Robust Reset Control

Using Adaptive / Iterative Learning Control

M.C.F. Ivens

Master of Science Thesis

TU Delft Delft University of Technology

Delft Center for Systems and Control

Robust Reset Control

Using Adaptive / Iterative Learning Control

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

M.C.F. Ivens

April 10, 2018

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology

Abstract

Reset controllers can outperform PID controllers and may introduce phase advantage compared to linear PID control. However, in general, reset controllers do not have the same steady state properties as linear controllers, like removing steady state errors. In case of model mismatches and disturbances, this may cause limit cycles (persisting oscillations) in the closed response when (constant) references are tracked. The occurrence of these limit cycles is very unwanted in mechatronic precision systems, since the response does not converge to the desired set-point.

To avoid / remove limit cycles in the response, some reset control methods are currently available. Examples are PI+CI controllers, reset controllers that reset to non-zero values and reset controllers with (adaptive) feedforward. Although the existing methods can be used to overcome the limit cycle problem, they are often dependent on the model of the system or are a trade-off between linear and nonlinear control. Hence, the existing methods are not very robust in general or do not use the full advantage of reset control.

In this thesis, a simple and robust fixed instant adaptive reset controller is developed for a minimum phase SISO system that can be stabilized by standard PID control. The presented adaptive reset algorithm detects if a limit cycle is present and adapts the after reset value in an iterative way until the limit cycle is removed. Although no mathematical proof is given, the idea behind the presented method is explained. Simulations and measurements are performed to show that the algorithm is able to get rid of limit cycles caused by model mismatches and constant input disturbances. Furthermore, it is shown that the adaptive algorithm can be applied to zero-crossing reset as well.

Table of Contents

List of Figures

Preface & Acknowledgments

This thesis on reset control is performed for the degree of Master of Science in Systems & Control at Delft University of Technology. The idea of doing my thesis on reset control came after a discussion with Hassan HosseinNia, PhD of the Precision and Microsystems Engineering department (PME). Initially, I was searching for a project on adaptive or iterative learning control, but Hassan HosseinNia made me interested in reset control as well. Hence, a project was formulated where both control topics could be combined. This resulted in this thesis on reset control where adaptive/iterative learning control is considered to overcome the problem of limit cycles that currently exists in reset control.

During my thesis, I discovered the advantages and variety of reset controllers that are available and discovered that reset control is not an as easy and straightforward control strategy as it looks. The mathematics involved with reset control could be quite complex and sometimes difficult to understand. Despite the hard times, I enjoyed working on this thesis and I am glad with the final obtained results.

Without the excellent support of my supervisors I was not able to finish this thesis. Therefore, my deepest thanks go to my supervisors Hassan HosseinNia, PhD and Niranjan Saikumar, PhD of the PME department. They were always there for me to support me and kept me motivated during the hard times of the project. Furhtermore, I would like to thank prof. dr. ir. Jan-Willem van Wingerden of the DCSC department for his support during my master program and his supervision.

I hope you enjoy reading this thesis.

Mark Ivens

Delft, April 10, 2018

Chapter 1

Introduction

PID controllers are one of the most used controllers in feedback control [1]. They are easy to implement and moreover, they are easy to understand and to tune. However, the performance that can be achieved by (linear) PID controllers is limited by the well-known waterbed effect (see e.g. [2]). In order to overcome the limitations of linear controllers, reset control has become a more popular research field in the past decades. It is for example used to control mechatronic systems like hard disk drives [3] and PZT positioning stages [4]. In this chapter, a short introduction to reset control will be given first. After that, one of the problems of reset control will be discussed, which will result in the problem statement of this thesis.

1.1 Reset controllers

A reset controller is a standard linear controller, e.g. PID, that resets (a subset of) the controller states when a certain reset law is satisfied. Between the resets, the controller behaves like a linear controller. Due to the reset actions, reset controllers can create a phase advantage. This makes it possible to obtain higher bandwidths, having less overshoot and/or a faster settling time compared to standard PID controllers, see for example [5], [6], [3] and examples in [2].

A simple and well-known reset controller is the Clegg Integrator (CI), which was introduced by J.C. Clegg in 1958 [7]. It is an integrator which resets its state to zero when the input is zero. A CI has a similar gain characteristic in terms of slope as a normal integrator, but due to the reset mechanism, it has a phase of -38° instead of -90°. To illustrate how a reset integrator operates, the output of a CI for a sinusoidal input is shown in Figure 1.1. As can be observed, the output resets to zero when the input crosses zero.

A generalization of the CI is a First Order Reset Element (FORE), which was proposed by Horowitz and Rosenbaum in [8]. It is a first order transfer function, which also resets its state when the error crosses zero. Besides the CI and FORE many other reset controllers are currently available. An example of a more general reset controller is given in [2], where the states of the controller can be reset partially.

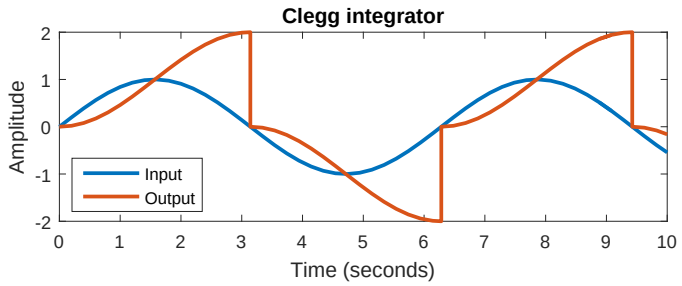

Figure 1.1: Clegg integrator

In general, a reset controller can be defined by equation (1.1). The first line in 1.1 describes the linear base dynamics of the controller (flow mode). The second line describes the value to which the integrator resets when the reset law is satisfied (jump mode). The third line defines the controller output and the last line the initial value. The reset law can be state and/or time dependent. The after reset value $x(t^+)$ may depend on the states of the controller, states of the plant, error signal and/or reference signal. See for example the reset controllers in [9], [4] and [10].

$$\begin{cases} \dot{x}(t) = Ax(t) + Be(t) & \text{if reset law} \neq \text{true} \\ x(t^+) = F(t) & \text{if reset law} = \text{true} \\ u(t) = Cx(t) + De(t) & \\ x(0) = x_0 & \end{cases} \tag{1.1}$$

As mentioned, resetting controller states may introduce a phase advantage which can make it possible to outperform linear PID controllers. Although this is the case, resetting the controller states can have some drawbacks. One of these drawbacks is discussed and illustrated in the next section and will result in the problem statement of this thesis.

1.2 Problem statement

Due to the reset actions of a reset controller, a reset integrator does not have the same steady state properties as a linear integrator has, [2, p.182]. If the integrator resets to a value which is not equal to the steady state value of the control signal, limit cycles may be present in the response. This might for example be due to model uncertainties and/or disturbances. The occurrence of limit cycles is very unwanted in precision systems, like wafer scanners and positioning stages, since the response does not converge to the desired set-point. To illustrate the problem, a few simple examples will be given for a first order system.

1.2.1 Resetting to zero and nonzero value

As a first example to illustrate the limit cycle problem, the influence of resetting to zero and nonzero values will be shown. For this purpose, a negative feedback loop as shown in Figure 1.2 is considered.

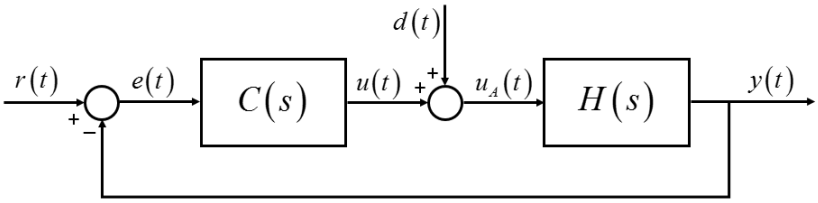

Figure 1.2: Negative feedback loop

Let $C(s)$ be a PI reset controller, given by (1.2) and let the plant be given by $H_1(s) = \frac{1}{s+0.5}$.

$$C := \begin{cases} \dot{x_c}(t) = e(t) & \text{if } \ e(t) \neq 0 \\ x_c(t^+) = 0 & \text{if } \ e(t) = 0 \\ u(t) = \frac{k_p}{\tau_i} x_c(t) + k_p e(t) & k_p = 2, \tau_i = 0.1 \end{cases} \tag{1.2}$$

The closed loop response for a unit step reference is shown in Figure 1.3. For comparison, also the response for the same PI controller without reset is shown. As can be observed, a limit cycle is present in the response controlled by the reset controller and steady state is not achieved.

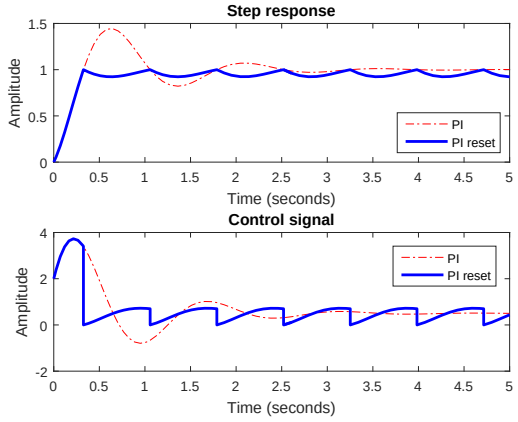

Figure 1.3: Step response of H_1 with $x_c(t^+) = 0$

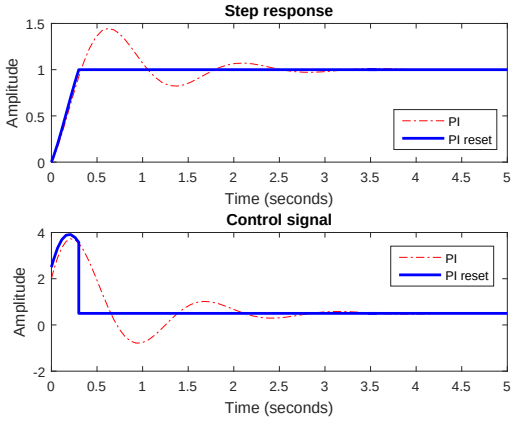

Figure 1.4: Step response of H_1 with $x_c(t^+) = 0.025$

The reason for the occurrence of this limit cycle is that the after reset value $x_c(t^+) = 0$ does not match the steady state value of the control signal of the closed loop linear base system (the system without reset applied). From the control signal of the normal PI controller, it can be observed that the steady state value of the control signal for H_1 is 0.5 instead of 0. Changing the after reset value in equation (1.2) to $x_c(t^+) = 0.5 \cdot \frac{\tau_i}{k_p} = 0.025$ and simulating

the system again leads to the results shown in Figure 1.4. As can be observed, the controller resets one time and the response stays at the desired set-point. A limit cycle is not present anymore, since the after reset value matches the steady state value of the control signal.

1.2.2 Model uncertainty

Suppose that the same controller as given in equation (1.2) is used but that due to model uncertainty the actual plant is given by $H_2 = \frac{1}{s+1}$ instead of $H_1 = \frac{1}{s+0.5}$. The response for $x_c(t^+) = 0.025$ and for $x_c(t^+) = 0.05$ are shown in Figure 1.5. As can be observed, resetting to $x_c(t^+) = 0.025$ result in a limit cycle. This is again caused by the fact that the after reset value does not match the steady state value of the control signal for H_2. By setting $x_c(t^+)$ to $x_c(t^+) = 1 \cdot \frac{\tau_i}{k_p} = 0.05$, the control signal is equal to 1 at reset, which is equal to the steady state value of the control signal. Hence, no limit is present in the response for $x_c(t^+) = 0.05$. In general, when there are no disturbances present, the steady state value of the control signal for a unit step reference is equal to the inverse DC gain of the plant.

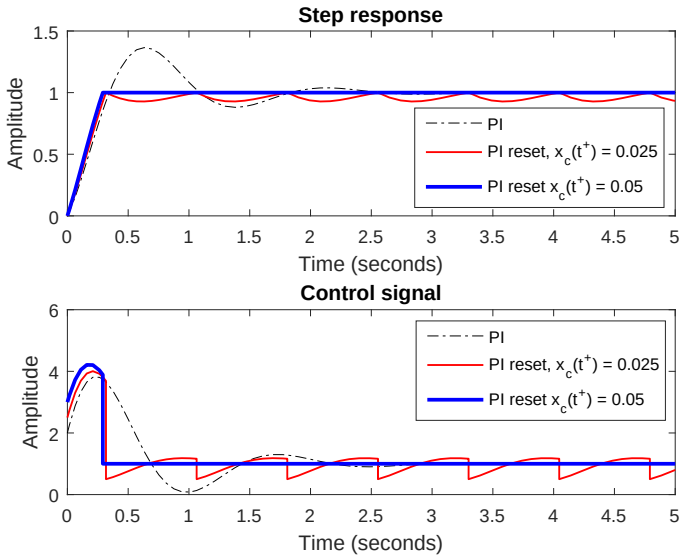

Figure 1.5: Step response of H_2 for wrong and right $x_c(t^+)$

1.2.3 Input disturbance

Limit cycles will also be present if a constant input disturbance at the plant input is present. This for reason that the control input to the plant changes by a constant value due to the disturbance. Hence, the after reset value should be chosen such that the input to the plant, after the disturbance is added, is equal to the steady state value of the control signal.

In order to illustrate the above, a simulation is performed for $H_1(s) = \frac{1}{s+0.5}$ with the PI reset controller as defined in (1.2) with $x_c(t^+) = 0.025$. This is the right after reset value when no disturbance is present. At $t = 5$ a constant disturbance with an amplitude of 0.5 is added at the plant input. The result is shown in Figure 1.6, which clearly shows that a limit cycle is present due to the added disturbance at $t = 5$.

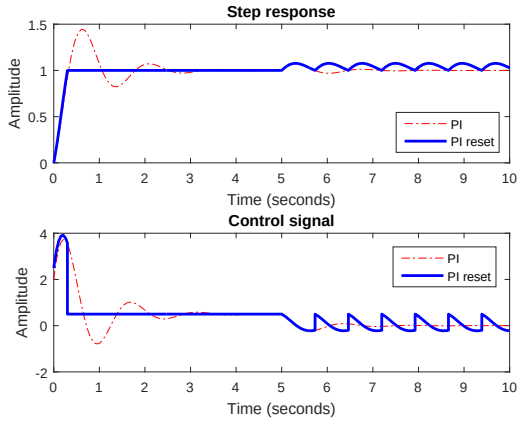

Figure 1.6: Constant input disturbance with amplitude of 0.5 applied at $t = 5$ and wrong $x_c(t^+)$ for disturbance rejection.

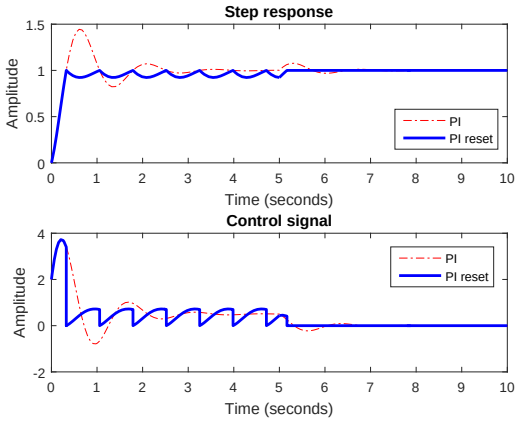

Figure 1.7: Constant input disturbance with amplitude of 0.5 applied at $t = 5$ and right $x_c(t^+)$ for disturbance rejection.

The control signal of the normal PI controller converges to 0 after the disturbance is applied. This is also what is expected since the steady state value of the control input without disturbance was 0.5 (see Figure 1.4). Due to the disturbance, the output of the controller has to be lowered by 0.5 to compensate for the disturbance. By performing the same simulation again, but using $x_c(t^+) = 0$, the result shown in Figure 1.7 is obtained. Clearly, the disturbance is removed. However, since $x_c(t^+) = 0$ is not the right after reset value as the disturbance is not present, a limit cycle is present during the first 5 seconds.

The examples in this section revealed two problems that occur when reset control is used. The first problem is that a reset integrator has not the same steady state properties as a linear integrator and cannot remove a steady state error in general. The second problem is that if the after reset value does not match the steady state value of the control signal, a limit cycle may be present in the response if the after reset value is not adapted.

The fact that reset integrators do not have the same steady state properties as a linear integrator and the fact that limit cycles may be present is not new. Currently, some methods are available to overcome these problems, like PI+CI controllers, reset controllers with a reset band [2], reset controllers with adaptive feedforward [11], [12] and reset control with ILC [9]. These methods will be discussed in more detail in Chapter 2. However, the aforementioned methods do not use the full advantage of reset control, are slowly adapting, or are complicated and difficult to tune and understand. Hence, finding an adaptive, robust and intuitive to tune reset controller is still an open problem.

1.3 Goal & objectives

The goal of this thesis is to design a robust adaptive / iterative learning reset controller algorithm that is able to remove limit cycles when (piecewise) constant references are being tracked. Furthermore, the algorithm should be able to reject constant input disturbances. In this thesis, the development the algorithm will be limited to minimum phase SISO system that can asymptotically be stabilized by standard PID control. In order to fulfill the goal, the following objectives are foreseen:

1. Literature review on existing reset control strategies.
2. Selecting a reset control method.
3. Proposing a method to detect limit cycles and an update law.
4. Testing the algorithm by performing simulations.
5. Testing the algorithm in practice on a real setup.

Chapter 2

Existing reset control methods

In this chapter, some available reset control methods will be mentioned. The advantages and disadvantages of these methods will be discussed as well. Some methods seem to overcome the limit cycle problem. Other methods do not (explicitly) focus on the occurrence of limit cycles, but might be interesting methods to use. For some methods, examples will be given in order to show that the methods are not robust.

2.1 PI+CI control

One of the available methods to avoid/remove limit cycles is a PI+CI controller. This controller is described in [13], [14] and [2] where also some tuning rules are given. The controller has the configuration shown in Figure 2.1. In fact, it is a PI controller, in which the integrator is parallel combined with a Clegg Integrator (CI). The parameter p_r is a tuning parameter, which can be seen as a trade-off between linear and reset control. This parameter can be fixed, but can also be made variable. Since a linear integrator is included, a PI+CI controller is able to remove steady state errors.

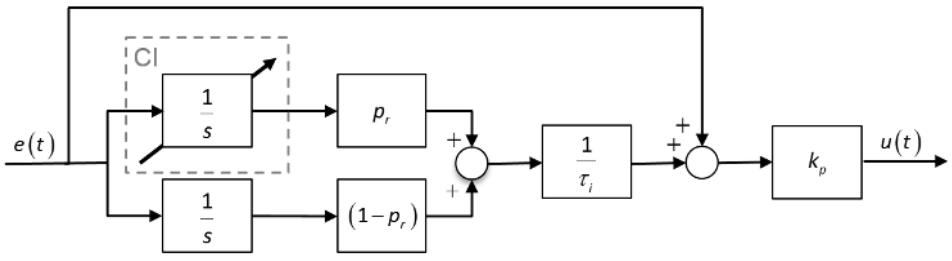

Figure 2.1: PI+CI controller structure

Due to the reset actions, a PI+CI controller can have phase advantage compared to a normal PI controller, see [2, p.185]. Hence, faster responses can be obtained compared to normal PI control. However, since the controller is a trade-off between linear and nonlinear control it does not use the full advantage of reset control and the response will not be as fast as can be. An example of a PI+CI controller can be found in Appendix A.1.

2.2 Reset control with reset band

Besides reset controllers that reset its states when the error signal crosses zero, there are also reset controllers that resets when the error signal enters a certain reset band, see [2]. These reset controllers are described by (2.1). The reset band surface B_δ is defined as $B_\delta = \{(x, y) \in \mathbb{R}^2 | (x = -\delta \wedge y > 0) \vee (x = \delta \wedge y < 0)\}$. The parameter δ is a nonnegative value that defines the reset band. The controller resets when its input (error signal) crosses $-\delta$ from below or δ from above.

$$C := \begin{cases} \dot{x}_r(t) = A_r x_r(t) + B_r e(t) & \text{if } (e(t), \dot{e}(t)) \notin B_\delta \\ x_r(t^+) = A_\rho x_r(t) & \text{if } (e(t), \dot{e}(t)) \in B_\delta \\ v(t) = C_r x_r(t) + D_r e(t) & \end{cases} \tag{2.1}$$

The advantage of using a reset band is that it may improve the stability and performance in systems with a time delay, since a reset band may improve the phase lead, see [15] and [2]. Besides using a fixed reset band, also a variable reset band can be used. In that case the reset band will be determined based on the error and the derivative of the error [2, section 3.4]. However, using a reset band is not a robust solution for the limit cycle problem, since limit cycles can still be present when the reset band is not chosen carefully. An example to illustrate this is given in Appendix A.2.

2.3 Limit cycle analysis

In [16], Baños et al. presented an approach to predict the amplitude and frequency of limit cycles when a CI or FORE with reset band is used. The presented method is applicable to closed loop systems without reference signal. To predict the amplitude and frequency of a possible present limit cycle, the describing function of the reset controller is used, which is denoted by $C_\delta(E, \omega)$. It is shown in the paper that the describing function of a FORE and a CI with a reset band depends on the ratio of the reset band and amplitude of the input δ/E.

