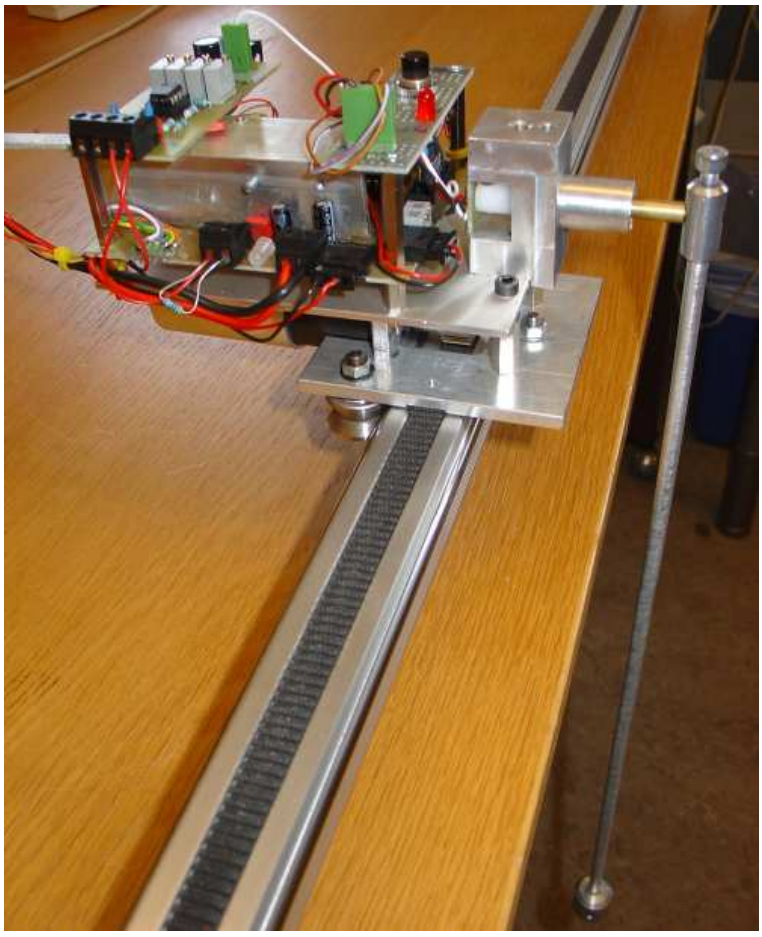
Nonlinear Control and Servo Systems

Laboratory Exercise 3
Optimal Control of Pendulum on Cart

Created: 2008 by Pontus Giselsson
Updated for Julia: 2018 by Mattias Fält
Latest update: December 10, 2018



1. Introduction

This laboratory exercise addresses optimal control applied to a free swinging pendulum that is attached to a cart. The laboration consists of three parts. The first part is on time optimal control of the cart. The second and third parts deal with time optimal control of the pendulum and cart together. In all three parts, the task will be to move the cart along the track in as short time as possible. The system should start and stop at rest and we have limited control actuation.

Important! There are 5 assignments in the lab. Number 1, 2, 4 and 5 have **home assignment** parts, which you will have to do before the lab.

The files you need for the lab can be downloaded from the course homepage. Initialize the lab by running the script `pend_init.m` in Matlab.

2. Modeling of a Pendulum on a Cart

2.1 System description

The system consists of a cart that is driven by a DC-motor and a free swinging pendulum that is attached to the cart. The system will be controlled by a cascaded controller, see Figure 1. The outer controllers, C_2 in Figure 1, that will be designed during the lab with different control objectives in mind, will use the acceleration reference to the inner loop as its control signal. We will use both feedback and feedforward control during the lab.

