Teleoperation over IP Network : Network Delay Regulation and Adaptive Control

P. Fraisse*, A. Lelevé**
* LIRMM, 161 rue Ada, Montpellier, 34000, France
**ICTT Batiment L. de Vinci 20 av A. Einstein Villeurbanne, 69621, France
Philippe.Fraisse@lirmm.fr, Arnaud.Leleve@insa-lyon.fr

Abstract

This paper proposes a solution to stabilize a remote manipulator robot (PUMA560) using IP network. A method is described to control this remote manipulator based on High-Order Sliding Mode so as obtain a product time delay bandwidth constant. Before introducing this control algorithm, we propose a Network Delay Regulator method (NDR) which eliminates the delay jitter in order to make estimation, prediction and control with a constant mean time delay (T_r). The stability condition of this algorithm is studied. Some simulation results illustrate the perfomance of the High-Order Sliding Mode to dynamically force the remote system according to the Network Round Trip Time (NRTT).

1 Introduction

There are situations when firms or laboratories have to resort to remote manipulation. Such cases appear when dangerous objects have to be handled [1] or/and when the environment is too aggressive for humans. Typical applications belong to the nuclear domain (for instance in the dismantling of a nuclear plant), deep-sea domain (work on underwater structures of oil rigs) and spatial domain (exploration of distant planets). Teleoperation has the supplementary advantage of giving the possibility of sharing an experiment between several operators located in distinct places. This way, heavy outdoor experimentations could be easily shared between several laboratories and costs could be reduced as much. However, long distance control of a remote system requires the use of different transmission media which causes two main technical problems in teleoperation : limited bandwidth and transmission delays due to the propagation, packetisation and many other events digital links may inflict on data [2]. Moreover bandwidth and delays may vary according to events occurring all along the transmission lines. In acoustic transmission, round-trip delays greater than 10s and bit-rates smaller than 10kbits/s are common. These technical constraints result in one hand in difficulties for the operator to securely control the remote system and, in the other hand, make classical control laws unstable. Many researches have proposed solutions when delays are small or constant [3], but when delays go beyond a few seconds and vary a lot as over long distances asynchronous links, solutions not based on teleprogramation are fewer because such delays make master and slave asynchronous and the control unstable.

Since 1997, our team has begun a research project on teleoperation. At this time, publications about teleoperation methods when delays are small and/or constant were numerous but there was a lack in long distance teleoperation literature, when delays are prohibitive and variable such as through Internet channels. Starting from an initial experiment with our mobile manipulator, our first work consisted in highlighting practical difficulties inherent to teleoperation of mobile manipulator.

We then began to develop a generic teleoperation model which we inserted in a simulation environment using *Matlab/Simulink*. Working with this tool, we proposed an architecture and a method [1] consisting in buffering data at its arrival in both master and slave. Associated with an adaptive predictor we could offer a real-time estimation of slave state over a virtual constant delay transmission link. In order to validate these results, we developed a software based on this model, which we applied on our mobile manipulator. The main drawback was

that this buffering method induced a very high global delay in short distance, due to an unsuitable transmission period. Therefore, we completed our model by a real-time network round-trip delay measure (*Network Round Trip Time, NRTT*) and forward prediction in order to adapt the transmission period (*Network Delay Regulator, NDR*) and buffer parameters to small frequency network behavior. First simulation results were presented in [4]. We propose in this paper an original remote decentralized controller on the manipulator based on High-Order Sliding Modes. This controller allows to adapt the manipulator dynamics in closed loop according to the (*NRTT*). We prove in the next section, under some specific assumptions, that same stability level is obtained if the product closed-loop bandwidth of the remote system by the *NRTT* is constant. We then propose to force dynamically each axis of the robot to verify this condition using the High-Order Sliding Modes algorithm whose the coefficients be adapted with respect to the *NRTT*. Stability is also proved into $3rd$ section. Afterwards, some simulations results are provided to illustrate the efficiency of this solution.

2 Teleoperation Environment

In this section we introduce the teleoperation architecture as well as the model of the experimental platform including the actuators of the manipulator robot, the network and the base station. Some assumptions are presented to reach the stability level condition. We also describe an original method based on *HSOM* allowing to dynamically force each actuator as a first order linear system.

2.1 Teleoperation Architecture

We consider a classical teleoperation diagram including two parts (the master alias Base and the slave alias Remote System) separated by a network transmission link. Figure 2 gives a global view of the teleoperation diagram and details what each part includes. We will use the term "Base" to refer to the host computer that

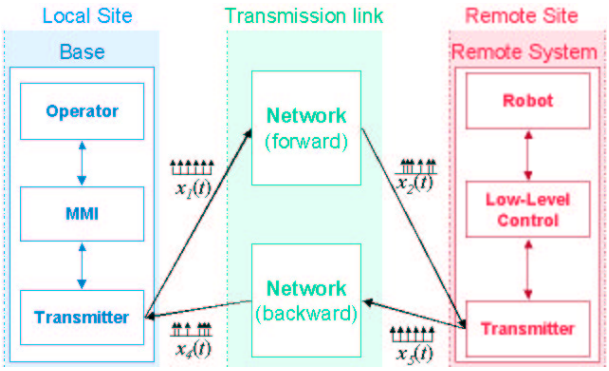


Figure 1: Teleoperation Architecture

features the operator and the man-machine interface (*MMI*). The "Remote System" will point out the system to be teleoperated, i.e. the transmitter and the low-level control associated with the robot. To simplify the reasoning, we will consider a monodimensional system. We just make the assumption that every axis of a complex system can be independently controlled, which is feasible by mean of computed torque methods, for instance. Signals exchanged between the master and the slave are periodically emitted back and forth every 200 ms before being delayed by the network both way. This period was the smallest our platform could handle in the first experiments.

