

State space controller for an inverted pendulum

Tobias Plüss

Description

Figure 1 shows the inverted pendulum. It consists of a slider which can be moved in direction x and a pendulum which is mounted on that slider.

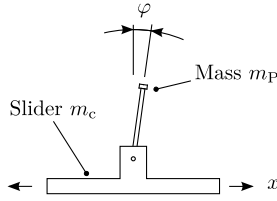


Figure 1: System block diagram

The controller's task now is to move the slider in such a way that the pendulum always points upwards. The controller's input signal is the position x . The slider will then be moved to that position and the pendulum is hold in the upper position.

Differential equations

Mechanical

For the slider's movement, the balance of forces

$$F_c(t) = F_M(t) - F_{P,h}(t) \quad (1)$$

applies, where $F_S(t)$ is the horizontal force on the slider, $F_M(t)$ is the motor's force and $F_{P,h}(t)$ the horizontal component of the pendulum's force.

For $F_c(t)$ itself, the differential equation

$$F_S(t) = m_c \ddot{x}(t) \quad (2)$$

applies which can be put into Equation 3. This results in the differential equation:

$$m_c \ddot{x}(t) = F_M(t) - F_{P,h}(t) \quad (3)$$

For the pendulum's center of mass, the differential equation

$$F_{P,h}(t) = m_P \ddot{x}_P(t) \quad (4)$$

can be written, where $x_P(t)$ is the pendulum's x position, for which

$$x_P(t) = x(t) + l \sin \varphi(t) \quad (5)$$

holds. When derived twice, we get

$$\ddot{x}_P(t) = \ddot{x}(t) + l \ddot{\varphi}(t) \cos \varphi(t) - l \dot{\varphi}^2(t) \sin \varphi(t) \quad (6)$$

bearing in mind the product and chain rules of differential calculus. Therefore,

$$F_{P,h}(t) = m_P (\ddot{x}(t) + l \ddot{\varphi}(t) \cos \varphi(t) - l \dot{\varphi}^2(t) \sin \varphi(t)) \quad (7)$$

and when put together with Equation 3, the following differential equation results:

$$(m_c + m_P) \ddot{x}(t) = F_M(t) - m_P (l \ddot{\varphi}(t) \cos \varphi(t) - l \dot{\varphi}^2(t) \sin \varphi(t)) \quad (8)$$

This equation describes the linear displacement both of the pendulum and the slider. It is a nonlinear differential equation which needs to be linearised before a state-space model can be developed. Since $\varphi = 0$ in the operating point, we can linearise the pendulum around that operating point. The following equations hold:

$$\sin \varphi \approx \varphi \quad (9)$$

$$\cos \varphi \approx 1 \quad (10)$$

Therefore, we can write the linearised differential equation:

$$(m_c + m_P) \ddot{x}(t) = F_M(t) - m_P l \ddot{\varphi}(t) \quad (11)$$

Next, we need to describe the coupling between slider and pendulum. This can be done with the torque. Basically, the torque is expressed through the equation

$$M = J \ddot{\varphi}(t) \quad (12)$$

but there is also an additional torque, resulting from the gravitational force

$$M_g = m_P l g \sin \varphi(t) \quad (13)$$

and a third torque resulting from the slider:

$$M_c = -m_P l \ddot{x}(t) \cos \varphi(t) \quad (14)$$

Note the minus sign. This is because this is the only torque whose direction is in opposition to the other two torques. When putting all together, we get the torque equation:

$$J \ddot{\varphi}(t) = m_P l g \sin \varphi(t) - m_P l \ddot{x}(t) \cos \varphi(t) \quad (15)$$

This is also a nonlinear differential equation. The inertial moment J of the pendulum is

$$J = m_p l^2 \quad (16)$$

so the complete and linearised differential equation is

$$m_p l^2 \ddot{\varphi}(t) = m_p l g \varphi(t) - m_p l \ddot{x}(t) \quad (17)$$

or, when a bit more simplified:

$$l \ddot{\varphi}(t) = g \varphi(t) - \ddot{x}(t) \quad (18)$$

Motor

Next step is to combine the mechanical model of the pendulum with that one of the motor. The motor current is

$$i(t) = \frac{1}{R} \left(v(t) - \frac{n(t)}{k_N} \right) \quad (19)$$

where $v(t)$ is the voltage applied to the motor, $n(t)$ is the motor speed and k_N is the motor speed constant. The motor speed $n(t)$ can be expressed by the slider speed $\dot{x}(t)$. If the motor speed unit is $[\frac{1}{\text{min}}]$, then the slider speed is

$$\dot{x}(t) = \frac{\pi r n(t)}{30} \quad (20)$$

so when put together, we get:

$$i(t) = \frac{1}{R} \left(v(t) - \frac{30 \dot{x}(t)}{\pi r k_N} \right) \quad (21)$$

