PID made simple or…

What the hell does the UHU controller and what are these parameters for ?

The UHU controller is a digital controller to drive a DC motor to a given position within a minimum time span and with minimal error. It provides a closed loop control with speed and position loopback which ensures that any deviations of the actual position to the set point are immediately compensated. This is done by applying a current to the motor to drive him in the needed direction.

If actual position and set point are identical and no torque is needed to hold the position, no current is applied to the motor.

The story starts to be interesting, when a new position is commanded to the controller or the motor shaft is turned against his holding torque. In this case, the controller has the task to get (back) to the set point as quick as possible and as precise as possible. The ideal current through the motor to achieve this goal has to be calculated and to be put out in form of a PWM (pulse width modulated) signal which is amplified for to drive the motor. The polarity of the current define the rotational direction, the amplitude defines the torque.

But how does the controller calculate the correct current ?

Here a few statements and assumptions:

- The bigger the difference between set point and actual position is, the more torque should be applied to get high velocities and to reach the destination as quick as possible.
- The bigger the displacement is, the more current has to be applied to get the desired torque
- When the actual position comes towards the set point, the current has to be reduced to prevent the motor to overshoot the desired position.

This leads us to the proportional parameter. The calculated position difference is multiplied with this parameter to get a current value which is then proportional to the distance to go.
The bigger this parameter is, the more current is put out to the motor at equal displacements. This makes the control loop more aggressive.

In principle we already have a loopback controller now. It is called P-controller.

Why don't we stop here ?
The problem with P-controllers is, that the current approaches zero for small deviations. Hence the motor is unable to make this little move and a desired position will never be reached accurately.

To be more precise, a second mechanism is needed:

- For small deviations the current value calculated by the P-controller hast to be increased by the time until the current is high enough to make the motor start.

That’s exactly what the I-controller does. He repeatedly multiplies the residual deviation with the I parameter and adds the results to the current value for the power stage until the motor starts to move. If the set point is overshot, the approach starts again from the other side.

And that's it ?
Not yet. The resulting PI-controller now does more or less what we wanted to achieve, at least in slow control processes. But it tends to oscillate with the higher P- and I- settings you need to get good dynamics.

Especially with the I-controller it's easy to understand: If friction makes the motor shaft blocking until a higher current value, the resulting torque makes the motor turn a bit more than the needed position difference. The same problem then occurs from the other side and so on.

Something like an electronic damping is needed. And this is where the D-controller comes in.

- If the motor makes a fast movement, the current value coming from the PI-controller has to be reduced accordingly. This is called negative feedback.

The D-controller multiplies the speed of the motor with the D parameter and subtracts the result from the current value coming from the PI-controller. This eliminates oscillations because it reduces speed and takes energy out of the oscillation process.
But there is a severe drawback: With higher speeds there is always some current deducted which would have been needed to reach the actual, moving set point. The tracking error increases with fast movements.

OK, the PID controller isn't worth it's money then?
You shouldn't see it this way. At limited speeds the PID controller is able to give precise and reliable results, with minimal position errors at least at the target positions. Quite a lot - if not the most - industrial controllers provide just these PID features.

Where is the beef ?
The UHU controller uses a trick to avoid this dynamic position error. This is the so called higher derivative feedback represented by the H-parameter. It does more or less the same as the D-parameter, while using acceleration values instead of the actual speed for the negative feedback.
Because of oscillations always coming along with high accelerations, the negative feedback of these dampens the control loop efficiently. But the undesired influence on the set points is limited to acceleration phases and continuous movements are not affected by position errors.

Nice. And how should I set the H-parameter ?
First you should try to approximate the highest P- and I-values to ensure good dynamics. With sufficiently powered systems you'll soon arrive at a point, where you feel a tendency to instability and where the loop is starting to oscillate. Start increasing the H-parameter along with further increasing of the P- and I-values. Dependent on the mechanical characteristics, your system might need some higher D-values as well.
You'll find an optimum, where a fast response is possible without seeing significant deflections of the tracking error in the terminal program.

Sometimes it looks like the D- and H-parameters are just reducing the values of P and I. This is of course not the case.
In fact, the first ones show curve characteristics and the latter straight lines. Hence you might even find different combinations of these values with comparable good results.