2.2 Teleoperation Platform

We use a terrestrial vehicle equipped with a 6 degree-of-freedom manipulator (d.o.f) [1]. It consists of a 6x6 vehicle fitted with a *PUMA* manipulator. Several control laws are featured: global motion of the whole mobile

manipulator, force-driven control laws, PID control law,... In this present work, we use a computed torque method [6] to control the arm. A dSpace © *DSP* board is dedicated to the low-level control of the 8 axis (6 for the arm and 2 for the vehicle). A laptop *PC* fitted with a wireless 2Mbps Ethernet network board takes care of the transmission and of a global control of the whole mobile manipulator. The operator sends and receives data to

(a) Mobile Manipulator (b) Transmission Loop

Figure 2: Experimental Setup

the remote site at least through his lab network. This data roams through the Internet if necessary, it reaches the remote site *LAN* and, at last, it is sent at 2Mbps through at 802.11b-like radio transmission link. Figure (2) pictures this trip. We have chosen *TCP* versus *UDP* because *TCP* is reliable: the received data is exactly the same as the sent data (no packet loss or disorder). The main drawback is an higher *NRTT* than with *UDP*, due to packet checking and reordering.

2.3 Modeling

The modelisation step describes the dynamic model including the network, the remote system and the base station. We consider that the remote manipulator robot as a set of linear subsystems. The nonlinearities (Gravity, Coriolis) are included in the disturbance term.

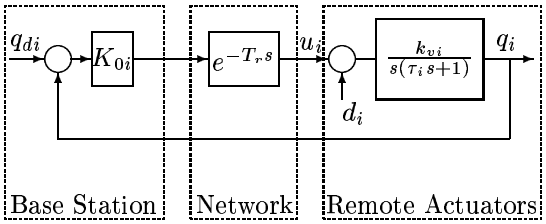


Figure 3: Transmission model

The linear model of the manipulator robot actuators can be expressed as :

$$\tau_i \ddot{q}_i(t) + \dot{q}_i(t) = k_{vi}\left(u_i(t) - d_i(t)\right) \tag{1}$$

where $\mathbf{u}(t) = [u_1..u_6]^t$ is the control vector, $\mathbf{q}(t) = [q_1..q_6]^t$ the position output vector, $\mathbf{d}(t)$ the disturbance vector and τ_i the time response of each actuator. The base station is always controlled by an human operator whose explicit modeling is complex. However, the stability study needs an explicit model. Also, we assume the base station as a proportionnal loop and the network reacts as a constant delay: either we use a channel that provides

constant delay T_r, or we use our *NDR* wich provides a constant virtual delay T_r. We obtain the following delayed equation :

$$\mathbf{u}(t) = \mathbf{K}_0 \mathbf{E}(t) e^{-T_r s} \tag{2}$$

with $\mathbf{E}(t) = \mathbf{q}_d(t) - \mathbf{q}(t)$ the tracking error and a constant global delay T_r.

2.4 Stability with constant time delay

The stability condition of the second-order system with constant time delay described by the above equations (1) and (2), provides an equation where the bandwidth of the remote system ($\beta_i = \frac{1}{\tau_i}$) according to the delay T_r is nonlinear [8]. However, in the case of the remote system is composed by a set of first-order systems such that :

$$\tau_i \dot{q}_i(t) + q_i(t) = k_{vi} u_i(t) \tag{3}$$

We then find the following stability condition :

$$T_r \leq \frac{\tau_i \arccos(-\frac{1}{K_{0i} k_{vi}})}{K_{0i} k_{vi} \sin \arccos(-\frac{1}{K_{0i} k_{vi}})} \tag{4}$$

Assume that gains K_{0i} and k_{vi} are constant, we can rewrite the above equation (4) as :

$$T_r \beta_i \leq C_i \tag{5}$$

where C_i is a constant positive value. In order to verify the equation (5) when T_r takes different values, the bandwidth β_i has to be adapted according to T_r.

3 Network Delay Regulation

This section will introduce you the low-level transmission architecture we have set up. It features a mean to easily compute an estimation of the state of the remote system (before backward data has reached the base) and also a prediction of movements at the moment when corresponding data is generated by the operator. This assumes delays have been made constant, which is the first subject we will develop.

3.1 Principle

In order to eliminate the delays jitter that is typical in asynchronous networks, we have thought of inserting two "Network Delay Regulator" (*NDR*), located at the input of both Base and Remote System. Each one stacks incoming data as it is asynchronously received; meanwhile each *NDR* unstacks previous data (in the same order : First-In, First-Out) at a constant sampling rate (T_r) common to the whole teleoperation structure. The results are that signals find again the initial shape they initially had. The main drawback of this operation is the growth of the mean delay. In our study, we have observed the network round-trip-time (*NRTT*) and the global round-trip-time (GRTT) including *NRTT* and both *NDRs* through time. Figure 4 illustrates the difference between *NRTT* and *GRTT*. In a first approach, we have assumed that both sites are too far to be synchronized so that we don't have access to single-trip-times. We have had to assume that forward and backward trip times can be deduced from *NRTT* by dividing it by 2. We are currently working on this subject; several solutions are conceivable: we could try to synchronize both hosts through the network using methods like [12] or using GPS clock up to a resolution of 500ns, which is quite enough for our experimentations.