To determine if a limit cycle is present, the Nyquist plot of the plant $P(j\omega)$ and the *critical locus* $-\frac{1}{C_\delta(1,\omega)}$ are created for different $\delta/E \in (0, 1)$. If $P(j\omega)$ and $-\frac{1}{C_\delta(1,\omega)}$ intersect and the intersection corresponds to the same frequency for both functions, a limit cycle exists. From this intersection, the amplitude and frequency of the limit cycle can be determined. A similar method for nonlinear systems controlled by a reset controller is proposed by Iwai et al. in [17].

Although the method of [16] seems an interesting method to predict limit cycles, it is only able to predict the limit cycle of closed loop systems without reference signal. The method highly depends on the model of the system. Furthermore it fails to predict the occurrence of limit cycles when a nonzero reference is used. This is shown in an example given in Appendix A.3.

Furthermore, since the method in [16] is based on the describing function of the reset controller (which considers only the first harmonic), it gives only accurate results when the input to the controller fulfills a low-pass condition. It also seems to be computational intensive to find the crossing, if there is one, where the frequency of both curves are equal.

Another proposed method to predict and analyze limit cycles when reset controllers with a reset band are used is presented by Barreiro et al. in [18]. In this method, instead of using the describing function of the reset controller, a Poincaré map of the reset system is used to predict limit cycles. However, also this method considers only the case of closed loop systems with zero reference and may not be very practical to use during online measurements.

2.4 Reset control with feedforward

A strategy to overcome the problems of steady state errors in reset control, is to combine reset control with (adaptive) feedforward control. This approach is found in several papers. The task of the feedforward controller is to correct the control signal in such a way that it reaches its required steady state value. In this section, two of such controllers are discussed.

2.4.1 Adaptive feedforward

In [11], Panni et al. developed an adaptive feedforward controller, which is combined with a FORE controller. The adaptive feedforward controller updates the value of the control signal at the reset instants, such that the control signal converges to the right steady state value in case of parametric model uncertainties. The adaptive feedforward controller developed by Panni et al. uses a vector containing independent basis functions and a vector containing parameters to be adapted. The authors prove and showed that it is possible to asymptoticly track constant references and reject constant disturbances at the plant input. However, it might be difficult to find the right basis functions and to determine which parameters have to be adapted.

Cordioli et al. presented in [12] a generalized version of the feedforward controller of Panni et al. to track smooth reference signals instead of constant references. The derivative of the reference signal has to be known for this purpose and is used by the feedforward controller. However, in the shown responses in [12] it seems that in the final obtained response, the reset controller does not reset anymore. The adaptive feedforward compensator seems to determine the control signal. In my opinion, since no reset seems to occur anymore in the final response, iterative learning control is a more easy approach to track smooth reference trajectories.

2.4.2 Iterative learning control

HosseinNia et al. [9] presented another approach to remove limit cycles. In their approach a PCI controller is combined with an Iterative Learning Controller (ILC). The PI controller resets to zero as the error signal is zero. The ILC is activated at the first reset instant. The task of the ILC in this approach is to correct the control signal, instead of the after reset value, such that the control cab achieve steady state. The update law defined in [9] is given in equation 2.2, where e^j is the error signal of trial j, T is the final time and $P(0)$ is the DC gain of the plant). In fact this update law adds a scaled version of the error signal to the current feedforward signal.

$$u_{ILC}^{j+1} = \begin{cases} 0 & \text{if } \ 0 \leq t < t_r \\ u_{ILC}^{j}(t) + \frac{1}{P(0)} e^j(t + \Delta) & \text{if } \ t_r \leq t < T \end{cases} \tag{2.2}$$

Although it is shown that the proposed method is able to remove / reduce the amplitude of the limit cycle, it takes around 200 iterations before the amplitude of the limit cycle has become small enough to be not significant anymore. Besides that, it is only applied on a first order system. For higher order systems, the presented method might not work or the update law has to be adapted. Another disadvantage is that it does not use the repeating pattern of the limit cycle within one trial and hence it might take longer than necessary to eliminate the limit cycle.

2.5 Adaptation of the after reset value

Instead of adapting the control signal by applying a feedforward signal, the control signal can also be influenced by adapting the after reset value directly. This way, it is possible to deal with model uncertainties and improve the transient response. Three methods that uses adaptation of the after reset value will be discussed in this section.

2.5.1 Improved reset controller

Zheng et al. [4], [19] designed an improved PI reset controller for a PZT positioning stage. This controller resets at fixed time instants and the after reset value is determined at every reset instant. A similar strategy is used by Guo et al. in [3]. The improved reset controller as proposed by Zheng et al. determines $x_r(t_k^+)$ at every reset instant, by minimizing the quadratic cost function given in equation (2.3), where P_0, P_1 and Q_0 are positive semidefinite tuning matrices and $e(t)$ the error signal. The controller equations are given in (2.4), where E_k, F_k and G_k are determined by the optimization, x_p are states of the plant, and r the constant reference to be tracked.

$$J_k = e^T(t_{k+1})P_0e(t_{k+1}) + \dot{e}^T(t_{k+1})Q_0\dot{e}(tk+1) + \int_{t_k}^{t_{k+1}} e(s)^T P_1 e(s)ds \tag{2.3}$$

$$RC := \begin{cases} \dot{x}_r = A_r x_r + B_r e, & t \neq t_k \\ x_r(t_k^+) = E_k x_p + F_k x_r + G_k r, & t = t_k \\ u = C_r x_r + D_r e \end{cases} \tag{2.4}$$

In Appendix A.4, a reproduction of the obtained results of Zheng et al. is shown, since this control strategy seems to be a promising method. The reproduction shows that although the overshoot and transient response are improved compared to standard PI and conventional reset control, a steady state error and limit cycle can still be present in case of small mismatches in the controller parameters. This clearly shows the sensitivity to parameter uncertainties when reset control is used.

Although the method is promising, the required optimization can be seen as disadvantage. This is for reason that the optimization depends on system matrices and tuning matrices. In case of model uncertainties, this may have an influence on the performance and limit cycles may appear as well. Furthermore, as stated in [2, section 1.7], the improved reset controller of Zheng et al. may be more sensitive to model uncertainties when an observer is needed to estimate states that cannot be measured.

Although this is the case, resetting at fixed time instants, seems to have some benefits. The number of reset actions can be influenced and it is sure that the controller will reset, no matter if the error crosses zero or not. This avoids the need of temporary regulation and creating aggressive transients like in [10].

2.5.2 General reset controller

In [20], HosseinNia et al. introduces a general reset controller that also resets at fixed time instants. This reset controller is mathematically represented by (2.5).

$$C := \begin{cases} \dot{x}_r(t) = A_r x_r(t) + B_r e(t), & t \neq t_k \\ x_r(t^+) = A_{Rr} x_r(t) + B_{Rr}(\frac{Kr(t) - D_r e(t_k)}{n_{\mathfrak{R}} c_r}), & t = t_k \\ u_r(t) = C_r x_r(t) + D_r e(t) \end{cases} \tag{2.5}$$

In this equation $A_{Rr} \in \mathbb{R}^{n_r \times n_r}$ selects the last $\mathfrak{R}$ states of $x_r(t)$, (which are the states that will be reset) and has the form $A_{Rr} = \begin{bmatrix} I_{n_{\hat{\mathfrak{R}}}} & 0 \\ 0 & 0_{n_{\mathfrak{R}}} \end{bmatrix}$, with $n_{\hat{\mathfrak{R}}} = n_r - n_{\mathfrak{R}}$ and $n_{\mathfrak{R}}$ the number of states that are reset. Furthermore $B_{Rr} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, $C_r = c_r \begin{bmatrix} 0 & 1 \end{bmatrix}, c_r \in \mathbb{R}$ where I and 0 are identity and zero matrices with proper dimension. The parameter K is a feedforward gain and is chosen equal to the inverse DC gain of the plant. This gain is multiplied by the reference signal $r(t)$. By doing this, the control output $u_r(t)$ at the reset instants becomes Kr, which is equal to the DC value of the control signal (when disturbances are not present).

HosseinNia et al. noticed that the control signal $u_r(t)$ of the improved reset controller of Zheng et al. in [4] tends to the inverse DC gain of the plant for a unit step reference. Since

the general method does not use an optimization and has less tuning parameters, this general method seems more easy to tune and implement. The performance of this general reset controller is also compared by HosseinNia et al. [20] with the improved reset controller of Zheng et al. [4]. The comparison shows that similar performance is achieved with the general fixed instant reset controller.

However, in the comparison made by HosseinNia et al. model uncertainties and disturbances are not considered. In case of model uncertainty, limit cycles can also be present when this controller is used. This is shown in an example given in Appendix A.5.

2.5.3 Robust hybrid PI controller

Recently, Scola et al. presented a simple and robust PI controller in [21]. This method is an adaptive version of the algorithm as proposed in [22] for a FORE. The controller in [21] can be represented by equation (2.6), where α is a tuning parameter that determines how fast the after reset value is adapted. Furthermore, $\zeta = x_r - x_{eq}$, $x_{eq} = Fr$ with F the inverse DC gain of the plant and τ is a time parameter to avoid Zeno solutions.

$$C := \begin{cases} \left.\begin{array}{lcl} \dot{x_r}(t) & = & e(t) \\ \dot{\tau} & = & 1 \end{array}\right\} & \text{if } 2e\zeta + \zeta^2 \geq 0 \text{ or } \tau \leq \rho \\ \left.\begin{array}{l} x_r(t^+) = x_r(t) - \alpha\zeta \\ \tau^+ = 0 \end{array}\right\} & \text{if } 2e\zeta + \zeta^2 \leq 0 \text{ and } \tau \geq \rho \\ u(t) = k_i x_r(t) + k_p e(t) \end{cases} \tag{2.6}$$

The results that are shown in [21] show that the algorithm is able to adapt the after reset value in such a way that steady state errors are removed. The presented method seems also easy to tune, since only one parameter for the adaptive part is present. However, it might be difficult to find a good value for α. Furthermore, the after reset value is adapted every time a reset is applied. In fact, the after reset value should only be adapted in case it results in a limit cycle. Besides that, the parameter α makes the controller a compromise between linear and nonlinear control, which may lead to a less fast transient response.

2.6 Non-overshoot reset control

Another available reset control method is presented by Zhao and Wang in [10] and Zhao in [23]. The algorithm in these papers makes it possible to achieve non-overshoot step responses while the rise time of the system can be tuned arbitrarily. The block diagram of the feedback loop used in [10] is shown in Figure 2.2.

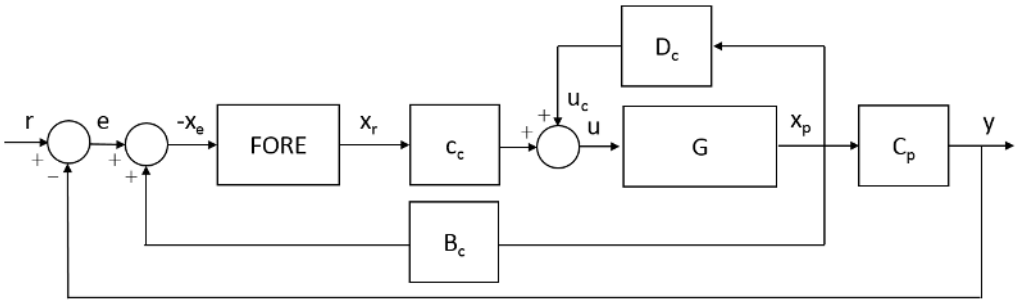

Figure 2.2: Control loop for nonovershoot reset controller.

The plant should be minimum phase and at least have one zero. The plant is transformed into state space form by applying the concept of zero dynamic transformation (see e.g. [24]). The matrices c_c, D_c seems to defined such that the original plant given by (2.7) is transformed into (2.8).

$$G(s) = \frac{b_m s^m + b_{m-1}s^{m-1} + ... + b_0}{s^n + a_{n-1}s^{n-1} + ... + a_0} \tag{2.7}$$

$$H(s) = \frac{\alpha^{h-1}}{s(c_{h-2}\alpha^{-1}s^{h-1} + c_{h-3}\alpha^{-2}s^{h-2} + ... + c_0\alpha^{1-h})} \tag{2.8}$$

In (2.8), α is a tuning parameter that defines the location of the poles of the transformed plant $H(s)$. The constants c_i are defined as $c_i = \frac{(h-1)!}{i!(h-1-i)!}$, where h is the relative degree of the original plant $G(s)$

The reset controller is an unstable FORE, with a state dependent reset law. The equations of the reset controller are given in (2.9) where $a_c, b_c > 0$, $\epsilon > 0$ is a parameter that defines the tilt of the boundary between the flow and the jump mode and ρ defines the minimum amount of time between two consecutive reset instants. Results of Nesic et al. [25] were used to prove stability.

$$\left.\begin{array}{rcl} \dot{\tau} & = & 1 \\ \dot{x}_r & = & a_c x_r + b_c(e + B_c x_p) \end{array}\right\} \text{ if } \epsilon(e + B_c x_p)^2 + 2(e + B_c x_p)x_r \geq 0 \text{ or } \tau \leq \rho \tag{2.9}$$

$$\left.\begin{array}{l} \tau^+ = 0 \\ x_r^+ = 0 \end{array}\right\} \text{ if } \epsilon(e + B_c x_p)^2 + 2(e + B_c x_p)x_r \leq 0 \text{ and } \tau \geq \rho$$

$$u_c = c_c x_r + D_c x_p$$

The matrix B_c is defined such that $x_r = \dot{x}_e$ and x_e is defined as $x_e = -(B_c x_p + e)$. It is proven in [10] that when α is chosen small enough and the initial conditions fulfill a certain condition, a stable non-overshooting response with zero steady state error is obtained.

The responses shown in [10] and [23] have indeed no overshoot and the rise time can be tuned arbitrarily by changing the controller parameters a_c and b_c. Therefore, this control strategy seems an interesting method. However, in practice a perfect cancellation of the zero dynamics will be difficult due to the presence of uncertainties. Furthermore, the state feedback part and the tuning of the rise time may lead to high control actions, which cannot be realized in practice and may lead to an unstable response.

2.7 Summarized

A PI+CI controller is able to remove/avoid limit cycles in the response, but it is a trade-off between linear and non-linear control. Hence, it does not use the full advantage of reset control. The reset controllers with a reset band may improve the performance, but is not a robust solution since limit cycles can still be present. The method in [16] can predict limit cycles, but only for autonomous systems controlled by reset controllers with a reset band.

The reset controllers with an adaptive feedforward controller [11] and [12] adapts the feedforward gain at every reset instant, which might not be needed. Besides that, it might be difficult to find right base functions for the adaptive part. Furthermore, in [12], the reset controller seems not to reset anymore in the final response, since the adaptive controller has learned the right control signal.

The ILC approach in [9] is able to remove / reduce the amplitude of limit cycles, but it does not use periodic pattern of limit cycles and hence the convergence is slower than it could be. Furthermore, only first order systems were considered. The hybrid PI controller in [21] is easy to tune and can be seen as a solution to overcome the limit cycle problem. However, the after reset value is changed at every reset instant, which is not necessary. Furthermore, the tuning parameter makes it a trade-off between linear and nonlinear control.

The non-overshoot reset controller in [23] and [10] is interesting, but not very realistic to use in practice. The reason for this is that the algorithm may lead to high control inputs. Furthermore, uncertainty is not considered, so probably limit cycles will be present in case of uncertainty. The general reset controller in [20] and the improved reset controller in [4] and [19] seems to be better options, since the responses obtained with these controllers are fast and has almost no overshoot. Moreover, they are easy to tune. However, limit cycles can still be present with these methods. Hence, they are not robust.

So, based on the studied literature, no robust solution for the occurrence of limit cycles in the closed loop response when tracking nonzero constant references exists. In the next chapter, a possible more robust solution for the limit cycle will be proposed.

Chapter 3

Proposed solution

In this chapter a solution for the limit cycle problem is proposed. First a reset control method is selected based on the reviewed literature. In section 3.2 a strategy to solve the limit cycle problem is introduced and explained. A simulation with the proposed algorithm will be performed in section 3.3 and preliminary results are given.

3.1 Selection of reset control method

Based on the studied reset control methods, the general PI reset controller with fixed reset instants (HosseinNia et al. [20]), given in (3.1) seems a promising method. It is a general and easy to tune method and can be extended to PID control as well. Furthermore, no optimization is required as is the case for the fixed instant reset controller of Zheng et al. in [4] and [19].

$$C := \begin{cases} \dot{x}_r(t) = e(t), & \text{if } t \neq t_k \\ x_r(t^+) = K\frac{\tau_i}{k_p} r(t) - \tau_i e(t), & \text{if } t = t_k \\ u_r(t) = \frac{k_p}{\tau_i} x_r(t) + k_p e(t) \end{cases} \tag{3.1}$$

Besides that, it is expected that a fixed instant reset controller is less sensitive to noise. This for reason that the controller is not resetting every time the error crosses zero, which might be due to noise as well. Also the need of including temporary regulation is avoided. Furthermore, stability of the closed loop system can more easily be guaranteed using the work of Zheng et al. [4], [19] and [26].

Another advantage is that the reset interval can be chosen, which makes it possible to have some influence on the period of a present limit cycle. Due to these advantages, a strategy to solve the limit cycle problem for fixed instant reset control will be developed. The final proposed algorithm will however also be tested for zero crossing reset. Since the principle of resetting is the same, it is expected that algorithm that will be proposed in the next section works for zero-crossing reset as well.

3.2 Strategy to solve the limit cycle problem

Since limit cycles are usually present when the after reset value does not match the steady state value of the control signal, an algorithm that adapts the after reset value is a good approach. This idea is already used in, [4], [19] and more recently in [21]. However, in these methods, the after reset value is changed every reset instant. In fact, the after reset value should only be adapted if a limit cycle is present. In (3.1), the after reset value can easily be adapted by changing gain K. Inspired by the methods in [4], [20], [9] and [21], the idea is to develop a robust algorithm that performs the following steps:

1. Detect if a limit cycle is present in the response during simulation/measurement (trial).
2. Adapt $x_r(t^+)$ by changing gain K if a limit cycle is present.
   Repeat step 1 (during trial).
3. Optional: use the new found K as initial value for K in a next run.
   Repeat step 1 and 2 during this new trial.

The above idea can be considered as a combination of adaptive and iterative learning control. During one measurement, the gain K will be *adapted* if a limit cycle is detected. After adapting gain K, it is checked during the same measurement if the limit cycle is removed. If not, K is adapted again. So the gain is adapted in an *iterative* way during one trial. If no (input) disturbances are present, the last found K can be used as initial value in a next run. This way, every new run the response becomes better (as long as disturbances are not present). In Figure 3.1, a block diagram of the algorithm is shown. In the next two subsections a method to detect limit cycles and an update law for K will be proposed and explained.

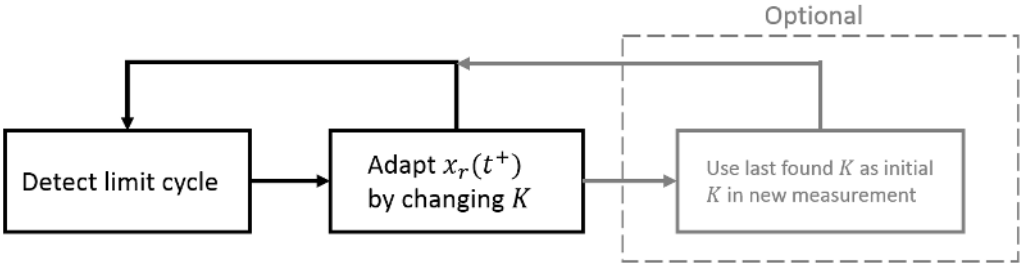

Figure 3.1: Block diagram of algorithm.

3.2.1 Detection of limit cycles

Since a limit cycles are periodic, a repetitive pattern is observable in the the response, but also in the control signal. If a limit cycle has a period of one reset interval (time between two resets), the jumps at two consecutive reset instants will be equal from the moment the limit cycle is present. This can be observed in Figure 1.5 and A.8. Hence, to detect limit cycles, jumps in the control signal at the reset instants can be used. Therefore, the idea is to detect a limit cycle by comparing two consecutive jumps in the control signal. In Figure 3.2, the idea of comparing jumps is illustrated. The red lines indicate the jumps that are compared. In this figure the zoomed control signal of Figure A.8 of Appendix A.5 is shown.

