Mec6313 Homework 3

Output feedback control for a P-P robot, designed using linear matrix inequalities

Stefan Bracher

(Version 2)

Presented to: El-Kébir Boukas

November 23, 2007

Index

1. Introduction

Following the two previous homeworks [1, 2], different output feedback controllers are designed for a pick-and-place robot. The gains are determined by solving linear matrix inequalities [3], using Yalmip [3] and Sedumi [4], considering nominal cases (design with no load mass) and uncertain cases, with a variable load-mass.
The performances are then compared to the state feedback controller of the previous homework [2].

2. The system

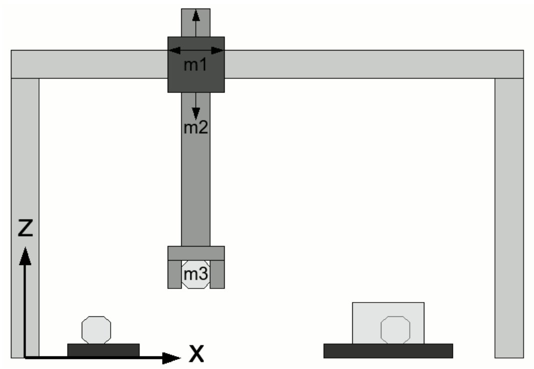

Figure 1: The pick and place robot

The system studied, a serial pick-and-place robot, is the same as in the previous homeworks. For details refer to [1, 2].

It's parameters are:
Nominal mass fist member: $m_1 = 1\,kg$
Nominal mass second member: $m_2 = 1\,kg$
Load: $m_3 = 0 - 0.5\,kg$
Nominal friction coefficients: $k_1 = k_2 = 0.1\,kg/s$

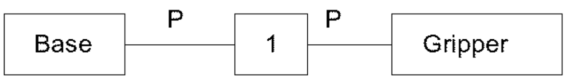

Figure 2: Kinematic chain

Gravity is regarded as a perturbation. The load mass m_3 is added as uncertainty to m_2.

States:	$\boldsymbol{x} = [\xi, x, \dot{x}, \zeta, z, \dot{z}]'$	$\dot{\boldsymbol{x}} = A\,\boldsymbol{x} + B\,\boldsymbol{u}$
Inputs:	$\boldsymbol{u} = [F_x, F_z]'$	$\boldsymbol{y} = C\,\boldsymbol{x} + D$
Output:	$\boldsymbol{y} = [\xi, x, \dot{x}, \zeta, z, \dot{z}]'$	

with

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \frac{-k_1}{(m_1+m_2)} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & \frac{-k_2}{m_2} \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{1}{(m_1+m_2)} & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & \frac{1}{m_2} \end{bmatrix}, \quad C = \text{To be determined}, \quad D = 0 \tag{2.1}$$

3. Static output feedback controller

In the previous homework, a state feedback controller was designed. To do this, it was assumed that the integrals ξ and ζ of both positions are known (And the matrix C is an identity matrix). Here the case is considered, where this is not so and only the positions and speeds can be measured.

The matrix C of the system is: $C=\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$

(3.1)

And the static output feedback controller has the form: $\boldsymbol{u}=K\,\boldsymbol{y}=K\,C\,\boldsymbol{x}$ [3]

(3.2)

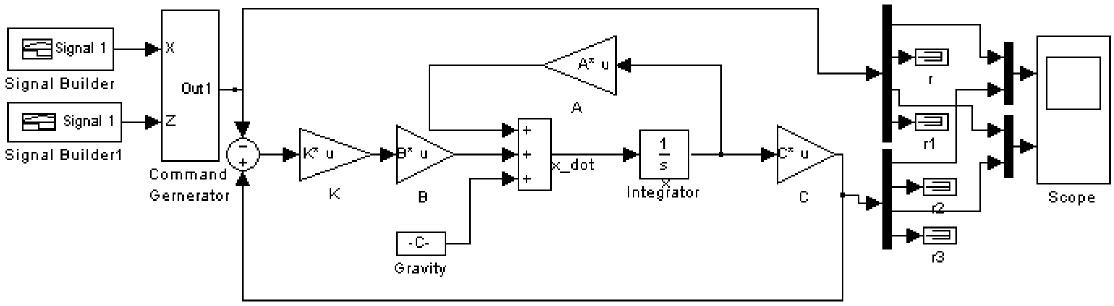

Figure 3: Simulink model of the static output feedback control

3.1 Nominal case

The linear matrix inequality (LMI) stability theorem for the nominal system $\dot{\boldsymbol{x}}=(A+BKC)\,\boldsymbol{x}$ is [3]:

