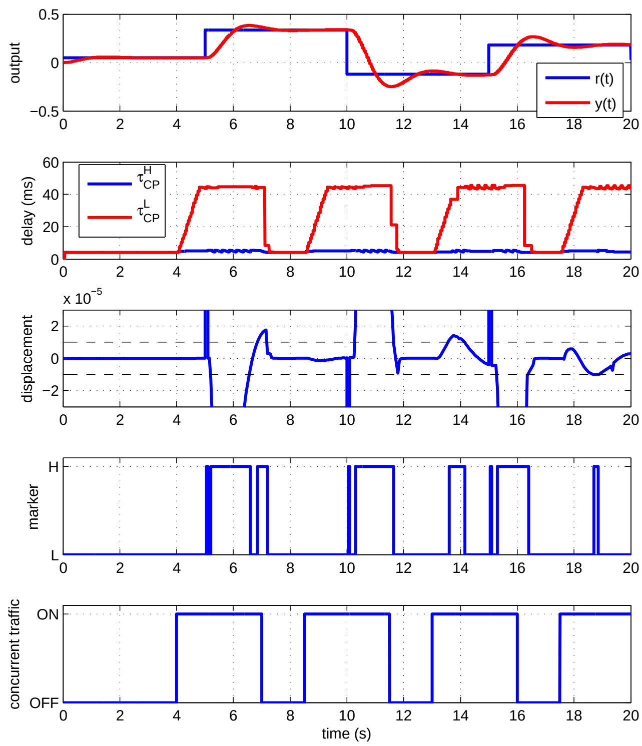

Fig. 6. Simulation results ($E = 10^{-5}$).

Marker strategy	threshold E	std tracking error	H policy share on application traffic
APM	10^{-6}	0.1020	47%
APM	10^{-5}	0.1028	25%
APM	10^{-4}	0.1029	10%

TABLE II

COMPARISON AMONGST DIFFERENT THRESHOLDS.

REFERENCES

[1] K.J. Åström and B. Wittenmark. *Computer-controlled systems: theory and design*. Prentice Hall New York, 1996.

[2] A. Antunes, P. Pedreiras, L. Almeida, and A. Mota. Dynamic rate and control adaptation in networked control systems. In *Industrial Informatics, 2007 5th IEEE International Conference on*, volume 2, pages 841–846. IEEE, 2007.

[3] D. Botturi, P. Fiorini, R. Muradore, and D. Quaglia. Simulation of Networked Control Systems with Applications to Telerobotics. In *European Control Conference (ECC)*, 2009.

[4] B. Brogliato, R. Lozano, B. Maschke, and O. Egeland. Dissipative systems analysis and control. *Theory and Applications, 2nd ed. London: Springer-Verlag*, 2007.

[5] C.C. Chen, A. Molin, and S. Hirche. Guaranteed Cost Control over Quality-of-Service Networks. In *European Control Conference*, 2009.

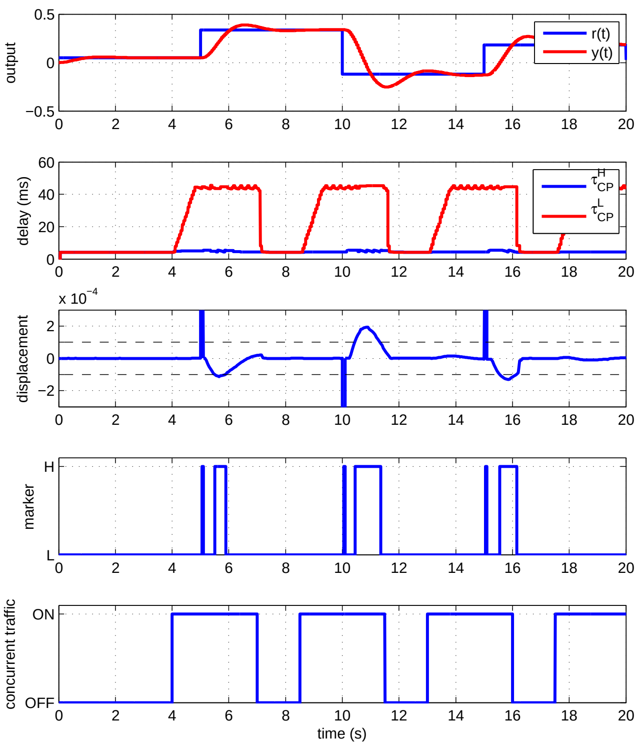

Fig. 7. Simulation results ($E = 10^{-4}$).

