Fachhochschule Nordwestschweiz
Hochschule für Technik

REPORT
SYSTEM CONTROL LABORATORY

Noah Huesser <noah.huesser@students.fhnw.ch>
Yohannes Measho <yohannes.measho@students.fhnw.ch>

May 25, 2016

Contents

1 Operating Basics

1.1 Introduction

To ensure a good understanding of controllers and controlling theory a laboratory experiment was performed. A motor was used whose speed had to be controlled. The step function was measured and analyzed at first. Knowing the step function it was very easy to implement a suitable PID controller. A second approach of analyzing the systems oscillation characteristics was performed as well.

1.2 Methods to dermine the controller parameters

There is many different approaches to determine the characteristics of the system and design a PID controller accordingly. For this experiment, the two described in the next two Sections were used.

1.2.1 Chien, Hrones, Reswick Method

To determine the characteristics of the system, a step is applied to the input. Then the output is observed.

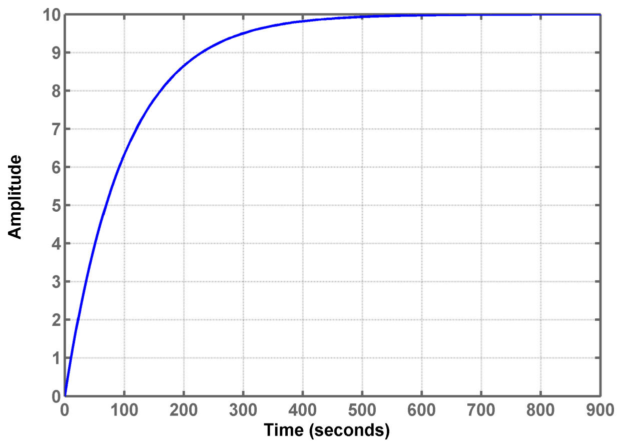


Figure 1: Step response of a PT_1 element

Using the turn tangent principle depicted in Figure 2, the parameters T_u, T_g and K_s can be derived from the step response.

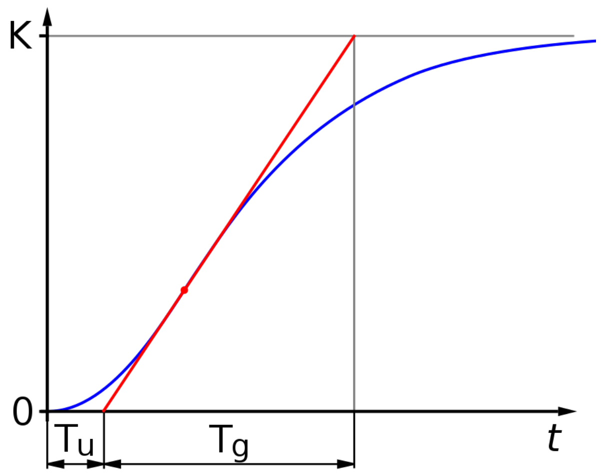


Figure 2: Turn Tangent Principle

Once those paramters are known, the PID parameters can be calculated as formulated in Table 1.

Controller Type	K_p	T_i	T_d
P	$0.3 * \frac{T_g}{T_u * K_s}$	-	-
PI	$0.35 * \frac{T_g}{T_u * K_s}$	$1.2 * T_g$	-
PID	$0.6 * \frac{T_g}{T_u * K_s}$	T_g	$0.5 * T_u$

Table 1: Chien, Hrones, Reswick Method

1.2.2 Ziegler-Nichols Method

To use this also called oscillation method the system characteristics are determined by bringing the system to the brink of oscillationby increasing K_p whilst the I and D parts remain zero. The parameters K_u and T_u are then the gain K_p and the period of the oscillating output.

The PID parameters then can be calculated according to Table 2.

Controller Type	K_p	T_i	T_d
P	$0.5 * K_{P.crit}$	-	-
PI	$0.45 * K_{P.crit}$	$0.85 * \tau_{crit}$	-
PID	$0.6 * K_{P.crit}$	$0.5 * \tau_{crit}$	$0.12 * \tau_{crit}$

Table 2: Ziegler-Nichols Method

2 Carrying out the experiment

2.1 Experimental setup

The experiment consisted of a motor that was to control, a tachometer and a controller. To make things more exciting, the system also featured disturbances, emulated by a switch coupling in some resistors.

The setup can be seen in Figure 3. The block diagram characterizing the system is depicted in Figure 4.

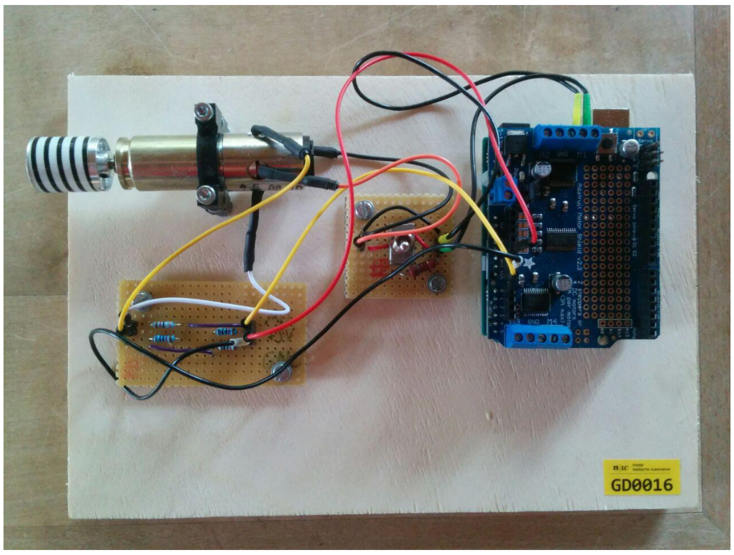
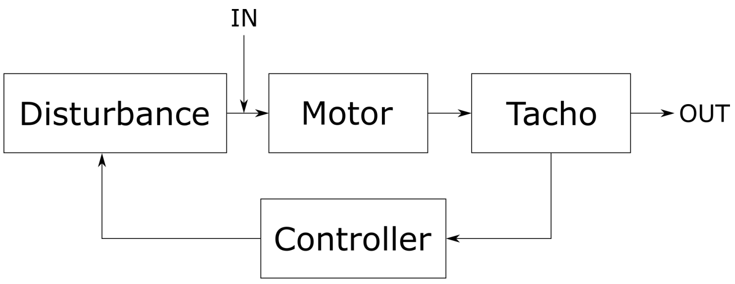