LMI stability theorem for nominal systems with output feedback controller [3]
There exists a static output feedback controller that stabilizes the nominal system if there exist symmetric and positive-definite matrices X > 0 and Y > 0 and Z that satisfy the following LMIs: • $X>0$ • $YC=CX$ • $XA^T+AX+BZC+C^TZ^TB^T<0$ The controller gain is given by $K=ZY^{-1}$

Solving these LMI's using Yalmip [4] and Sedumi [5], gives:

$$K=1.0\text{e+}007*\begin{bmatrix} -2.3636 & -0.0001 & 0.0000 & 0.0000 \\ 0.0000 & 0.0000 & -1.1706 & -0.0000 \end{bmatrix}$$

(3.1.1)

This gain is very high and thus it is very unrealistic to achieve in a physical system, as it will demand forces impossible to produce by real motors.

Nominal Case with reduced gains

To reduce the computed gains, we can, on purpose, reduce the stability of the system by subtracting $\Delta A = -0.0001 * I$ from A. The simulation will then show what the effect this has.

To computed K in this case is:

$$K = \begin{bmatrix} -493.4954 & -12.5369 & 0.0000 & 0.0000 \\ -0.0000 & -0.0000 & -246.7391 & -6.2170 \end{bmatrix} \tag{3.1.2}$$

Simulation results

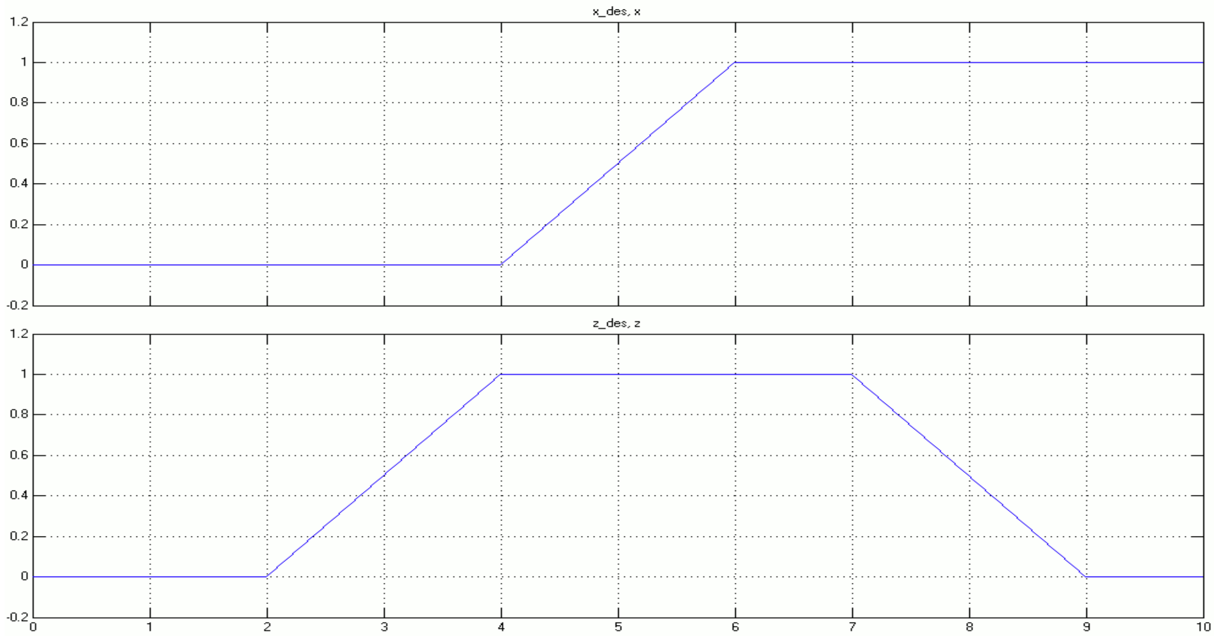

Figure 4: Simulink simulation result for static output feedback using K (3.1.1).The commanded trajectory (red) and the real positions (blue) are coincident.

