View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by INRIA a CCSD electronic archive server

AsynCar, a Radio-Controlled Vehicle for Asynchronous Experiments: Implementation of an Event-Based Cruise Control

Sylvain Durand, Julien Minet, Fermi Guerrero Castellanos, Nicolas Marchand

To cite this version:

Sylvain Durand, Julien Minet, Fermi Guerrero Castellanos, Nicolas Marchand. AsynCar, a Radio-Controlled Vehicle for Asynchronous Experiments: Implementation of an Event-Based Cruise Control. CCE 2011 - 8th International Conference on Electrical Engineering, Computing Science and Automatic Control, Oct 2011, Mérida City, Yucatan, Mexico. s.p. hal-00626001

HAL Id: hal-00626001

https://hal.archives-ouvertes.fr/hal-00626001

Submitted on 23 Sep 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

ASYNCAR, a Radio-Controlled Vehicle for Asynchronous Experiments
Implementation of an Event-Based Cruise Control

Sylvain Durand[1,2,3], Julien Minet[3,4], Jose Fermi Guerrero Castellanos[5], Nicolas Marchand[3,4]

[1] CRI PILSI, [2] INRIA, [3] GIPSA-lab, [4] CNRS – Grenoble, France.
[5] Faculty of Electronics, Autonomous University of Puebla (BUAP) – Puebla, Mexico.
E-mail: sylvain.durand@inrialpes.fr

***Abstract*— The main contribution of this paper is to develop an experimental platform in order to test some event-based control strategies. Contrary to the time-triggered fashion which calculates the control signal at each sampling time, an event-driven controller updates the control signal only when required. This theoretically allows to reduce the computational cost. In this paper, we propose to firstly test an asynchronous cruise control mechanism. Some first results clearly show a noticeable reduction of the mean control computation cost, which is really encouraging for developing such a platform.**

***Keywords*— Experimental platform, asynchronous cruise control, event-based control**

INTRODUCTION

The classical so-called discrete time framework of controlled systems consists in sampling the system uniformly in time with a constant sampling period h_{nom} and in computing and updating the control law every time instants $t_k = k \cdot h_{nom}$. This field, denoted the time-triggered case (or the synchronous case in sense that all the signal measurements are synchronous), has been widely investigated in [1], even in the case of sampling jitter or measure loss that can be seen as some asynchronicity. However, some works addressed more recently event-based sampling (also called asynchronous) where the control law is event-driven [2], [3], [4], [5], for instance when the output crosses a certain level $q_j = j \cdot q_{nom}$. Thus in this scheme, the term *sampling period* denotes a time interval between two consecutive level crossings of the measure, that is two successive sampling instants, and the sampling intervals are hence not equidistant in time anymore.

Many reasons are motivating the event-triggered systems and in particular because more and more asynchronous systems or systems with asynchronous needs are encountered. Actually, the demand of low-power electronic components in all embedded and miniaturized applications encourages companies to develop asynchronous versions of the existing time-triggered components, where a significant power consumption reduction can be achieved by decreasing the samplings and consequently the CPU utilization: about four times less power than its synchronous counterpart for the 80C51 microcontroller of Philips Semiconductors in [6] for example. An original and simple event-based PID control architecture was proposed in [7]. The suggested scheme updates the control signal only when required. Whereas an event was enforced with a mix of level crossings and a maximal sampling period (for stability reason) in the initial approach, this maximal period was then removed in [8] because, in fact, the Nyquist-Shannon sampling condition is no more consistent thanks to the level detection. Different *event-based PID algorithms without safety limit condition* were also developed. They clearly showed in simulation that the CPU cost can be considerably reduced without performance loss. Nevertheless, a safety limit is uselessly applied since then in the literature [9], [10], [11], [12] and, therefore, we propose to clearly highlight the efficiency of an approach without safety limit condition by implementing such a controller in a real-time testbed. The ASYNCAR experimental platform was thus especially developed to test some asynchronous techniques. It is a radio-controlled vehicle which embeds an event-based cruise control mechanism. The paper is structured as follows. Firstly, the platform is presented in section I. The event-based cruise control principle is then depicted in section II and some experimental results are provided in section III.

I. THE ASYNCAR PLATFORM

The ASYNCAR platform is depicted in Fig. 1. It is based on a *1/18-scale Mini Rock Crawler* radio-controlled car from LOSI [13], where some extra components were added to make possible the cruise control of the vehicle.