Figure 3: Experimental setup



$$\begin{tikzcd}
 & \text{IN} \arrow[d] & & \\
\boxed{\text{Disturbance}} \arrow[r] & \boxed{\text{Motor}} \arrow[r] & \boxed{\text{Tacho}} \arrow[r] \arrow[d] & \text{OUT} \\
 & \boxed{\text{Controller}} \arrow[ul] & \arrow[l] &
\end{tikzcd}$$

Figure 4: Block diagram of the system

2.2 Identifying the system

At first the characteristic curve of the system was recorded to learn more about the limitations of the system. This was done by measuring the outputs for a broad range of inputs.

The highest possible input was 13 Volts whilst the lowest was -13 Volts. This resulted in approximately 102 turns per minute in either clockwise or counter clockwise direction. The characteristic curve is plotted in Figure 5

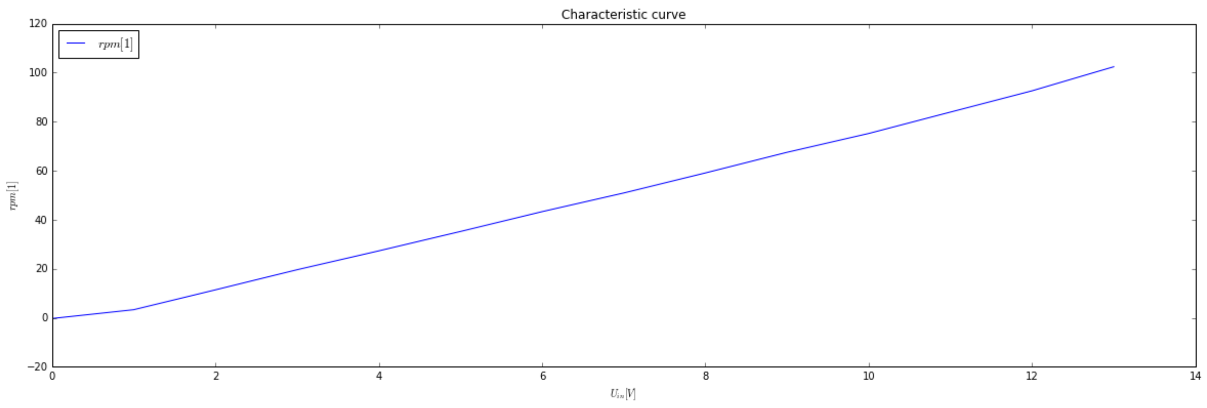


Figure 5: Characteristic curve

Knowing the limits of the system, an operating point of approcimately 7 Volts was chosen which results in around 50 turns. This operating point was chosen since a system is hard to control at its boundaries. This value is not to close to the upper limit and ensures that the controller has it's freedom.

Sadly a mistake was made with the settings and all the experiments were done at an operating point of 90 turns per minute. So this should be kept in mind in the further reading.

2.3 Carrying out the step experiment

To properly implement a PID-Controller using the Chien Hrones Reswick method, the curve seen in Figure 6 was analyzed and the parameters in Table 3 were determined.

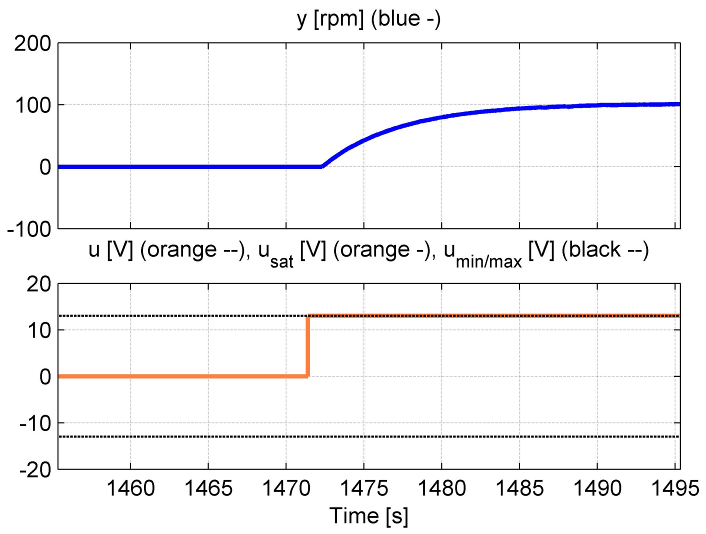


Figure 6: Step response of the system

Parameter	Value
K_s	10
T_u	1
T_g	2

Table 3: Chien, Hrones, Reswick Parameter

This was done using the approach of analyzing the step function of the system. This process is explained in Section 1.2.1.

2.4 Tinkering with the oscillation method

Using the oscillation approach, the critical gain and period in Table 4 were determined.

Parameter	Value
$K_{P,crit}$	1.4
τ_{crit}	$8/3$

Table 4: Ziegler-Nichols Parameter

Those parameters can be found by studying the curve seen in Figure 7 using the method explained in Section 1.2.2.