The positions for the gain (3.1.1) [see Figure 4] are coincident with the commanded positions x_des and z_des. This is not surprising with such a high gains, but merely a theoretical result, as, as said before, physically this will demand forces not produceable.

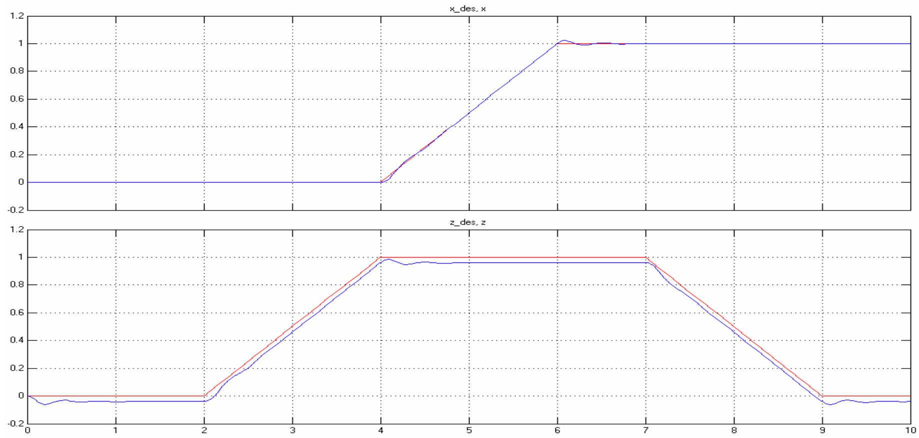

Figure 5: Simulink simulation result for static output feedback using K (3.1.2).Commanded trajectory (red), Real positions (blue).

Figure 5 shows that the result for the controller that was chosen to be "less stable" (3.1.2). The controller is still able able to command the x state, while in z direction with the continuous perturbation of the gravity, a static offset is resulting. This is not surprising as the controller, not knowing the integral of z, does not contain any integral part. This was the same in (3.1.1, but as the gains are so much higher, the static offset is not visible.

3.2 Uncertain case with load mass

The LMI theorem for norm-bounded uncertainties is:

Output feedback controller LMI stability theorem for systems with norm-bounded uncertainties [3]
There exists a static output feedback controller that stabilizes the uncertain system if there exist symmetric and positive-definite matrices X > 0 and Y > 0 and Z that satisfy the the following LMIs for all admissible uncertainties: • $X>0$ • $YC=CX$ • $\begin{bmatrix} J_u & XE_A^T & C^T Z^T E_B^T \\ E_A X & -\epsilon_A I & 0 \\ E_B Z C & 0 & -\epsilon_B I \end{bmatrix} < 0$ $J_u = XA^T + AX + BZC + C^T Z^T B^T + \epsilon_A D_A D_A^T + \epsilon_B D_B D_B^T$ The controller gain is given by $K = ZY^{-1}$

The uncertainties, due to the load mass, are the same as in [2]:

$$A=\begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -k_1*(\frac{1}{(m_1+m_2)}+\Delta_1) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & -k_2*(\frac{1}{m_2}+\Delta_2) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & \frac{-k_1}{(m_1+m_2)} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & \frac{-k_2}{m_2} \end{bmatrix} + D_A F_A E_A$$

with $D_A=\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{1}{(m_1+m_2+m_{3max})}-\frac{1}{(m_1+m_2)} & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & \frac{1}{(m_2+m_{3mac})}-\frac{1}{m_2} \end{bmatrix}$, $F_A(1,1)=[\,?\,]$, $E_A=\begin{bmatrix} 0 & 0 & -k_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -k_2 \end{bmatrix}$

(3.2.1)

$$B=\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{1}{(m_1+m_2)}+\Delta_3 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & \frac{1}{m_2}+\Delta_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{1}{(m_1+m_2)} & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & \frac{1}{m_2} \end{bmatrix} + D_B * F_B * E_B \quad \text{with} \quad D_B=\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \frac{1}{(m_1+m_2+m_{3max})}-\frac{1}{(m_1+m_2)} & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & \frac{1}{(m_2+m_{3mac})}-\frac{1}{m_2} \end{bmatrix},$$

