RTGL LAB REPORT 2

Alexander Murray

alexander.murray@students.fhnw.ch

alex.murray@gmx.ch

Emerson Lattmann

emerson.lattmann@students.fhnw.ch

emilat@msn.com

June 9, 2017

Contents

1 Introduction

Every engineer needs to have the knowledge of tuning and optimize a PID controller. He also needs to know the well proven rules of tuning a PID controller. Therefore the goal of this exercise is to apply the knowledge acquired to tuning a PID controller. Specially the tuning rules according to Ziegler-Nichols (oscillation method) and to Chien-Hrones-Reswick (step respibse method) are going to be used. Even if there are plenty of other tuning rules. The important two rules for this exercise will be the two mentioned before. This will be achieved by investigating an electrical motor and by optimizing the controllers parameters manually in closed loop. Furthermore the goal is to apply the knowledge to do a simulation in Matlab/Simulink.

2 Theory

The risk of instability in control circuits are caused by the delays. It is always necessary to ensure that a system is stable. There exist some simple tuning rules (proposed by Ziegler-Nichols and Chien, Hrones and Reswick) that allow us to calculate stable parameters for P, PI, and PID controllers.

2.1 System Characterisation

A systems complete dynamic behaviour around an operating point of interest can be determined by applying a step function – $\epsilon(t)$ – to its input and recording the system's output. The derivative of the resulting function is the system's *impulse response* and can be used to calculate optimal controller parameters.

Before measuring the step response we first have to select the step amplitude. If the step amplitude is too small, the plant behaves in an atypical way. This happens because of small disturbances which affect the step response. A good example for this is static friction. On the other hand if the step amplitude is too large, then non-linear effects will throw off the accuracy (and validity) of the result.

After a system's step response has been measured, it is necessary to characterise and determine its properties before it's possible to fit a transfer function to it. The method of characterisation used here is to find the point of inflection in the measured step response, through which a tangent is placed. The tangent's intersection points with the lower and upper horizontal bounds are used to determine the dead time T_u and the rise time T_g. Further, the amplitude K_s of the step response can be determined. This process is illustrated in figure 1.

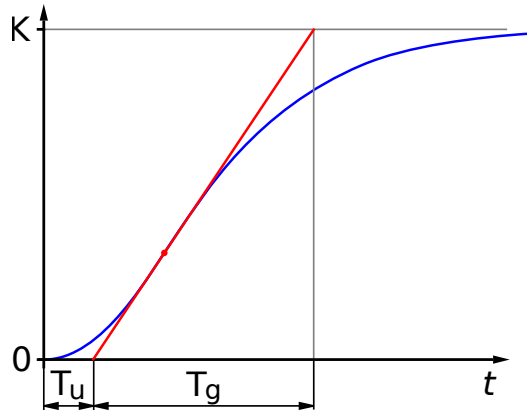
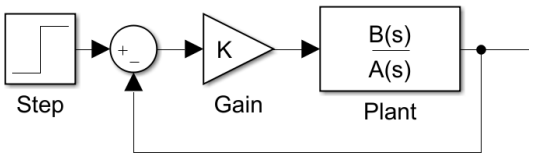


Figure 1: Example step response of a system and measurement of the angle of inflection for determining K_s, T_u and T_g. Image taken from Wikipedia[3].

It is important to note that this method of characterisation is valid for systems with an order of at least $n = 2$ and for systems that don't exhibit any overshoot.

2.2 Tuning Rules According to Ziegler-Nichols (Oscillation Method)

This method relies on a plant that can potentially become unstable. That is to say, the plant must be of higher order such that the phase crosses the $180°$ threshold at some point. If this is the case, then the plant can be placed in a closed loop with a single gain element (P-Controller) and the gain can be increased until the system's output begins to oscillate.



The gain at which this occurs is $K_{p,crit}$ and the period at which it oscillates is τ_{crit}.

The full procedure of the Ziegler-Nichols method is:

1. Use a pure P controller and set a small gain K_P.
2. Increasing the gain K_P until an undamped oscillation occurs.
3. Indentify the critical gain $K_{p,crit}$ and the critical period τ_{crit}
4. According to table 1 specifying the controllers parameter.

