General Problem Setting

The *Pendulum-Cart* system defines a classic problem in dynamics and control theory and is widely used as a benchmark for testing control algorithms. A simplified variation of this system is illustrated in Figure 2 where a mathematical pendulum is connected to a pair of only-rolling wheels with one translation degree of freedom. This models the *Wheeled Inverted Pendulum* system. An example for that is the Robot *Suricate* (Figure 1) built by the students at LRS / University of Kaiserslautern.

In the simplified model, the point mass m_p is connected to the cart with the mass m_c via a massless arm with the length L. q_1 is the displacement of the cart and q_2 is the angle of the pendulum. As input, u is assumed to be a force.

Abbildung 1: Suricate (JIM2)

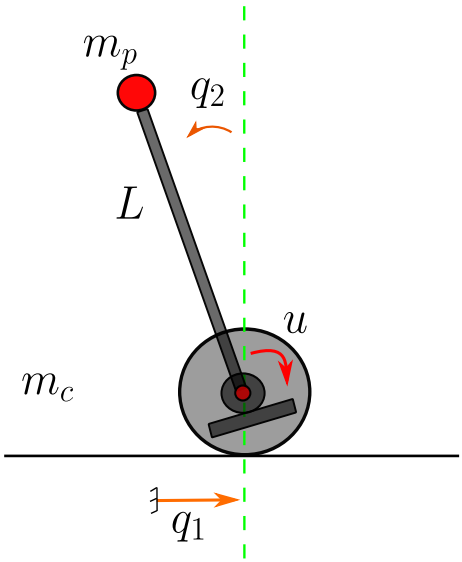

Abbildung 2: Pendulum-Cart System

In our exercises, we will handle the stabilization problem around the upper position of the pendulum. We will only consider the linearised model. The overall objective will be broken down into the following tasks:

- Creating the dynamical system using MATLAB.
- System Analysis, controller design using MATLAB tools and simulation.
- Design of state-space and optimal controllers and simulation.
- Design of state observers and investigating the closed loop including an observer.
- Time-discrete controller and observer design.

The results will be verified by simulation using SIMULINK.

Dynamic Model of the System

The equations of motion for the described system can be derived using different methods such as Lagrange-Formalism or Newton-Euler. The result is in any case a nonlinear system in the following form:

$$\boldsymbol{M}(\boldsymbol{q})\,\ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{g}_q\,\boldsymbol{u}$$

with

$$\boldsymbol{M}(\boldsymbol{q}) = \begin{pmatrix} m_c + m_p & -L\,m_p\,\cos q_2 \\ -L\,m_p\,\cos q_2 & L^2\,m_p \end{pmatrix}$$

$$\boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \begin{pmatrix} L\,m_p\,\dot{q}_2^2\,\sin q_2 \\ -L\,g\,m_p\,\sin q_2 \end{pmatrix} + \begin{pmatrix} d_1\,\dot{q}_1 \\ d_2\,\dot{q}_2 \end{pmatrix}\ ,\ \boldsymbol{g}_q = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

where d_1 and d_2 are the so-called *damping factors* representing friction in the cart displacement and the joint respectively.

A state-space representation can be derived using the equations above defining $\boldsymbol{x} = \begin{pmatrix} \boldsymbol{q} \\ \dot{\boldsymbol{q}} \end{pmatrix}$

$$\dot{\boldsymbol{x}} = \boldsymbol{F}(\boldsymbol{x}) + \boldsymbol{G}(\boldsymbol{x})\,u = \begin{pmatrix} \dot{q}_1 \\ \dot{q}_2 \\ -\boldsymbol{M}^{-1}(\boldsymbol{q})\,\boldsymbol{h}(\boldsymbol{q},\dot{\boldsymbol{q}}) \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \boldsymbol{M}^{-1}(\boldsymbol{q})\,\boldsymbol{g}_q \end{pmatrix} u$$

Linearising about the operation point $q_1^* = q_2^* = 0$ yields the linear state-space equation for the system

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\,\boldsymbol{x} + \boldsymbol{B}\,u \tag{1}$$

with

$$\boldsymbol{A} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \dfrac{g\,m_p}{m_c} & -\dfrac{d_1}{m_c} & -\dfrac{d_2}{L\,m_c} \\ 0 & \dfrac{g\,(m_c+m_p)}{L\,m_c} & -\dfrac{d_1}{L\,m_c} & -\dfrac{d_2\,(m_c+m_p)}{L^2\,m_c\,m_p} \end{pmatrix},\ \boldsymbol{B} = \begin{pmatrix} 0 \\ 0 \\ \dfrac{1}{m_c} \\ \dfrac{1}{L\,m_c} \end{pmatrix}.$$

We the following values:

$$m_c = 1.5,\ m_p = 0.5,\ g = 9.82,\ L = 1,\ d_1 = d_2 = 0.01$$

Tarefa 1

Na primeira tarefa, analisamos o sistema e comparamos as suas diferentes representações.