3.2 Queue Dimensioning

In order to fit the size of the *NDRs* queue to the network behavior, there is an initialization pattern at the beginning of teleoperation experiment. 10 frames are exchanged in order to measure the network round-trip delays. Theses measures are then analyzed; we use the maximum time (m_t) between two receptions (ie maximum

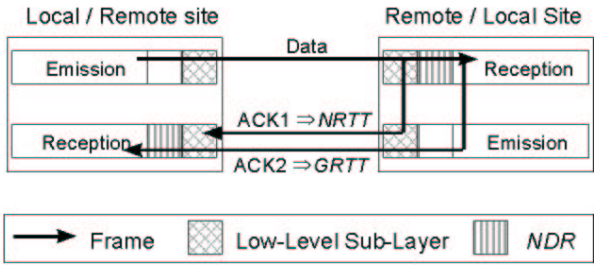


Figure 4: $NRTT$ and $GRTT$

difference between delays) of these $NRTT$ to set the nominal size (n_s) of the DVC queue, with a factor of security η :

$$n_s = 2 + E\left(\eta \frac{m_t}{T_r}\right)$$

When $\eta = 1$, the queues can handle delays corresponding to a maximum time between two receptions equal to m_t. Whenever this case occurs, the queue empties down to 2 samples, the minimum size we permit. $\eta = 1,3$ ensures that the queue won't go beyond this limit even if network delays deteriorate a bit. Moreover, this security factor limits the lack of precision of the $NRTT$ measure, due to the initialization pattern short duration.

3.3 Experimental Results

We have experimented the NDR at short distance (two hosts 75km apart) and at virtual long distance (a relay delayed network data with values corresponding to a transmission between Paris, France and New York, USA). Figure 5 allows to compare the Network Round-Trip-Time ($NRTT$) with the Global Round-Trip-Time ($GRTT$) in steady pattern. At long distance, statistics give a mean $GRTT$ to 7s (vs. 610ms for the $NRTT$) with a standard deviation of 12ms (vs. 500ms). There were 13 desired data samples in each queue which contained between 4 et

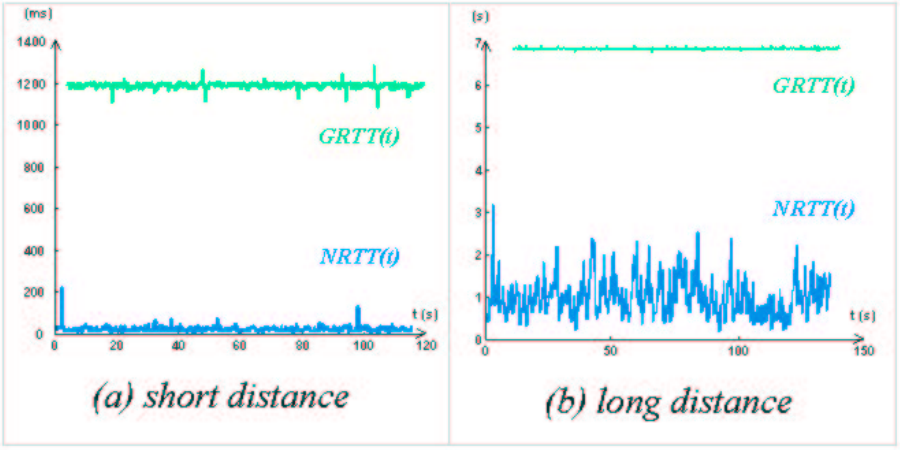


Figure 5: Comparison between $NRTT$ and $GRTT$

16 samples. At short distance, mean $GRTT$ grows up to 1,2s (vs 23ms for $NRTT$) and the standard deviations are identical. There were 2 desired samples in each queue which size varied between 1 and 3 samples. We observe

the following relation (s(t) is the evolution of the number of samples in each *NDR*) :

$$FRTT = 2max\,(s(t)) + NRTT$$

To put it in a nutshell, long distance results could be better with a real-time environment for *NDRs* which we didn't have at first for compatibility reasons. Nevertheless they inform us that the principle seems correct even if parameters might be better chosen to limit the embarrassing raising of the mean *GRTT*. At short distance, mean *RTT* is multiplied by 50 but it is difficult to have it smaller because the queue sizes vary between 1 and 3 samples. In fact, it is due to the sampling period T_r of 200ms which is 10 times as much as the mean *NRTT*. As the mean *GRTT* is a multiple of T_r , the only mean to decrease mean *GRTT* is to decrease T_r.

3.4 Enhancement : Dynamic Adaptation

As the latest conclusions let us foresee, we need to adapt the transmission period T_r and the nominal size (n_s) of the *NDR* queues. Consequently it is necessary to predict the evolution of network behavior in order to change the transmission parameters whenever they become out of proportions. A change of these parameters leads the system to pause the teleoperation task and then to go through a new transient pattern as at the first starting. This involves that every calculus made from these parameters (state and network prediction, control laws,...) are able to adapt themselves to the new parameters. Network behavior prediction is achieved this way: we use two discrete Kalman filters coupled with a triple integrator model and an autocorrelation $\phi[k] = \sigma_Q^2 e^{-a|k|T_r}$ [13]. This kind of association is typically used to predict movements of targets in problems of tracking. The first filter is used to predict the mean *NRTT* and the second one, the evolution of m_t, the maximum time between two receptions (directly used for resizing the NDR queues). Kalman matrices are then :

$$\mathbf{A} = \begin{bmatrix} 1 & T_r & \frac{T_r^2}{2} \\ 0 & 1 & T_r \\ 0 & 0 & 1 \end{bmatrix} \mathbf{Q} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \sigma_Q^2 \end{bmatrix} \begin{array}{ll} B = [0] & R = \sigma_Q^2 \\ H = [100] & \\ P_0 = I_3 \sigma_Q^2 & \end{array} \tag{6}$$