Motor torque is proportional to motor current with the torque constant k_M , so we can write:

$$M(t) = \frac{k_M}{R} \left(v(t) - \frac{30 \dot{x}(t)}{\pi r k_N} \right) \quad (22)$$

Last but not least, the slider force $F_M(t)$ can be expressed by the motor torque and the gear radius:

$$\begin{aligned} F_M(t) &= \frac{M(t)}{r} \\ &= \frac{k_M}{R r} \left(v(t) - \frac{30 \dot{x}(t)}{\pi r k_N} \right) \end{aligned} \quad (23)$$

Combination

Now we can combine the differential equations for the pendulum with the one for the motor. We get the following system of differential equations:

$$\begin{aligned} (m_c + m_p) \ddot{x}(t) + \frac{30 k_M}{\pi r^2 k_N R} \dot{x}(t) + m_p l \ddot{\varphi}(t) &= \frac{k_M}{R r} v(t) \\ l \ddot{\varphi}(t) + \ddot{x}(t) - g \varphi(t) &= 0 \end{aligned}$$

We use the following shorthands:

$$a = m_c + m_p \quad (24)$$

$$b = \frac{30 k_M}{\pi r^2 k_N R} \quad (25)$$

$$c = m_p l \quad (26)$$

$$d = \frac{k_M}{R r} \quad (27)$$

Then, we can rewrite the differential equation system as follows:

$$a \ddot{x}(t) + b \dot{x}(t) + c \ddot{\varphi}(t) = d v(t) \quad (28)$$

$$l \ddot{\varphi}(t) + \ddot{x}(t) - g \varphi(t) = 0 \quad (29)$$

In order to bring the system in state space form, we solve Equation 28 for $\ddot{\varphi}(t)$ and put that into Equation 29. We get

$$\ddot{\varphi}(t) = \frac{d}{c} v(t) - \frac{b}{c} \dot{x}(t) - \frac{a}{c} \ddot{x}(t) \quad (30)$$

and this results in

$$\frac{ld}{c} v(t) - \frac{lb}{c} \dot{x}(t) - \frac{la}{c} \ddot{x}(t) + \ddot{x}(t) - g \varphi(t) = 0 \quad (31)$$

when put into the second equation. Solving for $\ddot{x}(t)$ results in:

$$\ddot{x}(t) = \frac{g}{1 - \frac{la}{c}} \varphi(t) + \frac{lb}{c - la} \dot{x}(t) - \frac{ld}{c - la} v(t) \quad (32)$$

In the next step, we solve Equation 29 for $\ddot{x}(t)$ and put it into Equation 28. We get

$$\ddot{x}(t) = g \varphi(t) - l \ddot{\varphi}(t) \quad (33)$$

from Equation 29. When put into Equation 28, we get:

$$a g \varphi(t) - a l \ddot{\varphi}(t) + b \dot{x}(t) + c \ddot{\varphi}(t) = d v(t) \quad (34)$$

When solved for $\ddot{\varphi}(t)$, we get:

$$\ddot{\varphi}(t) = \frac{d}{c - al} v(t) - \frac{ag}{c - al} \varphi(t) - \frac{b}{c - al} \dot{x}(t) \quad (35)$$

Therefore, we can now write our state space model:

$$\begin{pmatrix} \dot{x}(t) \\ \dot{\varphi}(t) \\ \ddot{x}(t) \\ \ddot{\varphi}(t) \end{pmatrix} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{g}{1 - \frac{al}{c}} & \frac{bl}{c - al} & 0 \\ 0 & -\frac{ag}{c - al} & -\frac{b}{c - al} & 0 \end{pmatrix} \cdot \begin{pmatrix} x(t) \\ \varphi(t) \\ \dot{x}(t) \\ \dot{\varphi}(t) \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ -\frac{dl}{c - al} \\ \frac{d}{c - al} \end{pmatrix} \cdot v(t) \quad (36)$$

The parameters of the state space model are as follows:

$$\frac{g}{1 - \frac{al}{c}} = -\frac{g m_P}{m_c} \quad (37)$$

$$\frac{bl}{c - al} = -\frac{30 k_M}{k_N m_c \pi r^2 R} \quad (38)$$

$$\frac{ag}{c - al} = -\frac{g (m_P + m_c)}{l m_c} \quad (39)$$

$$\frac{b}{c - al} = -\frac{30 k_M}{k_N l m_c \pi r^2 R} \quad (40)$$

$$\frac{dl}{c - al} = -\frac{k_M}{m_c r R} \quad (41)$$

$$\frac{d}{c - al} = -\frac{k_M}{l m_c r R} \quad (42)$$

Actual parameters

In order to compute the matrices for the actual state space model, the values seen in Table 1 are used.