Type	K_P	T_i	T_d
P	$0.5 \cdot K_{P,crit}$	-	-
PI	$0.45 \cdot K_{P,crit}$	$0.85 \cdot \tau_{crit}$	-
PID	$0.6 \cdot K_{P,crit}$	$0.5 \cdot \tau_{crit}$	$0.12 \cdot \tau_{crit}$

Table 1: Table with controller parameters, according to the Ziegler-Nichols method (good disturbance rejection).

The advantage of the Ziegler-Nichols method is that it is simple to apply and because of that also easy to understand. On the other hand it is very risky because the control loop must be operated close to instability. Dependong on the plant being measured this can be very dangerous and very expensive.

A disadvantage is the need for the plant to be potentially unstable. Theoretically not all plants can be made to oscillate in a closed loop with a P-controller. The practical significance of this method is thus limited.

2.3 Tuning Rules According to Chien, Hrones and Reswick (Step Response Method)

This method is an improvement of the Ziegler-Nichols method and uses the system's step response characteristics T_u, T_g and K_s (discussed in section 2.1).

The procedure of the Chien-Hrones-Reswick method is:

1. Record the step response of the plant and measure the gain K_S, the delay time T_u and the compensation time T_g.
2. Determine the controller parameters according to the tuning rules (see tables 2 and 3).

Type	K_p	T_i	T_d
P	$0.3 \cdot \frac{T_g}{T_u \cdot K_s}$	-	-
PI	$0.6 \cdot \frac{T_g}{T_u \cdot K_s}$	$4 \cdot T_u$	-
PID	$0.95 \cdot \frac{T_g}{T_u \cdot K_s}$	$2.4 \cdot T_u$	$0.42 \cdot T_u$

Table 2: Table with controller parameters according to the Chien-Hrones-Reswick method (good disturbance rejection).

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

Table 3: Table with controller parameters according to the Chien-Hrones-Reswick method (aperiodic behaviour, good tracking).

Based on the ratio of $\lambda = \frac{T_u}{T_g}$, we can say how controllability the control plant is.

Ratio	Controllability	Difficulty
$\lambda < 0.1$	very good	small
$0.1 \leq \lambda < 0.2$	good	medium
$0.2 \leq \lambda < 0.4$	fair	large
$0.4 \leq \lambda < 0.8$	bad	very large
$\lambda \geq 0.2$	very bad	(special measurements required)

Table 4: Table showing approximate ratios and how "good" they are

3 Measurements

3.1 Measurement Setup

The setup consisted of 3 main components.

- Plant
- Controller
- PC with the user interface running under MATLAB

The plant itself consisted of the following elements.

- **Motor**: The motor used in the setup was a universally usable DC motor. The brushed DC motor is excited using permanent magnets.
- **Tachometer**: The speed sensor which is physically connected to the motor is a tachometer generator. It yields a voltage signal that is proportional to the motor speed.

The controller consisted of an Arduino UNO microcontroller along with a motor shield (Adafruit Motor Shield v2.3) and a voltage divider circuit. The motor shield is based on an H bridge and is capable of delivering the required current (as opposed to the Arduino UNO board itself). The voltage divider is required in order to transform the tachometer signal into a signal compatible with the analog input.

Microcontroller and PC are connected with a USB cable and communicate across the serial port.

Additionally, the laboratory setup includes a circuit that allows the simulation of an external disturbance (e.g. a sudden increase of the load driven by the motor) by changing the position of a switch.

The rpm could be changed by an input control voltage ranging from $-13V$ to $13V$.

3.2 Operating Point and Static Characteristics

To see the non-linear characteristics of the plant, we set various constant input voltages and waited for the motor to reach its settling speed. The settling speed is then recorded and plotted in function of the input voltage (see figure 2).

Of particular interest are the areas close to $0\,\mathrm{V}$ and the areas close to the maximum rotation speed. This is where the plant is not at all linear.

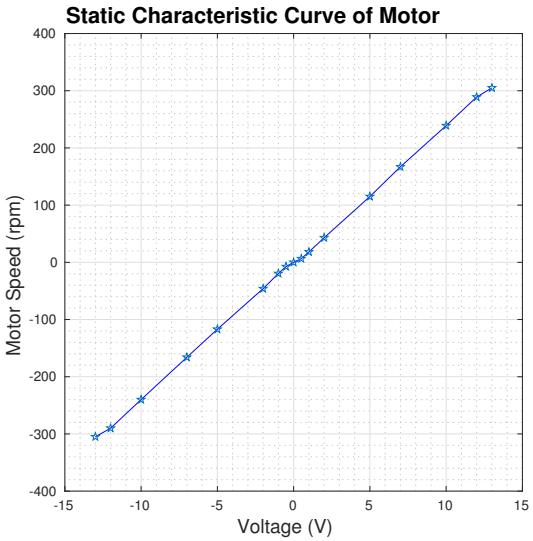


Figure 2: Static characteristic curve of the plant (motor): Input voltage vs settling rotation speed.