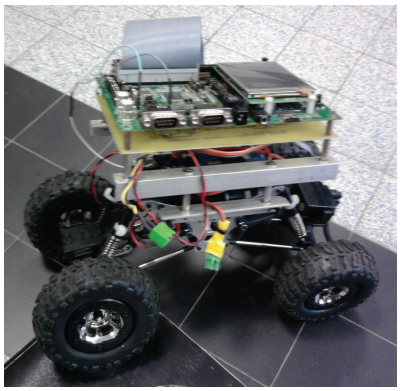


Fig. 1. The ASYNCAR platform.

A. The microcontroller

The main added component is a STM3210C-EVAL electronic card from STMICROELECTRONICS [14]. This evaluation board is represented in Fig. 2. The development kit

embeds a ARM microcontroller and several generic tools which will allow the implementation of a cruise control mechanism. It aims at providing a speed control signal to the car from some given setpoint and some measurements. Thus, different connection ports exist, such as RS232 or Ethernet ports. A LCD screen is also present and could be useful for debugging whereas a microSD card is available and will allow to store some log information.

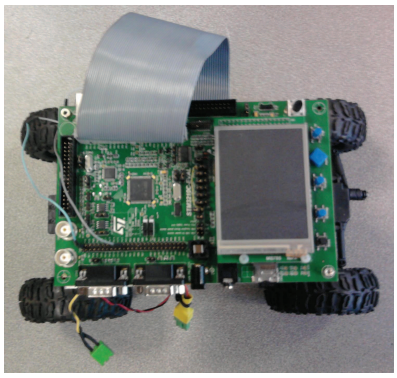


Fig. 2. The electronic card placed on top of the ASYNCAR platform.

B. The shaft encoder

A speed sensor is also required in the cruise control scheme. Therefore, a shaft encoder was directly connected to the drive-shaft of the car's motor in order to dynamically measure its current velocity.

C. The extra electronics

Another PCB was designed to make compatible the STM32 card with the car and the shaft encoder, adapting the voltage levels of the different components.

D. The testbed

A testbed was developed to fix the ASYNCAR on a hard platform in order to make easier the different experiments. This testbed is shown in Fig. 3. It allows to make indoor experiments (when only small place is available in a room) and some perturbations can be added on the wheel of the car. Nevertheless, the ASYNCAR can also run without this testbed, using a remote control for instance.

E. The data acquisition

The data acquisition is performed in a real-time framework and all data are stored in the microSD card. A network connection then allows to get these information in order to next analyze them off line.

II. EVENT-BASED CRUISE CONTROL

In fact, the motion of a vehicle can be controlled with a proportional-integral-derivative (PID) strategy. The cruise control principle is firstly introduced in subsection II-A. Then, whereas the classical time-triggered control scheme is called back in subsection II-B, some event-based approaches are introduced in subsections II-C and II-D.

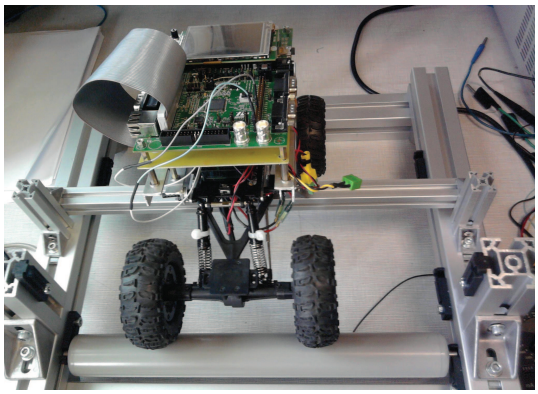


Fig. 3. The tesbed used to fix the ASYNCAR platform.