Parameter	Value	Units
g	9.81	$\left[\frac{m}{s^2}\right]$
k_N	317	$\left[\frac{1}{\text{min V}}\right]$
l_M	0.0302	$\left[\frac{Nm}{A}\right]$
l	280	[mm]
m_c	1.73	[kg]
m_P	0.175	[g]
r	12	[mm]
R	316	[mΩ]

Table 1: Values used for the individual model parameters

State space model

The inverted pendulum's state space model

$$\dot{\mathbf{x}} = \mathbf{A} \mathbf{x} + \mathbf{b} u \quad (43)$$

$$y = \mathbf{c}^T \mathbf{x} \quad (44)$$

uses the following parameters:

$$\mathbf{A} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -0.99234 & -11.556 & 0 \\ 0 & 38.580 & 41.273 & 0 \end{pmatrix} \quad (45)$$

$$\mathbf{b} = \begin{pmatrix} 0 & 0 & 4.6035 & -16.441 \end{pmatrix}^T \quad (46)$$

$$\mathbf{c}^T = \begin{pmatrix} 1 & 0 & 0 & 0 \end{pmatrix} \quad (47)$$

For the eigenvalues of \mathbf{A} , we get the equation

$$\det(s\mathbf{I} - \mathbf{A}) = 0 \quad (48)$$

which gives us the four following eigenvalues: $\{0, 6.02, -5.63, -11.95\}$. As we can see, the second eigenvalue has a positive real part and therefore, the system is – not surprisingly – unstable. The whole system – without controller – can be drawn as a block diagram, shown in Figure 2.

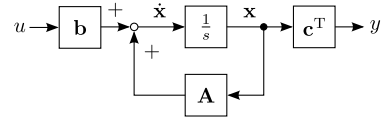


Figure 2: System block diagram

The state vector \mathbf{x} is defined as

$$\mathbf{x} = \begin{pmatrix} x & \varphi & \dot{x} & \dot{\varphi} \end{pmatrix}^T \quad (49)$$

where φ is the pendulum's angular displacement and x is the slider position. The derivatives are the angular speed and slider speed, respectively.

Observability

Observability of the system can be tested by means of the observability matrix \mathbf{Q}_B , which can be calculated with

$$\mathbf{Q}_B = \begin{pmatrix} \mathbf{c}^T \\ \mathbf{c}^T \mathbf{A} \\ \mathbf{c}^T \mathbf{A}^2 \\ \mathbf{c}^T \mathbf{A}^3 \end{pmatrix} \quad (50)$$

and if this matrix has full rank, i.e.

$$\text{rg } \mathbf{Q}_B = 4 \quad (51)$$

the system is observable. In this case, we get a observability matrix of

$$\mathbf{Q}_B = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & -0.9923 & -11.5564 & 0 \\ 0 & 11.4679 & 133.5509 & -0.9923 \end{pmatrix}$$

which has full rank and a determinant of $\det \mathbf{Q}_B \approx -0.9847$ which means the system is observable.

Controllability

Controllability is one other important property which should be tested before a state space controller is implemented. The controllability matrix \mathbf{Q}_S can be calculated with

$$\mathbf{Q}_S = \begin{pmatrix} \mathbf{b} & \mathbf{A} \mathbf{b} & \mathbf{A}^2 \mathbf{b} & \mathbf{A}^3 \mathbf{b} \end{pmatrix} \quad (52)$$

If this matrix has full rank (or in case of a SISO system, a nonzero determinant) the system is controllable. For this system, we get

$$\mathbf{Q}_S = \begin{pmatrix} 0 & 4.6 & -53.2 & 631.1 \\ 0 & -16.4 & 190 & -2830 \\ 4.6 & -53.2 & 631.1 & -7482.1 \\ -16.4 & 190 & -2830 & 33378.5 \end{pmatrix}$$

which also has full rank. Therefore, the system is not only observable, but also controllable.

Controllable standard form

In order to simplify controller design, the system's state space equations Equation 43 and Equation 44 are converted to the controllable standard form. To do this, we need the vector \mathbf{q}_S , which is the last row of the inverse controllability matrix, \mathbf{Q}_S^{-1} . We get

$$\mathbf{q}_S = \begin{pmatrix} -0.0062 & -0.0017 & 0 & 0 \end{pmatrix}$$

and now we are able to calculate the transformation matrix \mathbf{P} which is given by

$$\mathbf{P} = \begin{pmatrix} \mathbf{q}_S \\ \mathbf{q}_S \mathbf{A} \\ \mathbf{q}_S \mathbf{A}^2 \\ \mathbf{q}_S \mathbf{A}^3 \end{pmatrix} \quad (53)$$

The modified system matrices are now given by

$$\tilde{\mathbf{A}} = \mathbf{P} \mathbf{A} \mathbf{P}^{-1} \quad (54)$$

$$\tilde{\mathbf{c}}^T = \mathbf{c}^T \mathbf{P}^{-1} \quad (55)$$

which leads to the following modified system matrices:

$$\tilde{\mathbf{A}} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 404.89 & 38.58 & -11.56 \end{pmatrix}$$

$$\tilde{\mathbf{b}} = \begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix}^T$$

$$\tilde{\mathbf{c}}^T = \begin{pmatrix} -161.29 & 0 & 4.6 & 0 \end{pmatrix}$$

With this new state space model in controllable standard form, it is very easy to calculate the components for the controller.