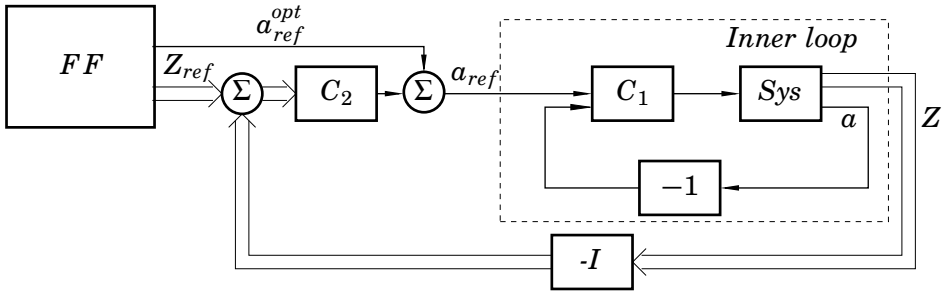


Figure 1 Control architechture

2.2 Nonlinear Model

The inner loop in Figure 1 consists of a cascade of a current loop and a PI-controlled velocity/acceleration loop. Without going into details the resulting cart dynamics can be modeled as a double integrator from acceleration reference a_{ref} to cart position p, that is

$$P = \frac{1}{s^2} a_{ref}$$

We are intereseted in the velocity and the position of the cart. To create a state space model of the cart we introduce position, p, as one state, and velocity, $\dot{p}$, as the other state. If we introduce

$$x = (p\ \dot{p})^T$$

the state equation becomes

$$\dot{x} = A_c x + B_c a_{ref} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} x + \begin{pmatrix} 0 \\ 1 \end{pmatrix} a_{ref} \tag{1}$$

A pendulum model is most easily obtained using Lagrange mechanics. The

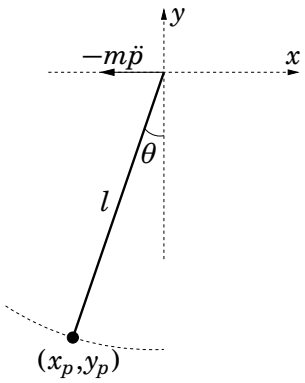


Figure 2 The pendulum

acceleration of the suspension point of the pendulum is equal to the acceleration of the cart, $\ddot{p}$. This creates an oppositely directed force, $F_x = -m\ddot{p}$, at the suspension point of the pendulum. This force acts along the negative x-axis, see Figure 2. The potential energy, V, and the kinetic energy, T, in the x-y coordinates are

$$V = mgy_p \ , \quad T = \frac{1}{2}m(\dot{x}_p^2 + \dot{y}_p^2)$$

where x_p and y_p are the pendulum end point coordinates, m is the pendulum mass and g is the gravitational acceleration. We introduce the generalized coordinate θ, which is the pendulum angle, see Figure 2. Note that we only need one generalized coordinate since the radius is constant and equal to the length of the pendulum[1], l. The relationsship between the coordinate systems is

$$\begin{aligned} x_p &= r_x(\theta) = -l\sin\theta \\ y_p &= r_y(\theta) = -l\cos\theta \end{aligned}$$

In the generalized coordinate, θ, the potential and kinetic energy become

$$V = -mgl\cos\theta \ \text{ and } \ T = \frac{1}{2}ml^2\dot{\theta}^2$$

The Lagrangian is $L = T - V$ and the Lagrange equation is

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \frac{\partial L}{\partial \theta} = (F_x \ 0) \cdot (\frac{\partial r_x(\theta)}{\partial \theta} \ \ \frac{\partial r_y(\theta)}{\partial \theta})^T$$

Calculation of the partial derivatives gives

$$\frac{d}{dt}(ml^2\dot{\theta}) + mgl\sin\theta = m\ddot{p}l\cos\theta$$

[1]For the cart-pendulum process in the lab exercise, the pendulum length, l, is 0.345 m.

which results in the following pendulum equation

$$\ddot{\theta} = -\frac{g}{l}\sin\theta + \frac{\ddot{p}}{l}\cos\theta$$

The reaction forces from the pendulum to the cart is attenuated by the inner controller. Thus an approximate model of the complete system dynamics is

$$\begin{pmatrix} \ddot{p} \\ \ddot{\theta} \end{pmatrix} = \begin{pmatrix} a_{ref} \\ -\frac{g}{l}\sin\theta + \frac{\ddot{p}}{l}\cos\theta \end{pmatrix} = \begin{pmatrix} a_{ref} \\ -\frac{g}{l}\sin\theta + \frac{a_{ref}}{l}\cos\theta \end{pmatrix} \tag{2}$$

where $\ddot{p}$ in the second equation has been replaced by the control signal a_{ref}.