To see how these non-linearities affect the step response of the plant, we performed two step response experiments, one in the non-linear region of $0\,\mathrm{V}$-$2\,\mathrm{V}$ and another in the fairly linear region of $2\,\mathrm{V}$-$10\,\mathrm{V}$. The results of these experiments are recorded in figure 3 (left) and are mapped onto the static characteristic curve (right).

When comparing the input amplitudes of both steps ($2\,\mathrm{V}$ and $8\,\mathrm{V}$, respectively) to the step response amplitudes (43 rpm and 197 rpm, respectively), we expect a perfectly linear system to satisfy the equation $\frac{43}{2} \stackrel{?}{=} \frac{197}{8}$, but this is clearly not the case.

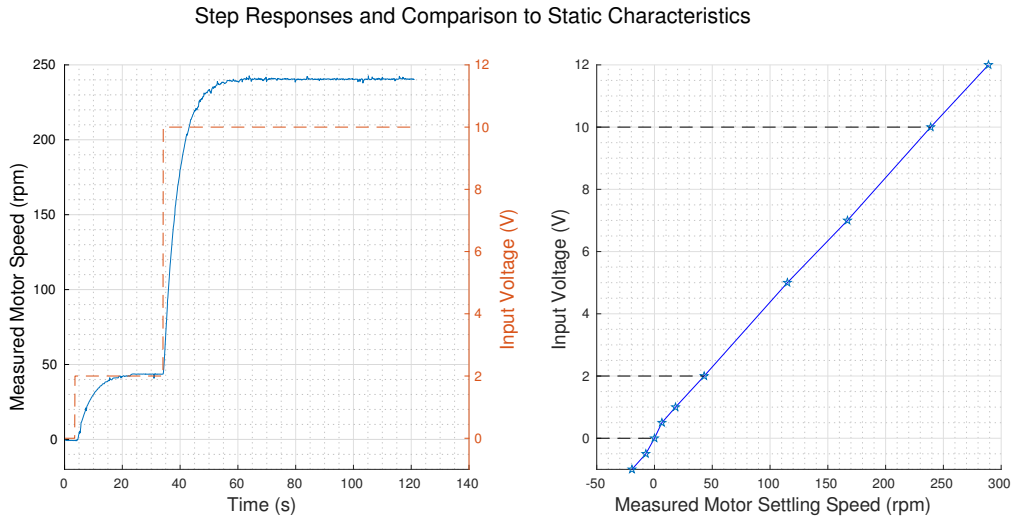


Figure 3: Two step response experiments, one in a non-linear region and the other in a fairly linear region. Notice how the step response amplitude differs between the two regions.

4 Simulations

4.1 Plant Characterisation

We decided to use the step response conducted in the fairly linear region of 2 V-10 V for further processing.

Using MATLAB, the measured step response is smoothed, the derivative is computed to find the point of inflection, and the tangent is calculated. The results of this are visualised in figure 4.

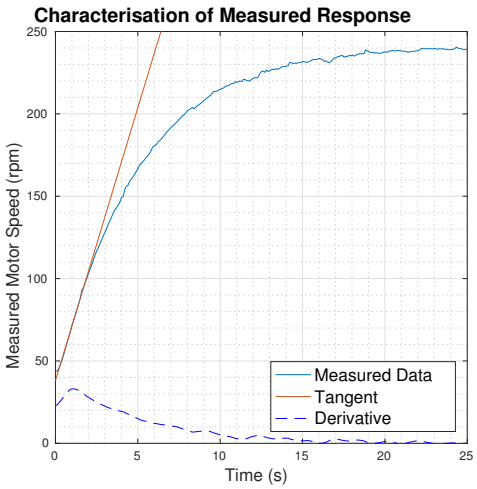


Figure 4: Calculated tangent of measured step response data.