State space controller

In order to hold the inverted pendulum at the desired point, a state space controller is needed. The

system's structure is shown in Figure 3 when such a controller is implemented and the control loop is closed.

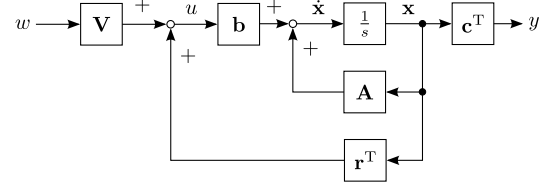


Figure 3: Closed loop system block diagram

The controller is represented by the constant (1×4) matrix \mathbf{r}^T , the matrix \mathbf{V} represents a prefilter. The prefilter is needed together with the controller in order to achieve stationary accuracy.

State space controllers can be designed using various methods, two of them being

- pole placement
- Matlab with the `lqr` or `acker` command.

In this document, we should show both of them.

Method of pole placement

As we have seen in the first section, we have two eigenvalues which are located at inappropriate coordinates. That is, one eigenvalue in the origin and one in the positive half of the s plane at 6.02. The eigenvalues for the modified system in controllable standard form are exactly the same.

These two inappropriate eigenvalues need to be shifted along the real axis until they have negative real parts. We choose -10 and -9 as the destination points for these two eigenvalues. The other eigenvalues are shifted to integer numbers. So the characteristic polynomial of the matrix $\tilde{\mathbf{A}} + \tilde{\mathbf{b}} \mathbf{r}^T$ should be as follows:

$$(s + 12)(s + 6)(s + 10)(s + 9)$$

which leads to the polynomial

$$s^4 + 37s^3 + 504s^2 + 2988s + 6480$$

when expanded. Now, from the coefficients of this polynomial, we can directly see the desired form of the matrix $\mathbf{A}_{CL} = \mathbf{A} + \tilde{\mathbf{b}} \mathbf{r}^T$:

$$\mathbf{A}_{CL} \stackrel{!}{=} \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -6480 & -2988 & -504 & -37 \end{pmatrix}$$

so we get

$$\mathbf{r}^T = \begin{pmatrix} -6480 & -3392.9 & -542.6 & -25.44 \end{pmatrix}$$

for the controller.

Simulations

Figure 4 shows two step responses of the system simulated with Simulink. The step occurs at position $t = 0.5$ s. The first step response is simulated when the system is controlled using the above designed controller, and the second step response is a simulation without any controller. As one can see, the uncontrolled system is unstable and things get out of hand very quickly here compared to the controlled system, which, however, has a stationary error. But at least the controlled system is no longer unstable.

In order to compensate the stationary error, the prefilter must be designed. In the case of a SISO system like this one, the prefilter is a scalar value. From the state space model $\tilde{\mathbf{A}}$, $\tilde{\mathbf{b}}$ and $\tilde{\mathbf{c}}^T$, we can obtain the transfer function

$$G(s) = \frac{4.604s^2 - 161.3}{s^4 + 37s^3 + 504s^2 + 2988s + 6480}$$

For such a transfer function, we can calculate the final value with some aid of the Laplace transform. The final value is

$$\lim_{t \rightarrow \infty} y(t) = \lim_{s \rightarrow 0} sG(s) \quad (56)$$

and when a step is used for the system input we get

$$\lim_{t \rightarrow \infty} y(t) = \lim_{s \rightarrow 0} s \frac{1}{s} G(s) = \lim_{s \rightarrow 0} G(s) \quad (57)$$

So, for this system, we can obtain a final value of

$$\lim_{s \rightarrow 0} G(s) \approx -0.0249$$

and therefore, the prefilter must have a gain of

$$V = \frac{1}{-0.0249} \approx -40.18$$

The step response achieved with state space controller and prefilter can be seen in Figure 5.