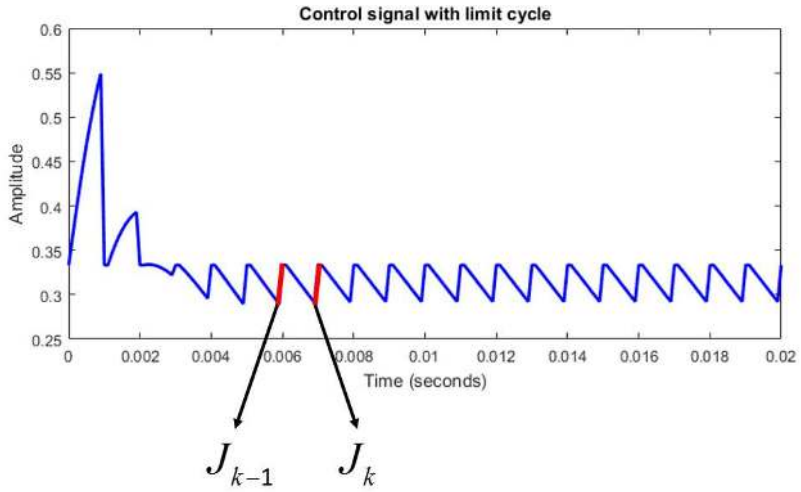

Figure 3.2: Detection of limit cycle in control signal.

Comparing two consecutive jumps will only work to detect a limit cycle if the period of the limit cycle is equal to one reset interval. However, for fixed instant reset control it is expected that this condition can be achieved by taking a small enough reset interval. Since the algorithm will be designed for fixed instant reset control in the first place, this is a reasonable assumption.

Due to numerical errors and noise that is present in practice, the jumps in case of a limit cycle will not exactly be equal. Hence, the condition in (3.2) will be considered to determine if a limit cycle is present. In this equation J_k denotes the jump in the control signal at reset instant k and the value ϵ is a small threshold value which can be seen as a tuning parameter.

$$0 \leq |J_k - J_{k-1}| \leq \epsilon \tag{3.2}$$

The above method might fail to detect limit cycles when zero-crossing reset is used, because limit cycles can have a period of more than one reset interval in that case. This may happen due to the fact that a reset can occur when the error is crossing zero from above and/or below. However, it is expected that the above method can also be used in that case, but instead of comparing two consecutive jumps, jumps that are spaced by the period of the limit cycle have to be compared. Another option is to force a certain minimum amount of time between two resets in order to realize a limit cycle with a period of one reset interval.

3.2.2 Adapting the after reset value

After detecting a limit cycle, the after reset value should be adapted. Normally, if the closed loop linear base system is asymptotically stable, an integrator steers the control signal to its required steady state value. When constant references are tracked, it can easily be verified that this steady state value is equal to $u_{ss} = Kr(t)$ with K the inverse gain of the plant and $r(t)$ the reference. This is also the underlying idea of the controller in [20]. However, if the integrator in equation (3.1) resets to a value which is not equal to the required steady state due to a wrong value for K, the reset actions are counteracting the steering process at a certain moment, resulting in a limit cycle.

Since it is assumed that the system is asymptotically stabilized by a linear PID controller, the control input $u_r(t)$ will converge to u_{ss} if no reset is applied. Hence, the idea is to adapt K to $\frac{u_r(t_k^-)}{r(t)}$ if a limit cycle is present, which is the value the control signal has before reset, divided by the value of reference.

By adapting K to $\frac{u_r(t_k^-)}{r(t)}$, the control signal $u_r(t)$ becomes $u_r(t^-)$ at reset, so in fact one reset instant is skipped. This gives the integrator one more reset interval of time to steer the control signal. At the same time K is adapted in the direction the integrator was steering the control signal before it was reset. This way, the after reset value will converge iteratively to the required steady state value, also in case a constant input disturbance is present. Hence, limit cycles are removed.

In Figure 3.3, the value used to adapt K is indicated by the green arrow, which is the value the control signal has before reset. This value should be divided by the reference signal, in order to have $u_r(t) = Kr(t)$ at reset in equation 3.1.

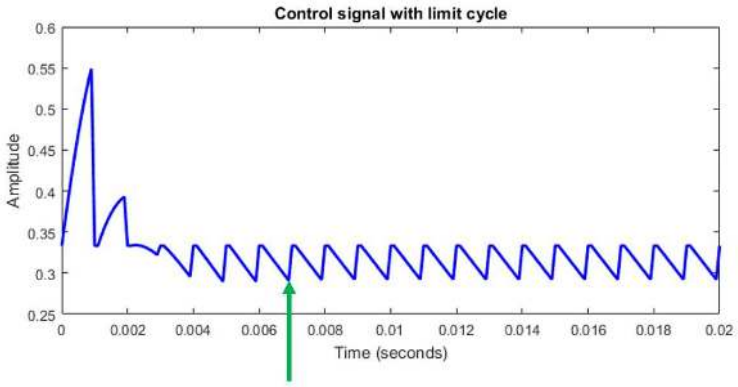

Figure 3.3: Control signal with limit cycle and value used to adapt K indicated.

3.2.3 Mathematical formulation of the proposed algorithm

A mathematical formulation of the proposed algorithm for the fixed instant PI reset controller is given in equation (3.3). The blue lines define the proposed adaptive part. It is assumed that the closed loop linear base SISO system is minimum phase and asymptotically stable. Furthermore, it is assumed that the limit cycles have a period of one reset interval.

$$C := \begin{cases} \dot{x}_r(t) = e(t) & \text{if } t \neq t_k \\ \left.\begin{array}{l} J_k = |u_r(t_k^-) - K_{k-1} r(t)| \\ K_k = \begin{cases} K_{k-1} & \text{if } |J_k - J_{k-1}| > \epsilon \text{ or } r(t) = 0 \\ \frac{u_r(t_k^-)}{r(t)} & \text{if } 0 \leq |J_k - J_{k-1}| \leq \epsilon \text{ and } r(t) \neq 0 \end{cases} \\ x_r(t^+) = K_k \frac{\tau_i}{k_p} r(t) - \tau_i e(t) \end{array}\right\} & \text{if } t = t_k \\ u_r(t) = \frac{k_p}{\tau_i} x_r(t) + k_p e(t) \end{cases} \tag{3.3}$$

Since the after reset value is updated in the direction of the linear control signal, the value of K also converges to the required steady state value. This makes it possible to track piecewise constant references and to reject constant input disturbances.

Notice that only one additional tuning parameter (ϵ) is introduced. Therefore, the proposed algorithm may seem comparable to the controller in [21]. However, in that algorithm the after reset value is updated every reset instant, while in the algorithm proposed in this thesis, the value is adapted only if a limit cycle is detected.

The above controller algorithm can be extended to PID control by connecting the PI controller in series with a lead/lag filter as shown in Figure 3.4. This way, K will still converge to the right value.

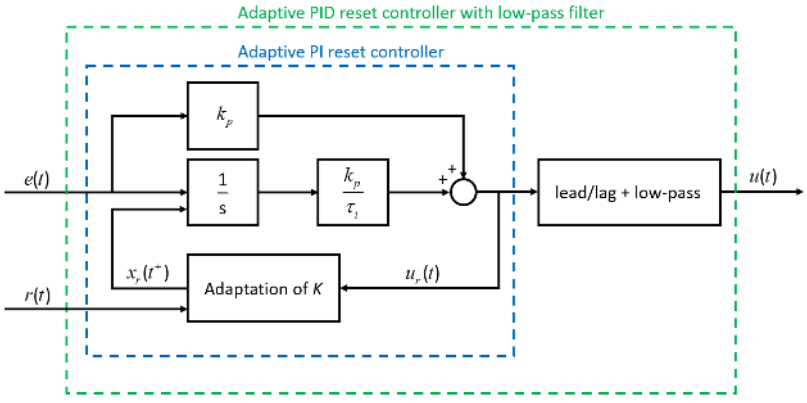

Figure 3.4: Structure of adaptive PID reset controller with low-pass filter.

3.3 Preliminary results

In the example given in Appendix A.5, it was illustrated that a limit cycle and a steady state error can be present when the controller as given in equation (3.1) is used. To see if the proposed idea works, the controller given in (3.3) is applied, with $K = \frac{1}{3}, k_p = 0.08$, $\tau_i = \frac{8}{3} \cdot 10^{-4}$, $\Delta t_k = 1$ ms and $\epsilon = 1 \cdot 10^{-3}$. For the plant $P(s) = \frac{3 \cdot 10^6}{s^2 + 1810s + 0.8 \cdot 10^6}$ is taken again, which is a slightly perturbed version of the plant used in Zheng et al. [4] and [19]. By performing a simulation with the new proposed adaptive controller, the results as shown in Figure 3.5 are obtained.

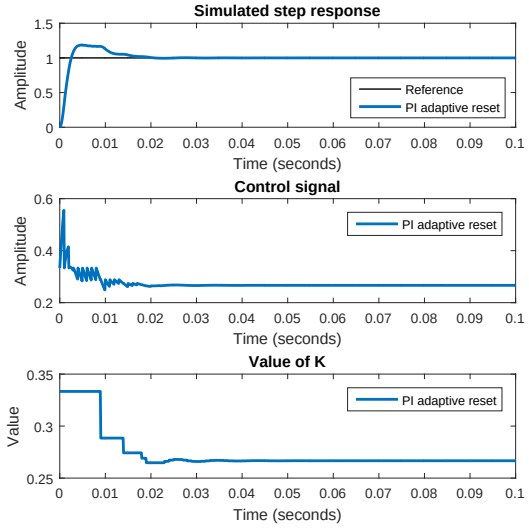

Figure 3.5: Step response with new reset controller.

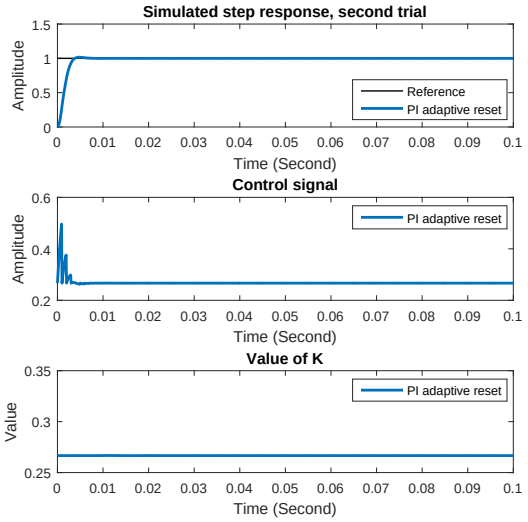

Figure 3.6: Step response with new reset controller, second trial.

It can be observed that the algorithm is able to remove the steady error and is also able to remove the limit cycle (periodic jumps with equal amplitude are not present in the control signal at the end). In Figure 3.6 it can be observed that the value of K converges to the right steady state value of $\frac{1}{P(0)} = \frac{0.8}{3} \approx 0.2667$.

If this value is used in a second run as initial value for K, the response in Figure 3.6 is obtained. The response has almost no overshoot and also the steady state error is removed. Using the last found value for K is a good idea if no large disturbances are present. In most precision systems small disturbances are seen from cross-coupling and ground vibrations, so in all those applications the last found value of K can be used. However, if large constant input disturbances are expected to be present, it is better to use the expected inverse DC gain of the setup as initial value for K.

To see what happens when a constant disturbance with an amplitude of 0.1 is added at the plant input at $t = 0.05$ seconds, a new simulation is performed. As initial gain, $K = \frac{1}{3}$ is taken. This gives the results shown in Figure 3.7. Clearly the algorithm is able to reject the disturbance. The gain K at the end of the simulation is converged to $K = 0.1667$. This is equal to the final value of the controller output, which is equal to the inverse DC gain of the plant minus the constant input disturbance of 0.1 scaled by the inverse of the reference (which is 1 in this case).

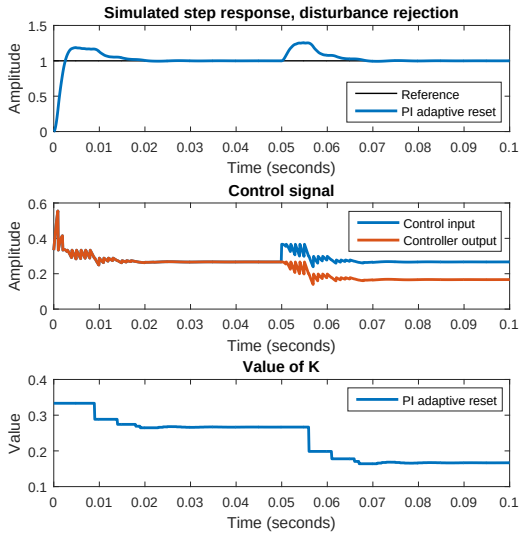

Figure 3.7: Disturbance rejection with adaptive reset controller.

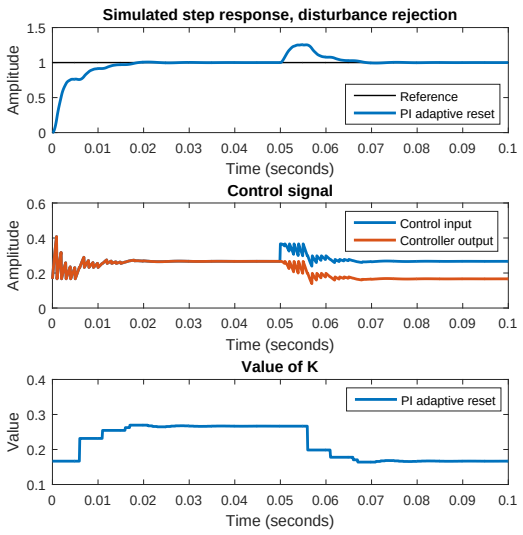

Figure 3.8: Disturbance rejection with adaptive reset controller, second trial.

If the final found $K = 0.1667$ is used as initial value for K in a second trial with the same constant input disturbance applied, the response shown in Figure 3.8 is obtained. Compared to the result shown in Figure 3.6, the response is slower. This is caused by the fact that during the first 0.05 seconds, the right gain is the inverse DC gain of the plant, which is $K \approx 0.2667$ and not $K = 0.1667$. Hence, during the first 0.05 seconds the algorithm adapts K first to $K = 0.2667$ and after the disturbance is applied, K is adapted back to $K = 0.1667$.

In fact, during the first 0.05 seconds in this second trial $K = 0.2667$ should be used and after $t = 0.05$ seconds, $K = 0.1667$, assumed the same disturbance is applied at the same time. Using such a time scheduled value for K might for example be useful in pick and place applications, which involve repetitive action. However, this procedure will not be incorporated in the this thesis and is left for future work.

Since the preliminary results in this section showed that the proposed algorithm is able to remove limit cycles, the algorithm will be tested in practice on a real setup. The setup that will be used is discussed in the next chapter.

Chapter 4

Setup & System identification

In this chapter, a description of the setup that will be used is given. After the system is described, a system identification is performed which will result in a transfer function of the setup. As will become clear after the system identification, the chosen setup needs a controller with integrator and hence it is a suitable setup to test the proposed adaptive reset controller on.

4.1 Description of the setup

The setup is shown in Figure 4.1. It is a 3-DOF system (x, y, θ) consisting of three Lorentz actuators (1A, 1B and 1C). These actuators can be used to control the position of the mass in the middle (3). This mass is connected to three additional masses (2). The encoder (4) measures the movement of one of the additional masses.

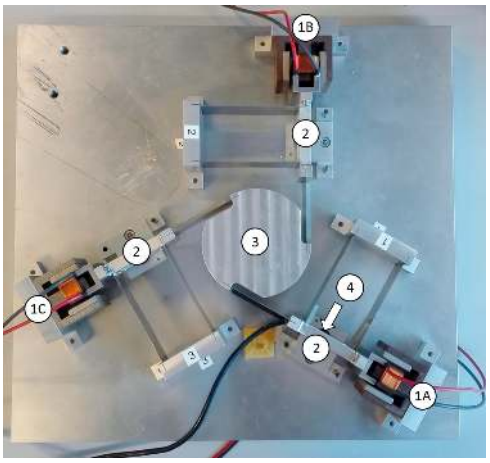

Figure 4.1: Setup on which proposed algorithm will be tested.

In this thesis, only actuator 1A will be actuated, so the system will be used as a SISO system. Therefore, the system will act as a collocated double mass spring damper system. Furthermore, since the position of only one mass is controlled, the other masses add additional dynamics at higher frequencies.

4.2 Hardware & software

The setup will be controlled by LabVIEW 2016 and a myRIO 1900 system, which will also log the data. The post-processing is done with MATLAB R2015b. Initially, dSPACE was used to control the setup, but the results obtained with dSPACE turned out to be noisy and not good enough. After wasting a lot of time trying to obtain good results with dSPACE, it was decided to switch to LabVIEW and a myRIO system.

The myRIO system is connected to an amplifier which powers the actuators of the setup. The encoder output of the setup is sent to the myRIO and the myRIO system interacts with a host computer. The LabVIEW files that will be used are separated in an FPGA part which runs on the myRIO and a target file, which is hosted by the computer. This way it is possible to interact with the system via the host computer. A scheme of the above described interactions is given in Figure 4.2.

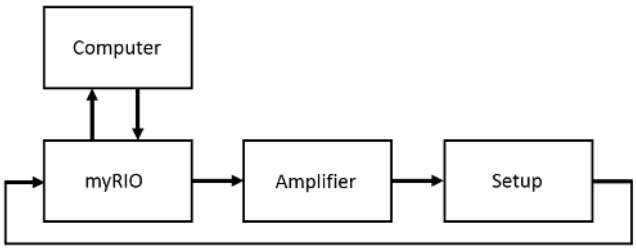

Figure 4.2: Hardware scheme

4.3 System identification

An open loop system identification is performed in order to obtain a mathematical model of the system. For the identification, the response of the system to a chirp input signal is measured. The frequency of the chirp is varied from 0.1 to 500 Hz and is increased by 3% every second. The amplitude of the chirp is set to 256 (raw value), which corresponds to an output value of the myRIO of approximately 1.25 Volt. Data is logged by the myRIO every $50 \cdot 10^{-6}$ seconds. The LabVIEW files used to perform the system identification were written by Niranjan Saikumar.

Based on the measured data, a Bode and coherence plot are created with MATLAB, which are shown in Figure 4.3. The MATLAB code can be found in Appendix B. Since the coherence is around 1 in the interested control region, the measurement can be used to estimate a linear model of the system. In the Bode plot, it can be observed that the there are two complex pole pairs and one complex zero pair. However, the main shape of the magnitude and phase curve looks similar to a collocated double mass-spring-damper system with only one complex pole pair and no zeros.

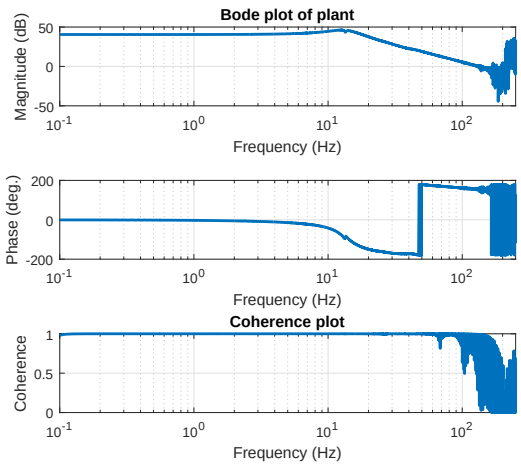

Figure 4.3: Identified Bode and coherence plot of setup.

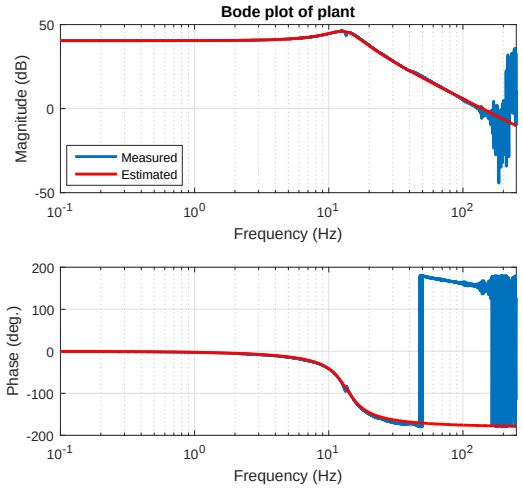

Figure 4.4: Comparison between measured and estimated plant.

Hence, a second order transfer function without zeros is estimated to describe the main dynamics of the system. A transfer function of the setup is estimated based on the estimated Bode plot data using MATLAB. The obtained second order transfer function is given in equation (4.1). This transfer function relates the raw output value of the myRIO to the number of encoder ticks. So the conversion of the raw input value to volts and the amplifier gain are included in this transfer function. For the estimation, only Bode plot data up to 50 Hz was considered. After 50 Hz, jumps in phase are present and the data becomes noisy, which would lead to a less accurate model estimation.

$$H(s) = \frac{7.606 \cdot 10^5}{s^2 + 46.58s + 7254} \tag{4.1}$$

In Figure 4.4 a Bode plot of the estimated transfer function is compared with the Bode plot based on the measured data. As can be observed, the second order model describes the main dynamics of the system pretty well. Furthermore, it can be observed that the controller needs an integrator in order to have a high loop gain at the lower frequencies and to be able to track constant references. Hence, this setup can be used to test the proposed adaptive reset controller on.