2.3 Linearized Model

The model of the cart dynamics is linear but the pendulum equation is non-linear. When linearizing the pendulum equation in the downward position, around $\theta = 0$, we get $\sin\theta \approx \theta$ and $\cos\theta \approx 1$. We introduce the state vector

$$z = (p \;\; \dot{p} \;\; \theta \;\; \dot{\theta})^T$$

Linearization of the full system dynamics, (2), results in the following state space system

$$\dot{z} = Az + Ba_{ref} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -g/l & 0 \end{pmatrix} z + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1/l \end{pmatrix} a_{ref} \tag{3}$$

3. Control of the Pendulum on Cart

The first part of this laboratory will be on time optimal control of the cart. The second part will address time optimal control of the cart and the pendulum. The third part uses dedicated optimization software to create time optimal control trajectories. To make it more interesting we will have additional constraints specified by areas where the end point of the pendulum must not enter.

3.1 Time-optimal Control (general case)

The linearized models will be used when calculating the time-optimal control problems. To avoid infinite control signals, we need control signal limitations, which is a physically natural constraint on the control design. The problem becomes

minimize t_f = minimize $\int_0^{t_f} 1\,dt$
subject to: $\dot{x} = A_o x + B_o u$
$|u| \leq u_{max}$
$x(0) = x_0$
$x(t_f) = x_{t_f}$

where x is some arbitrary state vector and u is some control signal. The Hamiltonian, H, becomes $H = 1 + \lambda^T(A_o x + B_o u)$. The Maximum Principle states that if we have optimal trajectories $u^*(t)$ and $x^*(t)$ then

$$\min_u H(x(t), u(t), \lambda(t)) = H(x^*(t), u^*(t), \lambda(t))$$

where

$$\dot{\lambda} = -\frac{\partial H}{\partial x} = -A_o^T \lambda$$

Since the only term that is dependent on u in H is $\lambda^T(t)B_o u(t) = \sigma(t)u(t)$, with $\sigma(t) \equiv \lambda^T(t)B_o$, H is minimized by choosing

$$u^*(t) = \begin{cases} -u_{max} & , \quad \sigma(t) > 0 \\ u_{max} & , \quad \sigma(t) < 0 \end{cases}$$

If σ is zero only a finite number of time points then the time optimal controller is a bang-bang controller. This is the case if (A, B_i) is controllable for every column B_i of B.

3.2 Time-optimal Control of the Cart

The model of the cart is found in (1) but is restated here for convenience.

$$\dot{x} = A_c x + B_c a_{ref} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} x + \begin{pmatrix} 0 \\ 1 \end{pmatrix} a_{ref} \tag{4}$$

where $x = (p\ \dot{p})^T$. The problem we are dealing with in this part of the lab is to move the cart along the track in as short time as possible. The cart should start at rest in one point, p_0, and stop at rest in the origin. The magnitude of the control signal is limited to a_{max} m/s^2. Mathematically this problem can be formulated as the following minimum time optimization problem

minimize t_f
subject to: $\dot{x} = A_c x + B_c a_{ref}$
$|a_{ref}| \leq a_{max}$
$x(0) = (p_0\ 0)^T$
$x(t_f) = (0\ 0)^T$

ASSIGNMENT 1

Home assignment:

- Use the theory in Section 3.1 to conclude that the time-optimal controller is a bang-bang controller. Decide the number of switches in the optimal control trajectory for the cart. That is, decide how many times $\sigma(t)$ changes sign. Assume that $p_0 \leq 0$ and determine an expression for the optimal a^*_{ref} (you do not yet have to find the points in time when σ changes sign).
- The solution to state equation (4) is

$$x(t_f) = e^{A_c t_f} x(0) + \int_0^{t_f} e^{A_c(t_f-\tau)} B_c a_{ref} d\tau \tag{5}$$

  where

$$e^{A_c t} = \begin{pmatrix} 1 & t \\ 0 & 1 \end{pmatrix}$$

  Set your calculated a^*_{ref} as control signal in (5) and calculate the switching time, t_1, and the final time, t_f.