Regarding the parameter K_s, a common pitfall is to say that K_s is the total change in the step response. This statement is in fact false, but happens to yield the correct value as long as the input step function has an amplitude of 1, which unfortunately is the only case studied in the majority of theory lectures. The parameter K_s is in fact the total change in output divided by the total change in input:

$$K_s = \frac{\Delta K_{out}}{\Delta K_{in}} \tag{4.1}$$

The calculated values of T_u, T_g and K_s turn out to be:

$$T_u = 0.1655$$
$$T_g = 5.9602$$
$$K_s = 197.42$$

4.2 System Identification using Dead-Time and PT1 elements

It is known that the motor has a dead-time built in, and by looking at the step response, one can say that it looks like a PT1 element might be a good approximation.

The dead time element and its Laplace transform is:

$$F_{T_t}(s) = \mathscr{L}\{\epsilon(t)(t - T_t) = e^{-sT_t}\} \tag{4.2}$$

The PT1 lag element and its Laplace transform is:

$$F_{PT1}(s) = \mathscr{L}\{\epsilon(t)e^{-sT_g}\} \tag{4.3}$$

These two elements are combined to obtain the model we will use to approximate the measured step response:

$$G_1(s) = F_1(s) \cdot F_2(s) = e^{-sT_t}\frac{K_s}{sT_g + 1} \tag{4.4}$$

By plugging in the parameters obtained from the characterisation step in section 4.1 (dead-time $T_t = T_u$) we obtain the following transfer function:

$$G_1(s) = e^{-0.165*s}\frac{24.68}{5.96s + 1} \tag{4.5}$$

Figure 5 shows the step response of the calculated transfer function $G_1(s)$ and compares it to the measured step response. It's pretty close, perhaps the parameter T_g can be adjusted so it rises faster. When doing this, however, the function no longer matches the measured data at the beginning where it starts rising. It's possible therefore that the measured system is of higher order. Adding more PT1 elements to our model can help make the approximation more exact, but for this experiment we will stick to a single PT1 element.

By looking at the Bode-Diagram of the model $G_1(s)$ (see figure 7) it is very easy to determine the critical gain $K_{s,crit}$. This is the point at which the phase exceeds $180°$. The amplitude at this point is about $-36\,\mathrm{dB}$, which means that $K_{s,crit} = 36\,\mathrm{dB}$, or:

$$K_{s,crit} = 10^{\frac{36\,\mathrm{dB}}{20}} \approx 63.1 \tag{4.6}$$

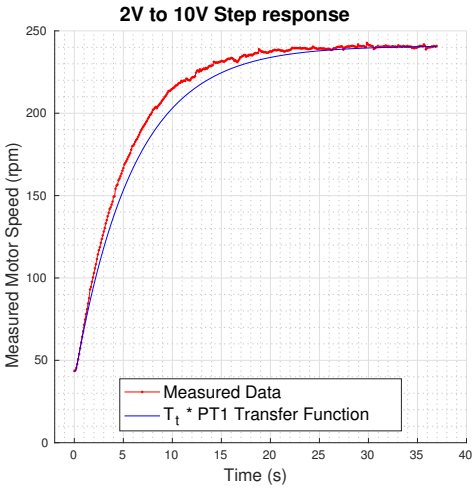


Figure 5: Step response comparison of calculated system $G_1(s)$ and the measured step response.

The closed loop transfer function of $G_1(s)$ with a P-controller is constructed:

$$T(s) = \frac{H(s)G_1(s)}{1 + H(s)G_1(s)} \tag{4.7}$$

Where $H(s)$ is simply the P controller. By setting $H(s) = K_{p,crit}$ and simulating a step response, we obtain a system that exhibits an undamped oscillation (see figure 6). Determining τ_{crit} is now simply a matter of measuring two points in this graph.

$$\tau_{crit} \approx 0.66 \tag{4.8}$$

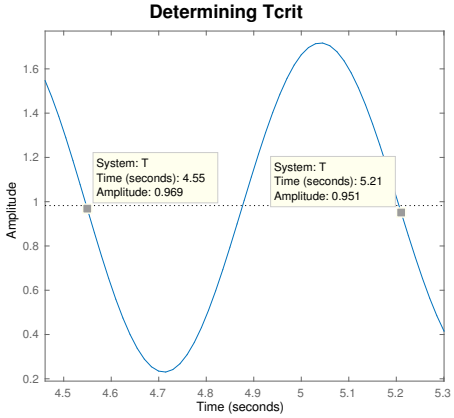


Figure 6: Step response of the closed loop transfer function with $K_p = K_{p,crit}$