A. Cruise control principle

As explained in introduction, event-based control is a computational cost-aware solution especially for all systems which do not need to be constantly controlled. For this reason, we decided to highlight such a scheme with the cruise control mechanism depicted in [15]. Indeed, the desired speed of the car is constant most of time and, in fact, a new control signal is only required when the setpoint changes or when the load (i.e. the slope of the road) varies. The equation of motion of the car is

$$m\dot{\nu} = F - F_d \tag{1}$$

where ν is the velocity and m the mass of the vehicle. The driving force F is generated by the engine, whose torque is proportional to a control signal $0 \leq u \leq 1$ that controls the throttle position and depends on engine velocity too.

$$F = \alpha_n u T_m \left[1 - \beta \left(\frac{\alpha_n \nu}{\omega_m} - 1\right)^2\right] \tag{2}$$

where the maximal torque T_m is obtained for a given engine speed ω_m and β. A physical interpretation of α_n, which depends on the gear ratio n, is the inverse of the effective wheel radius. On the other hand, the disturbance force F_d has three major components due to the gravity F_g, the rolling friction F_r and the aerodynamic drag F_a, which yields

$$F_d = F_g + F_r + F_a \tag{3}$$
$$\text{with} \quad \left| \begin{array}{l} F_g = mgsin(\theta) \\ F_r = mgC_r sgn(\nu) \\ F_a = \frac{1}{2}\rho C_d A \nu^2 \end{array} \right.$$

where g is the gravitational constant, C_r and C_d are the rolling friction and the shape-dependent aerodynamic drag coefficients respectively, ρ is the density of air, A is the frontal area of the vehicle and θ is the slope of the road, that is the disturbance. Such a system can then be approximated by a first-order system and so is quite simple a PID control.

B. Time-based control

The textbook PID controller in frequency domain is [16]

$$U(s) = K\left[E(s) + \frac{1}{T_i s}E(s) + T_d s E(s)\right] \tag{4}$$

where $U(\cdot)$ is the control signal and $E(\cdot)$ the error between the measured output of the controlled system and a given setpoint. K, T_i and T_d are some tunable parameters. A discrete time controller is finally obtained, that is

$$\begin{aligned} u_p(t_k) &= Ke(t_k) \\ u_i(t_k) &= u_i(t_{k-1}) + K_i h_{nom} e(t_k) \\ u_d(t_k) &= \frac{T_d}{T_d + Nh_{nom}} u_d(t_{k-1}) \\ &\quad + \frac{KT_dN}{T_d + Nh_{nom}} \Big[e(t_k) - e(t_{k-1}) \Big] \end{aligned} \tag{5}$$

where $K_i = {}^1\!/_{T_i}$. The proportional part $u_p(\cdot)$ was straightforward while the backward difference approximation was used for integral and derivative parts $u_i(\cdot)$ and $u_d(\cdot)$ respectively. A low-pass filter was also added in the derivative term to avoid problems with high frequency measurement noise, where N denotes the low-pass filter gain. Finally, t_k and t_{k-1} are the current and last sampling time respectively.

C. Årzén's event-based control

As explained in introduction, *Karl-Erik Årzén* initially proposed an original event-based PID controller in [7] in 1999. The basic setup consists in two parts: ***i)*** *a time-triggered event detector* used for the level-crossing detection and ***ii)*** *an event-triggered controller* which calculates the control signal. The first part runs with the constant sampling period h_{nom} – that is the same as for the corresponding conventional time-triggered controller – whereas the second part is driven by some requests sent by the event detector. These requests are provided when a new control signal has to be calculated and, therefore, the length of the varying sampling intervals $h(\cdot)$ for the control part is the time between two successive requests. Let τ_a denote the beginning time of the current control sample, that is the last time a request was sent by the event detector because the input signal crossed a level. Respectively, let τ_{a+1} denote the next time where a control signal will be calculated and so on. Note that the event-based sampling instant τ_a occurs at a discrete instant time t_k by construction. Furthermore, let $h(\tau_a)$ denote the sampling interval used to calculate the current control signal, i.e. $h(\tau_a) = \tau_a - \tau_{a-1}$. The initial Årzén's setup updates the control signal either ***i)*** *when the absolute error crosses a certain level*, that is when the current error crosses a certain limit, i.e. $abs\big(e(\tau_a)\big) > q_{nom}$, or ***ii)*** *if the maximal sampling period is achieved*, i.e. $h(\tau_a) \geq h_{max}$. This second condition was added to guarantee the stability by fulfilling the Nyquist-Shannon sampling condition.

Actually, the relative measurement was used instead of the absolute one in the initial proposal. Also, a small discretization improvement was also proposed (one could refer to [8] for further details).