At the lab: Edit `assignment1.m` and run it to calculate the switching times and the final time. Simulate the system using `cart.mdl` for different values of p_0 and a_{ref}. Note that the implemented model has been discretized in time, so the switching times are now a multiple of the sampling period. Is the control objective achieved?

Since we have two states it is interesting to regard the problem from a geometric point of view. Set $p_1 = p$ and $p_2 = \dot{p}$, for $a_{ref} = a_{max}$ we have

$$\dot{p}_1 = p_2, \quad \dot{p}_2 = a_{max}$$

The time variable can be eliminated by forming $dp_2/dp_1 = \frac{dp_2}{dt} / \frac{dp_1}{dt}$. This gives

$$\frac{dp_2}{dp_1} = \frac{a_{max}}{p_2}$$

Rearranging the terms and integrating give

$$\int dp_1 = \int \frac{p_2}{a_{max}} dp_2$$

with the solution

$$p_1 + C_1 = \frac{p_2^2}{2a_{max}} \tag{6}$$

When instead $a_{ref} = -a_{max}$ we get the solution

$$p_1 + C_2 = -\frac{p_2^2}{2a_{max}} \tag{7}$$

Phase plane trajectories for some values of C_1 and C_2, when $a_{max} = 3$ m/s^2, are plotted in Figure 3.

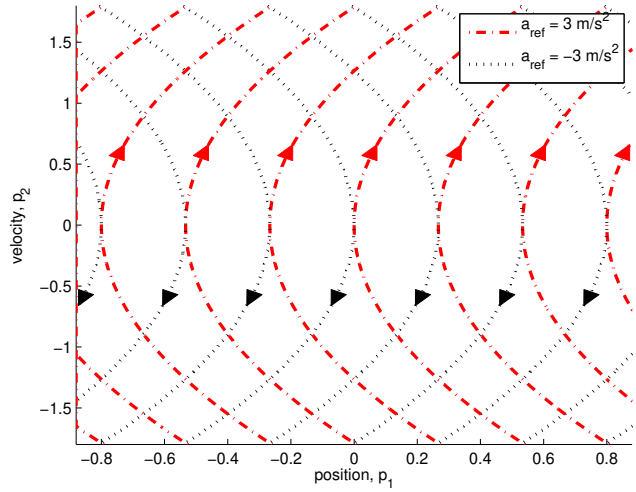


Figure 3 Trajectories for $a_{ref} = a_{max} = 3$ m/s^2 and $a_{ref} = -a_{max} = -3$ m/s^2

ASSIGNMENT 2

Home assignment:

- The system should be controlled to the origin. Decide graphically from Figure 3 the regions of the state space where a_{max} and $-a_{max}$ should be used respectively to achieve this. Draw the switching curve between the two regions.
- Determine an equation $f(p_1, p_2) = 0$ that describes how p_1 and p_2 relate to each other on the switching curve. (*Hint*: First determine the constants, C_1 and C_2, in (6) and (7). Then put the equations together. To do this you will need to use sign(p_2).)
- $f(p_1, p_2)$ takes negative values for points on one side of the switching curve, and positive values for points on the other side. Determine which side that gives positive and negative values respectively. We want a_{ref} to be a_{max} on one side of the switching curve and $-a_{max}$ on the other. Derive an expression for a_{ref}. (*Hint*: Use sign($f(p_1, p_2)$).)