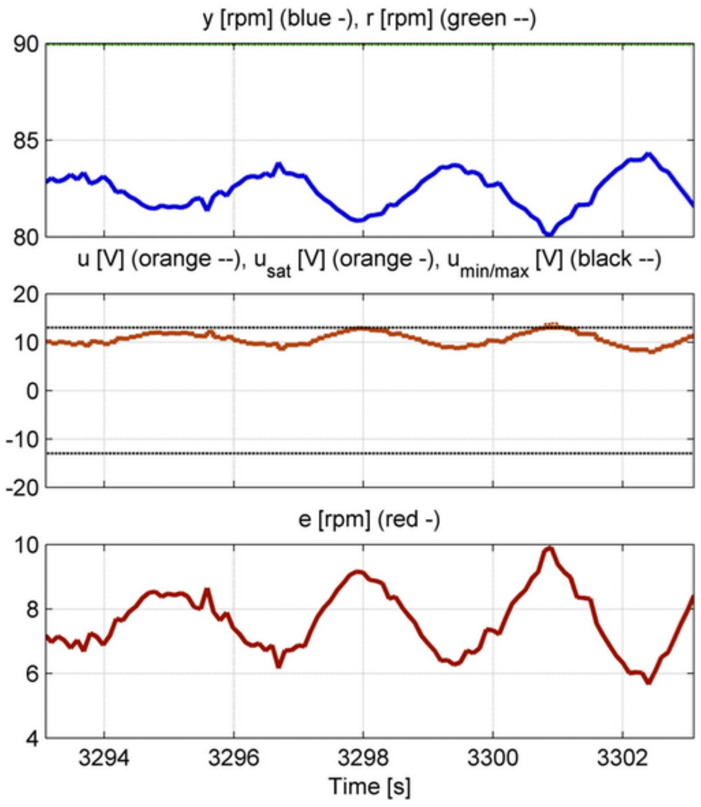


Figure 7: Critical gain $K_{P,crit}$ and τ_{crit}

3 Evaluation

For the three base types of controllers, P, PI and PID, different characteristic parameters are used in the following experiments. The experiments were mistakenly made using a set-point of 90 rpm instead of 50 rpm. Nevertheless, good assumptions about the performance of the controller can be made.

3.1 P-Controller

In the following three sections, the Ziegler-Nichols method was used. For that, $K_{p,rule} = 0.7$ was used. For the actual controller, K_P was a modification of $K_{p,rule}$.

First, a controller with

$$K_p = K_{p,rule} \cdot 0.2$$

was used. One can clearly see in Figure 8 that the deviation from the setpoint is huge.

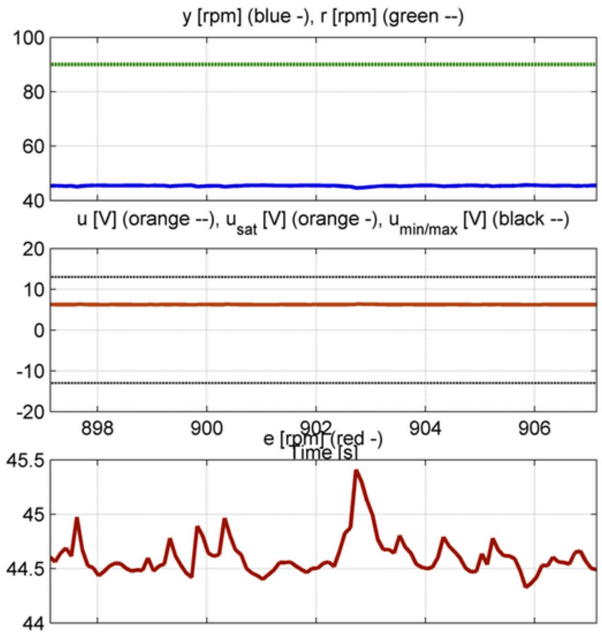


Figure 8: P-Controller with $K_p = K_{p,rule} \cdot 0.2$

To narrow the gap from the actual output to the set-point, a new value for K_P was used:

$$K_p = K_{p,rule}$$

The error is now much smaller as obtained in Figure 9, but still a good 15 percent off to the actual setting point.

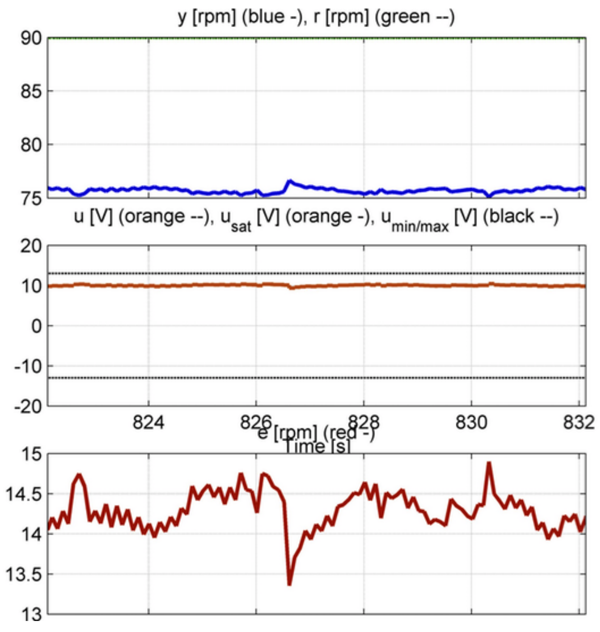


Figure 9: P-Controller with $K_p = K_{p,rule}$

Since the error decreased with increasing K_P, an even bigger value was tested:

$$K_p = K_{p,rule} \cdot 4$$

The experiments with the P-controller show that the error decreases with an increasing K_p. Considering Figure 10 the error did indeed decrease even more. Sadly now the actual value started to oscillate. To counteract this, in a next step, a PI-Controller will be tested.