$F_B(1,1)=[\,?\,]$, $E_B=\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

(3.2.2)

As in the nominal case, to get reasonable gains, the system has to be made a bit less stable by adding $\Delta A = -0.0001 * I$ to the system matrix A. This gives a gain K of:

$$K = \begin{bmatrix} -103.0739 & -21.7040 & -0.0000 & -0.0000 \\ -0.0000 & -0.0000 & -100.7948 & -13.2282 \end{bmatrix} \tag{3.2.3}$$

Simulation results

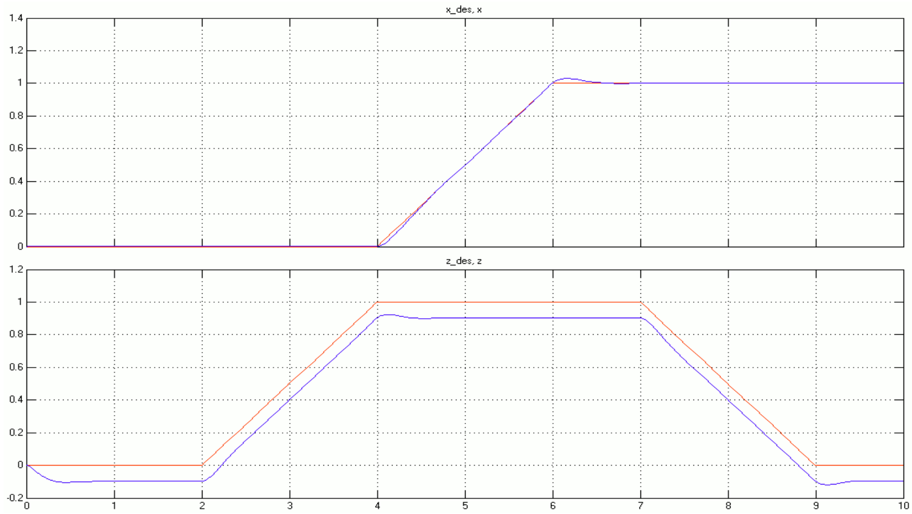

Figure 6: Simulink simulation result for static output feedback using K (3.1.3) and no load Commanded trajectory (red) and real positions (blue).

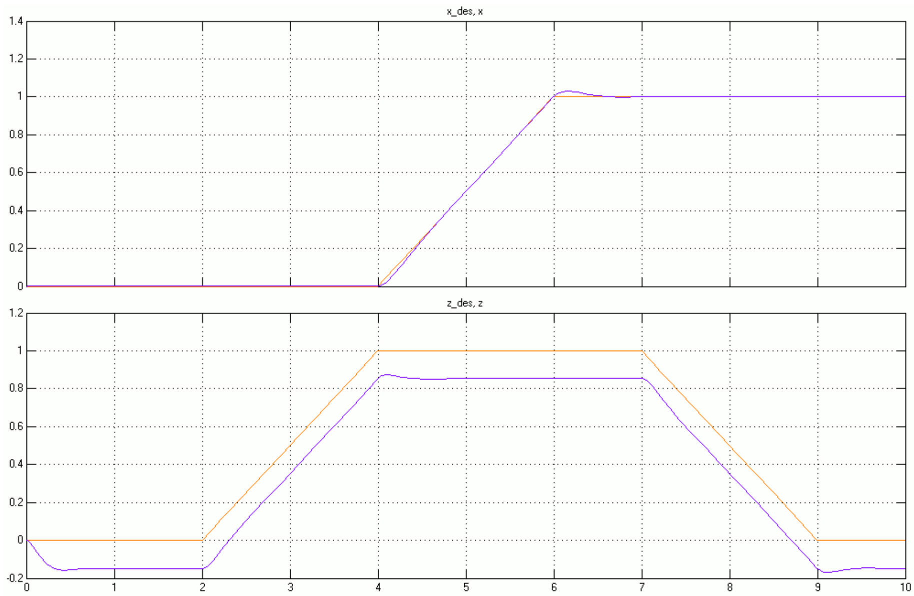

Figure 7: Simulink simulation result for static output feedback using K (3.1.3) and 0.5kg load Commanded trajectory (red) and real positions (blue).