At the lab:

- Type your derived controller in the block for embedded matlab code in `cart.mdl` (to avoid rapid switching when the system is close to the origin, linear state feedback will be used instead in a neighborhood of the origin). Switch controller and simulate the system to see if the control objectives are achieved. Phase plane trajectories can be plotted by the script `plot_cart_pp.m`. Set `initPosError` to see what happens if there is an error in the assumed initial position. Which controller performs best if the actual initial position is not p_0?
- Try the two controllers on the real system. To do this you need to change from "Simulation model" to "Real system" in `cart.mdl`. The "Real system" is found in `pend_lib.mdl`. Choose an initial position satisfying $-0.8 \leq p_0 \leq 0$.

3.3 Time-optimal Control of the Pendulum on Cart

The linearized state space model for the full system is found in (3), but is

restated here for convenience.

$$\dot{z} = Az + Ba_{ref} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -g/l & 0 \end{pmatrix} z + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1/l \end{pmatrix} a_{ref}$$

where $z = (p\ \dot{p}\ \theta\ \dot{\theta})^T$. The control objective is the same as in the previous section, that is to move the cart from one position on the track, p_0, to another position as fast as possible with limited control actuation. The system should be at rest both at the starting point and the final point. This is a time optimal control problem and from the introductory section we know that the optimal control trajectory is of bang-bang type, since the system is controllable. If the matrix A would have had real eigenvalues, then we would also know that the optimal control would have at most three switches, since it is a fourth-order system. Unfortunately, A has two non-real eigenvalues. Nevertheless, we will look for an optimal triple-switch bang-bang controller. That is

$$a^*_{ref}(t) = \begin{cases} a_{max} & , \quad 0 \le t \le t_1 \\ -a_{max} & , \quad t_1 \le t \le t_2 \\ a_{max} & , \quad t_2 \le t \le t_3 \\ -a_{max} & , \quad t_3 \le t \le t_f \end{cases}$$

The optimal switching times are calculated using (5) with appropriate system matrices. We will need e^{At} to do this (Note: $\sqrt{g/l} = \omega$).

$$e^{At} = \begin{pmatrix} 1 & t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \cos\omega t & \frac{1}{\omega}\sin\omega t \\ 0 & 0 & -\omega\sin\omega t & \cos\omega t \end{pmatrix}$$

Insert this into (5) and we will get

$$\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} p_0 \\ 0 \\ 0 \\ 0 \end{pmatrix} + \tag{8}$$

$$a_{max}\int_0^{t_1} \begin{pmatrix} t_f - \tau \\ 1 \\ \frac{1}{l\omega}\sin\omega(t_f - \tau) \\ \frac{1}{l}\cos\omega(t_f - \tau) \end{pmatrix} d\tau +$$

$$a_{max}(-\int_{t_1}^{t_2}(...)d\tau + \int_{t_2}^{t_3}(...)d\tau - \int_{t_3}^{t_f}(...)d\tau)$$

The primitive function is

$$F(\tau) = \begin{pmatrix} t_f\tau - \frac{\tau^2}{2} \\ \tau \\ \frac{1}{l\omega^2}\cos\omega(t_f - \tau) \\ -\frac{1}{l\omega}\sin\omega(t_f - \tau) \end{pmatrix}$$

Insert this into (8) and we get

$$\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} p_0 \\ 0 \\ 0 \\ 0 \end{pmatrix} + a_{max}(-F(0) + 2F(t_1) - 2F(t_2) + 2F(t_3) - F(t_f)) \quad (9)$$

The script `assignment3.m` finds the smallest t_1, t_2, t_3 and t_f that satisfy (9) for chosen a_{max} and p_0, thus giving us the time-optimal triple-switch controller. The optimal state and control trajectories that result from applying the optimal control trajectory are called z^{opt} and a_{ref}^{opt} respectively.

ASSIGNMENT 3

- Choose a_{max} and p_0 in `assignment3.m`, and run the script. Then simulate the system using `cart_pend.mdl`. Use the script `plot_pend.m` to plot or animate the resulting pendulum movements.
- Simulate again, but change initial values for the pendulum. The initial values are changed in `assignment3.m`. Why are the control objectives not met?