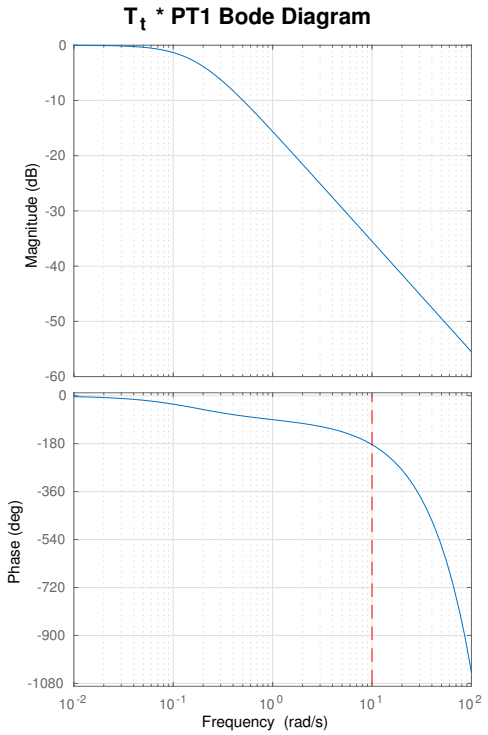


Figure 7: Bode-Plots of the model $G_1(s)$. The phase exceeds $180°$ at about $-36\,\mathrm{dB}$

4.3 System Identification using P. Hudzovic's method

An extensive report on the details of P. Hudzovic's method and the accompanied MATLAB simulations can be found here[2].

The method proposed by P. Hudzovic[1] is based on a model that approximates a plant with a series of PT1 elements multiplied together with varying time constants T_k to form a PTn element, $G_n(s)$.

$$G_n(s,r) = K_s \prod_{k=1}^{n} \frac{1}{1 + s \cdot T_k(r)} \tag{4.9}$$

The neat trick P. Hudzovic proposed was rather than having to find individual, independent time constants for each PT1 element – the effort of which greatly increases with higher orders of n – one should instead have a function $T_k(r)$ which spaces the time constants in a meaningful way. He defines:

$$T_k(r) = \frac{T}{1 - (k-1)r} \tag{4.10}$$

where the parameter r must be confined to the interval $0 \leq r < \frac{1}{n-1}$.

With this approach the problem has effectively been reduced to finding appropriate values for n, T, and r such that the step response of $G_n(s,r)$ approximates the measured step response as closely as possible.

It is not possible to *directly* calculate these values, however, it is possible construct a lookup-table from the equations 4.9 and 4.10 by calculating a (theoretically) infinite number of step responses for all values of n, T, and r, characterising each step response (i.e. determine T_u and T_g), and performing a reverse lookup on those results to find the parameters n and r. This method of reverse-lookup works because the lookup curves are monotonically increasing/decreasing (see figure 8).

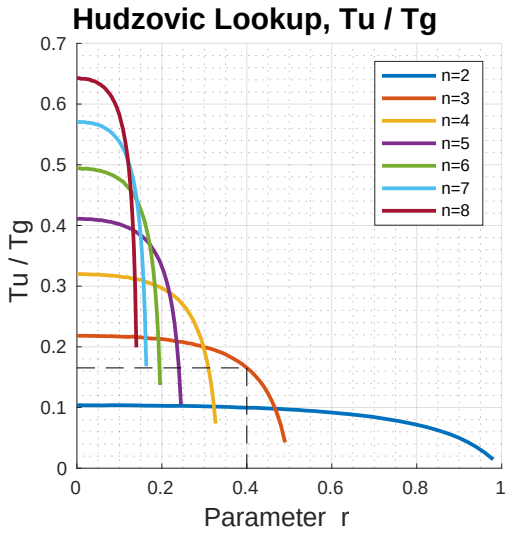
In practice, it is sufficient to calculate about 50 step response curves for each order n and interpolate between those points when performing the lookup.

Using MATLAB, the lookup table is constructed and the parameters n, r, and T are calculated based on the previously determined parameters T_u and T_g. The transfer function $G_2(s)$ is calculated and turns out to be:

$$G_2(s) = \frac{24.68}{1.159s^2 + 5.365s + 1} \tag{4.11}$$

Figure 9 shows the step response of the calculated transfer function $G_2(s)$ and compares it to the measured step response. $G_2(s)$ seems to be a much better approximation than the transfer function $G_1(s)$ obtained in section 4.2.