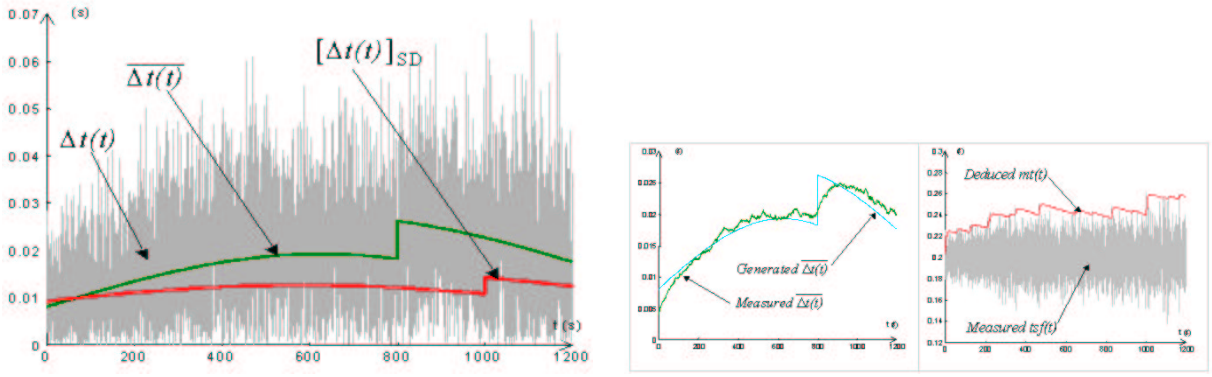
This filter gives us a prediction one step ahead, which corresponds to a few 100ms according to the transmission period. As we need to predict the network behavior farther ahead (a few seconds would be reasonable), we compute the prediction step of the Kalman filter 50 times: $\hat{x}_{k+1} = A^t \hat{x}_k$. As this extrapolation is noisy, we filter it with a low-pass filter ($\gamma = 50T_r$). This gives us an extrapolation which we have found good enough to be used for further parameter adaptation. The difficulties reside in measuring in one hand the mean *NRTT* when its standard deviation is high as in TCP transmissions, and in the other hand, the maximum time between two successive frames : $m_t(t)$. To measure the mean *NRTT*, worked out from the elapsed time between the emission of a frame and the reception of its corresponding acknowledgement, we simply use a low-pass filter with $\gamma = 250T_r$. This is a calculus we have found appropriate according to simulations we had done. To measure the evolution of $m_t(t)$, we compute the time between two successive frames $t_{sf}(t)$ and we keep the superior envelope of this signal. This computation consists in the following algorithm :

If	(t_{sf}[k]>e[k-1])
Then	e[k]=t_{sf}[k]
Else	e[k]=$\frac{e[k-1]}{v_d}$

Decreasing speed v_d has been set to 1,00005. As this envelope contains high frequencies, we filter it with a first order discrete low-pass filter with the bandwith $\gamma = 25T_r$.

3.5 Validation by Simulation

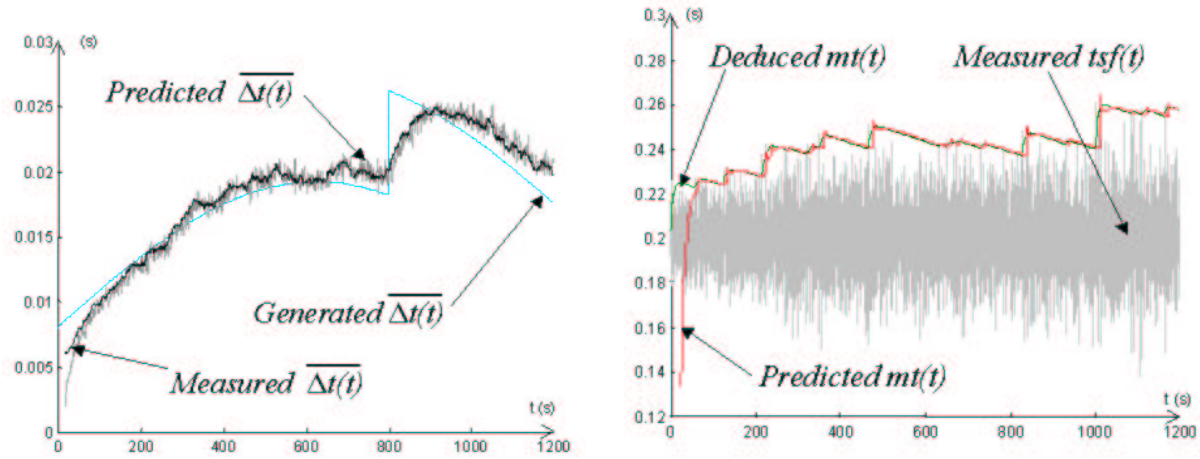
In order to validate our network behavior predictor, we have generated delays which mean value and standard deviation vary (see figure 6). We have chosen continuous sine variations with a discontinuity in order to represent long time variations of a real network. As we use a protocol that ensures data will be received in the same order as it has been sent, delays are modified to respect the right order of data reception. The prediction is tested during 20mn.



(a) (b)

Figure 6: Generated delays with mean and std. dev.

The first Kalman filter gives a prediction of the mean delays (ie $NRTT$) with an advance of $50T_r = 10s$. Figure 6 pictures the results. We can remark that the precision of the prediction depends a lot on the initial measure. In this case, the mean delays measure is a bit too slow when discontinuities happen. This prediction is then used to compute a new transmission period T_r at next parameter change.