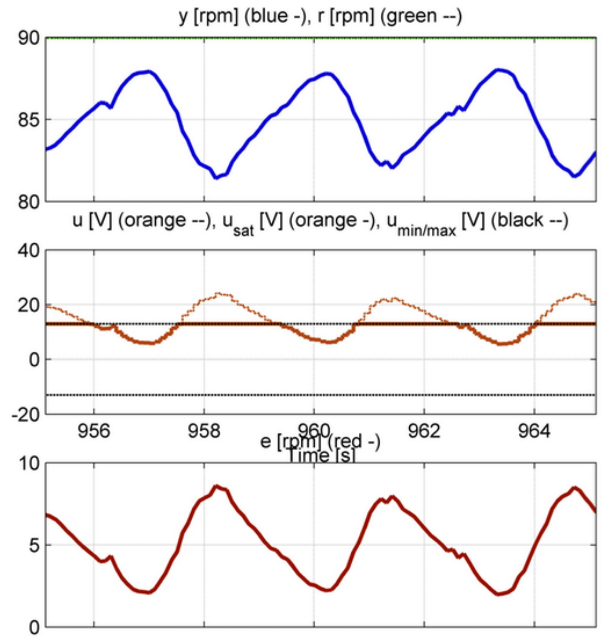


Figure 10: P-Controller with $K_p = K_{p,rule} \cdot 4$

3.2 PI-Controller

In this experiment, the increase of T_i should reduce the overshoot but in the same time increase the time used to reach the set-point whereas the decrease of T_i should reincrease oscillation with fast rise time that forces the controller to reach the setpoint in a much shorter time. Once again the Ziegler-Nichols method was used.

At first an experiment with

$$K_p = K_{p,rule} \text{ and } T_i = T_{i,rule} \cdot 0.2$$

was carried out. As seen in Figure 11 the RMS deviation from the setting point nearly vanished. Sadly the system still oscillates a lot.

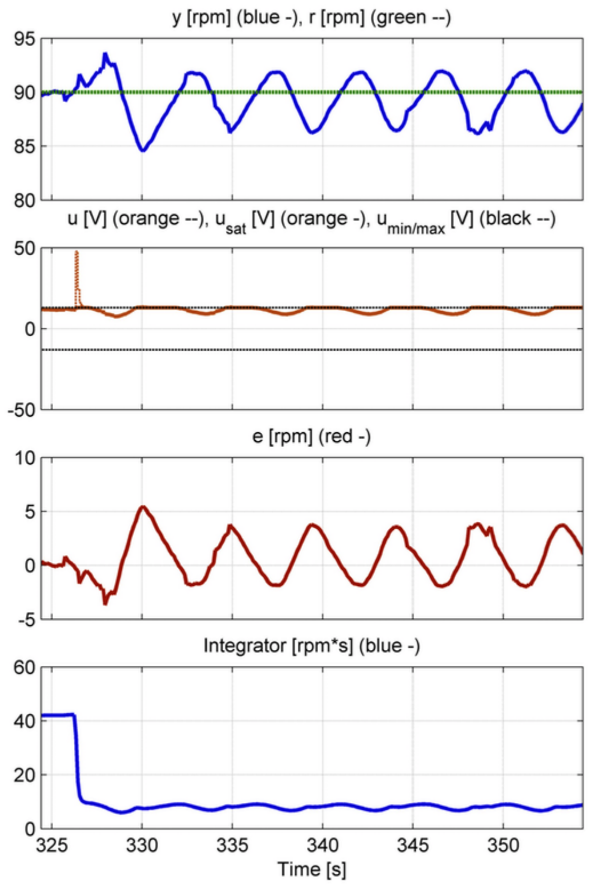


Figure 11: PI-Controller with $K_p = K_{p,rule}$ and $T_i = T_{i,rule} \cdot 0.2$

From Figure 12 the rise time for the above controller configuration can be extracted. It is quite some time which can surely be decreased.

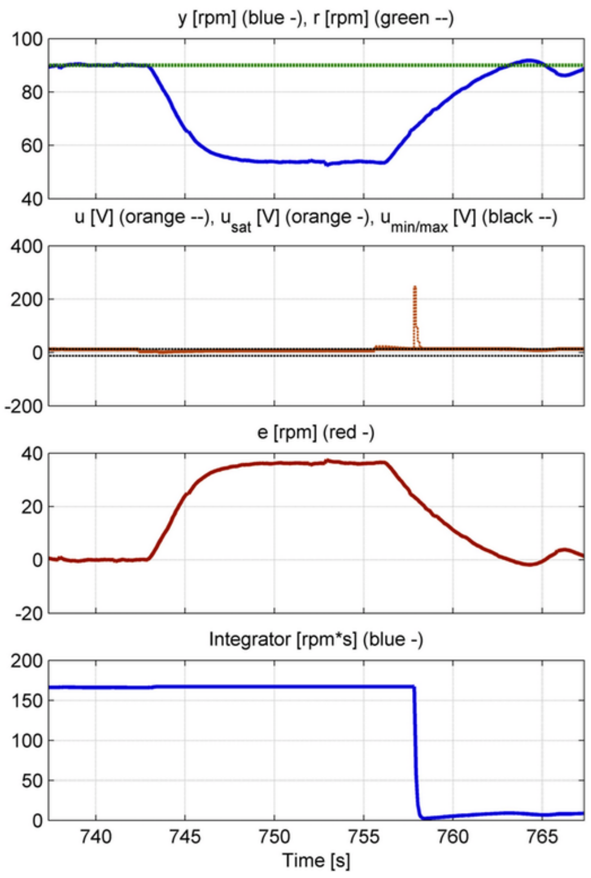


Figure 12: Rise time of a PI-Controller with $K_p = K_{p,rule}$ and $T_i = T_{i,rule} \cdot 0.2$

Now the paramters were set to

$$K_p = K_{p,rule} \text{ and } T_i = T_{i,rule}$$

In Figure 13 it can be observed that the error has now close to vanished. Sometimes the output still spikes but that can also be measurement errors.