By looking at the Bode-Diagram of the transfer function $G_2(s)$ (see figure 10), we see one potential issue: This system is a second order system, which means the phase never exceeds $180°$ and thus the parameter $K_{p,crit}$ cannot be determined for any finite value of the P-controller's gain! As a result, it will not be possible to use the method of Ziegler-Nichols for determining controller parameters.



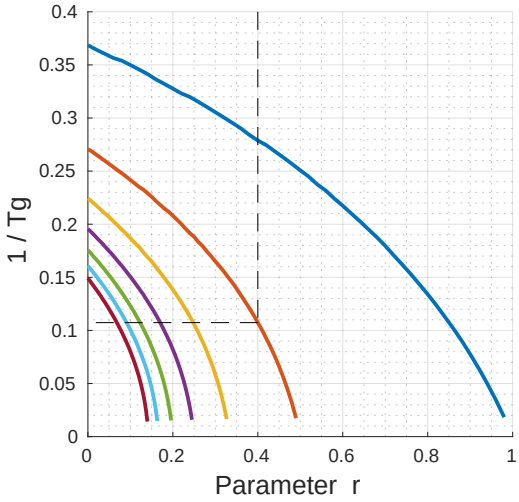


Figure 8: Lookup curves generated with P. Hudzovic's equations. By measuring T_u and T_g it is possible to look up the parameters n, r and T which can be used together with equation 4.9 to construct an accurate transfer function of the measured system.

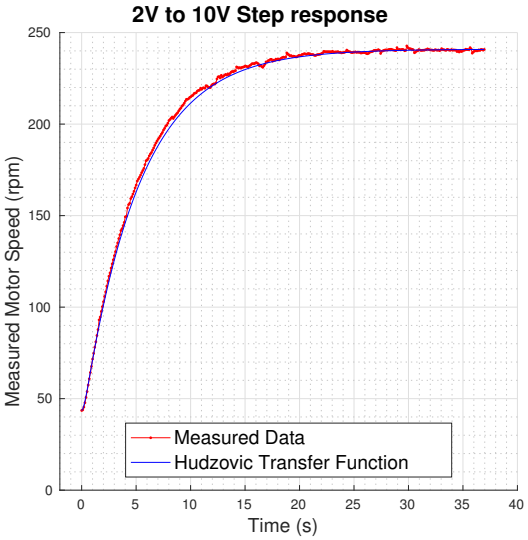


Figure 9: Comparison of the calculated step response function $G_2(s)$ (using P. Hudzovic's method) and the measured step response of the motor.

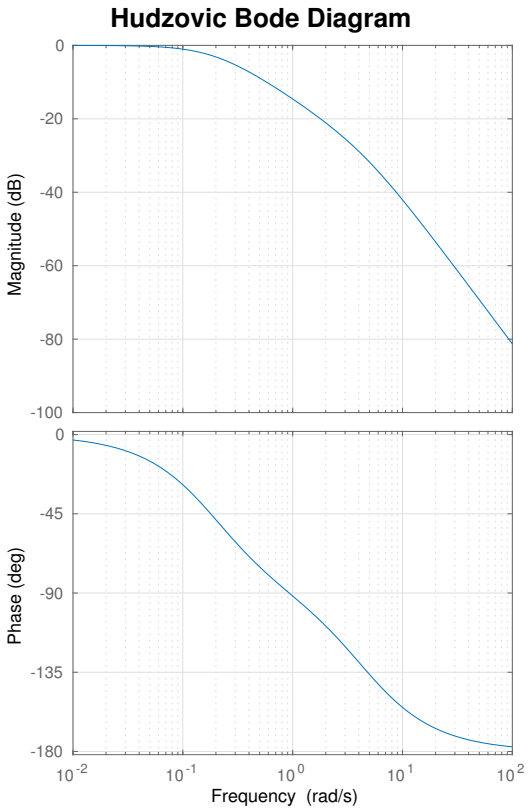


Figure 10: Bode-Diagram of the transfer function $G_2(s)$, acquired using P. Hudzovic's method

4.4 Controller Design using Dead-Time and PT1 element

Now that T_u, T_g, $K_{p,crit}$ and τ_{crit} have been determined, we use the formulas in tables 1, 2 and 3 to create various P, PI and PID controllers. The resulting closed loop step functions are plotted in figures 11, 12 and 13.