(a) (b)

Figure 7: Mean delays prediction

The second Kalman filter gives a 10s prediction of $m_t(t)$, in other words, $m_t(t + 10s)$. The results showed in figure 7 are good enough to be used to prevent NDR queue emptying 10s before it would happen. It leaves enough time for the system to get in pause mode.

3.6 Computation of New Parameters

Experimentations at short distance showed that the average $GRTT$ was equal to 1,2s versus 23ms for $NRTT$. As $T_r = 200ms$, the time spent in each NDR was equal to $3 \times T_r$. Therefore we had $GRTT = 2 \times 3 \times T_r + NRTT$. In order to have a good resolution during the regulation, T_r should be at most equal to $NRTT$, but transmission frequency is limited by the network bandwidth (which may vary) and by the speed of computer hosted in the local and remote site. The choice of T_r will then depend on the platform used for the teleoperation. In our

case, we have decided to make the following calculus: $T_r = max\left(50ms, \frac{NRTT}{3}\right)$ For our short distance, we would obtain an average $GRTT$ equal to $2 \times 3 \times 50ms = 300ms$ versus 1,2s in first experiments. And at long distance $2 \times 16 \times \frac{610}{3}ms \approx 3,9s$ (T_r would be equal to 200ms so the queues would have the same sizes as in our experimentations) versus 7s. Concerning the nominal size of the NDR queues, we keep the same calculus for η_s as in paragraph 3.1.

4 Algorithm based on High-Order Sliding Modes

One solution to dynamically force the response of the remote actuators to react as a first-order is the sliding modes control law [9]. This method, when the sliding condition is verified, dynamically force the response of the system such that the sliding surface $s = 0$. However, the sliding condition is obtained with a discontinuous control vector which causes a chattering phenomenon at the equilibrium point. Several works try to vanish this problem, but only one method gives a satisfactory result. This method is called High-Order Sliding Modes $HOSM$ [10]. The main goal is the constraint given by the equation $s(x) = 0$ has to be a sufficiently smooth constraint function. The discontinuity does not appear in the first $^{(r-1)th}$ total time derivative (cf. 7).

$$s = \dot{s} = \ddot{s} = ... = s^{r-1} = 0 \tag{7}$$

The relative degree r is defined as :

$$\frac{\partial s^i}{\partial u} = 0, \ \frac{\partial s^r}{\partial u} \neq 0 \tag{8}$$

with $i = 1..(r-1)$. In the case of $r = 1$ in order to avoid the chattering phenomenon, the 2-sliding mode control can be used and the time derivative of the plant control $\dot{u}(t)$ may be considered as the actual control vector. In the 2-sliding mode the control vector $u(t)$ is continuous and the chattering phenomenon disappears.

4.1 Control Law Strategy

Let consider the state representation of the equation (1).

$$\begin{cases} \dot{x}_{1i} = x_{2i} \\ \dot{x}_{2i} = -\frac{1}{\tau_i}x_{2i} + \frac{1}{\tau_i}k_{vi}u_i(t) \end{cases} \tag{9}$$

Where $x_{1i} = q_i$ with $i = 1..6$ is the joint position of the i^{th} actuators, τ_i is the time response of the motor and $u_i(t)$ the control variable. We propose a first-order surface equation with the desired output value $x_{1di} = 0$:

$$s_i(x_i) = -x_{2i} - \lambda_i x_{1i} \tag{10}$$

The 2-sliding mode control considers the time derivative control variable $\dot{u}(t)$ as the control vector. One solution is obtained expressing $\ddot{s}$ with equation (9). The second one is achieved considering the time derivative state model (9) such that the time derivative of the control variable $\dot{u}_i(t)$ appears within $\dot{s}$. We obtain in the second case, the following model:

$$\begin{cases} \dot{x}_{1i} = x_{2i} \\ \dot{x}_{2i} = x_{3i} \\ \dot{x}_{3i} = -\frac{1}{\tau_i}x_{3i} + \frac{1}{\tau_i}k_{vi}\dot{u}_i(t) \end{cases} \tag{11}$$

Working on the new model defined previously (eq.11), we propose a new sliding s_y surface depending on the $s_i(x_i, t)$ function as :

$$s_{yi} = y_{2i} + \alpha_i y_{1i} \tag{12}$$

where $y_{2i} = \dot{s}_i$ and $y_{1i} = s_i$. This sliding surface can be expressed according to the state variables as :

$$s_{yi} = -x_{3i} - (\alpha_i + \lambda_i)x_{2i} - \alpha_i\lambda_i x_{1i} \tag{13}$$

The time derivative control vector $\dot{s}_i(t)$ can be written as the sum of two functions :

$$\dot{u}_i(t) = \dot{u}_{eqi} + \Delta\dot{u}_i \tag{14}$$

where $\dot{u}_{eqi}$ is the equivalent control to achieve $\dot{s}_i = 0$:

$$\dot{u}_{eqi} = \frac{\tau_i}{k_{vi}}\left(\frac{x_{3i}}{\tau_i} - (\lambda_i + \alpha_i)x_{3i} - \alpha_i\lambda_i x_{2i}\right) \tag{15}$$

and $\Delta\dot{u}_i$ is the discontinuous second term used to compensate the unmodelled dynamics of the system :

$$\Delta\dot{u}_i = K_i sgn(s_{yi}) \tag{16}$$

where K_i is the maximum input value of the actuators. The η-reachability equation defined below (17) is a necessary condition for a finite time convergence.

$$s_{yi}\dot{s}_{yi} \leq -\eta|s_{yi}| \tag{17}$$

The equations (15) and (16) verify the condition (17) which ensures the finite time convergence. Integrating equation (17), we obtained the following result :

$$t_{ei} \leq \frac{\alpha_i\lambda_i\tau_i}{k_{vi}K_i} \tag{18}$$

where t_e is the upper bound for the time convergence such that the surface $s_i(t_{ei}) = 0$. Using equations (14), (15) and (16), the global control vector $u_i(t)$ can be expressed as :

$$u_i(t) = u_{eqi}(t) + \int K_i sgn(s_{yi})dt$$

This control vector is continuous and the chattering phenomenon disappears.