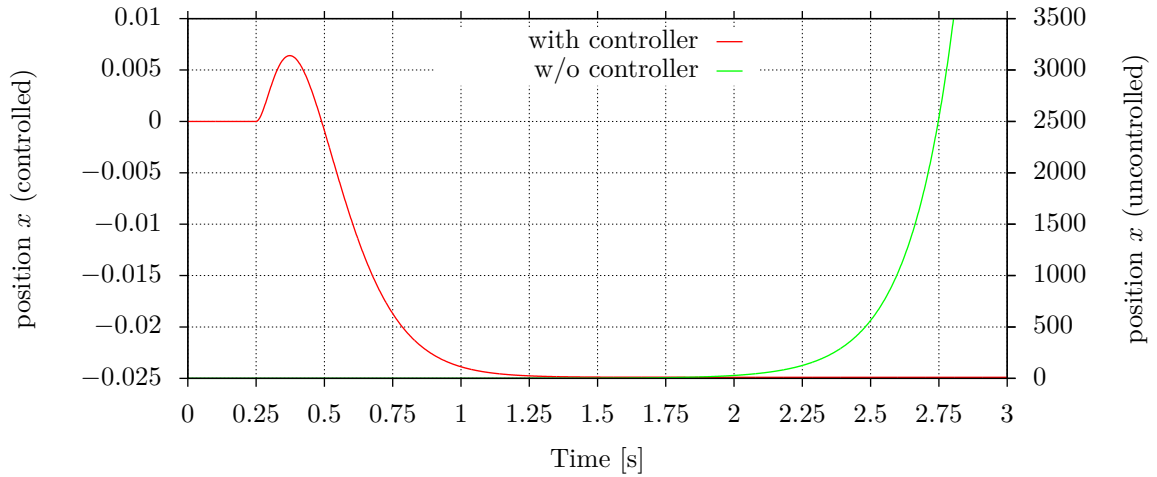


Figure 4: Simulated step responses of the inverted pendulum with and without controller

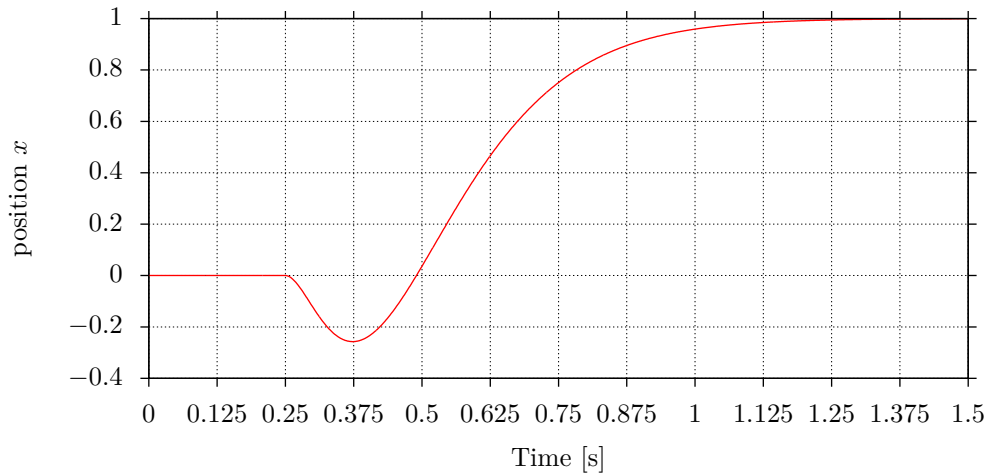


Figure 5: Simulated step response with prefilter

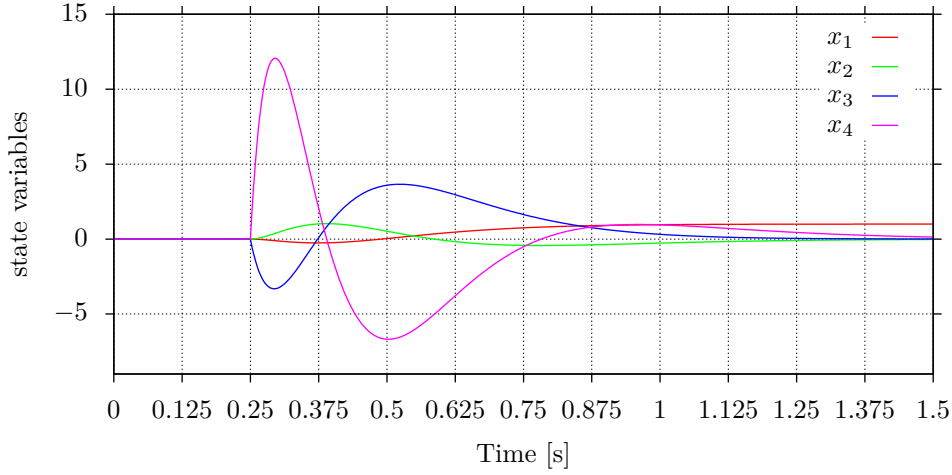


Figure 6: Simulated values of the state variables

In Figure 6 the four state variables are plotted over the time in order to verify that no extreme values occur. E.g. if the state feedback would have too large gains, it produces too large amplitudes at the actuator.