Results are similar to the nominal case (Section 3.1) and robustness for the whole range of admissible loads (0-0.5kg) is given. Again a static offset is resulting because the controller without integral part, is not able to compensate for static gravity effect. The offset is of course bigger with the load as without the load, as the produced force is bigger.

4. Observer-based output controller

A possible solution to the problems related to static output feedback is to do an observer-based output feedback controller of the form [3]:

$$\dot{\hat{\boldsymbol{x}}}=A_{obs}\hat{\boldsymbol{x}}+B_{obs}\hat{u}+L(C_{obs}\hat{\boldsymbol{x}}-\boldsymbol{y})$$
$$\boldsymbol{u}=K\hat{\boldsymbol{x}}$$

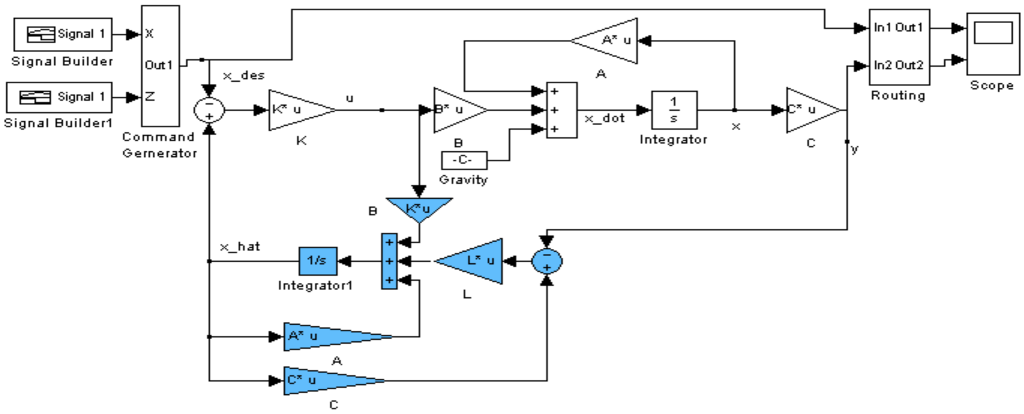

Figure 8: Simulink model of the observer-based output feedback control

(4.1)

4.1 LMI Approach

The matrix inequality theorem (LMI) to find the observer gain L as well as the controller gain K that stabilize the system is:

LMI stability theorem for nominal systems with observer-based output feedback stabilization [3]
If there exist symmetric and positive matrices X>0 and Q>O and matrices Y_C and Y_O such that the following LMIs hold 1. $AX+XA^T+BY_C+Y_C^TB^T+B*B^T<0$ 2. $\begin{bmatrix} A^TQ+QA+Y_OC+C^TY_O^T & K^T \\ K & -I \end{bmatrix}<0$ Then the controller gains that stabilize the system are given by: $K=Y_CX^{-1}$ $L=Q^{-1}Y_O$ The first LMI has to be solved in order to obtain the controller gain K, which is then used to find the observer gain L.

Trying to solve the second LMI using Yalmip [4] and Sedumi [5] however fails*. This does not mean that there are no gains L that stabilize the system, only that the LMI for the system in question is too restrictive or that the algorithms uses are not adequate.

*See observer_output_feedback_nominal.m

4.2 Scientific guess approach

In fact it is quite easy to find some gains K and L thatstabilize the system by scientific guessing.

As seen in figure 9, the system can be stabilized with the following gains::

$$K = \begin{bmatrix} -640.8073 & -259.3156 & -31.2388 & 0 & 0 & 0 \\ 0 & 0 & 0 & -324.2877 & -131.2287 & -15.7623 \end{bmatrix}$$

(4.2.1)

$$L = \begin{bmatrix} -10 & 0 & 0 & 0 \\ -200 & 0 & 0 & 0 \\ 0 & -100 & 0 & 0 \\ 0 & 0 & -10 & 0 \\ 0 & 0 & -200 & 0 \\ 0 & 0 & 0 & -100 \end{bmatrix}$$

(4.2.2)