4.2 Stability

Let define a Lyapunov function such that :

$$V(\mathbf{X}, t) = \frac{1}{2}\mathbf{X^t P X}$$

with $\mathbf{X} = [x_1, x_2, x_3]^T$ the state vector and $\mathbf{P}$ a matrix defined by :

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & -\alpha_i\lambda_i \\ 0 & \alpha_i\lambda_i - \frac{\alpha_i\lambda_i}{\tau_i} & -(\alpha_i + \lambda_i) \\ -\alpha_i\lambda_i & -(\alpha_i + \lambda_i) & -1 \end{bmatrix}$$

The time derivative term of the Lyapunov function can be written as :

$$\dot{V} = (-\alpha_i - \lambda_i + \frac{1}{\tau_i})x_{2i}^2 + \frac{\alpha_i\lambda_i}{\tau_i}x_{1i}x_{3i} - \frac{k_{vi}K_i}{\tau_i}|s_{yi}|$$

where V is positive defined and $\dot{V}$ is negative defined under the following condition :

$$\frac{\alpha_i\lambda_i}{\alpha_i + \lambda_i} < \frac{1}{\tau_i} \tag{19}$$

We therefore conclude, under the stability condition (eq. 19) that the system is exponentially stable.

5 Simulation Results

First of all, we validate in simulation the theorical results obtained previously (stability, finite time convergence). Afterwards, we introduce global simulation results made up of base station, network and remote system using the *HOSM* control law whose the sliding surface coefficients are adapted according to *NRTT*, verifying equation (5).

5.1 High-Order Sliding Mode Algorithm

A Simulink model based on the equivalent model of the remote actuators defined by the equation (11) allows to verify the stability equation. This condition achieved in (19) can be rewritten as :

$$\alpha_i < \frac{\lambda_i}{\lambda_i \tau_i - 1} \tag{20}$$

We have performed the model by tuning the coefficients λ_1 and α_1 in order to reach the stability limit. We present the figure (8) these values.

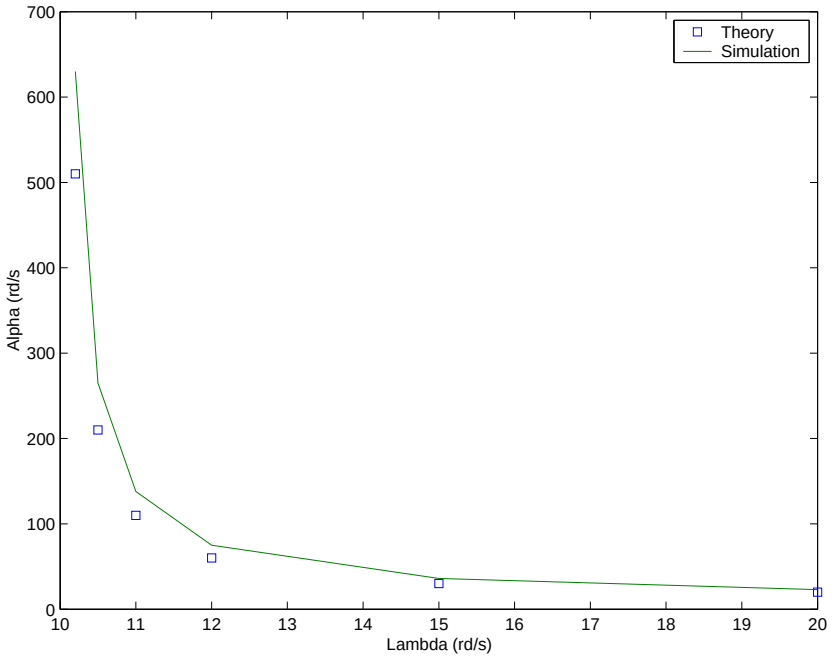
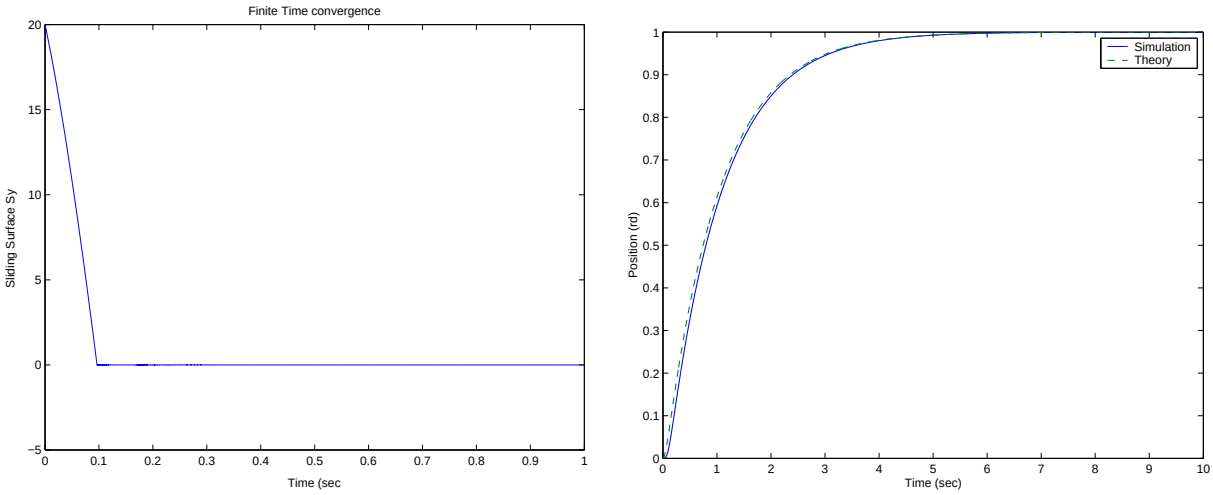