D. Event-based control without safety limit condition

We proposed in [8] to remove the safety limit condition $h(\tau_a) \geq h_{max}$ – in order to improve and simplify the event-based setup (and because the Nyquist-Shannon sampling condition is no more consistent in asynchronous systems thanks to the level detection) – which results in only computing the control signal when required from a performance point of view. However, this modification requires to change the integral part in the control law, i.e. $u_i(\tau_a) = u_i(\tau_{a-1}) + K_i he(\tau_a)$, where $he(\tau_a) = h(\tau_a)e(\tau_a)$ afterwards denotes the integral gain. Indeed, the product $h(\cdot)e(\cdot)$ can become huge in absence of event or when the setpoint varies, causing some important overshoots. Actually, a steady-state interval can be divided into ***i)*** a first part where the sampling interval increases a lot but the error remains small and ***ii)*** a second part where the error becomes very large but only during a few instant. Therefore, the product $he(\cdot)$ does not cause any problem anymore since $h(\cdot)$ and $e(\cdot)$ compensate themselves each other. Finally, a bound of the integral gain is

$$he(\tau_a) \leq \big[h(\tau_a) - h_{nom}\big] q_{nom} + h_{nom} e(\tau_a) \tag{6}$$

This observation is taken into account in the proposed algorithms in [8], whose best ones are summarized before testing them on the experimental platform in section III.

Saturation of the integral gain

This algorithm consists in bounding the product $he(\cdot)$ after a long steady-state interval in order to reduce the overshoots. Thus, the product is saturated such that

$$he(\tau_a) = \big[h(\tau_a) - h_{nom}\big] q_{nom} + h_{nom} e(\tau_a) \tag{7}$$

Exponential forgetting factor of the sampling interval

Another method consists in adding a forgetting factor of the sampling period so that, after a long steady-state interval, the value of $h(\cdot)$ is reduced enough to not impact the control signal too much. An exponential function is chosen to decrease its impact as the elapsed steady-state time increases, that is

$$h_{exp}(\tau_a) = \big[h(\tau_a) - h_{nom}\big] \exp\Big(h_{nom} - h(\tau_a)\Big) \tag{8}$$

The new sampling interval $h_{exp}(\cdot)$ is thus used in the integral part, such that

$$he(\tau_a) = h_{exp}(\tau_a) e(\tau_a) \tag{9}$$

At the end, this function leads to have a nominal sampling period during the transients, i.e. when $h(\tau_a) = h_{nom}$, and an exponential decreasing sampling period during the steady-state intervals.

Hybrid algorithm

This algorithm is a mix between the two previous ones. Thus, we propose to use the exponential forgetting factor into the algorithm with saturation, which leads

$$he(\tau_a) = \big[h_{exp}(\tau_a) - h_{nom}\big] q_{nom} + h_{nom} e(\tau_a) \tag{10}$$

III. Experimental results

In this section, we propose to test the different event-based PID strategies on the AsynCar platform.

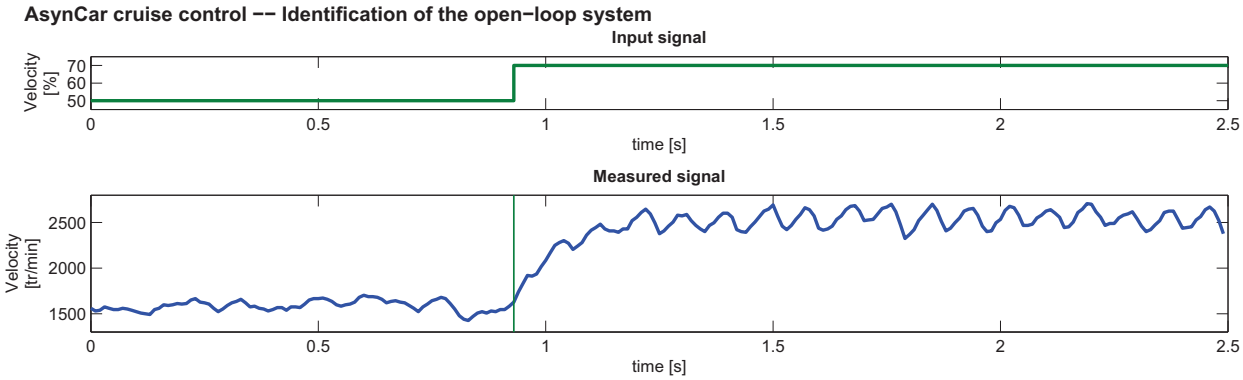


Fig. 4. Identification of the system response when applying a step in input of the ASYNCAR's motor.