The P-controller obtained from the Ziegler-Nichols method is quite unbelievable. The motor requires about 30 seconds to reach its end speed, so why is it able to reach its target value after just three seconds? By simulating what happens on the output of the controller (the voltage that gets fed into the motor), we see what's going on (see figure 14).

The voltage levels being used to control the motor grossly exceed the capabilities of the driver and the maximum rating of the motor. One would have to decrease the K_p parameter in this controller until the voltage reaches acceptable levels again in order to make this controller feasible. The issue then, though, is that the target value doesn't get reached any more.

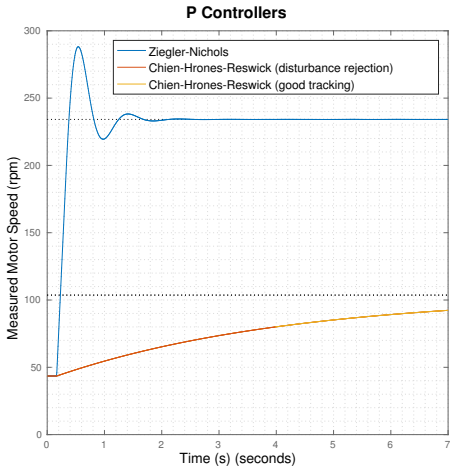


Figure 11: Various P controllers

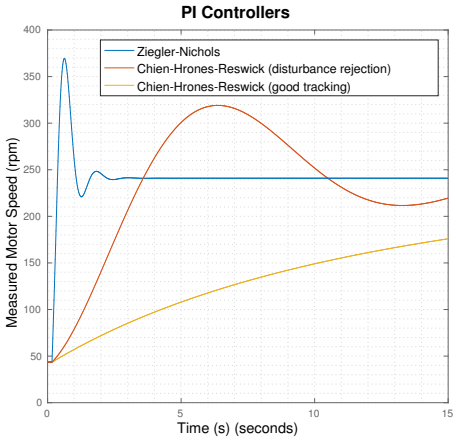


Figure 12: Various PI controllers

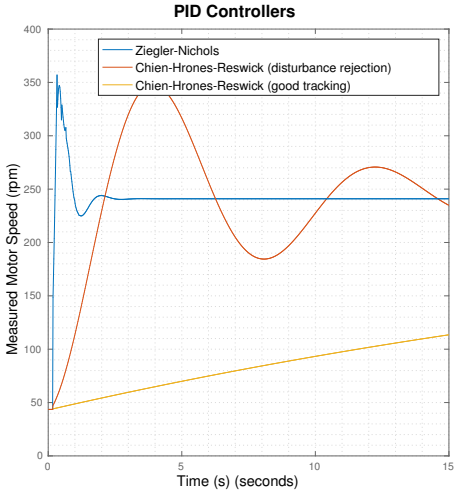


Figure 13: Various PID controllers

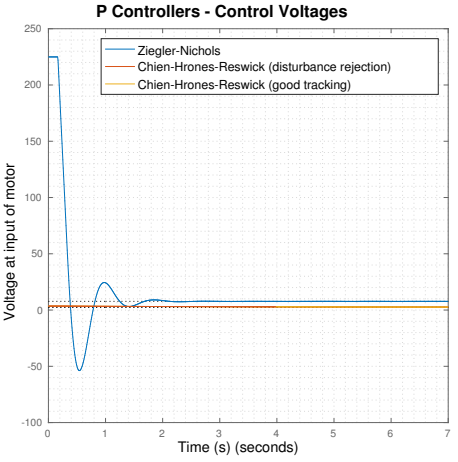


Figure 14: Simulated voltage on the input of the motor

4.5 Simulation with simulink

A simulation was done using Simulink to compare the tracking behaviour between simulation and experiment. Unfortunately the time in the laboratory was not enough to finish the exercise. So we didn't have the time to analyse the tracking behaviour of the electrical motor in the laboratory. However, in Simulink the tracking behaviour was analysed and it can be seen in figure 17. The best value for the PT_1-element we determined to be 5.962, so our PT_1-element is of the form:

$$G(s) = \frac{1}{5.962s + 1} \tag{4.12}$$

For the dead time element we took the value 0.1655. In figure 15 there are 5 different Block diagram. the first 3 diagram represent the simulation model of the plant. The last two Block diagram represent an implemented PID controller with and without a tracking behaviour.