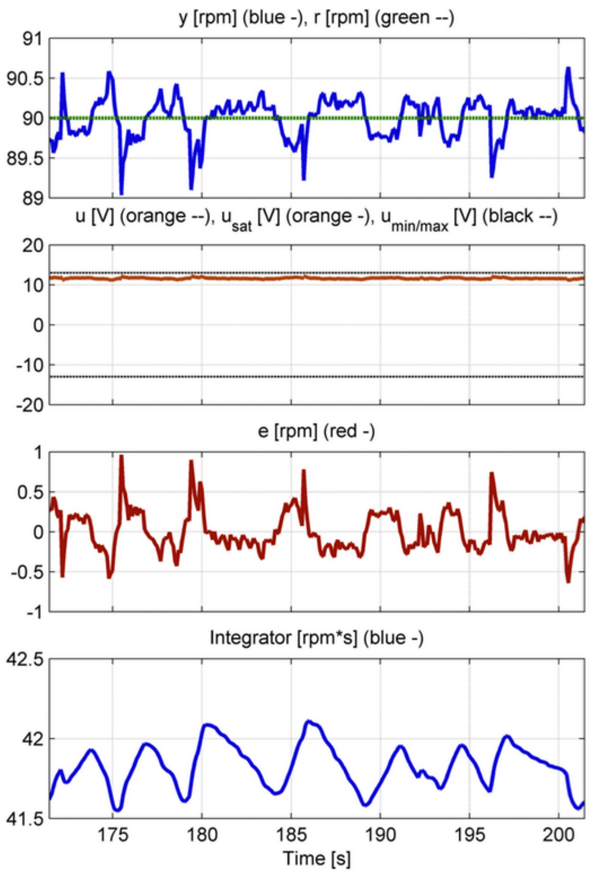


Figure 13: PI-Controller with $K_p = K_{p,rule}$ and $T_i = T_{i,rule}$

The rise time is still about the same as before, as seen in Figure 14.

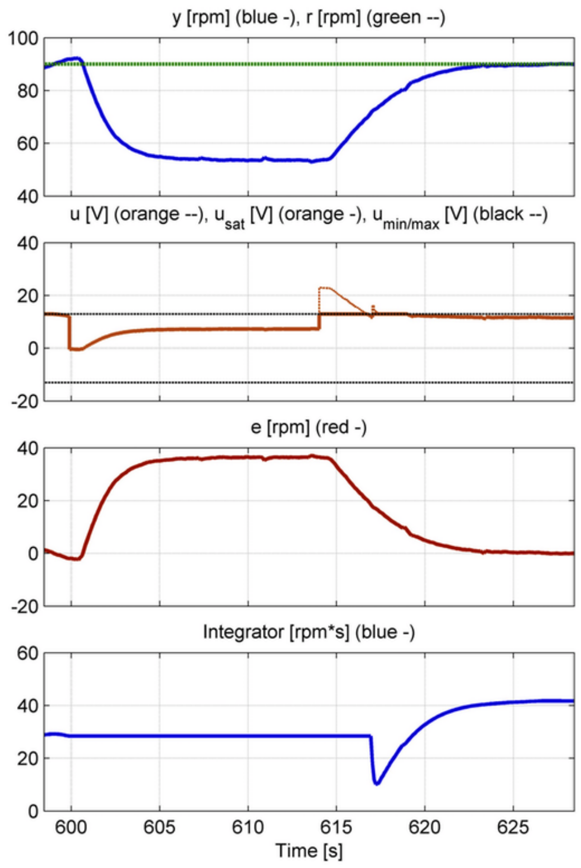


Figure 14: Rise time of a PI-Controller with $K_p = K_{p,rule}$ and $T_i = T_{i,rule}$

Last but not least the values

$$K_p = K_{p,rule} \text{ and } T_i = T_{i,rule} \cdot 4$$

were tested. The error did not worsen but the rise time quadrupled! (Consult Figures 15 and 16) This is not wanted and a lower T_i should be selected for sure.