A. System model and controller's parameters

The model of the velocity of a vehicle was introduced in subsection II-A but, in fact, an open-loop identification easily gives a first-order transfer-function between the measured speed and the control signal. The system response is represented in Fig. 4, which leads to

$$H(s) = \frac{G}{1+Ts} \tag{11}$$

where $G = 0.45$ is the steady-state gain. In fact, the system velocity varies from 0 to about $3000\,tr.min^{-1}$ (measured by the encoding shaft when the input is $100\,\%$) but, afterwards, we prefer to work with a percentage of the maximum speed. Note that a negative value is also possible to inverse the direction of the vehicle. A conversion is finally applied to achieve a pulse width modulation (PWM) signal whose duty cycle is $1.5\,ms \pm 0.5\,ms$ with a period equal to $5\,ms$. On the other hand, $T = 180\,ms$ is the time constant of the car. Note that a short time delay induced by the car's technology is neglected. Furthermore, the measured signal is quite noisy (due to the mechanics) and, for this reason, we next propose to add a digital filter using a weighted average of the measured velocity ν. This yields

$$\tilde{\nu}(t_k) = (1-\kappa)\tilde{\nu}(t_{k-1}) + \kappa\nu(t_k) \tag{12}$$

where $\tilde{\nu}$ is the estimated velocity (then used in the control algorithms) and $\kappa = 0.1$ is the weighted value. Also, in order to be as reactive as possible, we suggest to not apply this filtering during the transients or when a perturbation occurs. A solution consists in avoiding the estimation when the variation of the measured velocity becomes important. That is when $\nu(t_k) - \nu(t_{k-1}) > \Delta\nu$, where $\Delta\nu = 6\,\%$ is a parameter fixed by the designer.

Finally, the parameter's values of the different controllers are obtained by pole placement of the closed-loop system with the time-triggered controller. They yield $K = 1$, $T_i = 250$, $T_d = 1.1$ and $N = 20$. The nominal sampling interval is $h_{nom} = 10\,ms$. The event-based controllers are then designed with these same values and they will finally try to be as closed as possible of the time-triggered closed-loop shaping. Also, the maximum sampling interval needed in the Årzén's controller is $h_{max} = 100\,ms$ and the detection level used in all event-driven strategies is $q_{nom} = 4\,\%$. Additionally, an anti-windup mechanism is added since the control signal can only vary from $0\,\%$ to $+100\,\%$ of the maximum speed in order to prevent windup when the actuator is saturated. At the end, the extra term is

$$u_w(\tau_a) = K_a h(\tau_a)\Big[u(\tau_{a-1}) - u_{sat}(\tau_{a-1})\Big] \tag{13}$$

where $u_{sat}(\cdot)$ is the saturated value of the control signal and $K_a = 0.2$ is another tunable parameter. The control law finally becomes

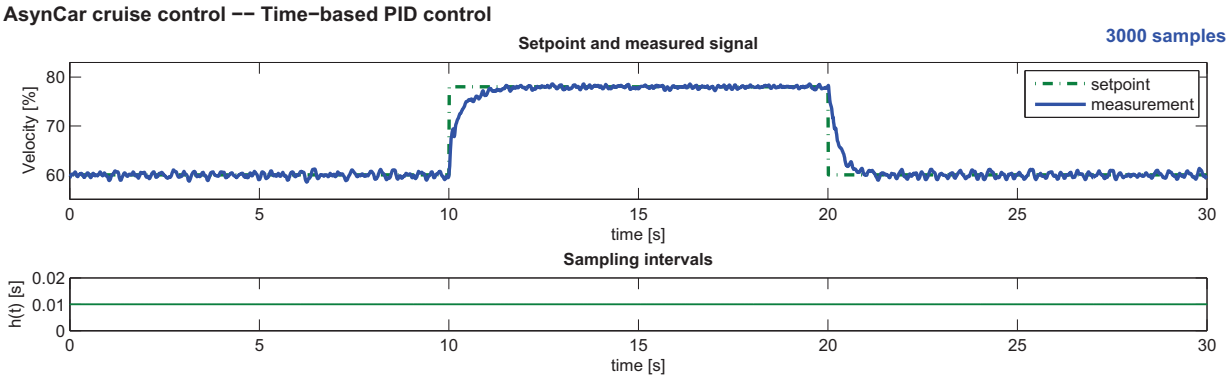
$$u(\tau_a) = u_p(\tau_a) + u_i(\tau_a) + u_d(\tau_a) + u_w(\tau_a) \tag{14}$$

where proportional, integral and derivative terms were defined in (5).