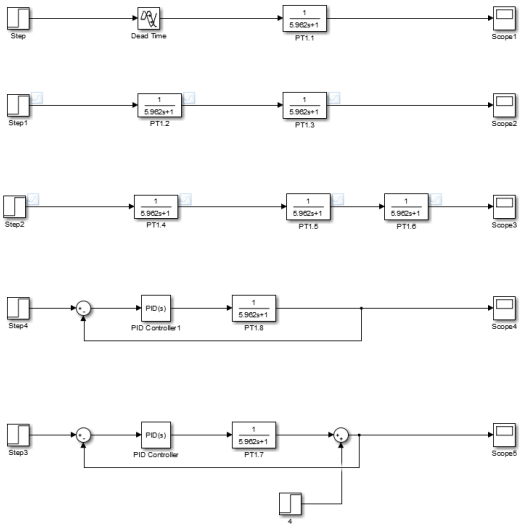


Figure 15: Block diagram of the Simulink simulation

The best PID controller we obtained during the experiment have the values (Proportional P, Integral I and Derivative D) :

$$P = 0.330$$
$$I = 0.660$$
$$D = 0.330$$

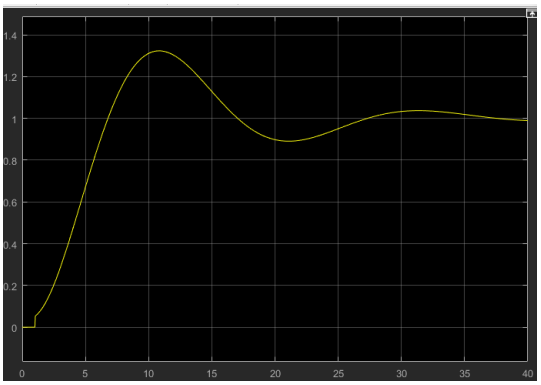


Figure 16: The system without tracking.

The smallest tracking behaviour we obtained with the values:

$$P = 0.330$$
$$I = 0.660$$
$$D = 0.330$$

which are the same as for the normal PID controller. In picture 17 one can see that the tracking behaviour have a change of 180 degree. That means that the electrical motor change his direction.

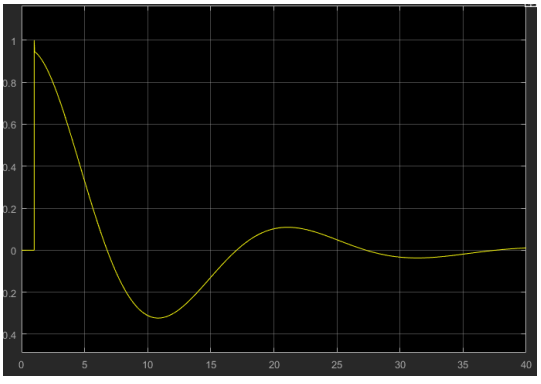


Figure 17: Tracking behaviour of the system.

5 Discussion

The methods proposed by Ziegler-Nichols and Chien-Hrones-Reswick certainly don't yield good controllers by a long stretch, but they can serve as suitable starting points. Of course, if you lack the facilities or knowledge to optimise a controller properly, these methods are fairly easy to implement. In fact, you don't even need a computer to calculate appropriate controllers, these methods can be used with a pen and a paper and a little bit of fiddling.

One of the drawbacks of the Ziegler-Nichols method is that it doesn't work for systems that can't be made unstabled in a closed loop with a P-controller. Such was the case when using P. Hudzovic's system identification technique, which yielded a PT2 element.

Choosing a deadtime element and PT1 element combination as the model got around this issue, as a deadtime element will *always* become unstable at higher frequencies.

References

[1] P. Hudzovic. "Algorithm for step response parameterization". In: *Proceedings of RIP 1994 Process – Control Conference with international participation Horni Becva*. 1994, pp. 145–148.

[2] Alex Murray (TheComet93). *Report for "mathematical laboratory", FHNW Windisch*. https://github.com/thecomet93/mlab. 2016.

[3] *Wikipedia: Faustformelverfahren (Automatisierungstechnik)*. https://de.wikipedia.org/wiki/Faustformelverfafhren_%28Automatisierungstechnik%29. accessed 2017-05-30.