Figure 8: Stability limit

This simulation result is achieved with $\tau_1 = 0,1s$, $K_1 = 15V$ and $k_{v1} = 1rd/V$. We also added the theorical points of the stability condition. These curves show a similar behavior between the simulator and the theorical result. The finite time convergence observation is carried out with $\alpha_1 = 1rd/s$ and $\lambda_1 = 20rd/s$. Using previous values of the actuator model and equation (18), we achieve $t_{e1} = 0,13s$. This simulation result is showed figure 9(a) where ($t_e \sim 0,1s$). The time step response of the remote system is then forced dynamically by the sliding surface $s_{y1}(t)$ (eq.(13)). Reaching the sliding straight, the time response is defined by the solution of the ordinary differential equation $s_{y1}(t) = 0$. In the case of this step response ($x_{1d}(t) = u(t)$), the time solution can be expressed as :

$$x_{1T}(t) = u(t)\left(1 - \frac{\alpha_1}{\alpha_1 - \lambda_1}e^{-\lambda_1 t} + \frac{\lambda_1}{\alpha_1 - \lambda_1}e^{-\alpha_1 t}\right)$$



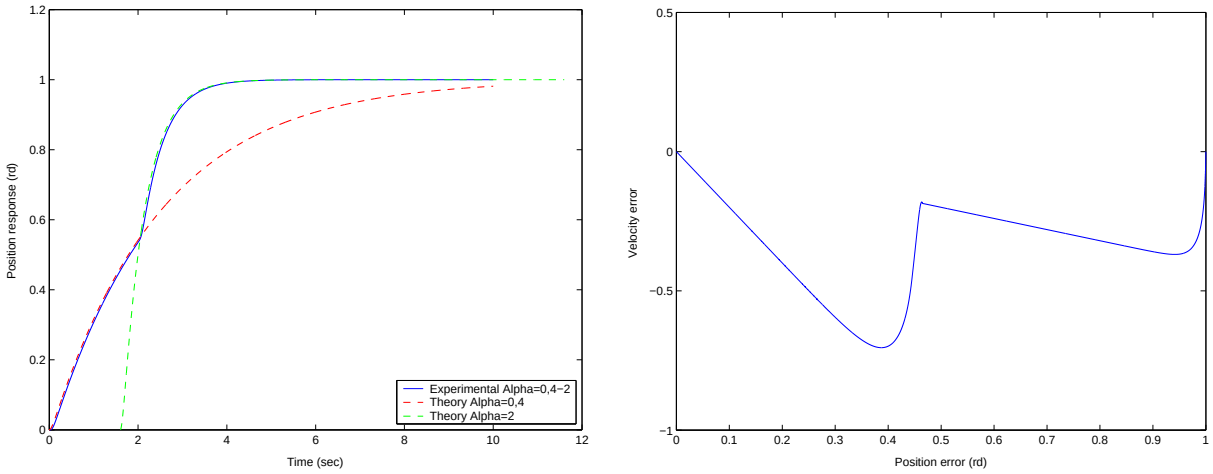
(a) Finite time convergence (b) Position Response

Figure 9: Step Response

If the coefficient $\alpha_1 << \lambda_1$, the time response of x_{1T} is equivalent to the first-order time response. This theorical time response $x_{1T}(t)$ is showed figure 9(b) versus the simulation result. The simulation time response $x_{1T}(t)$ is closed to the theorical time response of the sliding surface. That proves the position trajectory is always defined by the sliding surface when $t > t_e$. It is an important result given in the worst case when the operator sends brutal step orders in a teleoperation task.

5.2 Switching of sliding surfaces

When the constant delay T_r is changed by the *NDR*, the *HOSM* algorithm has to modify each α_i. Figure (10) shows a time step response of a sliding surface switching with two values of α_1 : $\alpha_1(1) = 0,4rd/s$ and $\alpha_1(2) = 2rd/s$ with $\lambda_1 = 20rd/s$. The experimental curve is very close to theorical responses. This remark allows



(a) Time Response Switching of Surface(b)

Figure 10: Switching of Surface

us to conclude that the sliding condition is always verified even during the switching period. Figure (10) presents

the switching of sliding surface with $-x_{21}(t) = f(1 - x_{11}(t))$.

5.3 Teleoperation with time delay

We have performed this control algorithm based on *HOSM* method on a simulation of teleoperation task. The operator sends some desired position steps (see figure (11)) to the remote manipulator.

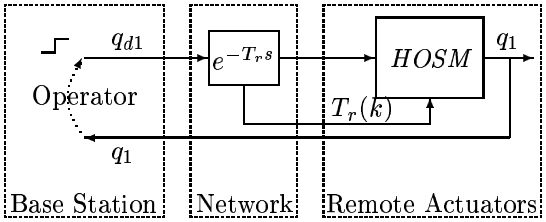


Figure 11: Teleoperation task

We assume that the network delay (T_r) takes two different values during the simulation time : $T_r(1) = 1s$, $t \leq 12s$ and $T_r(2) = 250ms$, $t > 12s$. This can happen when the network behavior changes drastically and the *NDR* has to switch to a new T_r to be able to provide a constant virtual delay anew.