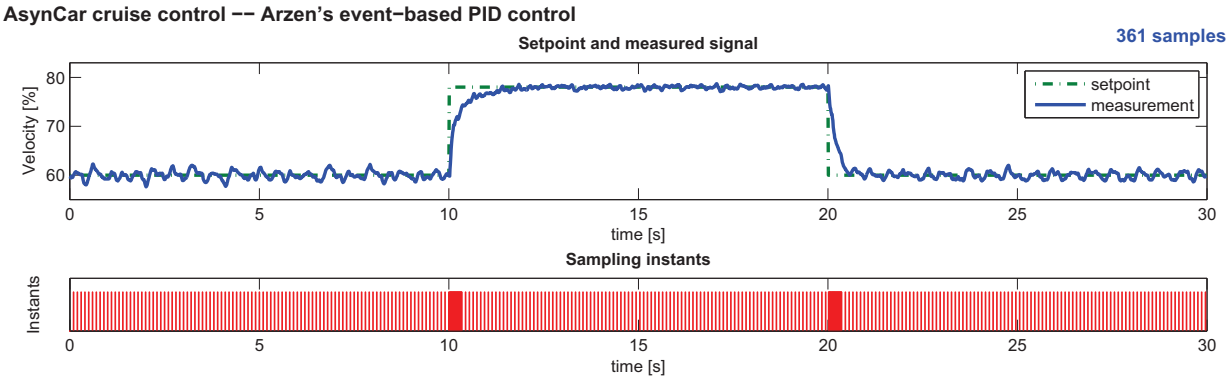
B. Comparison of the algorithms

The first experiment runs the conventional approach. The results are represented in Fig. 5(a), where the top plot shows the setpoint and the measured velocity. The bottom plot shows the sampling intervals which are, of course, all equal to h_{nom} in this time-triggered case. The number of samples needed to perform the testbench is also indicated in the right-top corner of the figure. The system is then tested with the Årzén's controller. The experimental results are shown in Fig. 5(b), where the bottom plot now refers to the sampling instants: an event is drawn each time the control signal is updated. This representation will be preferred in the next figures. The control law is now event-driven when the measured signal crosses a given level, that is why a lot of samples occur during the transients. Moreover, a maximal sampling interval was also introduced enforcing an event even if the measurement remains unchanged. The resulting behavior can be seen during the steady-state intervals where some samples hit every $0.1\,s$. This principle allows to considerably reduce the number of samples anyway (about $88\,\%$ less than in the conventional case) with similar final performance.

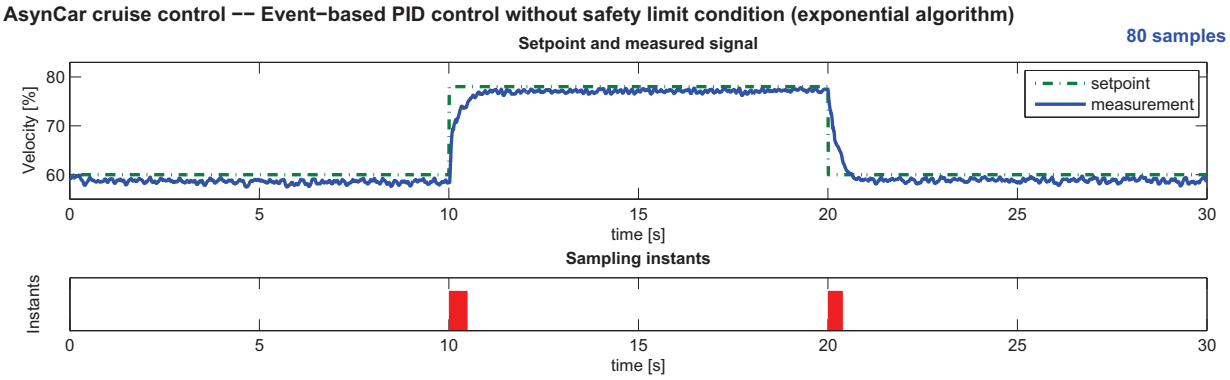
Our proposal finally consists in removing the safety limit condition in order to decrease again the number of samples. In this paper we only test both algorithms ***i)*** with an exponential forgetting factor of the sampling interval