Chapter 5

Results

In this chapter, several experiments are performed to investigate if the proposed algorithm works in simulation and in practice. In Section 5.1 the final controller algorithm is presented first. In Section 5.2 the results for the fixed instant adaptive reset controller are presented. In Section 5.3, the algorithm is tested for zero-crossing reset.

5.1 Final controller algorithm

The final controller is a slightly extended version of the in Chapter 3 proposed algorithm. In Chapter 3, a fixed instant adaptive PI reset controller was presented. The setup will however be controlled by a PID base controller with low-pass filter in order to obtain bandwidth of 60 Hz. To overcome the addition of the derivative action and low-pass filter, the PID controller is split into a PI part and a lead/lag part with low-pass filter as was shown in Figure 3.4. This way, the proposed adaptive PI reset controller structure can still be used and it the process of adapting K will still work. In Figure 5.1, the way in which the final controller will be implemented is shown again, where $e(t)$ represents the error signal and $r(t)$ the reference signal. The structure of the PID controller that will be used, is given in equation (5.1).

$$\underbrace{k_p\left(1+\frac{2\pi f_i}{s}\right)}_{\text{PI}}\underbrace{\left(\frac{\frac{s}{2\pi f_d}+1}{\frac{s}{2\pi f_t}+1}\right)\left(\frac{1}{\frac{s}{2\pi f_l}+1}\right)}_{\text{lead/lag + low-pass filter}} \tag{5.1}$$

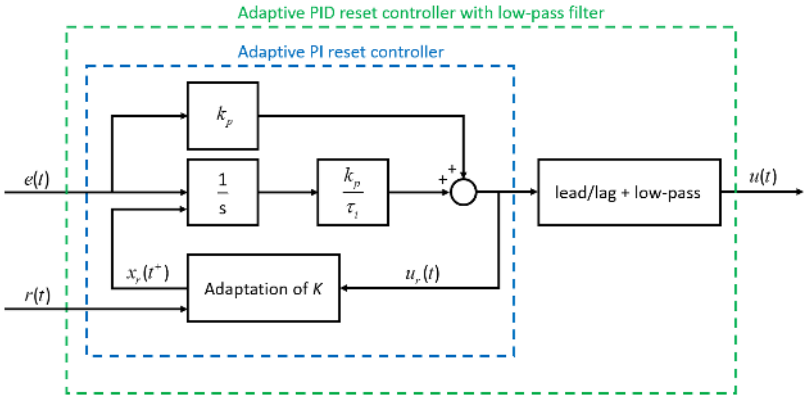

Figure 5.1: Structure of adaptive PID reset controller with low-pass filter.

Splitting the base controller into a PI controller and lead/lag compensator with low-pass filer leads to the final controller equations given in equation (5.2), where the blue lines indicate the adaptive part. These controller equations will used for simulations and measurements in this chapter.

$$C_{PIreset} := \left\{ \begin{array}{ll} \dot{x}_r(t) = e(t) & \text{if } t \neq t_k \\ J_k = |u_r(t_k^-) - K_{k-1} r(t)| & \\ K_k = \begin{cases} K_{k-1} & \text{if } |J_k - J_{k-1}| > \epsilon \text{ or } r(t) = 0 \\ \frac{u_r(t_k^-)}{r(t)} & \text{if } 0 \leq |J_k - J_{k-1}| \leq \epsilon \text{ and } r(t) \neq 0 \end{cases} & \text{if } t = t_k \\ x_r(t^+) = K_k \frac{\tau_i}{k_p} r(t) - \tau_i e(t) & \\ u_r(t) = \frac{k_p}{\tau_i} x_r(t) + k_p e(t) & \end{array} \right. \tag{5.2}$$

$$C_{DLF} = \underbrace{\left(\frac{\frac{s}{2\pi f_d} + 1}{\frac{s}{2\pi f_t} + 1}\right)\left(\frac{1}{\frac{s}{2\pi f_l} + 1}\right)}_{\text{lead/lag + low-pass filter}}$$

5.1.1 Control loop and parameters

The control loop that will be used for simulations and measurements is shown in Figure 5.2. In this figure $r(t)$ represents the reference, $e(t)$ the error, $u(t)$ the controller output, $u_A(t)$ the amplifier input, $d(t)$ a disturbance and $y(t)$ the output. Furthermore, $H(s)$ represents the plant (including the amplifier) and $C(s)$ represents the controller. The controller will be implemented as a discrete controller and will be discretized using zero-order-hold. The parameters of the PID controller are tuned using the rules of thumb given in [27]. The cut-off of the low-pass filter is set to 600 Hz and the gain k_p is tuned such that a stable controller and a bandwidth around 60 Hz is obtained.

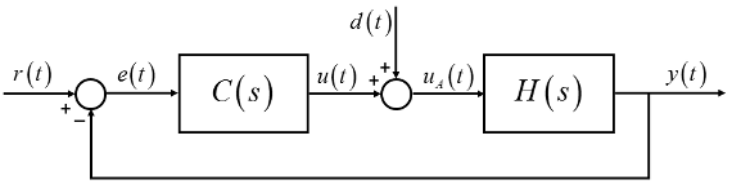

Figure 5.2: Negative feedback loop.

In Table 5.1, the controller, simulation and measurement parameters are listed. These parameters are used for simulation and measurements that are performed in the remainder of this chapter. All simulations are performed in MATLAB SIMULINK R2015b. The LabVIEW files to perform the measurements are based on the LabVIEW files of Niranjan Saikumar.

Table 5.1: Controller, simulation and measurement paramaters.

Parameter	Value	Units	Description
f_i	6	Hz	Break-point of integrator
f_d	20	Hz	Beginning point of differential action
f_t	180	Hz	Taming frequency of differential action
f_l	600	Hz	Cut-off frequency of low-pass filter
k_p	0.06	(encoder ticks)$^{-1}$	Proportional gain
τ_i	$\frac{1}{2\pi f_i}$	s	Time constant of integrator
T_s	0.4	ms	Sampling time
Δt_k	4	ms	Time between fixed reset instants $\Delta t_k = t_k - t_{k-1}$
ϵ	0.1	-	Threshold to decide if two consecutive jumps are considered as a limit cycle.

A sample time of 0.4 ms is chosen, since it turned out that this is the fastest sampling time that can be used in the Control & Simulation Loop of LabVIEW in combination with the myRIO system. When a higher sampling rate is used, computations are not finished in time anymore.

The threshold ϵ is set to 0.1. The results obtained for different ϵ are shown in Figure 5.3 and 5.4. The responses are measured for $K = 0.5K_{DC}$ and $K = 1.5K_{DC}$, where $K_{DC} = 0.009537207467789$ is the inverse DC gain of the identified plant in equation (4.1). As can be observed, the adaptation will be slow if ϵ is too small. By taking a large value for ϵ, K will be updated every reset instant, also when there is no limit cycle present. A value of 0.1 is a good compromise.

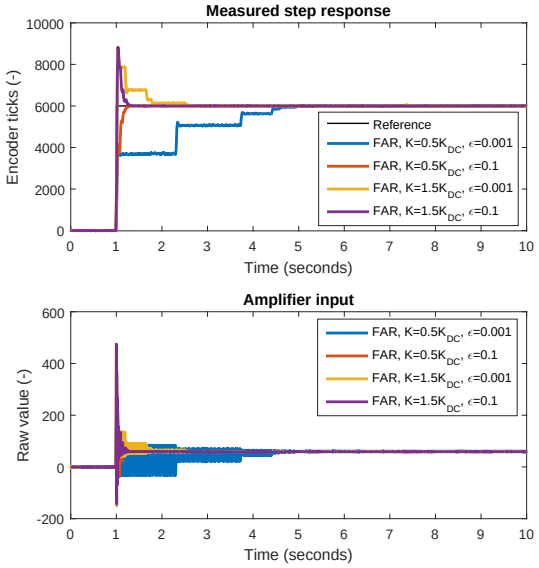

Figure 5.3: Influence of ϵ on step response.

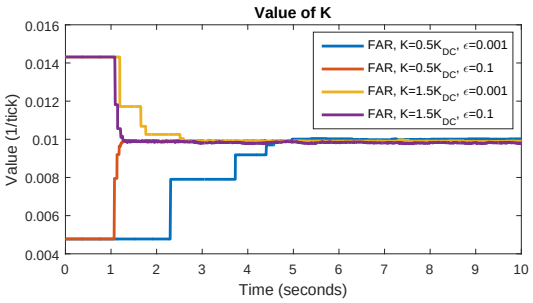

Figure 5.4: Influence of ϵ on adaptation speed of K.

The reset interval is set to $\Delta t_k = 4$ ms. This value is chosen based on several test simulations for different values, see Figure 5.5 and 5.6. Using a large Δt_k will result in a more linear response, while for a very small Δt_k the integrator is integrating for a short period before it is reset, so in that case the integral action is not really used anymore. It turned out that for $\Delta t_k = 4$ ms the response is smooth, while still having an active integral action.

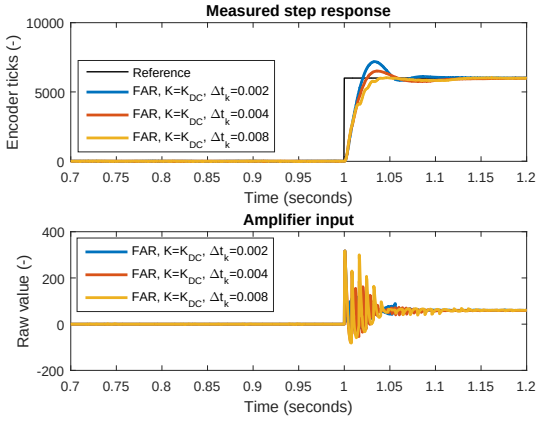

Figure 5.5: Influence of Δt_k on step response.

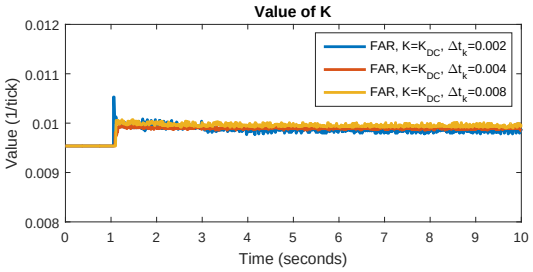

Figure 5.6: Influence of Δt_k on adaptation speed of K.

5.2 Fixed instant (adaptive) reset control

In this section the simulation and measurement results for the fixed instant (adaptive) reset controller are shown. First, step responses will be considered. After that, the ability to reject constant input disturbances is studied. Furthermore, it is investigated if the controller is able to track a piecewise constant trajectory. All mentioned values for K in the legends of the plots indicate the *initial value* for K. The Simulink implementation to obtain the simulation results can be found in Appendix C.

5.2.1 Step responses

The response for a step reference of 6000 encoder ticks applied at $t = 1$ second is simulated and measured. First, simulations and measurements are performed for the case without adaptive algorithm applied. After that, the same experiment is performed with adaptive algorithm.

Step response without adaptive reset algorithm

The step response is simulated and measured for $K = K_{DC}$, $K = 0.5K_{DC}$ and $K = 1.5K_{DC}$, where $K_{DC} = 0.009537207467789$ is the inverse DC gain of the identified plant in equation (4.1). The response is simulated and measured for 10 seconds. The results for 0.9 to 1.2 seconds are shown in Figure 5.7. In the figures, fixed instant reset control is abbreviated as FR.

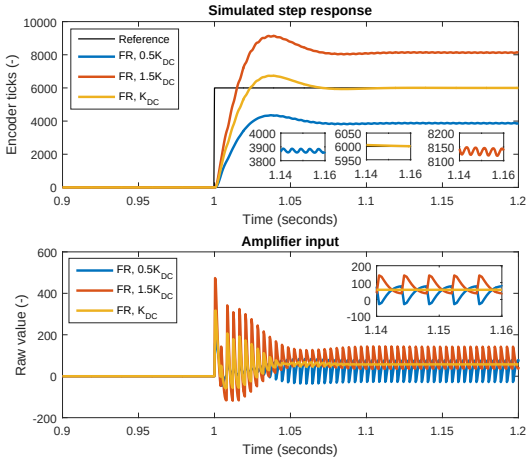

Figure 5.7: Simulated step response, fixed instant reset control without adaptation.

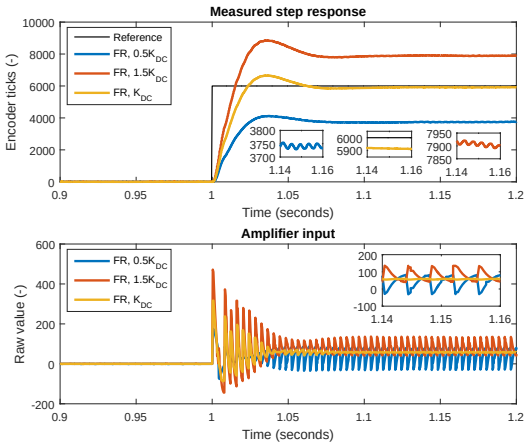

Figure 5.8: Measured step response, fixed instant reset control without adaptation.

As can be observed, the simulated and measured response are almost identical. For a wrong value of K ($K = 0.5K_{DC}$ and $K = 1.5K_{DC}$), a large steady state error is present. Furthermore, in the control signal a periodic saw-tooth pattern with equal amplitude is observable. This indicates that a limit cycle is present, although this limit cycle is not clearly observable in the response due to its small amplitude. In the *measured* response, also a steady state error is present for $K = K_{DC}$, even as a small limit cycle.

This is caused by the fact that K_{DC} is based on the identified transfer function and is not exactly equal to the right inverse DC value of the setup for each measurement. So without the ability to adapt K the response is sensitive to model uncertainties.

Step response with adaptive reset algorithm

The above experiment is repeated, but this time the proposed adaptive reset algorithm is applied. The responses were simulated and measured again for a duration of 10 seconds. The results for 0.5 to 2 seconds are shown in Figure 5.9 and 5.10. The fixed instant adaptive reset controller is abbreviated as FAR.

As can be observed, the simulation and measured results show similar behavior. In simulation and practice, the fixed instant adaptive reset controller removes the limit cycle and steady state error over time for all three considered initial values for K. In the measured response, small fluctuations around the desired set-point of 6000 are observable. Also in the measured control signal, small fluctuations are observable. These fluctuations are caused by noise and by the fact that the output of the myRIO is mapped into an integer value. However, these fluctuations are not a limit cycle, since they are non-periodic. In the simulation results these fluctuations are not present due to the absence of noise and the fact that the control signal is not mapped into an integer value.

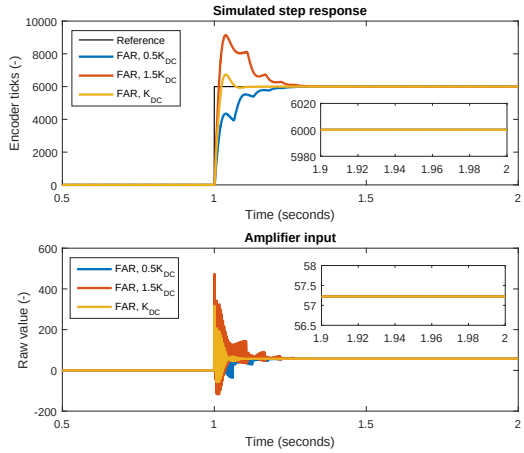

Figure 5.9: Simulated step response, fixed instant reset control with adaptation.

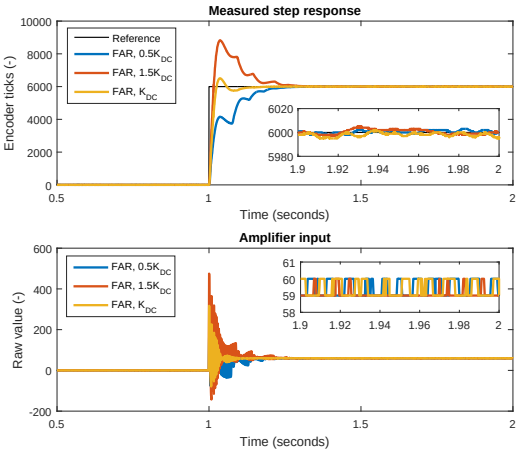

Figure 5.10: Measured step response, fixed instant reset control with adaptation.

In Figure 5.11 the simulated adaptation of K is shown for the three different initial values of K for the complete duration of 10 seconds. For all three cases, K converges exactly to the inverse DC gain of the plant in equation (4.1) as should be the case. For initial $K = K_{DC}$, the value of K stays (almost) constant. The value of K for initial $K = K_{DC}$ is however slightly adapted around $t = 1.1$. This is caused by the used threshold of $\epsilon = 0.1$. Around $t = 1.1$ the difference between two consecutive jumps in the output of the PI part of the controller turns out to be less than 0.1 (see also Figure 5.1 and equation 5.2). Hence these jumps are considered as a limit cycle and K is adapted. By using a smaller value for ϵ, this adaptation will not happen.

Since noise is present in practice, two consecutive jumps will never be exactly equal if a limit cycle is present. Hence, taking a too small value for ϵ in practice will prevent the algorithm to adapt K. A smaller value for ϵ could be used in simulation, since noise was not added. However, in Section 5.1 it was shown that $\epsilon = 0.1$ is a good value in practice. Therefore, this value is used in simulation as well in order to compare the results.

Although K is slightly adapted for initial K_{DC} in simulation, the final found value of K is exactly equal to K_{DC}. Also for the other initial values for K, the value converged to K_{DC} in simulation. This shows that the proposed adaptation law for K works in simulation for the used PID base controller with low-pass filter.

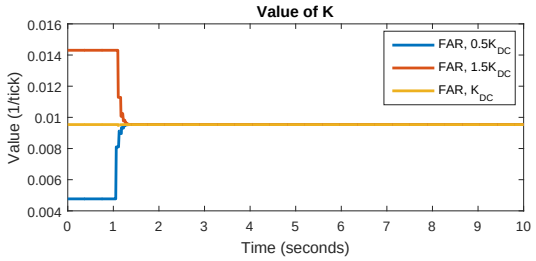

Figure 5.11: Simulated adaptation of K, fixed instant reset.

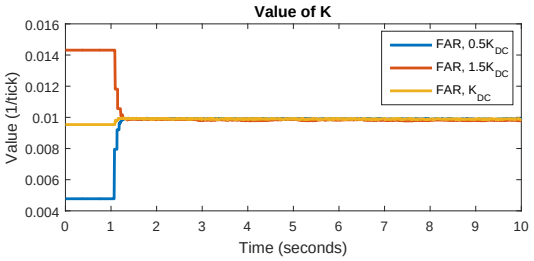

Figure 5.12: Measured adaptation of K, fixed instant reset.

In 5.12 the measured adaptation of K is shown. It can be observed the algorithm adapts K to almost the same value for all three considered cases. For initial K_{DC}, the value of K is slightly adapted, since K_{DC} is not exactly equal to the inverse DC gain of the identified model of the setup. Furthermore, due to noise, nonlinearity of the amplifier and setup and the used threshold of $\epsilon = 0.1$, the final value of K will never be the same for each measurement. Although this is the case, the adaptive reset algorithm adapts K in the right way, since limit cycles and steady state error are removed as observed in Figure 5.9 and 5.10.

Step response, second trial

Based on the results obtained with the adaptive reset algorithm, the value of K at $t = 10$, for the case started with K_{DC} is taken as initial value for K in a second measurement. This value is $K = 0.009842$ for the measured response. For the simulated response $K = K_{DC}$, which will give the same response as already shown. Hence, only a new measurement is performed for initial $K = 0.009842$. The results are shown in Figure 5.13 and 5.14. For comparison, also the response of the measurement for initial $K = K_{DC}$ is shown.

As can be observed in Figure 5.13, the response with the adapted K as initial value is slightly faster and has a slightly higher overshoot. This is what is expected since the initial K is slightly higher than the initial K_{DC} used in the first measurement and is closer to the right inverse gain of the system. The response converges to the desired set-point of 6000 encoder ticks, although a small fluctuation is observable due to noise. The fluctuations in the control signal are also caused by noise and the fact that the output of the myRIO (amplifier input) is an integer value.

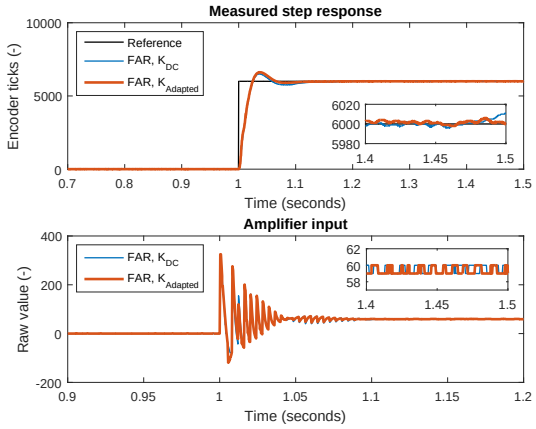

Figure 5.13: Measured step response, second trial.