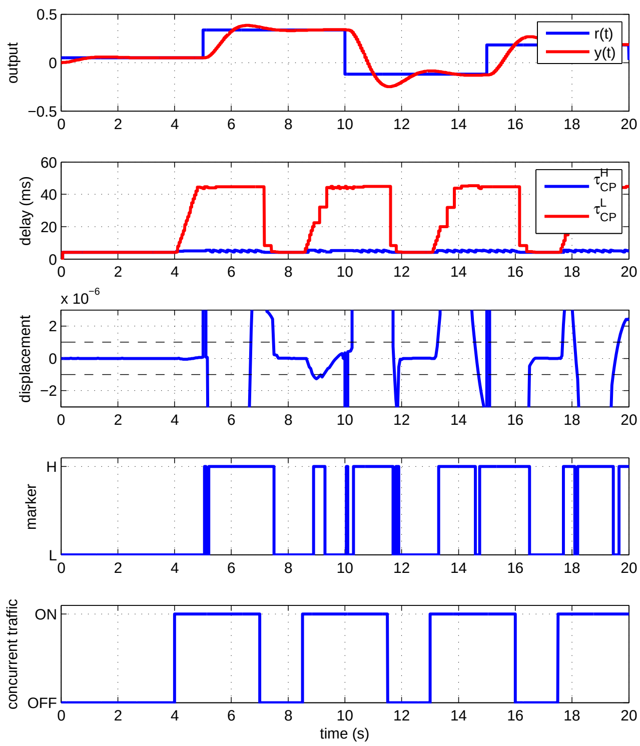

Fig. 8. Simulation results ($E = 10^{-6}$).

[6] M.B.G. Cloosterman, N. van de Wouw, W.P.M.H. Heemels, and H. Nijmeijer. Stability of networked control systems with uncertain time-varying delays. *Automatic Control, IEEE Transactions on*, 54(7):1575–1580, july 2009.

[7] J.C. De Martin and D. Quaglia. Distortion-based packet marking for MPEG video transmission over DiffServ networks. In *IEEE International Conference on Multimedia and Expo*, 2001.

[8] W. Feng, D. Kandlur, D. Saha, and K. Shin. Maintaining end-to-end throughput in a differentiated-services internet. *IEEE/ACM Transactions on Networking*, 7(5):685–697, October 1999.

[9] D. Görges, M. Izák, and S. Liu. Optimal Control and Scheduling of Networked Control Systems. In *Proc. 48th IEEE Conference on Decision and Control, Shanghai, China*, 2009.

[10] V. Gupta, A.F. Dana, J.P. Hespanha, R.M. Murray, and B. Hassibi. Data transmission over networks for estimation and control. *IEEE Transactions on Automatic Control*, 2006.

[11] J.P. Hespanha, P. Naghshtabrizi, and Y. Xu. A survey of recent results in networked control systems. *Proceedings of the IEEE*, 95(1):138–162, 2007.

[12] P.F. Hokayem and M.W. Spong. Bilateral teleoperation: An historical survey. *Automatica*, 42(12):2035–2057, 2006.

[13] N. Kottenstette and P.J. Antsaklis. Stable digital control networks for continuous passive plants subject to delays and data dropouts. In *Decision and Control, 2007 46th IEEE Conference on*, pages 4433–4440. IEEE, 2007.

[14] N. Kottenstette, X. Koutsoukos, J. Hall, J. Sztipanovits, and P. Antsaklis. Passivity-based design of wireless networked control systems for robustness to time-varying delays. In *2008 Real-Time Systems Symposium*, pages 15–24. IEEE, 2008.

[15] D.A. Lawrence. Stability and transparency in bilateral teleoperation. *IEEE Transactions on Robotics and Automation*, 9(5):624–637, 1993.

[16] J. Lee, S. Bohacek, J.P. Hespanha, and K. Obraczka. Modeling communication networks with hybrid systems. *IEEE/ACM Transactions on Networking*, 15(3):630–643, 2007.

[17] Hongbo Li, Mo-Yuen Chow, and Zengqi Sun. Optimal stabilizing gain selection for networked control systems with time delays and packet losses. *Control Systems Technology, IEEE Transactions on*, 17(5):1154–1162, sept. 2009.

[18] G. Lu. Issues and technologies for supporting multimedia communications over the internet. *Computer Communications*, 23(14):1323–1335, August 2000.

[19] A. Molin and S. Hirche. On LQG Joint Optimal Scheduling and Control under Communication Constraints. In *Proc. 48th IEEE Conference on Decision and Control, Shanghai, China*, 2009.

[20] K. Nichols, V. Jacobson, and L. Zhang. A Two-bit Differentiated Services Architecture for the Internet. *RFC 2638*, July 1999.

[21] G. Niemeyer and J.J.E. Slotine. Stable adaptive teleoperation. *IEEE Journal of oceanic engineering*, 16(1):152–162, 1991.

[22] L. Schenato. Optimal estimation in networked control systems subject to random delay and packet drop. *IEEE transactions on automatic control*, 53(5):1311–1317, 2008.

[23] L. Schenato, B. Sinopoli, M. Franceschetti, K. Poolla, and S.S. Sastry. Foundations of control and estimation over lossy networks. *Proceedings of the IEEE*, 95(1):163–187, 2007.

[24] J. Silvestre, L. Almeida, R. Marau, and P. Pedreiras. Dynamic qos management for multimedia real-time transmission in industrial environments. In *Emerging Technologies and Factory Automation, 2007. ETFA. IEEE Conference on*, pages 1473–1480. IEEE.

[25] A.J. van der Schaft. *L2-gain and passivity techniques in nonlinear control*. Springer Verlag, 2000.