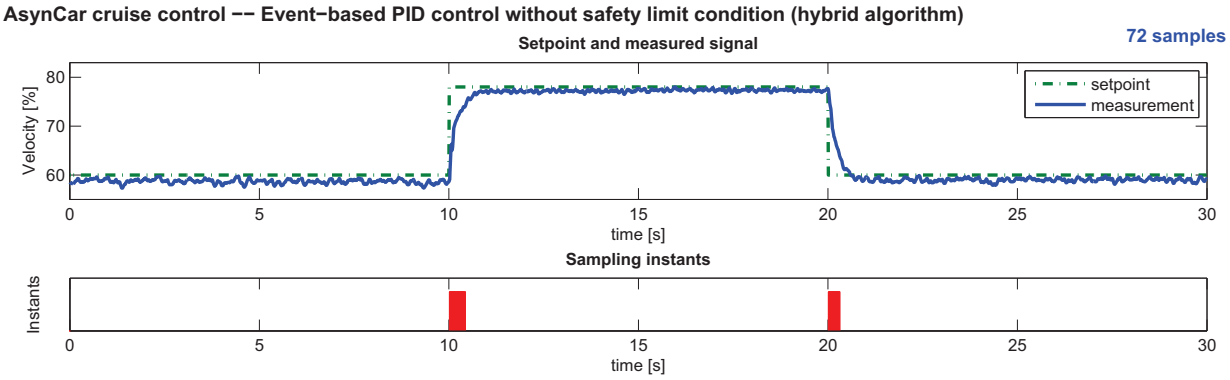
(a) Classical time-triggered PID control.



(b) Årzén's event-based PID control.



(c) Event-based PID control without safety limit condition: algorithm with an exponential forgetting factor of the sampling interval.



(d) Event-based PID control without safety limit condition: hybrid algorithm.

Fig. 5. Control of the velocity of the AsynCar: comparison of the existing techniques and the event-based proposals without safety limit condition.

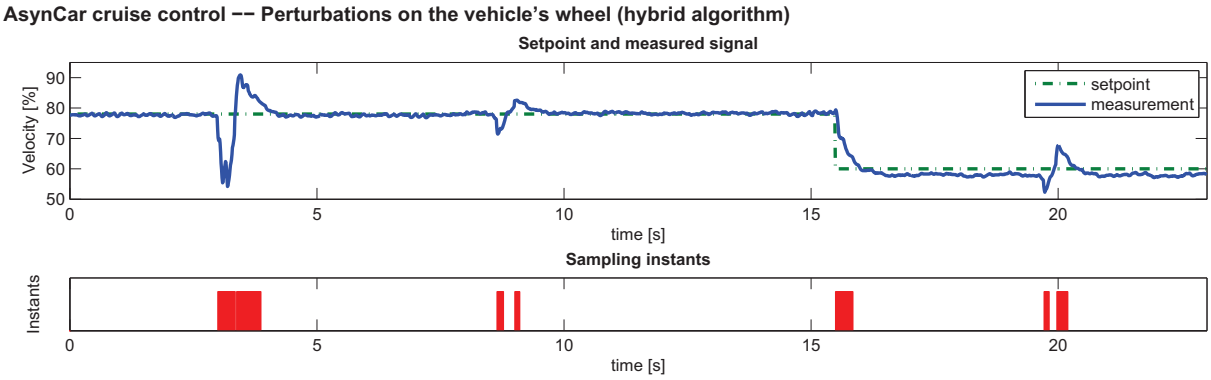


Fig. 6. Analysis of the hybrid algorithm when applying some perturbations on the wheel of the AsynCar.

and ***ii)*** with hybrid strategy. The experimental results are provided in Fig. 5(c) and (d) respectively. These algorithms without safety limit condition allow to considerably reduce the number of samples (about $97\,\%$ and $80\,\%$ of samples less than in the conventional and Årzén's cases respectively). The vehicle still tracks the given setpoint with a system response as fast as previously, but one could remark a steady-state error due to the level detection. Actually, its value is important because of the noise (even after numerical filtering). Consequently, the event-based proposed approaches have to make a tradeoff between performance and computational cost. At the end, these results for a simple first-order system are very encouraging and advantages of the asynchronous scheme are clearly demonstrated.