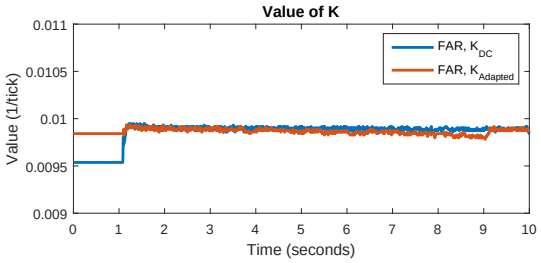

Figure 5.14: Measured step response, second trial.

In Figure 5.14 the adaptation of K for the complete measurement duration is shown. The value of K is slightly adapted during this second measurement in order to overcome nonlinearities of the system and the presence noise. However, no large adaptations are observable. This experiment shows that it can be beneficial to use the last found K as initial value for K in a next measurement.

5.2.2 Disturbance rejection

To check whether or not the adaptive algorithm is able to reject constant input disturbances, a step disturbance is applied at the plant input. A step reference of 6000 encoder ticks is taken again. For the disturbance, a step with an amplitude of 30 (raw value) is applied. This value is around 50% of the steady state control signal. First, the case without adaptation of K is considered. After that, the same experiment will be performed with the adaptive algorithm.

Disturbance rejection without adaptive reset

In case of a perfect model and in the absence of disturbances, using $K = K_{DC}$ will result in perfect step reference tracking and no adaptation of K is required. However, in case of model mismatches and disturbances, this is not the case anymore. Therefore, the purpose of this experiment is to show the need of an adaptive algorithm in case disturbances are present.

A measurement with input disturbance is performed for initial $K = K_{DC}$. This value may not exactly be equal to the inverse DC gain of the setup and may result in a steady state error and limit cycle already before the constant input disturbance is applied. This is not an issue, since the disturbance will create another limit cycle and a steady state error in that case.

The obtained measurement results are shown in Figure 5.15 and 5.16. The *'amplifier input'* is the perturbed control signal (so with disturbance included). As can be observed, a limit cycle is present after the disturbance is applied. This is clearly observable in the control signal, since a periodic pattern with equal amplitude is present from that moment on. Besides that, a large steady state error becomes present in the response. This is very unwanted.

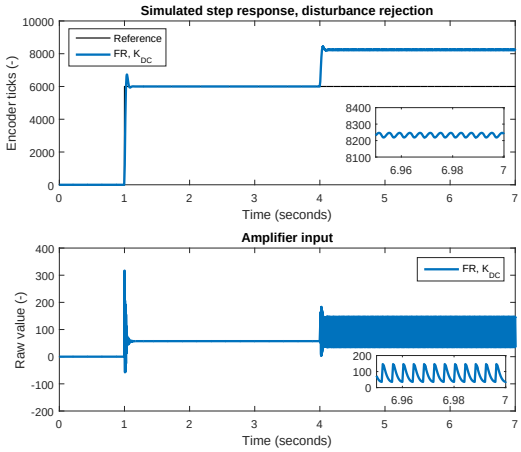

Figure 5.15: Simulated disturbance rejection, without adaptation.

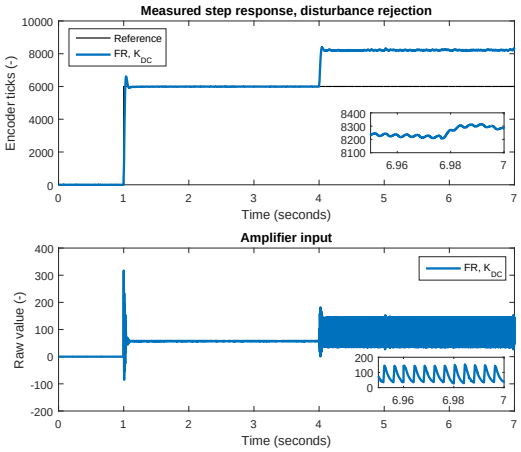

Figure 5.16: Measured disturbance rejection, without adaptation.

Disturbance rejection with adaptive reset

The above experiment is repeated, but this time with the adaptive algorithm enabled. As initial value $K = K_{DC}$ is taken again. Since the adaptive algorithm is enabled, the value of K can adapt to the right value before the disturbance is applied at $t = 4$. Instead of only adding a positive step disturbance of 30 at $t = 4$, also a step disturbance of -30 is applied at $t = 7$ in order to see if the controller is able to adapt in both cases. The simulation and measurement results are shown in Figure 5.17 and 5.18. As can be observed, it take some time to reject the disturbances, but they are rejected.

In Figure 5.19 and 5.20 the adaptation of K is shown. To reject the first disturbance, K should adapt to $K = K_{DC} - \frac{\text{disturbance}}{r(t)} \approx 0.004537$. In simulation, this is exactly the case. In the measurement, K is adapted to $K \approx 0.004547$ at $t = 7$, which is close to the expected value. To reject the second disturbance, the value of K should finally be adapted back again to the value before the first disturbance was applied. In simulation, $K = K_{DC}$ again at $t = 10$. In te measurement $K = 0.009567$ at $t = 4$, so between $T = 7$ and $t = 10$ the value should adapt to this value. This is indeed the case, since the value is at $t = 10$ is $K = 0.009567$ again. Due to the fact that the algorithm is able to adapt K to the required value, the limit cycle and steady state error are removed even as the applied disturbances.

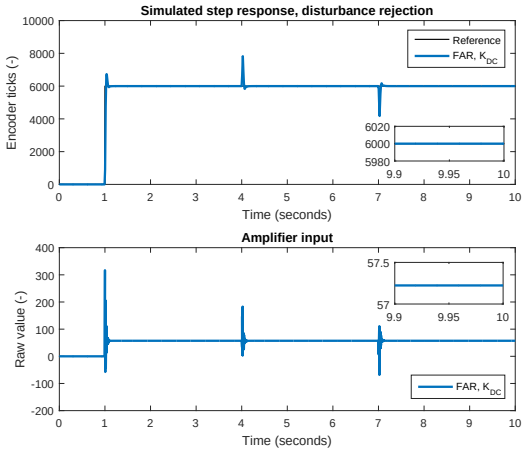

Figure 5.17: Simulated disturbance rejection, with adaptation.

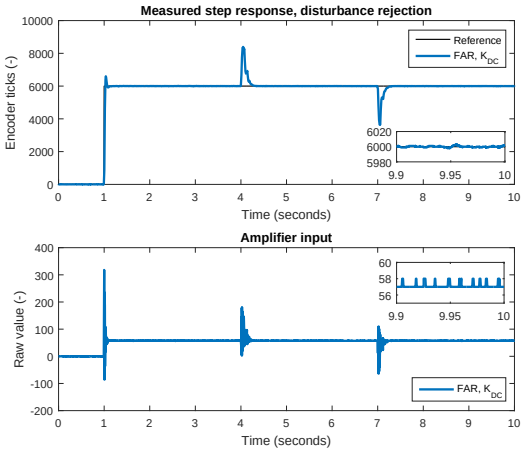

Figure 5.18: Measured disturbance rejection, with adaptation.

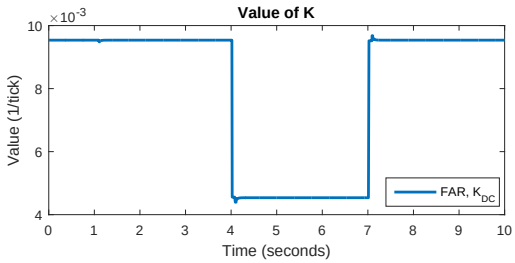

Figure 5.19: Simulated adaptation of K, fixed instant reset.

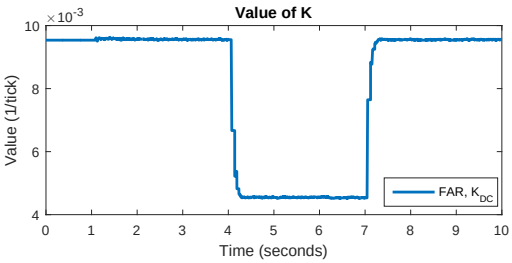

Figure 5.20: Measured adaptation of K, fixed instant reset.

5.2.3 Piecewise constant reference tracking

In the foregoing experiments, it is shown that the adaptive reset controller is able to adapt K in case of a constant step reference. In this section, a last experiment is performed where a piecewise constant reference is used. At $t = 1$, a reference of 6000 encoder ticks is applied. At $t = 4$, the reference is changed to -3000 encoder ticks (3000 encoder ticks in the opposite direction), at $t = 7$ to 4500, at $t = 10$ to -6000 and at $t = 13$ to 0. The response is measured for initial $K = 0.5K_{DC}$, $K = 1.5K_{DC}$ and $K = K_{DC}$. The simulation and measurement results are shown in Figure 5.21 and 5.22.

The results in both figures show that the reference signal is tracked for all three cases. For the cases with initial $K = 0.5K_{DC}$ and $K = 1.5K_{DC}$, the algorithm has to adapt K first to the correct value. Therefore, the response at $t = 1$ is different than at the other instants where the reference changes.

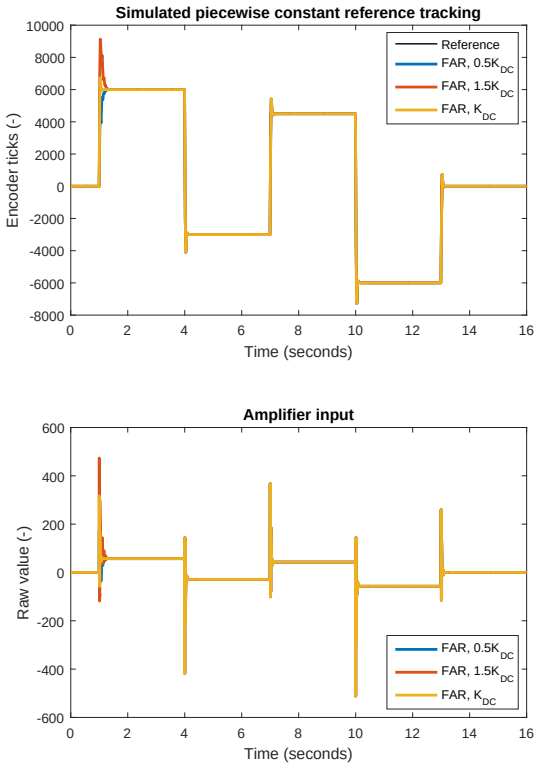

Figure 5.21: Simulated piecewise constant reference tracking with fixed instant adaptive reset controller.

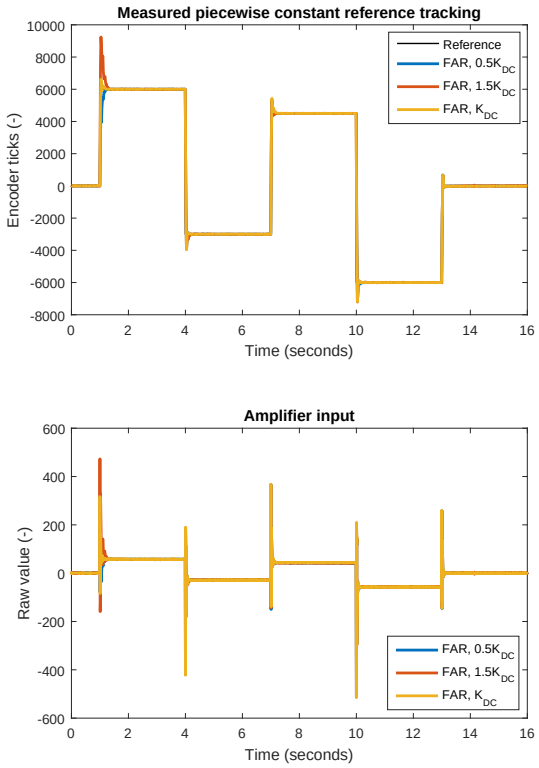

Figure 5.22: Measured piecewise constant reference tracking with fixed instant adaptive reset controller.

In Figure 5.23 and 5.24, the adaptation of K is shown. It can be observed that at $t = 1$, K starts adapting. In simulation, K converges to K_{DC} as should be the case. Around the instants where the reference changes, K is slightly adapted, also when K_{DC} is used. In the measured response, it is better observable that K is adapted around the moments the reference changes. This is caused by the used threshold of $\epsilon = 0.1$. Lowering this value will avoid this, but if an uncertainty or noise is present, the adaptation process of K will be slower or K will not be adapted at all as explained earlier.

The measurement also shows that K is more often adapted and does not exactly converge to the same value for the three measurements. This is caused by noise and nonlinearity of the amplifier and setup. However, the reference is tracked, without a steady state error and limit cycle, which would not be the case if K was not adapted or not adapted to the right value.

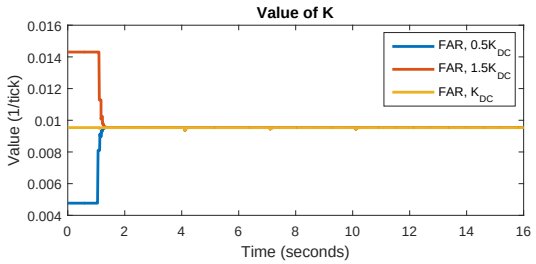

Figure 5.23: Simulated adaptation of K, fixed instant reset.

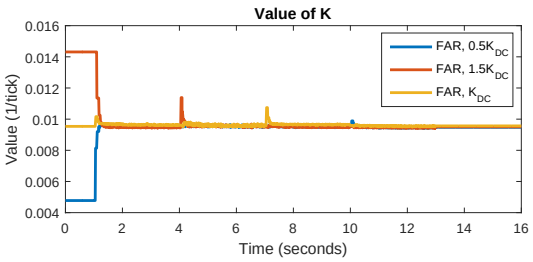

Figure 5.24: Measured adaptation of K, fixed instant reset.

5.3 Zero-crossing (adaptive) reset control

Since zero-crossing reset is most popular in literature, the adaptive algorithm will also be tested for zero-crossing reset. The proposed adaptive algorithm is initially developed for a fixed instant reset controller, but since the principle of resetting is the same, it is expected that the adaptation process will work for zero-crossing reset as well. In this section, the same experiments as done for the fixed instant reset controller will be performed in order to investigate if this is indeed the case.

For the experiments, the same tuned PID controller with low-pass filter is used, so the controller parameters stay the same, see Table 5.1. However, the reset law is changed to $e(t) = 0$. Furthermore, to avoid resetting every sample instant and limit the number of resets due to noise, the minimum amount of time between two resets (τ) is set to $\rho = 0.0012$ seconds, which is three times the used sampling time of 4 ms. Hence, the PI part of the adaptive PID reset controller is changed to (5.3). Notice that the update process for K (blue lines) is not changed.

$$C_{PIreset} := \begin{cases} \dot{x}_r(t) = e(t) & \text{if } e(t) \neq 0 \text{ or } \tau < 0.0012 \\ \left.\begin{array}{l} J_k = |u_r(t_k^-) - K_{k-1} r(t)| \\ K_k = \begin{cases} K_{k-1} & \text{if } |J_k - J_{k-1}| > \epsilon \text{ or } r(t) = 0 \\ \frac{u_r(t_k^-)}{r(t)} & \text{if } 0 \leq |J_k - J_{k-1}| \leq \epsilon \text{ and } r(t) \neq 0 \end{cases} \\ x_r(t^+) = K_k \frac{\tau_i}{k_p} r(t) - \tau_i e(t) \end{array}\right\} & \text{if } e(t) = 0 \text{ and } \tau \geq 0.0012 \\ u_r(t) = \frac{k_p}{\tau_i} x_r(t) + k_p e(t) \end{cases} \tag{5.3}$$

Due to the fact that zero-crossings might be caused by noise as well, band limited white noise is added to the control signal in the simulations. For this purpose the 'band limited white noise' block in SIMULINK is used with the following settings: noise power 0.001, sample time (of the noise only) 0.01 seconds and seed 23343. The Simulink implementation to obtain the simulation results can be found in Appendix D.

5.3.1 Step responses

In this section, the step responses for a reference of 6000 encoder ticks are shown for the zero-crossing PID reset controller. First simulation and measurement results are given for the case without adaptation. After that, the same experiment is performed with adaptive algorithm. Since it is conventional in zero-crossing reset to reset to zero, $K = 0$ is taken as initial value. All values of K mentioned in the plot legends indicate the initial value for K that is used.

Step response without adaptation

The simulation and measurement results for a step of 6000 encoder ticks applied at $t = 1$ are shown in Figure 5.25 and 5.26. Clearly, a limit cycle is present in the responses. This is also what is expected, because the after reset value is not right for $K = 0$. Since K is not adapted, the limit cycle stays present.

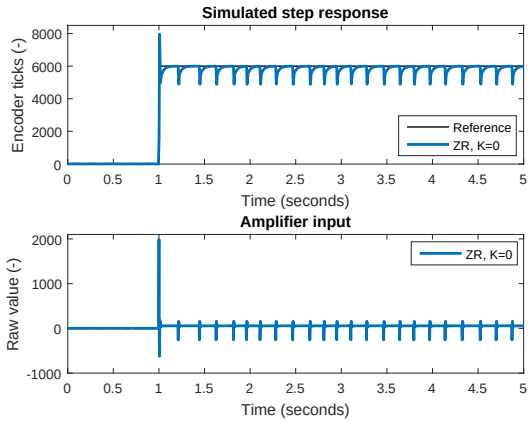

Figure 5.25: Simulated step response for zero-crossing reset without adaptation.

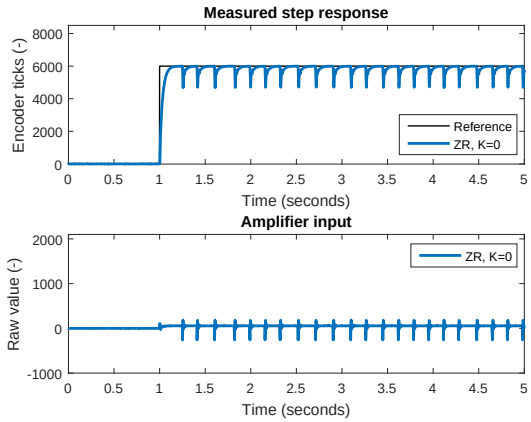

Figure 5.26: Measured step response for zero-crossing reset without adaptation.

A difference between the simulated and measured response is the overshoot. In the simulated response, an overshoot is present, while there is not in the measured response. The reference is zero from $t = 0$ to $t = 1$. In this interval, the error may cross zero due to noise and a reset occurs. Due to the restriction of 0.0012 seconds between two resets, the next reset may occur after $t = 1$, which can increase the control signal. This is what happened in simulation.

However, if due to noise the error is less than 0 at $t = 1$ and the last reset is more than 0.0012 seconds ago, the error crosses zero at $t = 1$ since the reference changes to 6000 at that time instant. Hence the integrator resets to zero directly at $t = 1$. This may lower the control signal and result in a slower response with less overshoot. This is what happened in the measurement and therefore, no overshoot is present in the measured response.

To show that the above explanation is right, a new simulation is performed for different noise settings (power 0.001, sample time 0.1 seconds, seed 23326). For these noise settings, a reset occurs at $t = 1$. The results are shown in Figure 5.27. As can be observed, no overshoot is present due to this reset at $t = 1$.

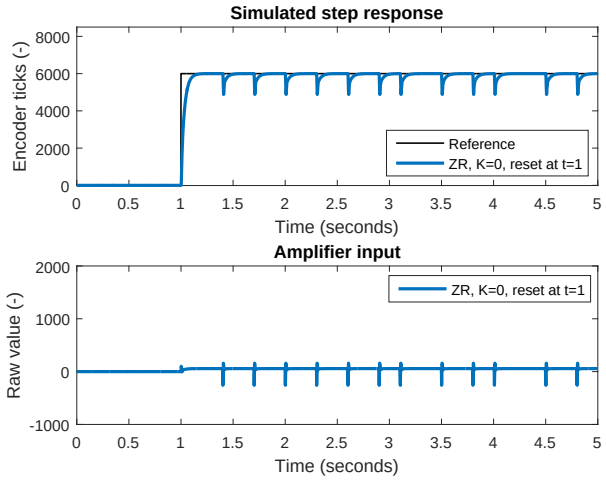

Figure 5.27: Simulated step response for zero-crossing reset without adaptation, different noise.

Step response reset with adaptation

The obtained results for a step of 6000 encoder ticks at $t = 1$ with the adaptive algorithm enabled are shown in Figure 5.28 and 5.29. As can be observed, the limit cycle is removed in simulation and also in the measured response. The difference in overshoot can still be explained by the presence of noise and the restriction to have at least 0.0012 seconds between two resets.