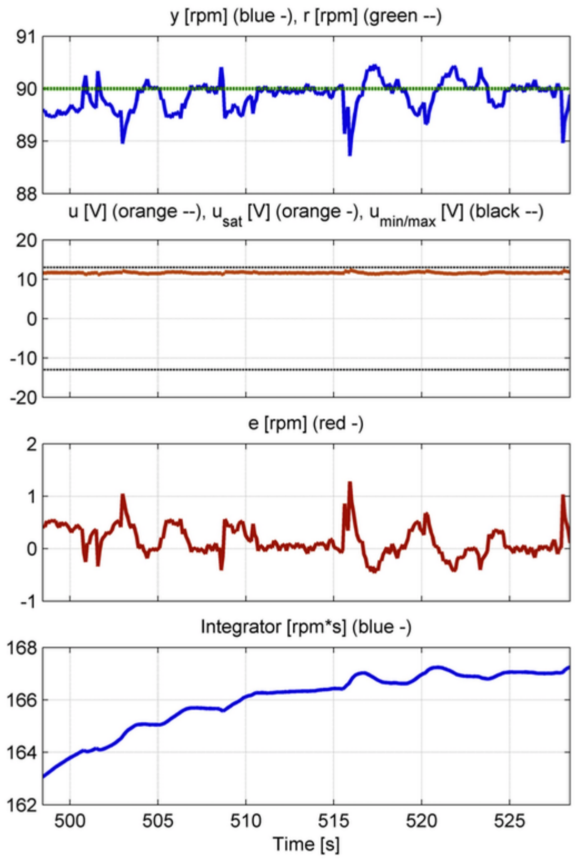


Figure 15: PI-Controller with $K_p = K_{p,rule}$ and $T_i = T_{i,rule} \cdot 4$

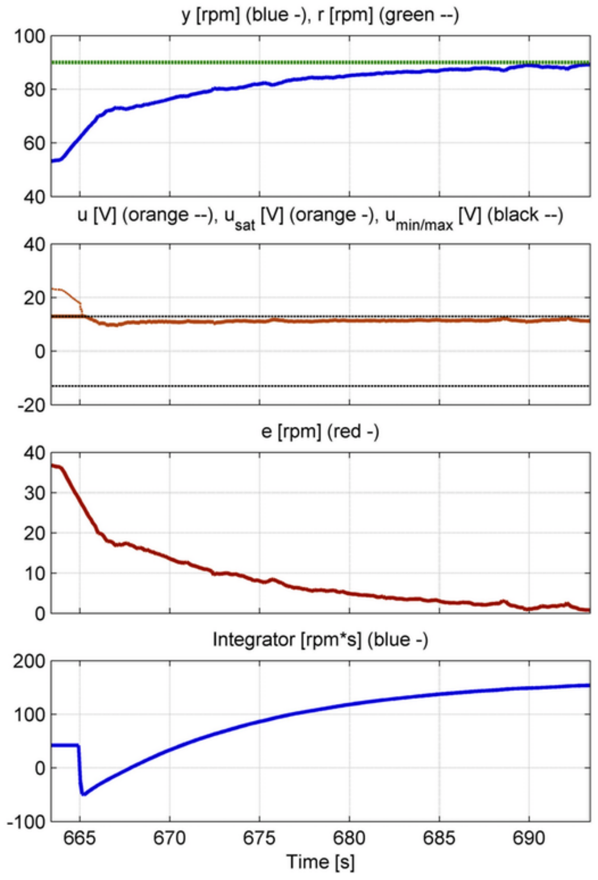


Figure 16: Rise time of a PI-Controller with $K_p = K_{p,rule}$ and $T_i = T_{i,rule} \cdot 4$

3.3 PID-Controller

To improve the behavior of the controller even more, a full PID controller was tested now. Both, the Ziegler-Nichols and Chien-Hrones-Reswick methods of tuning PID controllers have been used for the following part of the experiment.

To carry out the experiment using the Ziegler-Nichols method, the values in Table 5 were used. The resulting behavior can be observed in Figure 17

Parameter	Value
T_d	0.32
T_i	1.33
K_p	0.84

Table 5: Ziegler-Nichols Parameter

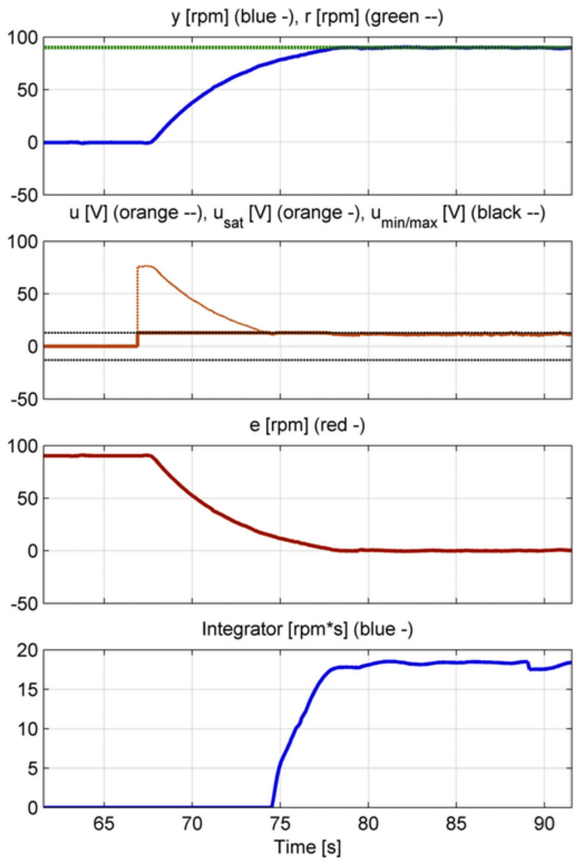


Figure 17: PID-Controller with the Ziegler-Nichols tuning rule

For the Chien, Hrones, Reswick experiment, the values in Table 6 were used.