C. Perturbations and robustness

The AsynCar is then submitted to some perturbations – slowing down the wheels – in order to see if the proposed scheme still works when the system does not perform as well as expected. The experimental results for the hybrid algorithm are depicted in Fig. 6. The system reacts as soon as a perturbation occurs at about 3, 9 and $20\,s$ (because the measurement crosses the detection level). The asynchronous PID control without safety limit condition hence allows the velocity to track the reference even in case of perturbations. The robustness is hence demonstrated in practice.

Conclusions and Future Works

An experimental platform was introduced, called the AsynCar platform, which aims at testing some event-based control schemes. In this paper, an asynchronous cruise control mechanism was presented in order to drive the radio-controlled vehicle with a given speed setpoint. The practical implementation clearly gives good performance with a minimum of samples: more than $97\,\%$ of samples less than with the classical PID controller. The advantage of an asynchronous scheme is hence highly highlighted and the encouraging results strongly motivate to continue developing event-based control strategies. Next step is to develop a full asynchronous scheme where the communication with the remote control will be based on an event-based scheme too.

Acknowledgment

This project is part of a *collective project* in the ENSE3 engineer school curriculum (from Grenoble university), where a eight-students team works on a project with some technical and team management constraints [17].

References

[1] K.J. Åström and B. Wittenmark. *Computer Controlled Systems, 3rd Edition*. Prentice Hall, 1997.

[2] K.J. Åström and B. Bernhardsson. Comparison of Riemann and Lebesque sampling for first order stochastic systems. In *Proceedings of the 41st IEEE Conference on Decision and Control*, 2002.

[3] J.H. Sandee, W. Heemels, and P.P.J. van den Bosch. Event-driven control as an opportunity in the multidisciplinary development of embedded controllers. In *Proceedings of American Control Conference*, pages 1776–1781, 2005.

[4] J. Lunze and D. Lehmann. A state-feedback approach to event-based control. *Automatica*, 46:211–215, 2010.

[5] A. Anta and P Tabuada. To sample or not to sample: Self-triggered control for nonlinear systems. *IEEE Transactions on Automatic Control*, 55:2030 – 2042, 2010.

[6] H. Van Gageldonk, K. van Berkel, A. Peeters, D. Baumann, D. Gloor, and G. Stegmann. An asynchronous low-power 80C51 microcontroller. In *Proceedings of the 4th International Sympsonium on Advanced Research in Asynchronous Circuits and Systems*, pages 96–107, 1998.

[7] K-E Årzén. A simple event-based PID controller. In *Preprints of the 14th World Congress of IFAC*, Beijing, P.R. China, 1999.

[8] S. Durand and N. Marchand. Further results on event-based PID controller. In *Proceedings of the European Control Conference*, 2009.

[9] J. Sánchez, M. Guarnes, S. Dormido, and A. Visioli. Comparative study of event-based control strategies: An experimental approach on a simple tank. In *Proc. of the European Control Conference*, 2009.

[10] J. Sánchez, M. Guarnes, and S. Dormido. On the application of different event-based sampling strategies to the control of a simple industrial process. *Sensors*, 9:6795–6818, 2009.

[11] W. Heemels, J.H. Sandee, and P.P.J. van den Bosch. Analysis of event-driven controllers for linear systems. *International journal of control*, 81:571–590, 2009.

[12] H. Mounier, A. Cela, S.I. Niculescu, and J. Wang. Event driven intelligent PID controllers with applications to motion control. In *accepted to the 18th world congress of IFAC*, 2011.

[13] `http://losi.com/`.

[14] `http://www.st.com/`.

[15] K.J. Åström and R.M. Murray. *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press, 2008.

[16] K.J. Åström and T. Hägglund. *PID controllers: theory, design, and tuning, 2nd Edition*. The Instrumentation, Systems, and Automation Society, 1995.

[17] P. Drogo, N. Bihler, S. Cattey, S. Dupuis, T. Delaforge, A. Aujoulat, Y. Claude, H Kajdas, J. Minet, and S. Durand. Design of an experimental platform for an asynchronous cruise control (in French). ENSE3 collective project, technical report, 2011.