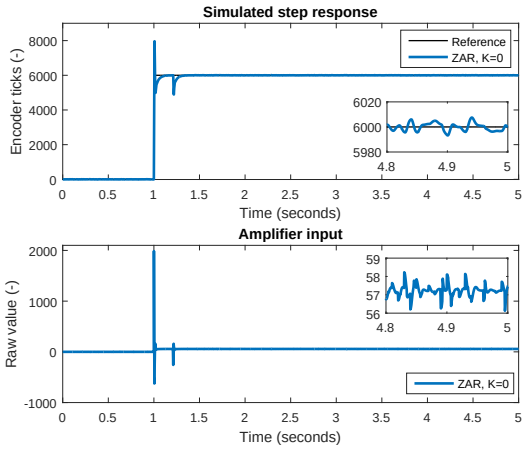

Figure 5.28: Simulated step response for zero-crossing reset with adaptation.

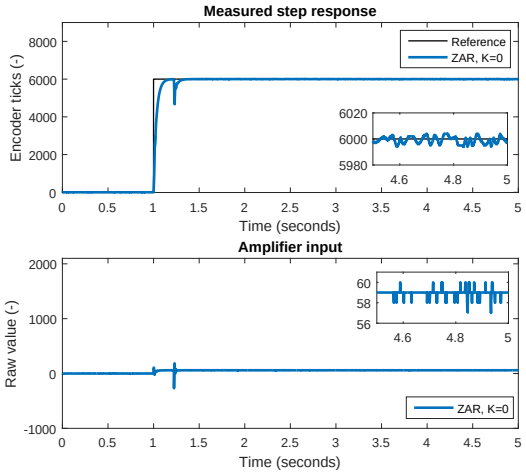

Figure 5.29: Measured step response for zero-crossing reset with adaptation.

The adaptation of K for the complete experiment duration is shown in Figure 5.30 and 5.31. In simulation, K is adapted at $t = 1.443$. In Figure 5.25 it can be observed that this is the moment the limit cycle has completed one cycle. Hence, the adaptive algorithm adapts K at that moment, as expected. In the measurement, K is also adapted around $t = 1.424$. In Figure 5.26 it can be observed that this instant corresponds also to the instant where the limit cycle would normally repeat itself in the measurement. Hence, also in this case, K is adapted in the right way. Furthermore, the convergence of K is faster for zero-crossing reset than for the fixed instant reset controller. It takes only one iteration before the right value is found.

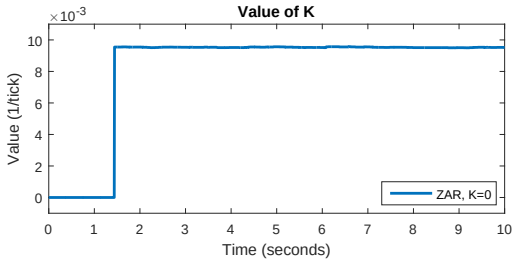

Figure 5.30: Simulated adaptation of K, zero-crossing reset.

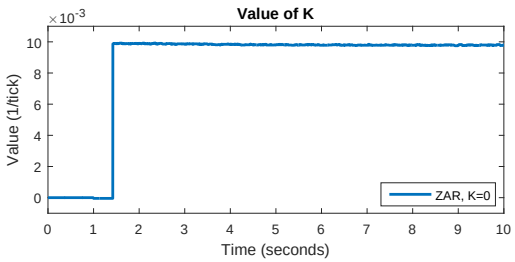

Figure 5.31: Measured adaptation of K, zero-crossing reset.

5.3.2 Step response, second trial

The value of K at $t = 10$ is used as initial value for K in a second measurement. This value is $K = 0.009525738974238$ in simulation (which is close to K_{DC}) and $K = 0.009796$ in the measurement. Again, a step response for 6000 encoder ticks is measured. For comparison, also the response for the case with initial $K = 0$ is shown again.

As can be observed, in Figure 5.32, the response with the adapted value for K as initial value has a higher overshoot, but is faster than started with K_{DC}. This is caused by the fact that the controller does not directly reset at $t = 1$ and the initial value of K is higher than $K = 0$. Due to this, it turns out that the control signal is increased at the first reset after $t = 1$, which results in a higher overshoot.

In Figure 5.33 the measured response is shown. It can be observed that the response with the adapted K as initial value for K is faster than for initial $K = 0$. However, it may happen that the controller resets directly at $t = 1$ if the error crosses zero at $t = 1$ and the last reset is more than 0.0012 seconds ago. This reset at $t = 1$ decreases the control signal, which leads to a slower response. If this reset does not occur, it turns out for the considered controller and setup that the first reset after $t = 1$ increases the control signal. This leads to a faster response, but with a higher overshoot. Although this is the case, the results show that a limit cycle is not present anymore in this second trial for initial $K = K_{Adapted}$.

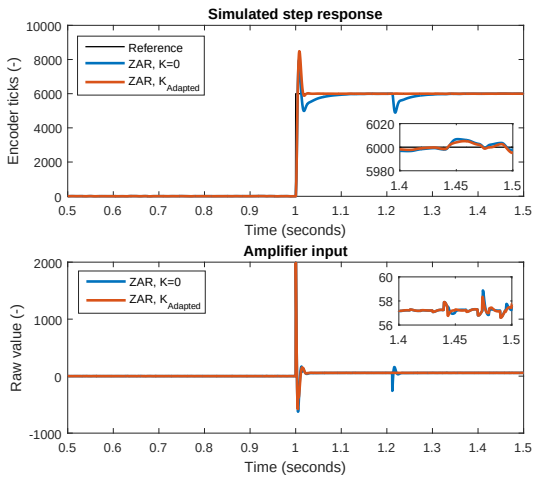

Figure 5.32: Simulated step response, second trial.

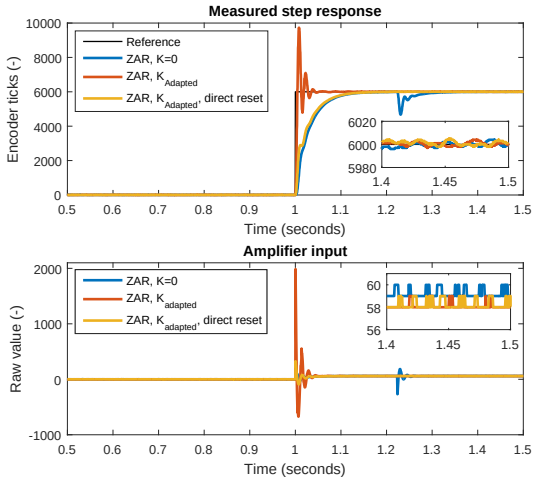

Figure 5.33: Measured step response, second trial.

Figure 5.34 and 5.35 show the adaptation of K. In simulation, K stays (almost) constant over time as expected, since $K_{Adapted}$ is almost equal to K_{DC}. In the measurement, it can be observed that K is adapted to a slightly lower value than started with. This is caused by noise and nonlinearity of the amplifier and setup.

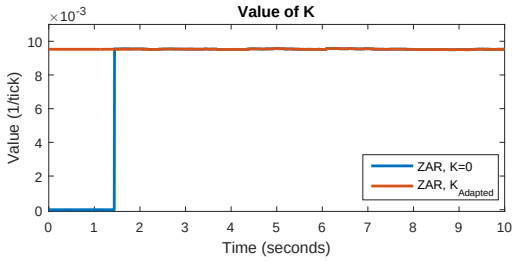

Figure 5.34: Simulated adaptation of K, zero-crossing reset.

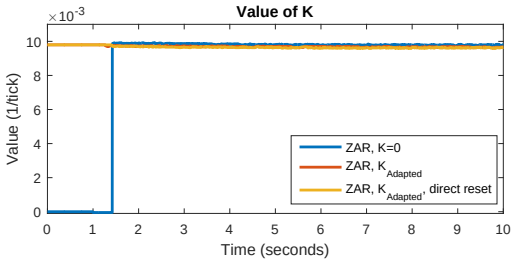

Figure 5.35: Measured adaptation of K, zero-crossing reset.

5.3.3 Disturbance rejection

To investigate if the PID reset controller is able to reject a constant input disturbance in case of zero-crossing reset, a step disturbance is added to the control signal. The reference is still set to 6000 encoder ticks and as initial value $K = 0$ is taken. The simulation and measurement results are shown for the case without and with adaptation of K.

Disturbance rejection without adaptation

The disturbance rejection results for the case without adaption are shown in Figure 5.36 and 5.37. The step input disturbance with an amplitude of 30 is applied at $t = 4$. As expected, a limit cycle is present in the obtained responses, since K is set to $K = 0$ and is not adapted. After the disturbance is applied, the amplitude of de limit cycle decreases, because the disturbance brings the amplifier input closer to its required steady state in this case.

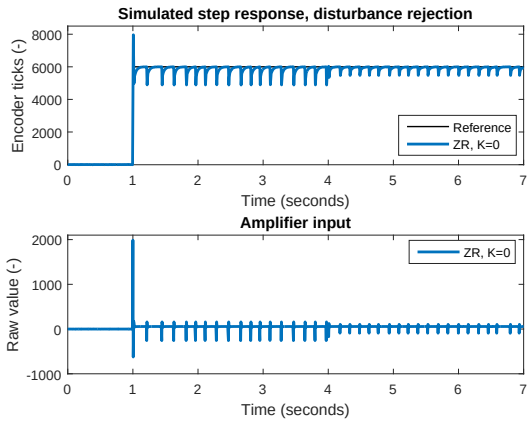

Figure 5.36: Simulated disturbance rejection, without adaptation.

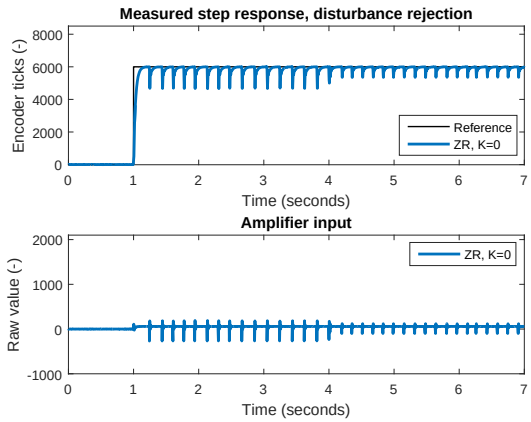

Figure 5.37: Measured disturbance rejection, without adaptation.

Disturbance rejection with adaptation

The disturbance rejection is again investigated for zero-crossing reset, but this time with adaptation of K. An input step disturbance with an amplitude of 30 is applied at $t = 4$ and at $t = 7$ an input step disturbance of -30 is applied. The obtained simulation and measurement results are shown in Figure 5.38 and 5.39.

As can be observed, in simulation it takes longer before the disturbances are rejected. This is caused by the added white noise and the used threshold of $\epsilon = 0.1$. As long as the difference between two consecutive jumps in the output of the PI part of the controller is not smaller than 0.1, K is not adapted. Hence, it may take some time before the condition to adapt K is fulfilled and the disturbance is rejected. In practice, the noise is not the same as in simulation and the adaptation condition is fulfilled earlier. This results in an earlier adaptation of K and a faster rejection of the disturbance. Although this is the case and the speed of rejection of a constant input disturbance is a bit unpredictable, limit cycles are removed.

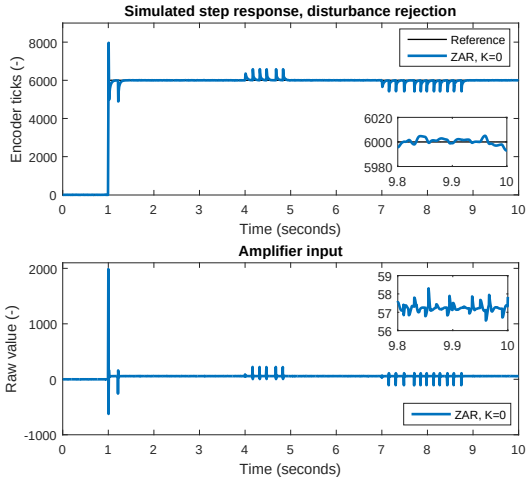

Figure 5.38: Simulated disturbance rejection, with adaptation.

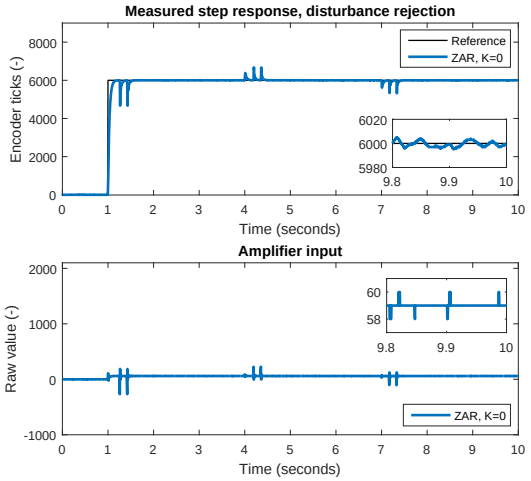

Figure 5.39: Simulated disturbance rejection, with adaptation.

The adaptation process of K in simulation and practice is shown in Figure 5.40 and 5.41. In simulation, K is first adapted to approximately the inverse DC gain of the plant around $t = 1.443$ as should be the case since no significant disturbance is present until $t = 4$. However, since noise is added, the value of K will not be exactly equal to K_{DC}. At $t = 4.966$, the value of K is adapted to $K \approx K_{DC} - \frac{\text{disturbance}}{r(t)} \approx 0.0045$ as is to be expected. Then at $t = 7$ the second step disturbance of -30 is applied and K is adapted again to $K \approx 0.0095 \approx K_{DC}$ at $t = 8.924$. This is exactly what should happen, since the two applied disturbances will cancel each other, hence K should be adapted to the correct value for the case no disturbance is applied, which is (very close to) K_{DC} in simulation. Wwithout noise added, it is exactly equal to K_{DC}.

In the measurement the same adaptation process of K can be observed. The only difference is that the adaptation takes place at different time instants. This is caused by the fact that the limit cycle has a different period in practice, since the resets occur at different instants due to noise. At $t = 1.584$, K is adapted to 0.009842. Then at $t = 4.515$ the value of K changes from 0.009888 to 0.004837, which is around the theoretical expected value. At $t = 7.49$ the value is adapted from $K = 0.004822$ to 0.009811. At $t = 10$, $K = 0.009842$ which is close to the value K was adapted to at $t = 1.584$ before any disturbance was applied. Furthermore, K is also slightly adapted over time although the disturbance stays constant. This is caused by noise, since the update condition for K can be satisfied by noise due to the used threshold of $\epsilon = 0.1$.

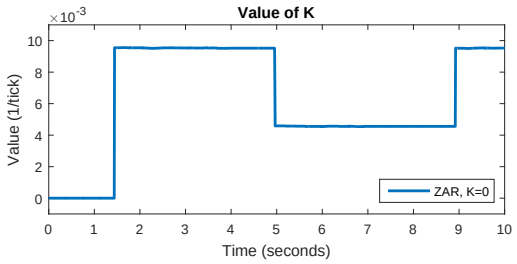

Figure 5.40: Simulated adaptation of K, zero-crossing reset.

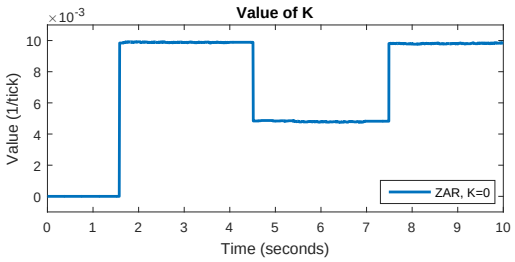

Figure 5.41: Measured adaptation of K, zero-crossing reset.

5.3.4 Piecewise constant reference tracking

The last experiment that is performed is tracking a piecewise constant reference. The same reference as used for the fixed instant adaptive reset controller is used. The initial value of K is set to 0. This resulted in the results shown in Figure 5.42 and 5.43. As can be observed, the trajectory is properly tracked in simulation and in practice. There are some differences in overshoot observable at the instants the reference changes. These differences are due to the fact that the controller resets when the error crosses zero and at least 0.0012 seconds is passed after the last reset.

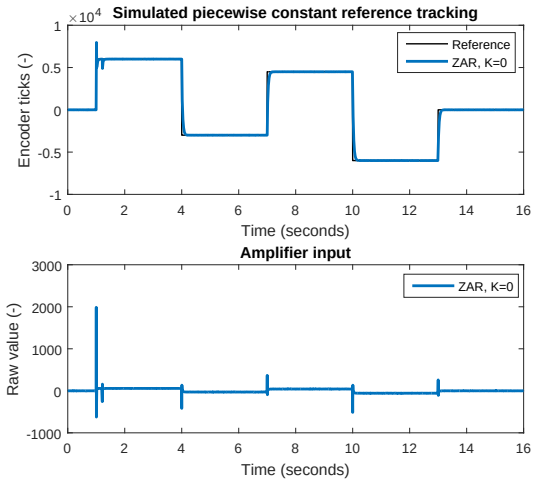

Figure 5.42: Simulated piecewise constant reference tracking, zero-crossing reset.

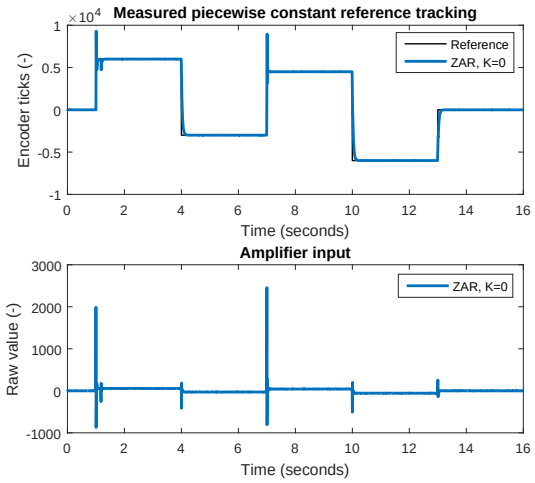

Figure 5.43: Measured piecewise constant reference tracking, zero-crossing reset.

In simulation, it turns out that the controller resets at the moments the reference changes, except for the first step due to te applied restrictions. A reset at the time instants where the reference changes, results in a lower control signal and a lower overshoot. In the measurement, the same behavior is observable, but at $t = 7$, the controller does not reset due to the applied restrictions. It turns out that the control signal is increased by the first reset after $t = 7$, which leads to a higher overshoot. So, the responses for the zero-crossing reset controller is sensitive to the moment the reference changes and if a reset is directly applied at that moment or not. This is in fact also the case for fixed instant reset control, but there it can be predicted if a reset will occur at the moment the reference changes.

In Figure 5.44 and 5.45 the adaptation of K is shown. The value of K is adapted from $K = 0$ to $K \approx K_{DC}$ and is fluctuating a bit around this value due to the added noise. In the measurement, K is also close to this value and is also fluctuating a bit. The value of K stays almost constant after it is adapted to K_{DC} as should be the case.

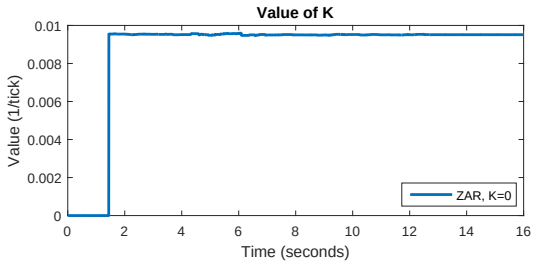

Figure 5.44: Simulated adaptation of K, zero-crossing reset.

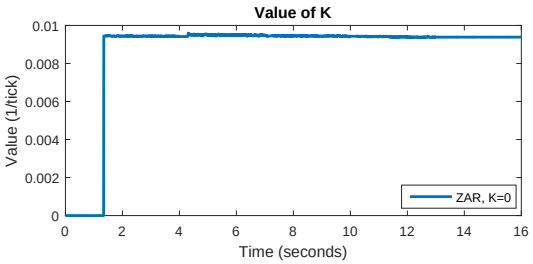

Figure 5.45: Measured adaptation of K, zero-crossing reset.

Although the above experiments show that the algorithm is able to adapt K and remove limit cycles also in case of zero-crossing reset, it cannot be guaranteed that it works for every minimum phase SISO system. For the system and tuned controller used in this thesis it turned out that the proposed algorithm is able to remove limit cycles when zero-crossing reset is used. However, in case a limit cycle has a period of more than one reset interval, this might not be the case as explained in Chapter 3. Increasing the minimum amount of time between the resets or by comparing jumps spaced by the period of the limit cycle have to be considered in that case.

Chapter 6

Conclusion

The goal of this thesis was to develop a robust reset controller algorithm for minimum phase SISO systems that is able to reject limit cycles in case of model uncertainty and constant input disturbances when tracking piecewise constant references. The proposed adaptive PID reset controller in this thesis adapts the after reset value if two consecutive jumps in the control signal are (almost) equal. The after reset value is changed by adapting a gain K to the value the output of the PI part of the controller has before reset, scaled by the inverse of the reference.

The presented algorithm is based on the assumptions that the system is a minimum phase SISO system that can asymptotically be stabilized by a standard PID controller and that a present limit cycle has a period of one reset interval. Although no mathematical proof is given, an explanation behind this approach was given. Based on obtained simulation and measurement results, it can be concluded that the proposed fixed instant adaptive PID reset controller is able to remove limit cycles when uncertainties in the steady state gain are present. Besides that, the fixed instant adaptive reset controller is able to adapt the after reset value in such a way that constant input disturbances are rejected as well.