In Figure 7 we can see a step forth and back. As we can see, the position x always first moves towards the opposite position than desired. This is because the system given is not a minimum-phase system. Nonminimal phase systems contain some allpass path.

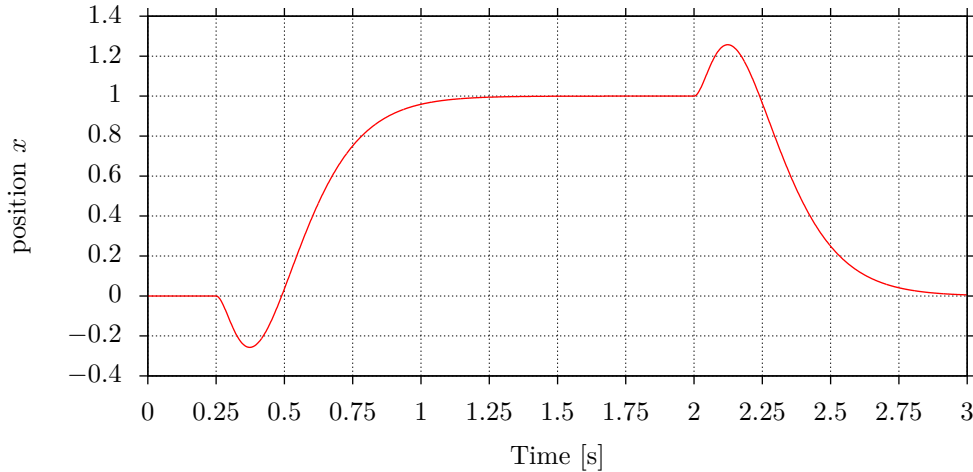


Figure 7: Simulated step response back and forth

Using Matlab

In Matlab, we can easily check controllability and observability with the following code snippet:

```
% observability matrix
Qb = [c; c*A; c*A^2; c*A^3];

% controllability matrix
Qs = [b A*b A^2*b A^3*b];
rank(Qb)
rank(Qs)
```

Also, a state space controller can be developed using the built-in function `acker` which implements Ackermann's formula to calculate the controller com-

ponents. With

```
acker(A, b, [-12 -6 -10 -9])
```

the controller matrix \mathbf{r}^T is directly returned, assuming \mathbf{A} and \mathbf{b} are the state space matrices in controllable standard form. The system's poles are placed at -12 , -6 , -10 and -9 , as before. Since the `acker` function in Matlab assumes negative feedback, the signs of the controller components are the opposite of that ones which we calculated before, that is

```
ans =
1.0e+03 *
6.4800    3.3929    0.5426    0.0254
```

but if we look closer at these values, they should look like before.

Another possible solution would be an LQR controller. It could be calculated using the `lqr` command in Matlab. The LQR controller, also known as Riccati controller, is an optimised controller. It can be calculated by minimising the optimisation criterion

$$J = \int_0^{\infty} \mathbf{x}^T \mathbf{Q} \mathbf{x} + \mathbf{u}^T \mathbf{R} \mathbf{u} dt \quad (58)$$

which leads to a so-called algebraic Riccati equation. It can be solved using numerical methods. For a SISO system, the value \mathbf{R} is not a matrix but a scalar R . The larger \mathbf{R} or R is, respectively, the slower is the controller.

The matrix \mathbf{Q} is a diagonal matrix. It is called weighting matrix, since we can express a weighting for each state variable. This weighting expresses how important that particular state variable is for the controller. We choose

$$\mathbf{Q} = \begin{pmatrix} 7000 & 0 & 0 & 0 \\ 0 & 8000 & 0 & 0 \\ 0 & 0 & 300 & 0 \\ 0 & 0 & 0 & 200 \end{pmatrix}$$

and

$$R = 10$$

as starting points. With a few iterations, we can obtain quite interesting controllers. The corresponding step responses can be seen in Figure 8.