1. Dentro de um script MATLAB, construa o modelo de espaço de estado do sistema usando as matrizes A, B, C e D pelo comando *ss* do Matlab. Suponha o ângulo q2 como sendo a saída do sistema.

2. Use comandos do Matlab como *pole, eig e zero* para analisar o sistema. O sistema é estável? Aproxime para ZERO os polos e zeros que são muito pequenos (tendendo a zero), e obtenha a nova função de trasferência. O que você constatou agora? Por que houve cancelamento de zeros e polos?

3. Converta o sistema de "espaço de estados" para"função de transferência" usando o comando em Matlab *ss2tf*. Construa um segundo sistema, idêntico a este exceto pela saída que será agora a posição q1, ao invés do ângulo q2 (ou seja, mudando a matriz C). Obtenha também a função de transferência desse segundo sistema usando *ss2tf*.

4.Compare os dois sistemas (o de saída q1 e o de saída q2), em termos de polos e zeros. Eles são iguais? Se houver diferença, como você explicaria?

Tarefa 2

Nesta tarefa, consideramos os resultados da tarefa 1 para continuar com a análise do sistema e um primeiro projeto de controlador. Considere as matrizes do sistema A, B, C e D do sistema de espaço de estados que tem como saída q2.

1. Crie um modelo em SIMULINK e simule a dinâmica do sistema linearizado usando a representação em blocos A,B,C,D.

2. Use o comando *rlocus* para traçar o diagrama do lugar das raízes do sistema (aqui vc deve usar a função de transferência do sistema). A partir do plot to lugar das raízes, você acha que um controlador proporcional simples "P"consegue estabilizar a malha fechada? Por quê?

3. Abra a ferramenta SISO-TOOL do Matlab com o comando *sisotool*. Tente encontrar um controlador C(s) para estabilizar o sistema. Este controlador deve ser uma função de

transferência com dois zeros e dois polos. Use um integrador, um pólo real e um par de zeros complexos. Se preferir, construa a função de transferência desse controlador C(s), multiplique-a pela função de transferência da planta P(s), e trace o lugar das raízes da malha aberta C(s).P(s), para achar um ganho estabilizante (usando a função *rlocfind*). Exporte o controlador para a área de trabalho para usá-lo na simulação em Simulink.

4. Adicione o controlador projetado ao seu modelo linearizado em SIMULINK e simule. Considere o ângulo inicial do pêndulo q2(0) = 10◦ (verifique se o modelo pede em graus ou em radianos!). Veja se a saída q2 converge para zero como desejado. Todos os estados também convergem para zero? Se você simular a malha fechada por um tempo suficiente (por exemplo, 20 s ou mais), parece que algo está dando errado... Como você explicaria isso? O que deve-se mudar para evitar este problema?

Tarefa 3

No controlador projetado na última tarefa, detectamos um problema com a estabilidade total do sistema. Embora o mapeamento de entrada-saída do sistema parecesse estar estabilizado usando um controlador adequado, e embora o ângulo do pêndulo tenha convergido para zero, a posição do carrinho (q1) porém divergiu! Nesta tarefa, vamos investigar o problema, apresentando uma solução.

1. Verifique a Controlabilidade e a Observabilidade do sistema do pêndulo das tarefas anteriores (tanto pro modelo com saída q1, como pro modelo com saída q2). Que diferença você observa quando a posição do carrinho q1 é a saída?

2.Para o modelo com saída q1, projete um controlador de realimentação de estados K para alocar os pólos da malha fechada para posições desejadas (escolha os 4 polos). Investigue a resposta ao degrau da malha fechada no modelo SIMULINK linearizado.

3. Altere a posição dos pólos de malha fechada desejados e repita o procedimento da anterior subtarefa.

4. Projete um controlador LQR ideal usando *lqr* e repita a subtarefa anterior.

5. Modifique o LQR (mudando as matrizes Q e R) e compare o efeito disso no comportamento do sistema. Observe como varia a entrada de controle (u) em cada caso.

6. Teste seu controlador no sistema não linear ”real” fornecido em arquivo Simulink (modelo pêndulo completo). Para qual faixa de ângulos iniciais q2(0) o controlador ainda consegue estabilizar o sistema? O que poderia fazer o controlador funcionar para um ângulo maior (ou mesmo para qualquer valor de ângulo inicial)?

Tarefa 4

O controlador anterior assume o conhecimento de todos os estados. Na prática, isso é dificilmente alcançável. Em vez de aplicar vários sensores para medir todos os estados, um observador de estados é frequentemente usado para estimá-los a partir da saída do sistema. Dentro desta tarefa vamos estender o controlador anterior integrando um observador de estados.

1. Projete um observador de estados usando a alocação de pólos. Note que o observador deve ser mais rápido que o controlador para evitar comportamentos indesejados em malha fechada.

2. Teste seu controlador no sistema não linear ”real” fornecido on-line. Tudo funciona corretamente? Se não, qual pode ser o motivo? Neste caso tente ajustar o controlador para tornar a malha fechada estável.