Since the control objectives are not met we will introduce feedback. In Assignment 2 we were able to analytically derive an optimal feedback law based on the switching curve in the phase plane, but this approach is not feasible when we are now considering a fourth-order system. We will instead use model predictive control (MPC) to take care of when the state trajectories, z, deviate from their optimal trajectories, z^{opt} and also to penalize when the control signal, a_{ref}, deviates from the optimal one, a_{ref}^{opt}, to get a smooth controller. Since we have a maximum possible cart acceleration of 7 m/s^2, our control signal should stay within this limit, $|a_{ref}| \leq 7$. The track is a bit more than 1 meter long. We want our controller to take care of this limitation as well. If we define our starting position, p_0, to be 0.1 m from the left end of the track we get that $p + p_0 \leq 0.9$ m and $p + p_0 \geq -0.1$ m since $p = 0$ at p_0. To use linear MPC we linearize the system around the optimal trajectory and discretize the result. The resulting models are

$$\begin{aligned}
\Delta z(k+1) &= \Phi(k)\Delta z(k) + \Gamma(k)\Delta a_{ref}(k) \\
\Phi(k) &= \begin{pmatrix} 1 & h & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \cos\omega_k h & \frac{1}{\omega_k}\sin\omega_k h \\ 0 & 0 & -\omega_k \sin\omega_k h & \cos\omega_k h \end{pmatrix} \\
\Gamma(k) &= \begin{pmatrix} \frac{h^2}{2} \\ \omega_k h \\ -\frac{b_k}{\omega_k^2}(\cos\omega_k h - 1) \\ \frac{b_k}{\omega_k}\sin\omega_k h \end{pmatrix} \\
\omega_k &= \sqrt{\frac{g\cos\theta^{opt}(k)}{l}} \\
b_k &= \frac{\cos\theta^{opt}(k)}{l}
\end{aligned}$$

where $\Delta z(k) = z(k) - z^{opt}(k)$ and $\Delta a_{ref}(k) = a_{ref}(k) - a_{ref}^{opt}(k)$. It turns out that the discretized model depends on the angle of the pendulum. Since we know at which angle the pendulum should be at every time step, $\theta^{opt}(k)$, we use this angle in our linearized model. This is a good approximation, if the actual pendulum angle, $\theta(k)$, is not too far away from the optimal, $\theta^{opt}(k)$.

Before we can state the optimization problem that the MPC-controller should solve at each time step, we need to define some matrices

$$
\begin{aligned}
Z(k) &= \begin{pmatrix} z(k+1) \\ \vdots \\ z(k+N) \end{pmatrix} & A_{ref}(k) &= \begin{pmatrix} a_{ref}(k) \\ \vdots \\ a_{ref}(k+N-1) \end{pmatrix} \\
Z^{opt}(k) &= \begin{pmatrix} z^{opt}(k+1) \\ \vdots \\ z^{opt}(k+N) \end{pmatrix} & A_{ref}^{opt}(k) &= \begin{pmatrix} a_{ref}^{opt}(k) \\ \vdots \\ a_{ref}^{opt}(k+N-1) \end{pmatrix} \\
\Delta Z(k) &= Z(k) - Z^{opt}(k) & \Delta A_{ref}(k) &= A_{ref}(k) - A_{ref}^{opt}(k)
\end{aligned}
$$

In addition we also need a vector for the predicted positions, $P(k)$ = blockdiag([1 0 0 0])$Z(k)$. The block diagonal matrix picks every fourth element of $Z(k)$ since they contain the predicted positions of the cart. Equivalent vectors are produced for the position deviations from the optimal positions, $\Delta P(k)$ and for the optimal positions, $P^{opt}(k)$. At each time step, k, the MPC-controller should solve the following optimization problem

$$
\begin{aligned}
&\min_{\Delta A_{ref}(k)} \ \Delta Z(k)^T Q \Delta Z(k) + \Delta A_{ref}(k)^T R \Delta A_{ref}(k) \\
&\text{subject to: } A_{ref}(k) = \Delta A_{ref}(k) + A_{ref}^{opt}(k) \leq \mathbf{7} \\
&\qquad\qquad\quad A_{ref}(k) = \Delta A_{ref}(k) + A_{ref}^{opt}(k) \geq -\mathbf{7} \\
&\qquad\qquad\quad P(k) = \Delta P(k) + P^{opt}(k) + p_0 \mathbf{1} \leq \mathbf{0.1} \\
&\qquad\qquad\quad P(k) = \Delta P(k) + P^{opt}(k) + p_0 \mathbf{1} \geq -\mathbf{0.9}
\end{aligned}
$$