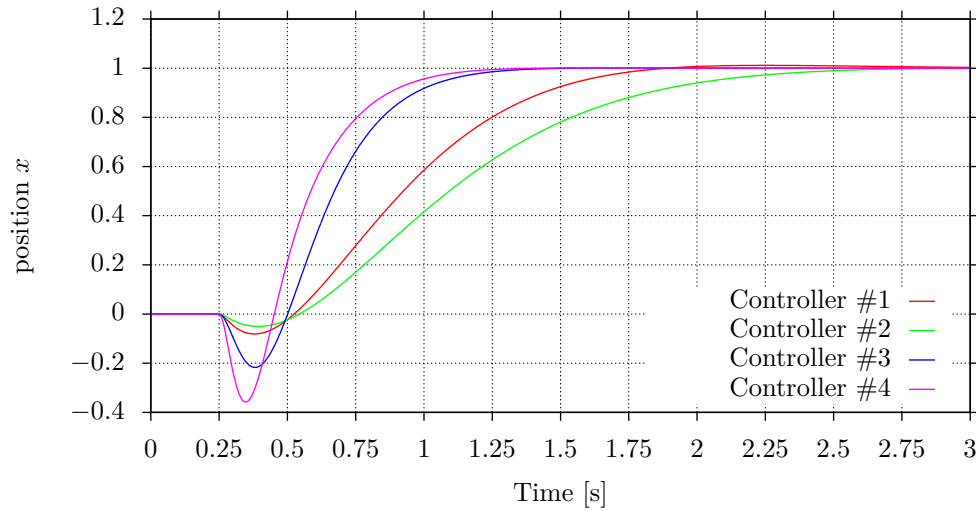


Figure 8: Simulated step responses for various Riccati controllers

Measurements

The first controller implemented by pole placement works fine, but has some slight oscillations. With the Riccati controller we get better results. The best result can be achieved with

$$\mathbf{Q} = \text{diag}(9000, 4000, 0, 0)$$

and

$$R = 2$$

which can easily be found by some iterations. Since the model is only a approximation, there is no exact way to calculate the controller coefficients. The corresponding controller's step response is visible in Figure 9. Figure 10 is a very similar diagram, but here, the position is shown over a much longer period.

The controller has the coefficients

$$\mathbf{r}^T = (67.08 \quad 86.61 \quad 36.55 \quad 12.48)$$

which results in the poles at

$$s = \{-21.24 \pm 18.74j, -3.06 \pm 2.02j\}$$

for the closed loop.

As we can see, in the simulation there are no oscillations. However, in the real system, there are very slight oscillations since the smallest disturbance would tilt the pendulum.

Figure 11 shows the step at the setpoint, as well as the angle. Figure 12 is a similar diagram, but shows the angle for a much longer period.

At http://files.kooltek.net/hslu/inv_pend.mp4 it is possible to watch the inverted pendulum with controller in action.

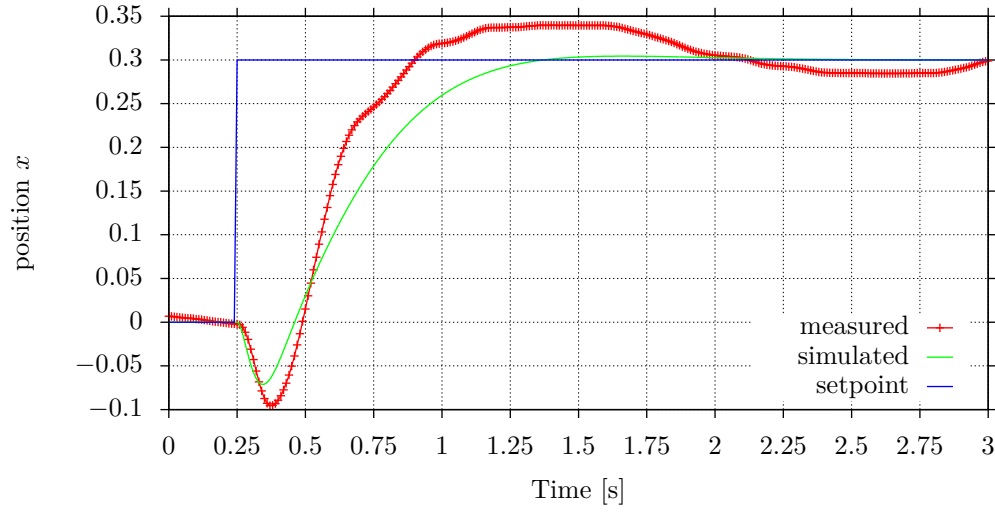


Figure 9: Comparison between simulated Riccati controller and the measured step response