The controller was also tested for zero-crossing reset. The results showed that the algorithm is also able to adapt K in the right way for zero-crossing reset, at least for the setup and controller used. It can however not be concluded that the algorithm works for zero-crossing reset in general. The reason for this is that limit cycles can have a period of more than one reset interval when zero-crossing reset is used. In that case, it does not work to compare two consecutive jumps to detect a limit cycle. Although this might be the case, the shown results prove that the proposed adaptive algorithm works for fixed instant reset control and might work for zero-crossing reset control as well.

Chapter 7

Recommendations

The results obtained with the proposed adaptive algorithm in this thesis showed that the adaptive reset controller is robust against model uncertainties. Furthermore, the controller is able to reject constant input disturbances. Although the results are promising, the algorithm can still be improved to make it more robust and to improve the performance.

Although constant input disturbances are rejected, the sensitivity and rejection speed can be improved. A suggestion to improve the disturbance rejection is to detect and log the time instant at which a disturbance is applied and the moment when it is (almost) rejected. If also the values of K are logged at these instants, they can be used in a next run in order to speed up the disturbance rejection. In order to detect disturbances, a disturbance observer is probably needed or changes in the error signal can be used. This procedure will make the controller more valuable for pick and place applications.

A second suggestion for future work is to investigate what the most optimal time instants are to reset the controller states in case fixed instant reset control is used. For this, the work of Guo et al. in [3] can be considered. Another suggestion is to test the algorithm for smooth references and trajectories, non-constant disturbances and for different types of systems. Furthermore, giving a mathematical proof of the proposed algorithm can also be considered as part of future work.

Appendix A

Examples of existing strategies

In this appendix examples of existing reset control methods are given. The examples will show that the current existing strategies are not robust or does not really provide a solution for the limit cycle problem.

A.1 Example of PI+CI controller

Consider a standard negative feedback loop as was shown in Figure 1.2. Let $C(s)$ represent a PI+CI controller as shown Figure 2.1 with $k_p = 2$, $\tau_i = 0.1$, $x(t^+) = 0$ and $p_r = 0.5$ and consider a plant given by $H(s) = \frac{1}{s+0.5}$

Simulating the closed loop system for a unit step reference gives the results shown in Figure A.1. For comparison also the case for $p_r = 0$ (only linear integration) and for $p_r = 1$ (only CI) is shown. As can be observed, the response with PI+CI control is faster than the with a normal PI controller and has less overshoot.

Furthermore it can be observed that although an oscillation is present in the PI+CI response, the amplitude decreases over time. This is due to the linear integrator, that takes care of the steady state error (as long as $p_r \neq 1$) and hence avoids the occurrence of a limit cycle. It can also be observed that when the linear integrator is not included in the controller ($p_r = 1$), a limit cycle is present which stays present. Since the controller is a trade-off between linear and nonlinear control, the response with PI+CI is not as fast as could be with pure reset.

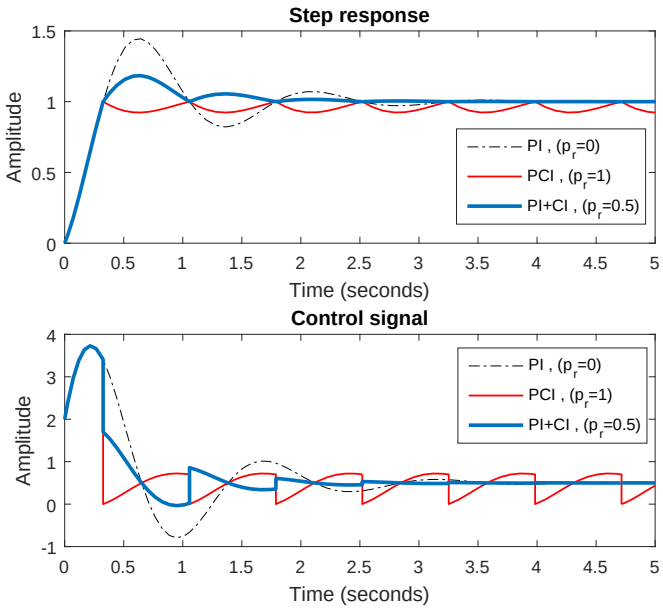

Figure A.1: Response with PI+CI controller

A.2 Example of reset controller with reset band

This example illustrates the concept of reset control with a reset band as presented in [2]. Consider a negative feedback loop as was shown in Figure 1.2. A PI reset controller with a reset band of $\delta = 0.1$ is used, which is defined in (A.1). Let the plant be given by $H(s) = \frac{1}{s+0.5}$. Simulating the system gives the response shown in Figure A.2. As can be observed, a limit cycle is present at the lower bound of the reset band.

$$C := \begin{cases} \dot{x_c}(t) = e(t) & \text{if } (e(t), \dot{e}(t)) \notin B_\delta \\ x_c(t^+) = 0 & \text{if } (e(t), \dot{e}(t)) \in B_\delta \\ u(t) = \frac{k_p}{\tau_i} x_c(t) + k_p e(t) & k_p = 2, \tau_i = 0.1 \end{cases} \tag{A.1}$$

However, when a reset band of $\delta = 0.2$ is used, a limit cycle is not present, see Figure A.3. So, using a reset band can avoid limit cycles, but it depends on the value of the reset band, which makes it less robust against model uncertainties. Furthermore, as can be observed in Figure A.3, one reset is applied and after that, the response stays within the reset band where the controller acts just like a linear controller, which makes the transient response less fast than it could be.

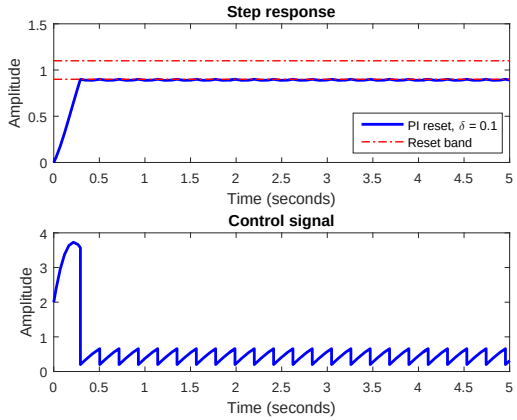

Figure A.2: Response with PI reset controller with reset band, $\delta = 0.1$

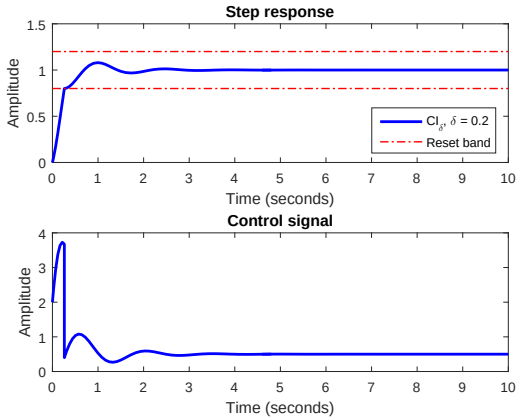

Figure A.3: Response with PI reset controller with reset band, $\delta = 0.2$

A.3 Limit cycle analysis

This example will show that the limit cycle analysis as proposed by [16] is not accurate for non-autonomous systems. Consider the negative feedback loop as was shown in Figure 1.2. Let the plant be given by $H = \frac{1}{s+0.5}$ and take a CI with reset band, which has a describing function as given in (A.2) (see [2, p.132]). The normal time domain equations are given in A.3.

$$CI_\delta(E,\omega) = \frac{1}{j\omega}\left(1 + \frac{j4(\sqrt{1-(\frac{\delta}{E})^2})}{\pi} e^{jsin^{-1}(\frac{\delta}{E})}\right) \tag{A.2}$$

$$CI_\delta := \begin{cases} \dot{x_c}(t) = e(t) & \text{if } (e(t), \dot{e}(t)) \notin B_\delta \\ x_c(t^+) = 0 & \text{if } (e(t), \dot{e}(t)) \in B_\delta \\ u(t) = x_c(t) \end{cases} \tag{A.3}$$

Using the method of [16] for $\omega = 0.5$, $\omega = 1$ and $\omega = 1.5$ gives the plot shown in Figure A.4. There are no crossings, so based on this analysis no limit cycles should be present. However, the closed loop step response for the considered system with a reset band of $\delta = 0.1$ shows the presence of a limit cycle as can be seen in Figure A.5.

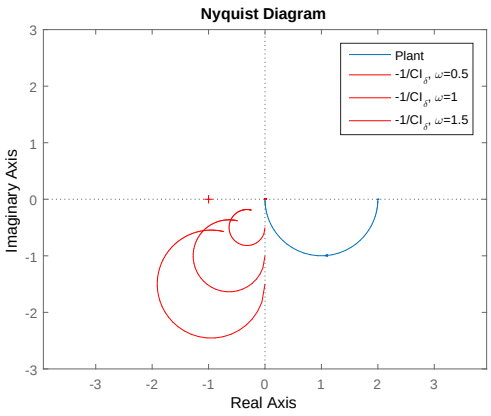

Figure A.4: Limit cycle analysis

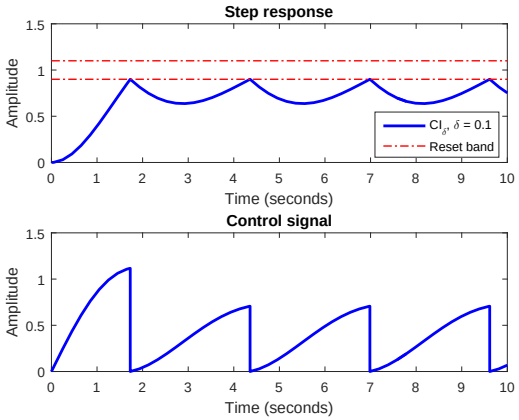

Figure A.5: Response with CI_δ

A.4 Example of improved reset controller

The work of Zheng et al. [4] is reproduced here, since it is an interesting control strategy. A constant reset interval of $\Delta t_k = 1$ ms is taken. The plant as used in [4] is defined as (A.4), with $a_1 = 10^6$, $a_2 = 1810$ and $b = 3 \cdot 10^6$. The reset controller is given by (A.5), with $k_p = 0.08$ and $k_i = 300$ (which is a PI controller with reset). Furthermore, for the cost function $P_0 = 2.1$, $Q_0 = 10^{-6}$ and $P_1 = 0$ is taken, which according to [4] leads to $E_1 = -2.8 \cdot 10^{-4}$, $E_2 = -6.8 \cdot 10^{-7}$ and $G = 0.0014$.

$$P := \begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -a_1 x_1 - a_2 x_2 + bu \\ y_p = x_1 \end{cases} \tag{A.4}$$

$$RC := \begin{cases} \dot{x}_r = e, & t \neq t_k \\ x_r(t_k^+) = E_1 x_1 + E_2 x_2 + Gr, & t = t_k \\ u = k_i x_r + k_p e \end{cases} \tag{A.5}$$

The reproduced results of the work of Zheng et al. with the aforementioned values is shown in A.6, where it is also compared with conventional reset PI reset control (reset to zero) and standard PI control. As can be observed, the response with the improved reset controller is faster than the response obtained with a normal PI controller. Furthermore, the response has (almost) no overshoot.

If conventional reset is applied, a limit cycle is present in the response. With the improved reset controller there seems to be no limit cycle at first sight, but a detailed view in Figure A.7 shows that a limit cycle with small amplitude is present, even as a steady state error. Probably this comes due to the fact that the authors mentioned rounded off values for the controller parameters in the paper. This clearly shows the sensitivity to parameter uncertainties when reset control is used.

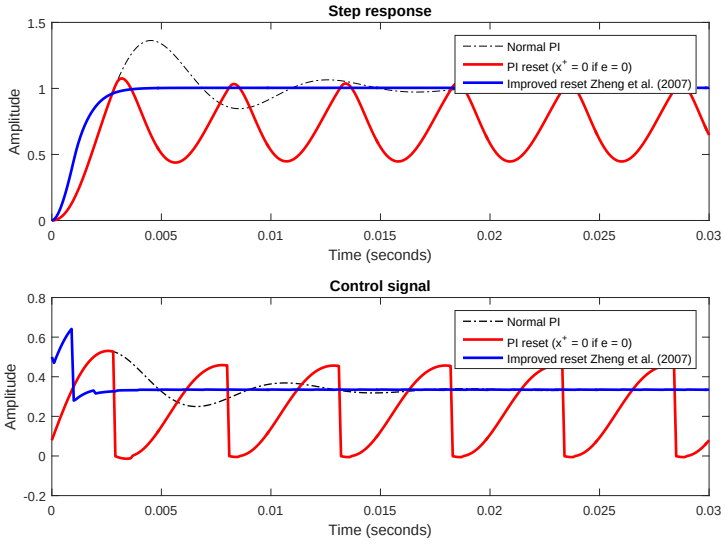

Figure A.6: Reproduced results of the improved reset controller of Zheng et al. (2007).

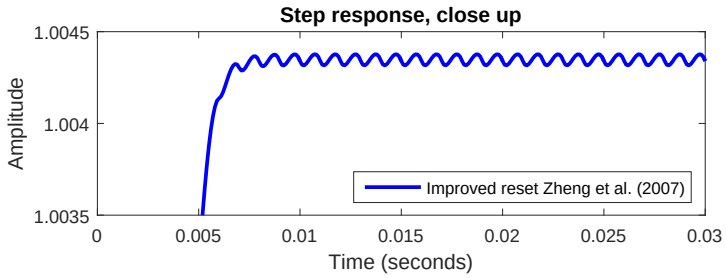

Figure A.7: Limit cycle in step response with improved reset controller.

A.5 Example of general fixed instant PI reset control

This example is performed to check the robustness of the general fixed instant reset controller as proposed by HosseinNia et al. in [20]. Consider the fixed instant reset controller given by (A.6) that was used in [20], with $K = \frac{1}{3}, k_p = 0.08, \tau_i = \frac{8}{3} \cdot 10^{-4}$ and $\Delta t_k = 1$ ms. In fact it is a PI reset controller which resets at fixed time instants. This controller was tuned for a second order system, given by $P(s) = \frac{3 \cdot 10^6}{s^2 + 1810s + 1 \cdot 10^6}$, which was used in [4].

$$C := \begin{cases} \dot{x}_r(t) = e(t), & \text{if } t \neq t_k \\ x_r(t^+) = K\frac{\tau_i}{k_p} r(t) - \tau_i e(t), & \text{if } t = t_k \\ u_r(t) = \frac{k_p}{\tau_i} x_r(t) + k_p e(t) \end{cases} \tag{A.6}$$

Now consider the situation that the actual plant is given by $P(s) = \frac{3 \cdot 10^6}{s^2 + 1810s + 0.8 \cdot 10^6}$. Then the gain $K = \frac{1}{3}$ is not equal anymore to the steady state value of the control signal (the right value is $K = \frac{0.8}{3}$ for this case). Simulating the system with the controller as defined in (A.6) gives the response shown in Figure A.8. As can be observed, a limit cycle and a large steady state error is present. The limit cycle is best observed in the control signal. Hence, this method is not robust against model uncertainties.

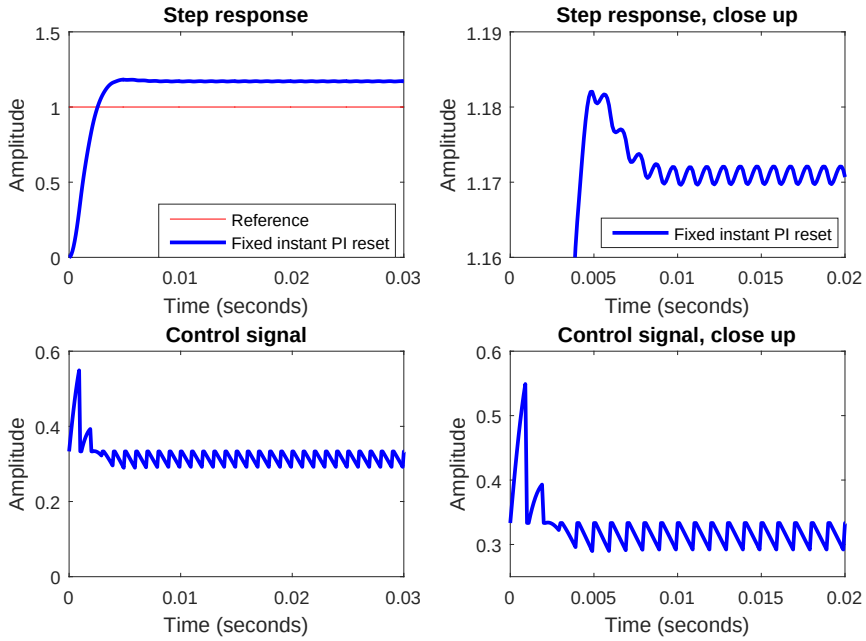

Figure A.8: Step response for fixed instant reset control in case of model mismatch.

Appendix B

M-file for System Identification

```
%% System identification
% This file is used to obtain a transfer function of the setup, based on
% an open loop input output measurement for a 0.1 - 50 Hz chirp signal.

close all
clear all
clc

% Load data, plot input signal and measured output
load('IdentificationData.mat')
Ts=50e-6;    % Sample time [s]

% Plot identification data
figure
subplot(2,1,1)
plot([0:length(Input)-1]*Ts,Input)
title('Input')
xlabel('Time (seconds)')
ylabel('Raw value')
subplot(2,1,2)
plot([0:length(Output)-1]*Ts,Output)
title('Ouput')
xlabel('Time (seconds)')
ylabel('Encoder ticks')

%% Estimate transfer function
Ts=Ts;        % Sample time
Fs=1/Ts;      % sampling frequency
Fmax=Fs/2;    % Nyquist frequency
[Tio, Fhz]=tfestimate(Input(1:end),Output(1:end),[],[],[],1/Ts);
M=abs(Tio);              % Magnitude
Prad=(angle(Tio));       % Phase in radians
Frads=2*pi*Fhz;          % Frequency in rad/s
```

```
34 MdB_dB=20*log10(M);     % Magnitude in dB
35 Pdeg=180/pi*Prad;       % Phase in degrees
36
37 %% Plots of estimated frequency response function
38
39 % Bode plot
40 figure
41 subplot(3,1,1)
42 semilogx(Fhz,mag2db(abs(M)),'LineWidth',2); hold on
43 xlim([0.1 250])
44 xlabel('Frequency (Hz)')
45 ylabel('Magnitude (dB)')
46 title('Bode plot of plant')
47 grid on
48
49 subplot(3,1,2)
50 semilogx(Fhz, Pdeg,'LineWidth',2);
51 hold on
52 xlim([0.1 250])
53 xlabel('Frequency (Hz)')
54 ylabel('Phase (deg.)')
55 grid on
56
57 % Coherence plot
58 [C,w]=mscohere(Input,Output,[],[],[],1/Ts);
59 subplot(3,1,3)
60 semilogx(w, C,'LineWidth',2);
61 xlim([0.1 250])
62 title('Coherence plot')
63 ylabel('Coherence')
64 xlabel('Frequency (Hz)')
65 grid on
66
67 %% Estimate transfer function
68 data= idfrd(Tio(1:5243),Fhz(1:5243)*2*pi,Ts);
69 G=tfest(data,2,0)
70 [num,den]=tfdata(G)
71 G=tf(num,den)
72 [m,p,w]=bode(G,Fhz);
73
74 % Compare identified transfer function with measurement
75 figure
76 subplot(2,1,1)
77 semilogx(Fhz,mag2db(abs(M)),'LineWidth',2); hold on
78 semilogx(w/(2*pi),squeeze(20*log10(m)),'r','LineWidth',2);
79 xlim([0.1 250])
80 xlabel('Frequency (Hz)')
81 ylabel('Magnitude (dB)')
82 title('Bode plot of plant')
83 legend('Measured', 'Estimated')
84 grid on
85
86 subplot(2,1,2)
```

```
87 semilogx(Fhz, Pdeg,'LineWidth',2); hold on
88 semilogx(w/(2*pi),squeeze(p),'r', 'LineWidth',2);
89 xlim([0.1 250])
90 xlabel('Frequency (Hz)')
91 ylabel('Phase (deg.)')
92 grid on
```

Appendix C

Simulink model of fixed instant (adaptive) PID reset controller

In this appendix the m-code and Simulink implementation of the fixed instant adaptive PID reset controller can be found. These files are used to obtain the simulation results for the fixed instant reset controller that are shown in Chapter 5.