The Q- and the R-matrices are user chosen relative costs between the different states and the control. The optimization problem is solved at every time instant, k, and the control $a_{ref}(k) = \Delta a_{ref}(k) + a_{ref}^{opt}(k)$ is applied.

ASSIGNMENT 4

Home assignment: Try to understand how the Q and R-matrices affects the control. Discuss how large and small values of R affect the aggresiveness in the control signal. Also discuss how large and small values of Q affect the corresponding state trajectory errors.

At the lab:

- Open `assignment4.m` and choose Q and R-matrices, for the MPC-controller. Also choose sampling time, h, and cost horizon, N. The prediction horizon of the MPC-controller is Nh. This should be at least 0.7 s to get good predictions of the pendulum behaviour. The computation time for the MPC-controller, found in the scope `execTime` in the Simulink-model, must be less than $h/2$ s. Then choose p_0 and a_{max}. Simulate the system with different initial conditions on the pendulum. Recalibrate Q, R, h and N until you are satisfied. Remember to examine the control signal as well as the controlled signals, since we have limited actuation. Use the script `plot_pend.m` to plot or animate the resulting pendulum movements. Edit the file to specify animation speed.
- When you are satisfied with your design, try your controller on the real process. Run the system with and without the MPC feedback. Also run the system with different initial conditions on the pendulum.

3.4 Time-optimal control with obstacle avoidance

In the final part of the lab, we will try to reach the origin in a time-optimal manner while avoiding a pendulum obstacle. Figure 4 shows how we want the pendulum to move. The pendulum should start at rest at $p = -0.8$ m. The goal is to reach $p = 0$ m with the constraint that the end point of the pendulum must not enter the ellipse. When at $p = 0$ the pendulum should be at rest, meaning that all states should be zero. This movement should be performed in as short time as possible. The control signal, *a_ref*, is constrained to be between -5 and 5 m/s^2, its derivative, *a_ref_dot*, is constrained to be between -100 and 100 m/s^3 in the optimization. Since the track is limited we also need the cart position to satisfy $-0.9 \leq p \leq 0.1$.

This maneuver will require large pendulum movements, so the system will no longer be described by the linearized model with sufficient accuracy. We will thus base our control design on the nonlinear model (2). Deriving the optimal open-loop control trajectory analytically when considering the nonlinear dynamics and nonconvex obstacle constraint is not feasible. There are many tools that can solve such nonlinear optimization problems. The dedicated optimization software JModelica.org[2], is one such tool (and has been used in the lab before), which can give us optimal open-loop trajectories. However, its linux support is lacking, so in this lab we will be using the package `NLOptoControl.jl` [3] in the open source programming language Julia[4] instead. We will then use the same linear MPC framework as before to

[2]For more information about the open-source software JModelica.org, visit `http://www.jmodelica.org` .

[3]The package can be found at `https://github.com/JuliaMPC/NLOptControl.jl`

[4]Julia is a new programming language, designed for scientific computing. The syntax is similar to MATLAB, but the language is much more powerful, see `https://julialang.org/`.