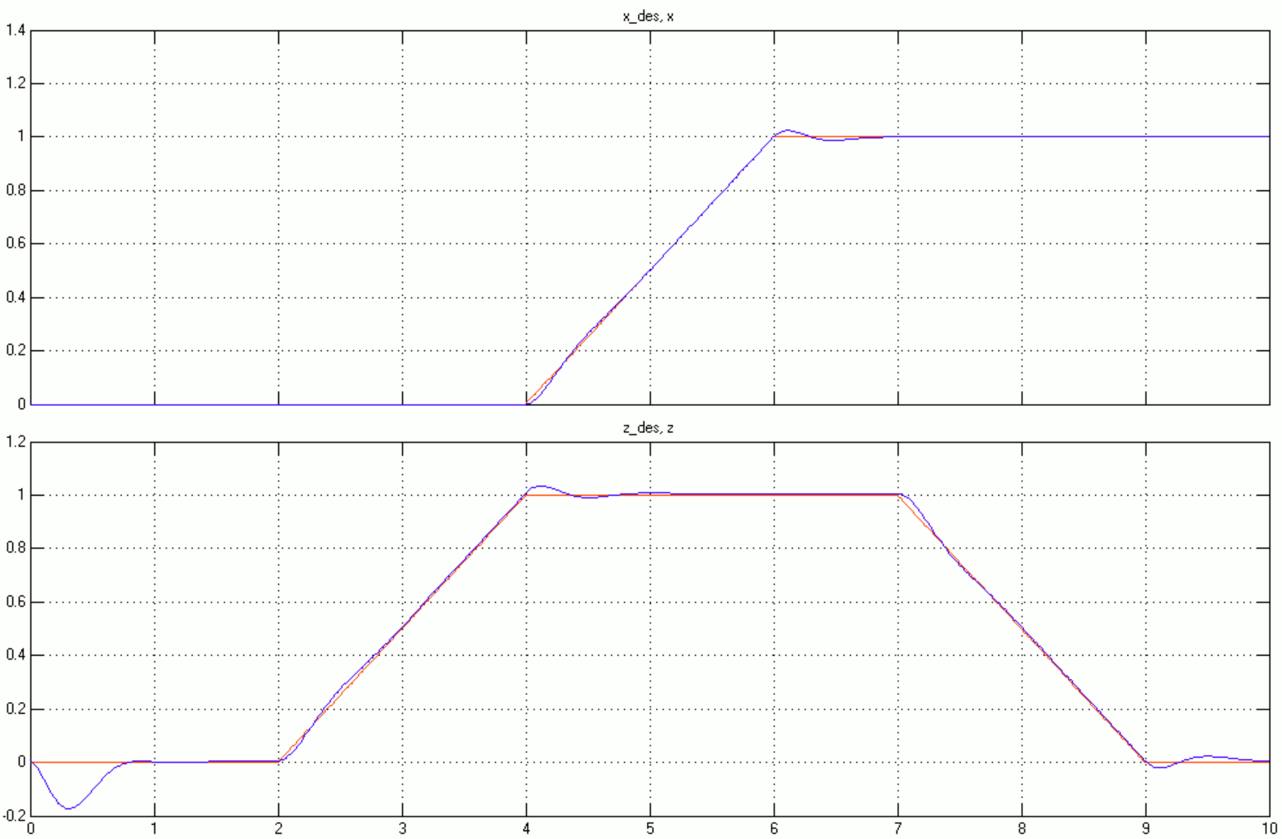

Figure 9: Simulink simulation result for observer-based output feedback using K (4.2.1), L (4.2.2) and 0.5kg load. Commanded trajectory (red) and real positions (blue).

With the observer in place, there is no static offset as the integral of the positions is known. For t<1s, the z-position does not follow the desired trajectory as the gravity makes a sudden jump at the begin of the simulation from zero to 9.81m/s². The controller first has to saturate. Once this is done and gravity is kept constant, the real positions follow the desired ones. It has to be noted that such a temporary error will result whenever the load mass of the robot is changed.