C.1 M-file with model parameters

Below, the m-file with the model parameters is given. This file determines the discrete transfer functions of the plant and the controller that are used in the Simulink model.

```
%% Simulation parameters for fixed instant (adaptive) PID reset control

% Plant
s=tf('s');
G=7.606e5/(s^2+46.58*s+7254);

% PI (reset) controller parameters
BW=60;              % Desired bandwidth                    [Hz]
kp=0.06;            % Proportional gain
fi=BW/10;           % Break-point of integrator            [Hz]
fd=BW/3;            % Beginning point of differentiator    [Hz]
ft=BW*3;            % Taming frequency of differentiator   [Hz]
fl=600;             % Cut-off frequency of low-pass filter [Hz]
ti=1/(2*pi*fi);     % Time constant of integrator          [s]

Cr=kp/ti;   % Parameter of PI reset controller
Dr=kp;      % Parameter of PI reset controller

K=1/dcgain(G);  % Initial value for gain K
```

```
epsilon=1e-1;   % Threshold to decide if two consecutive jumps are
    considered as a limit cycle
dtk=4e-3;       % Reset interval

% Lead/Lag + low-pass filter
DLP=(s/(2*pi*fd)+1)/(s/(2*pi*ft)+1)*(1/((s/(2*pi*fl))+1));

% Sample time and discrete transfer functions
Ts=0.0004;                                  % Sample time [s]
[numG,denG]=tfdata(c2d(G,Ts),'v');          % Discretized plant
[numDLP,denDLP]=tfdata(c2d(DLP,Ts),'v');    % Discretized Lead/Lag + low-
    pass

% All other parameters like reference signal and disturbance(s) have to
    be
% changed in the Simulink model itself.
```

C.2 Simulink implementation

In Figure C.1, the overall control loop is shown. In Figure C.2, the implementation of the fixed instant adaptive PID controller itself is shown.

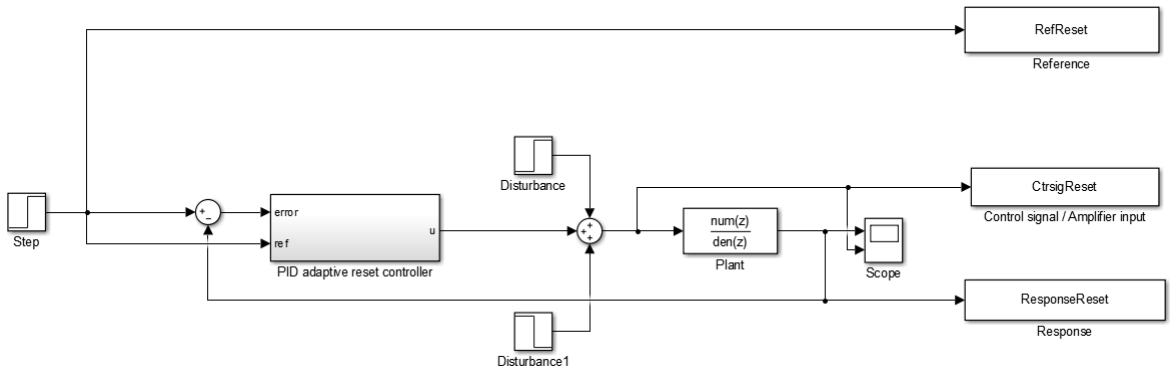

Figure C.1: Simulink implementation of control loop with fixed instant adaptive PID reset controller.

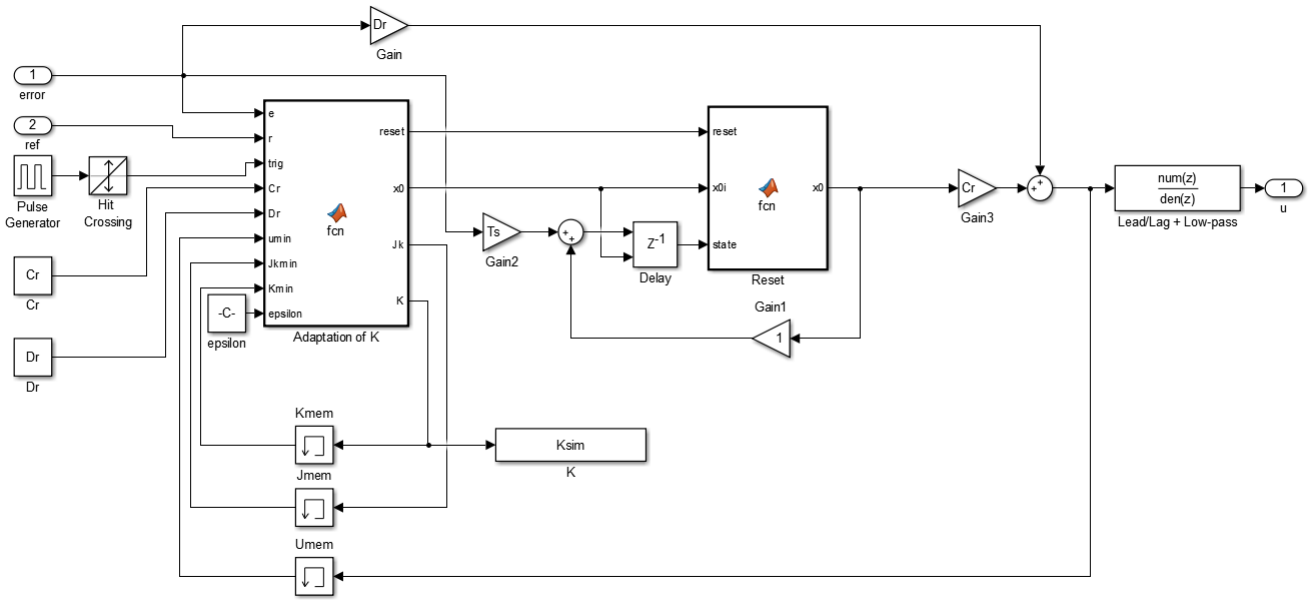

Figure C.2: Simulink implementation of fixed instant adaptive PID reset controller.

C.3 Function file to adapt K and to reset the integrator

Below, the two function files that are used by the Simulink model are given. The first function shows how K is adapted and the after reset value is determined. The second one shows how the integrator is reset.

C.3.1 Function to adapt K and to determine after reset value

```
function [reset,x0,Jk,K] = fcn(e,r,trig,Cr,Dr,umin,Jkmin,Kmin,epsilon)
% This function determines if two jumps are considered as a limit cycle,
% adapts K and determines the after reset value.

% Inputs
% e = error signal
% r = reference
% trig = reset controller if true (or 1)
% Cr = kp/ti (PI controller parameters)
% Dr = kp    (PI controller gain)
% umin = output of PI controller before it was reset
% Jkmin = value of jump in control signal at previous reset instant
% Kmin = current value of gain K, before adaptation
% Epsilon = threshold to decide if two consecutive jumps are considered
    as limit cycle

% Outputs
% reset = is a reset applied (trig is true)
% x0 = value to which the integrator resets.
% Jk = value of the current jump in the control signal, before K is
```

```
% adapted.
% K = gain

if trig==1
   Jk=umin-Kmin*r;
    if abs(Jk-Jkmin)<=epsilon
       if r~=0
        K=umin/r; % To simulate without adaptation use K=Kmin
       else
        K=Kmin;
       end

    else
        K=Kmin;
    end

else
    Jk=Jkmin;
    K=Kmin;
end

reset=double(trig);
x0=(K/Cr*r-Dr/Cr*e);
```

C.3.2 Function to reset integrator

```
function x0 = fcn(reset,x0i,state)
% This function resets the state of the integrator if the reset law is
% satisfied

% Inputs
% reset = if 1, integrator resets
% x0i = after reset value
% state = current value of the integrator

% Ouput
% x0 = output of the integrator

if reset==1
    x0=x0i;
else
    x0=state;
end
```

Appendix D

Simulink model of zero-crossing (adaptive) PID reset controller

In this appendix the m-code and Simulink implementation of the zero-crossing adaptive PID reset controller can be found. These files are used to obtain the simulation results for the zero-crossing reset controller that are shown in Chapter 5.

D.1 M-file with model parameters

Below, the m-file with the model parameters is given. This file determines the discrete transfer functions of the plant and the controller that are used in the Simulink model.

```
%% Simulation parameters for zero-crossing (adaptive) PID reset control

% Plant
s=tf('s');
G=7.606e5/(s^2+46.58*s+7254);

% PI (reset) controller parameters
BW=60;              % Desired bandwidth                    [Hz]
kp=0.06;            % Proportional gain
fi=BW/10;           % Break-point of integrator            [Hz]
fd=BW/3;            % Break-point of differentiator        [Hz]
ft=BW*3;            % Cut-off frequency of differentiator  [Hz]
fl=600;             % Cut-off frequency of low-pass filter [Hz]
ti=1/(2*pi*fi);     % Time constant of integrator          [s]

Cr=kp/ti;   % Parameter of PI reset controller
Dr=kp;      % Parameter of PI reset controller

K=0;            % Initial value for gain K
```

```
epsilon=1e-1;   % Threshold to decide if two consecutive jumps are
    considered as a limit cycle
rho=0.0012;     % Minimum amount of time between resets [s]

% Lead/Lag + low-pass filter
DLP=(s/(2*pi*fd)+1)/(s/(2*pi*ft)+1)*(1/((s/(2*pi*fl))+1));

% Sample time and discrete transfer functions
Ts=0.0004;                                  % Sample time [s]
[numG,denG]=tfdata(c2d(G,Ts),'v');          % Discretized plant
[numDLP,denDLP]=tfdata(c2d(DLP,Ts),'v');    % Discretized Lead/Lag + low-
    pass

% All other parameters like reference signal and disturbance(s) have to
    be changed in the Simulink model itself.
```

D.2 Simulink implementation

In Figure D.1, the overall control loop is shown. In Figure D.2, the implementation of the zero-crossing adaptive PID controller itself is shown.

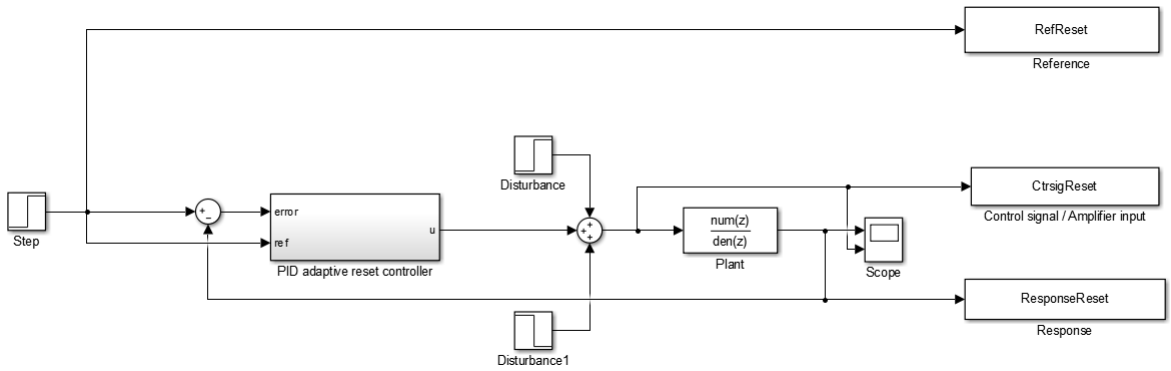

Figure D.1: Simulink implementation of control loop with zero-crossing adaptive PID reset controller.

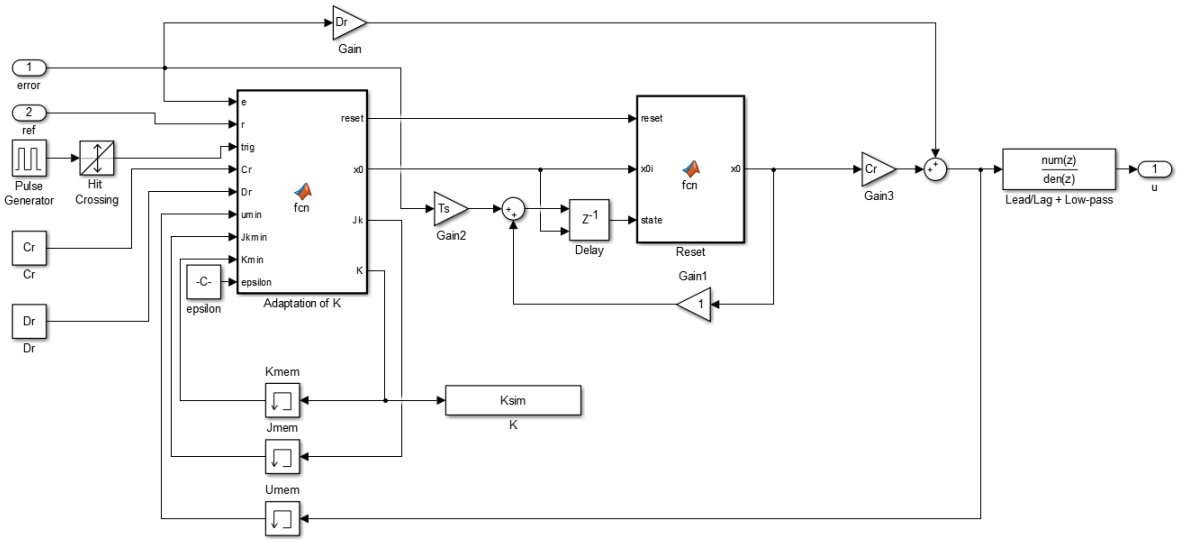

Figure D.2: Simulink implementation of zero-crossing adaptive PID reset controller.

D.3 Function file to adapt K and to reset the integrator

Below, the two function files that are used by the Simulink model are given. The first function shows how K is adapted for the zero-crossing reset controller and how the after reset value is determined. The second function shows how the integrator is reset.

D.3.1 Function to adapt K and to determine after reset value

```
1  function [reset,x0,Jk,K,to] = fcn(e,r,trig,Cr,Dr,umin,Jkmin,Kmin,epsilon,
       rho,t,ti)
2  % This function determines if two jumps are considered as a limit cycle,
3  % adapts K and determines the after reset value.
4
5  % Inputs
6  % e = error signal
7  % r = reference
8  % trig = reset controller if true (or 1)
9  % Cr = kp/ti (PI controller parameters)
10 % Dr = kp    (PI controller gain)
11 % umin = output of PI controller before it was reset
12 % Jkmin = value of jump in control signal at previous reset instant
13 % Kmin = current value of gain K before adaptation
14 % Epsilon = threshold to decide if two consecutive jumps are considered
       as limit cycle
15 % rho = minimum amount of time between resets
16 % t = current time
17 % ti = time of previous reset
18
19 % Outputs
20 % reset = is a reset applied (trig is true)
```

```
% x0 = value to which the integrator resets.
% Jk = value of the current jump in the control signal, before K is
% adapted.
% K = gain
% to = time instant of previous reset


% Check if it is allowed to reset
if (t-ti)>=rho && trig==1
    trig=1;
else
    trig=0;
end

% Adaptation of K
if trig==1
   to=t;
   Jk=umin-Kmin*r;
    if abs(Jk-Jkmin)<=epsilon
       if r~=0
        K=umin/r; % To simulate without adaptation use K=Kmin
       else
        K=Kmin;
       end

    else
        K=Kmin;
    end

else
    Jk=Jkmin;
    K=Kmin;
    to=ti;
end

reset=double(trig);
x0=(K/Cr*r)-(Dr/Cr*e);
```

D.3.2 Function to reset integrator

```
function x0 = fcn(reset,x0i,state)
% This function resets the state of the integrator if the reset law is
% satisfied

% Inputs
% reset = if 1, integrator resets
% x0i = after reset value
% state = current value of the integrator

% Ouputs
% x0 = output of the integrator

if reset==1
    x0=x0i;
else
    x0=state;
end
```

Bibliography

[1] K. J. Åström and T. Hägglund, "The future of PID control," *Control engineering practice*, vol. 9, no. 11, pp. 1163–1175, 2001.

[2] A. Baños and A. Barreiro, *Reset control systems*. Springer Science & Business Media, 2011.

[3] Y. Guo, Y. Wang, L. Xie, H. Li, and W. Gui, "Optimal reset law design of reset control systems with application to HDD systems," in *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pp. 5287–5292, IEEE, 2009.

[4] J. Zheng, Y. Guo, M. Fu, Y. Wang, and L. Xie, "Improved reset control design for a PZT positioning stage," in *Control Applications, 2007. CCA 2007. IEEE International Conference on*, pp. 1272–1277, IEEE, 2007.

[5] O. Beker, C. Hollot, Y. Chait, and H. Han, "Fundamental properties of reset control systems," *Automatica*, vol. 40, no. 6, pp. 905–915, 2004.

[6] G. Zhao, D. Nesic, Y. Tan, and J. Wang, "Open problems in reset control," in *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*, pp. 3326–3331, IEEE, 2013.

[7] J. Clegg, "A nonlinear integrator for servomechanisms," *Transactions of the American Institute of Electrical Engineers, Part II: Applications and Industry*, vol. 77, no. 1, pp. 41–42, 1958.

[8] I. Horowitz and P. Rosenbaum, "Non-linear design for cost of feedback reduction in systems with large parameter uncertainty," *International Journal of Control*, vol. 21, no. 6, pp. 977–1001, 1975.

[9] S. H. HosseinNia, I. Tejado, B. M. Vinagre, and Y. Chen, "Iterative Learning and Fractional Reset Control," in *ASME 2015 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pp. V009T07A041–V009T07A041, American Society of Mechanical Engineers, 2015.

[10] G. Zhao and J. Wang, "Existence and design of non-overshoot reset controllers for minimum-phase linear single-input single-output systems," *IET Control Theory & Applications*, vol. 9, no. 17, pp. 2514–2521, 2015.

[11] F. S. Panni, H. Waschl, D. Alberer, and L. Zaccarian, "Position regulation of an EGR valve using reset control with adaptive feedforward," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 6, pp. 2424–2431, 2014.

[12] M. Cordioli, M. Mueller, F. Panizzolo, F. Biral, and L. Zaccarian, "An adaptive reset control scheme for valve current tracking in a power-split transmission system," in *Control Conference (ECC), 2015 European*, pp. 1884–1889, IEEE, 2015.

[13] A. Banos and A. Vidal, "Design of PI+ CI Reset Compensators for second order plants," in *Industrial Electronics, 2007. ISIE 2007. IEEE International Symposium on*, pp. 118–123, IEEE, 2007.

[14] A. Baños and A. Vidal, "Design of reset control systems: the PI+ CI compensator," *Journal of Dynamic Systems, Measurement, and Control*, vol. 134, no. 5, p. 051003, 2012.

[15] A. Vidal and A. Banos, "QFT-based design of PI+ CI reset compensators: application in process control," in *Control and Automation, 2008 16th Mediterranean Conference on*, pp. 806–811, IEEE, 2008.

[16] A. Baños, S. Dormido, and A. Barreiro, "Limit cycles analysis of reset control systems with reset band," *Nonlinear Analysis: Hybrid Systems*, vol. 5, no. 2, pp. 163–173, 2011.

[17] M. Iwai and T. Ushio, "Prediction of limit cycles in nonlinear systems with reset controllers using describing function," in *Control, Automation and Systems (ICCAS), 2015 15th International Conference on*, pp. 913–918, IEEE, 2015.

[18] A. Barreiro, A. Baños, and S. Dormido, "Reset control systems with reset band: Well-posedness and limit cycles analysis," in *Control & Automation (MED), 2011 19th Mediterranean Conference on*, pp. 1343–1348, IEEE, 2011.

[19] J. Zheng, Y. Guo, M. Fu, Y. Wang, and L. Xie, "Development of an extended reset controller and its experimental demonstration," *IET Control Theory & Applications*, vol. 2, no. 10, pp. 866–874, 2008.

[20] S. H. HosseinNia, I. Tejado, D. Torres, B. M. Vinagre, and V. Feliu, "A general form for reset control including fractional order dynamics," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 2028–2033, 2014.

[21] I. R. Scola, M. M. Quadros, and V. J. Leite, "Robust hybrid pi controller with a simple adaptation in the integrator reset state," *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 1457–1462, 2017.

[22] L. Zaccarian, D. Nesic, and A. R. Teel, "Set-point stabilization of siso linear systems using first order reset elements," in *American Control Conference, 2007. ACC'07*, pp. 5808–5809, IEEE, 2007.

[23] G. Zhao, "Model-based reset control for overcoming performance limitations of linear feedback," in *Decision and Control (CDC), 2015 IEEE 54th Annual Conference on*, pp. 2241–2246, IEEE, 2015.

[24] M. A. Henson and D. E. Seborg, *Nonlinear process control*. Prentice Hall PTR Upper Saddle River, New Jersey, 1997.

[25] D. Nesic, A. R. Teel, and L. Zaccarian, "Stability and performance of SISO control systems with first-order reset elements," *IEEE Transactions on Automatic Control*, vol. 56, no. 11, pp. 2567–2582, 2011.

[26] Y. Guo, Y. Wang, L. Xie, and J. Zheng, "Stability analysis and design of reset systems: Theory and an application," *Automatica*, vol. 45, no. 2, pp. 492–497, 2009.

[27] R. M. Schmidt, G. Schitter, and A. Rankers, *The Design of High Performance Mechatronics-: High-Tech Functionality by Multidisciplinary System Integration*. IOS Press, 2014.