Parameter	Value
T_d	0.5
T_i	2
K_p	0.154

Table 6: Chien, Hrones, Reswick Parameter

The systems behavior using these parameters can be studied in Figure 18

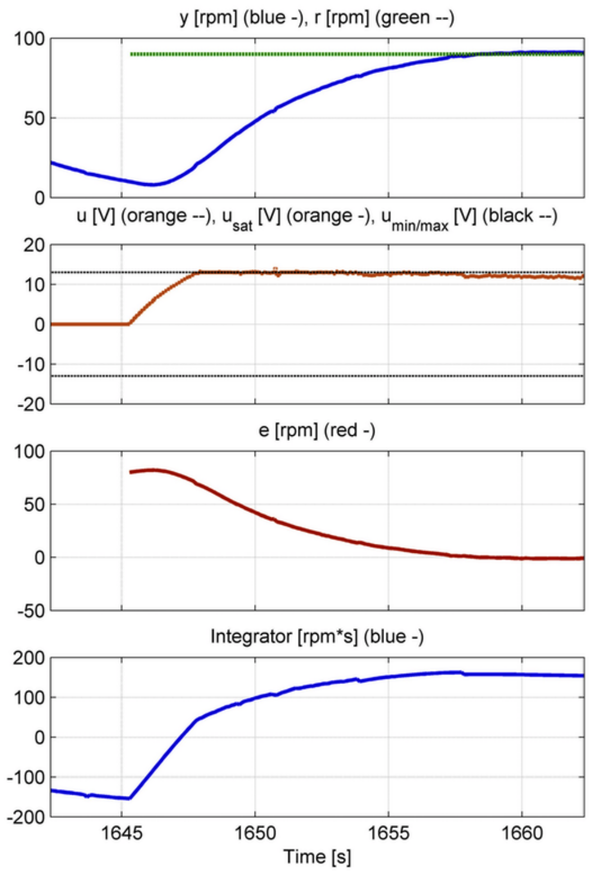


Figure 18: PID-Controller with the Chien-Hrones-Reswick tuning rule

Both methods of creating a controller yielded similar results. It was then decided to go with the Chien, Hrones, Reswick approach and tinker with it and try and optimize the resulting controller. Sadly the rise time could not really be reduced. For that to happen T_i possibly should have been reduced a lot more whilst the K_P and T_d should have increased a lot. The disturbance rejection capability is okay, but the system is still very prone to disturbances. This behavior can be seen in Figure 19

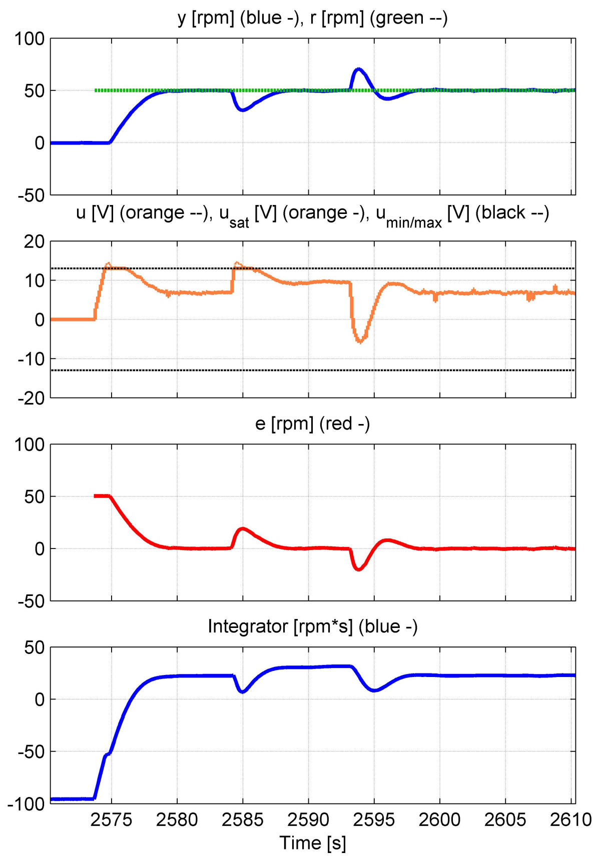


Figure 19: Error rejection of the tuned controller

4 Simulations

4.1 Characteristics of the system

To verify the results in practice, a SIMULINK model was created whose step response resembles the one of the experimental setup as close as possible.

In Figure 20 the blockdiagram of the system can be observed. It's step response is depicted in Figure 21