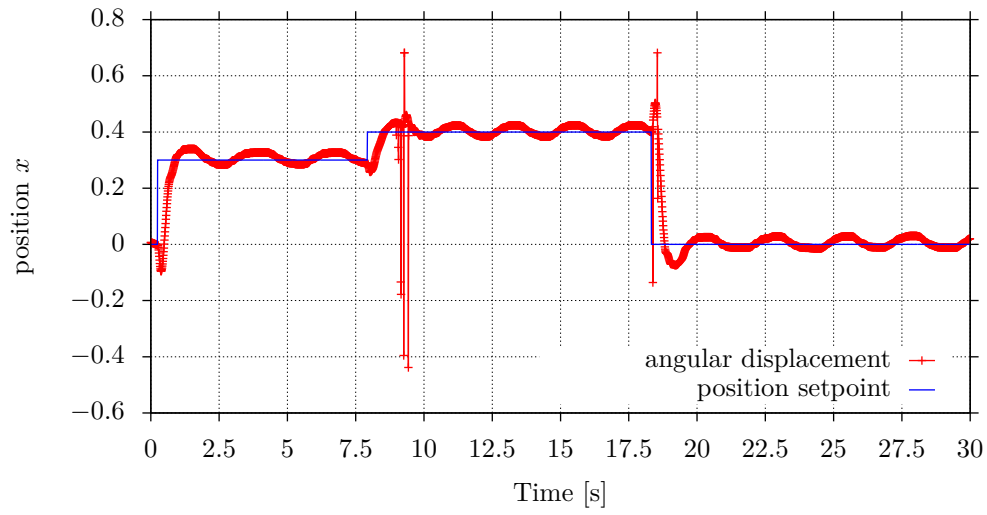


Figure 10: Angular displacement during a step back and forth

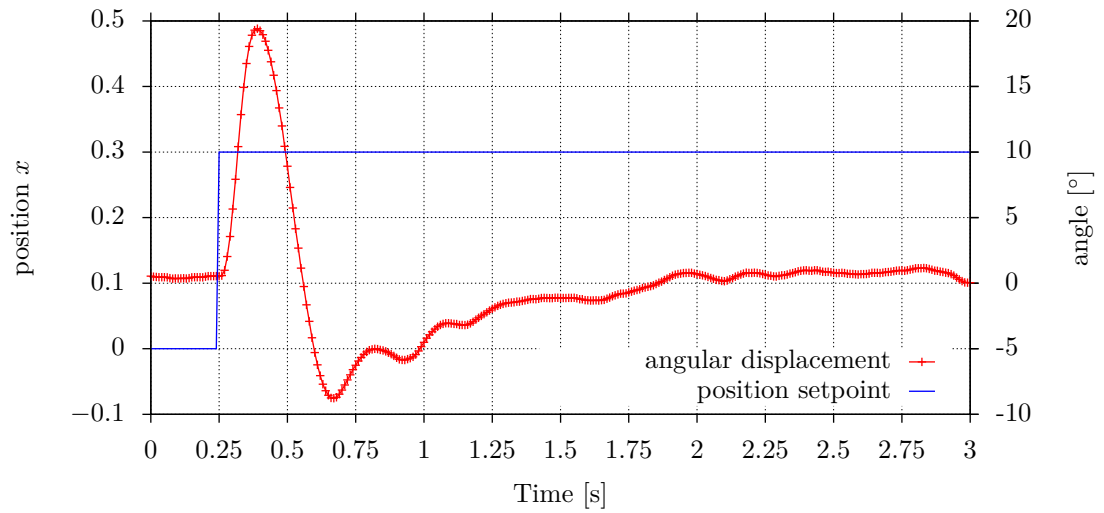


Figure 11: Angular displacement during a step

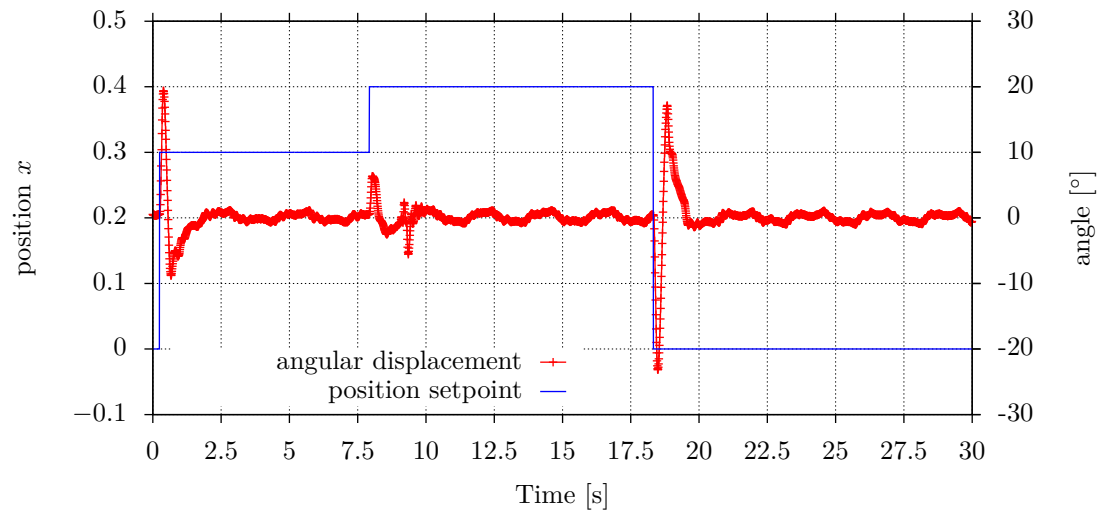


Figure 12: Angular displacement during a step back and forth

References

- [1] O. Föllinger: Regelungstechnik 2. Aufl. S. 281-373.
- [2] Dr. Th. Holzhüter, FH Hamburg: Zustandsregelung.
- [3] J. Lunze: Regelungstechnik; Bd. 1 und 2.