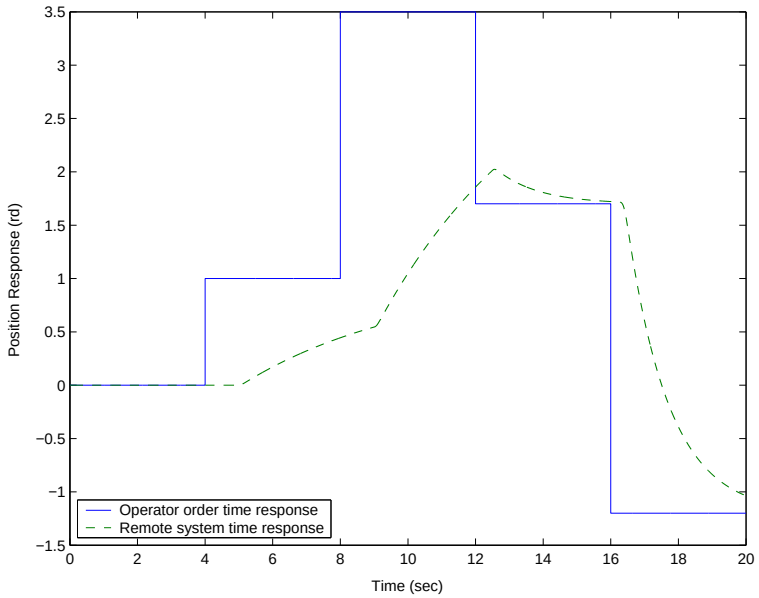


Figure 12: Effects of the it HOSM on $q_1(t)$

The equation (5) gives a solution to keep a same stability level when the average delay changes. Choosing $C_1 = 0,2$ for the base station, we obtain two values of the sliding surface coefficient $\alpha_1(1) = 0,2rd/s$ and $\alpha_1(2) = 0,8rd/s$. Figure (12) illustrates a teleoperation task using a *HOSM* algorithm. The bandwidth of the remote system is adapted with respect to the network delay. The operator has always a stable system and if the mean delay tends to infinity then the manipulator bandwidth (α) will tend to zero.

6 Conclusion

The method we have proposed in this paper concerns a global stable solution to remote control a manipulator robot via an asynchronous network such Internet. This method assumes a slow step varying delay. This way, it can be used as an upper layer of previous low-level works such as our *NDR*, upon a real network with jitter. This study should be implemented on experimental setup in fews months to validate these results. This control method associated with the previous results (*NDR*), provides an efficient tool for the telerobotics systems using TCP/IP network.

References

[1] A. Lelevé, P. Fraisse, A. Crosnier, P. Dauchez, F. Pierrot, "Towards Virtual Control of Mobile Manipulators", *Proc. of the IEEE ICRA'98*, Leuven, Belgium, pp.2971-2976, 1998.

[2] R. Oboe, P. Fiorini, "Internet-based telerobotics: problems and approches", *Proc. International Conference on Advanced Robotics*, Monterey, USA, pp765-770, July 1997.

[3] G. Niemeyer, J.J. Slotine, "Designing Force Reflecting Teleoperators with large time delays to appears as virtual tools", *Proc. of the IEEE ICRA'97*, pp. 2212-2218, 1997.

[4] A. Lelevé, P. Fraisse, P. Dauchez, "Telerobotics over IP Networks: Towards a low-level Real-Time Architecture", *Proc. of the IROS'01, IEEE/RSJ*, Maui, Hawaii, USA. Oct.2001.

[5] P. Fraisse, A. Leleé, "Teleoperation with Time Delay : Adaptive Control Law based on High-Order Sliding Modes" *Submitted of the ICRA'03, IEEE*, Taipan, Taiwan, May 2003.

[6] E. Freund, "Fast nonlinear control with arbitrary pole placement for industrial robots and manipulators", *The International Journal of Robotics Research*, Vol. 1(1), pp.65-78, 1982.

[7] A. Lelevé, P. Fraisse, P. Dauchez, F. Pierrot, "Teleoperation through the Internet : Experimental Results with a Complex Manipulator", *Proc. of the International Conference on Robotics*, Tokyo, Japan,pp. 63-70, 1999.

[8] A. Lelevé, "Contributions à la téléopération de robots en présence de délais de transmission variables", *Thèse de doctorat, Université Montpellier II, in french*, December 2000.

[9] V. Utkin,"Variable structure systems with sliding modes", *IEEE Trans. on Automatic Control*, Vol. AC-22, No2, pp212-222, 1977.

[10] L. Fridman, A. Levant, "High-Order Sliding Modes", in Sliding Modes control in Engineering, *Ed. W. Perruquetti, J.P. Barbot, Marcel Dekker, Inc., New York*, pp.53-101,2002.

[11] K. Hirai, Y. Satoh, "Stability of a system with Variable Time Delay", *IEEE. Trans. on Automatic Control*, Vol. AC-25, N°3, June 1980.

[12] S. Moon, P. Skelley, D. Towsley, "Estimation and Removal of Clock Skew from Network Delay Measurements", *Proc. of IEEE Infocom'99, New-York*, March 1999.

[13] F. Janabi-Sharifi, V. Hayward, C-S. J. Chen, "Novel Adaptive Discrete-Time Velocity Estimation Techniques and Control Enhancement of Haptic Interfaces", *http://citeseer.nj.nec.com/22148.html*