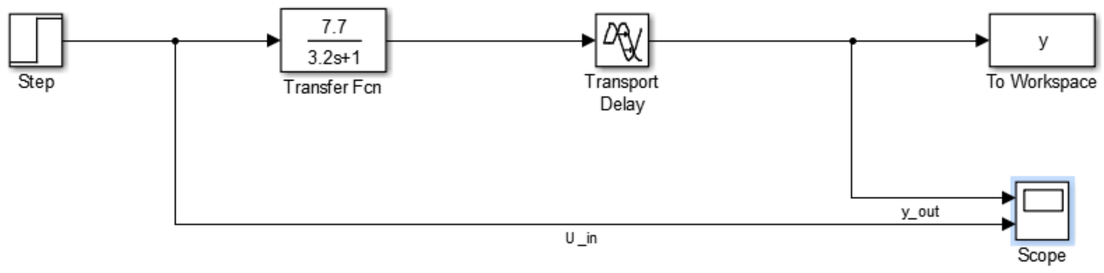


Figure 20: Block diagram of the simulation

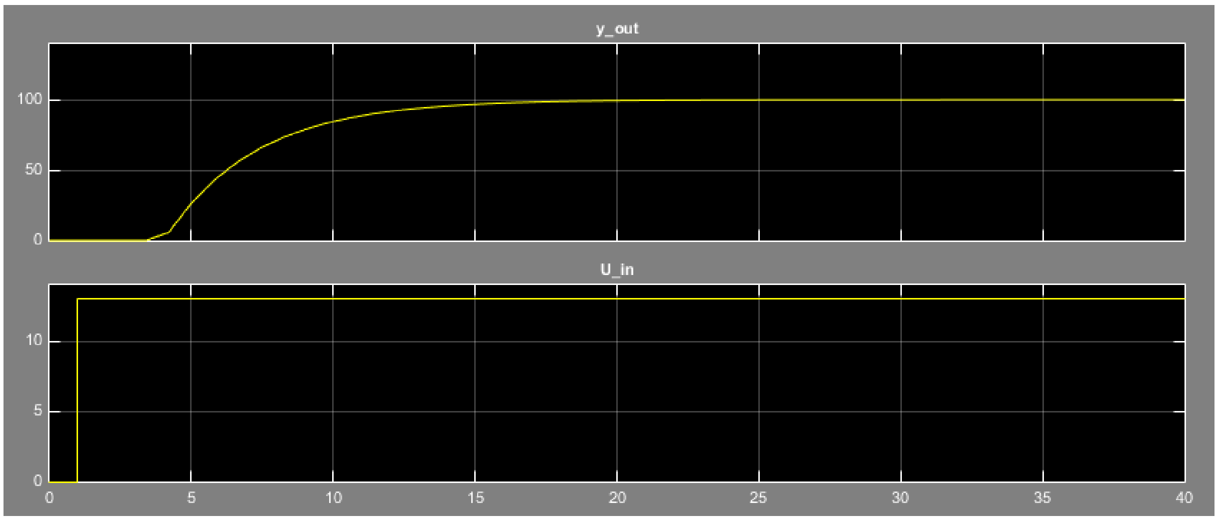


Figure 21: Step response the system

4.2 Controlling the system

After the system was close enough to the one form the experimental setup, a PID controller was added according to Figure 22. At first the output started to raise to infinity very quickly, using the PID parameters determined in the experimental setup. One problem was that the error fed into the PID-Controller was in units of rpm whilst the input to the system was expected in volts. Then a saturation limitter was added to have the maximum output of the PID-Controller be max [-13, 13]. The PID-Controller then behaved nicely as seen in Figure 23. Unfortunately it misses the setpoint by 10 percent. All tries to make this behaviour better were a failure. Tinkering with the tuning function of MATLAB produced much better results as seen in Figure 24. It seems like the integrator part of the PID values is way too high from the value gained in the experiments and the integrator of the MATLAB PIDÂăcontroller gets to a point of no return where it will always strive to gain even more which is why the system stays at it's upper limit.

The final MATLAB-tuned PID values can be obtained from Table 7.

Parameter	Value
K_p	0.0566
T_i	0.0171
T_d	0

Table 7: Final MATLAB-tuned PID values

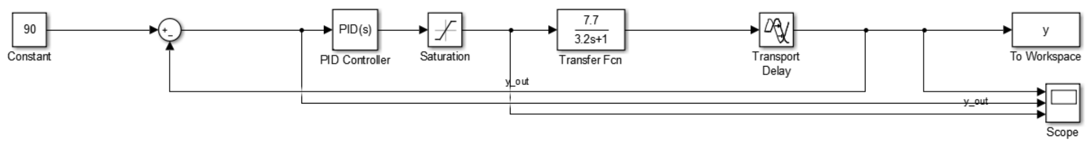


Figure 22: The simulation system with a PID-Controller added

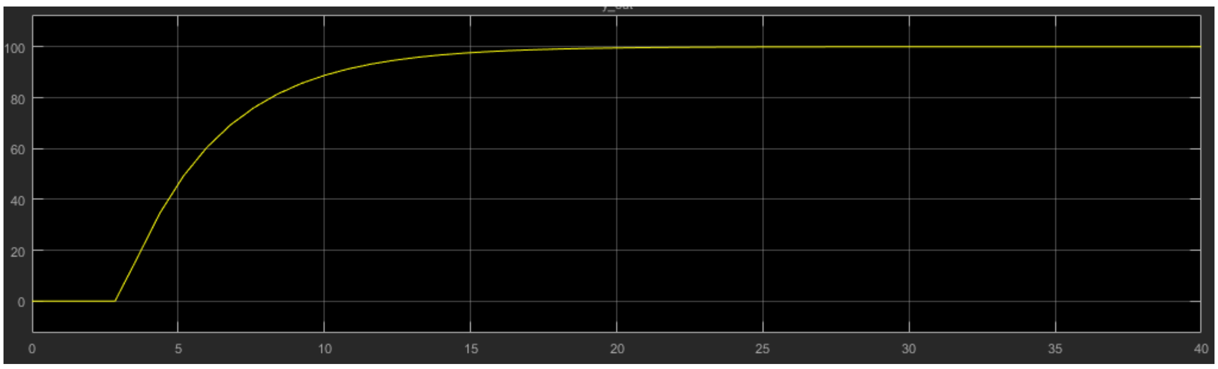


Figure 23: Behavior of the system controlled by the experimental controller

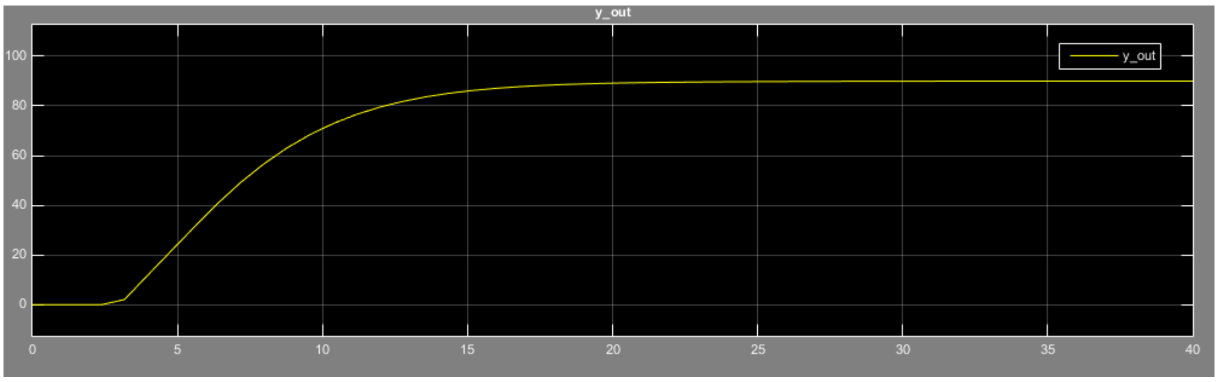


Figure 24: Behavior of the MATLAB-tuned, controlled system

5 Conclusion

As written in previous parts of the the report, the P-Controller does not work standalone. It gets close to the operating point or even very close but with strong oscillation.

To counteract this Effect a PI-Controller is needed. This type of controller should be enough to control the system. The oscillation is rejected and the controller reaches it's set-point. The only disadvantage is the long rise time.

To counteract this effect once again, a PID-Controller is used. With the PID-Controller a slightly better rise time can be achieved. It comes with increased noise tho. Whilst the T_d value makes the controller react faster, it also worsens the noise of the system.

To conclude it can be stated that a PI-Controller should be good enough for the purpose of controlling this system. Slightly better results can be achieved using a PID-Controller.