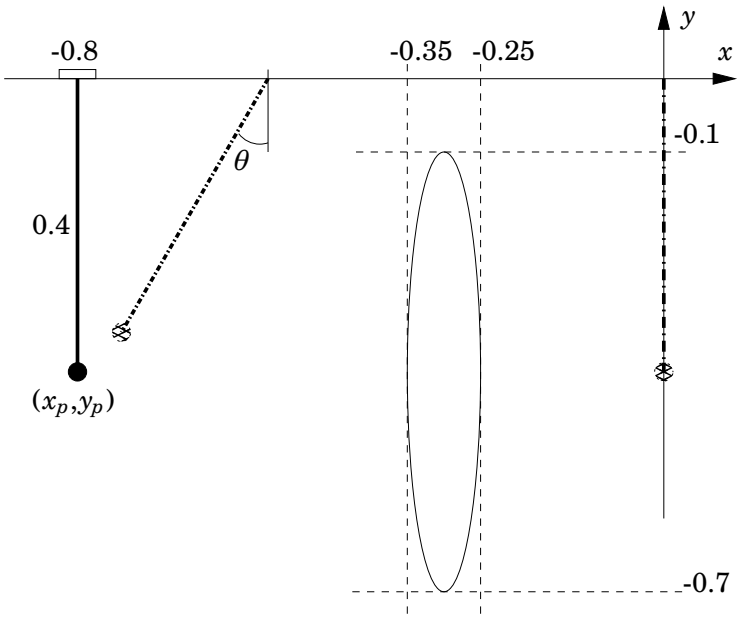


Figure 4 Constraints on the pendulum

follow these trajectories.

`NLOptControl.jl` allows us to specify the dynamics and constraints of the system, it will then discretize the system, formulate an optimization problem and call `Ipopt`[5] to solve the problem.

The model of the nonlinear pendulum on cart, (2), and the optimization problem are specified in the Julia file `pendulum.jl`. The states in the model are the cart position, which is p, the cart velocity, *pd*, the pendulum angle, *theta*, and the pendulum angle velocity, *thetad*. To be able to constrain the derivative of the control signal a_{ref}, we let a_{ref} be another state, *ar*, and consider its derivative, *ard*, to be the input.

[5]Ipopt is an open source software package for large-scale and nonlinear optimization, see `https://projects.coin-or.org/Ipopt`

ASSIGNMENT 5

Home assignment: State the above specified optimization problem mathematically.

At the lab:

- Open the directory which you downloaded from the course homepage in the beginning of the lab by typing
  ```
  > cd /path/to/lab-directory/
  ```
  where you have to type your specific directory.
- Initialize the julia environment
  ```
  > ./setupjulia
  ```
- In the julia terminal, add the nessesary packages by running
  ```
  julia> include("setup.jl")
  ```
  This will install all the necessary packages, precompile them and open a Julia environment in the terminal, where Julia commands and scripts can be executed. Then, open the Julia file `pendulum.jl` with any text editor and specify the missing constraints and model information from the home assignment. The initial conditions are already given in this file. The file `example_problem.jl` contains a simple example of a double integerator as reference, if the syntax for defining the model seems confusing.
- In the Julia terminal, run
  ```
  julia> include("pendulum.jl")
  ```
  This runs the script you were editing and numerically solves the optimization problem. This might take a bit longer the first time you run it. The solver has to be provided with an initial guess. This is needed because the optimization problem is rather tough and some "guidance" is needed. The initial guess file, `initial_guess.csv`, contains a feasible, but not optimal, solution to the same problem. The initial guess file is created by dividing the problem into two separate optimization problems. The first starts from the beginning and ends when the pendulum is just above the obstacle. The second starts where the first ends and ends at the terminal constraints. (The code for generating the initial guess is in `calculate_initial_guess.jl`, but this has already been run in advance of the lab). The final optimization result is found in the file `pendulum_result.csv`.
- There are three functions provided for inspecting the results
  ```
  julia> plotAll(prob)
  julia> plot_trajectory()
  julia> create_animation()
  ```
  The first plots all the states, the second plots the trajectory and obstacle, and the third will generate an animation.
- Open `assignment5.m` and respecify your MPC parameters from Assignment 4. Run `assignment5.m`, then simulate the system using `cart_pend.mdl`. Simulate your system both in open and closed loop. Redesign your MPC controller if you are not satisfied with the result. The script `plot_optim` plots the pendulum trajectory and the elliptic constraint.
- Test your design on the real process.