5. Performance comparison

5.1 Previous results for the state feedback controller

In the previous homework [2], a state feedback controller was designed for the same system. Using a K matrix of about the same dimensions*:

$$K=\begin{bmatrix} -422.4529 & -263.3477 & -48.3451 & -0.0000 & -0.0000 & -0.0000 \\ 0.0000 & 0.0000 & 0.0000 & -184.3762 & -115.2124 & -21.1981 \end{bmatrix} \quad (5.1)$$

*the bigger the $\Delta A = a * I$ added to A, the faster is the response speed, but the higher the gains K and thus the demanded forces.

It gives the following response to the same trajectory command as used to test the other controllers:

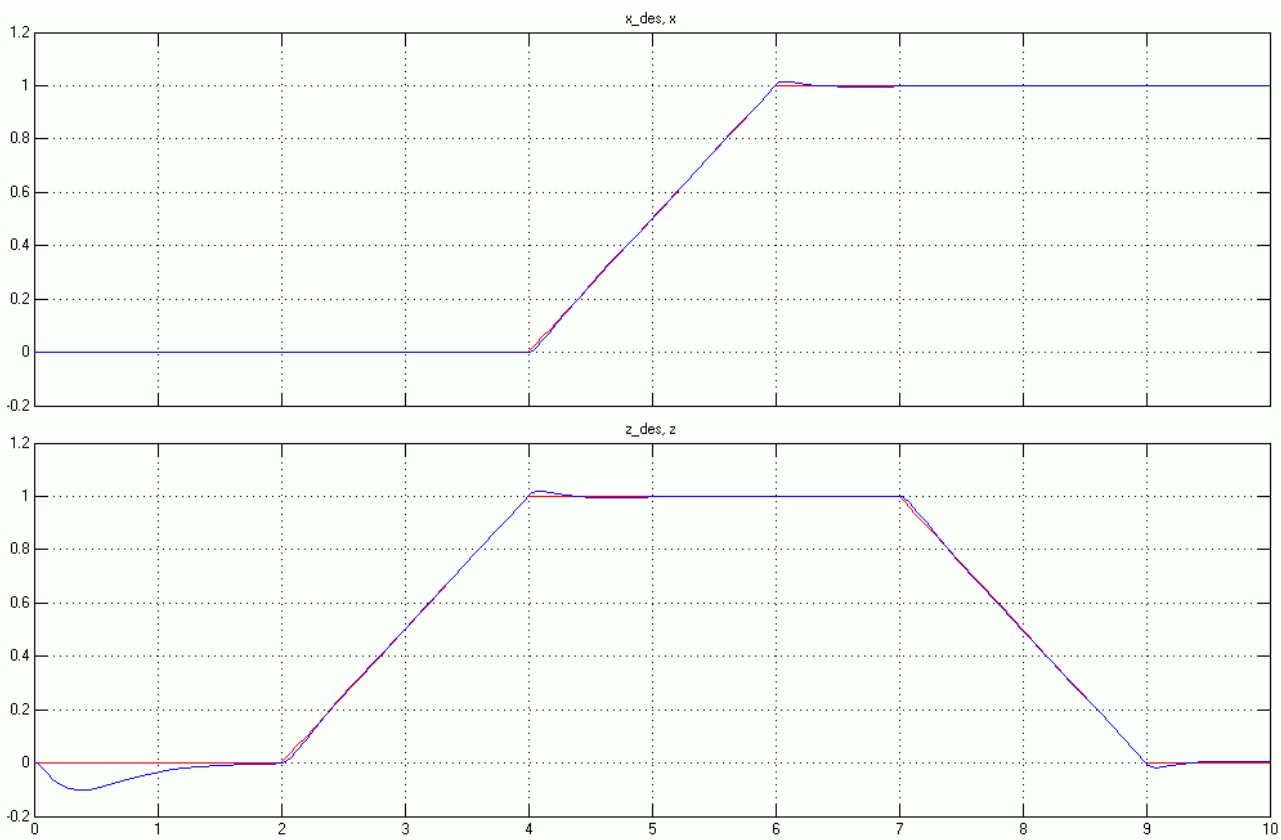

Figure 10: Simulink simulation result for state feedback using K (5.1) and 0.5kg load Commanded trajectory (red) and real positions (blue).

As in figure 10, the controller needs some time to correct the error introduced by the gravity at t=0. But once gravity is compensated, the control works.

5.2 Static output feedback vs. state feedback

The main advantage of the static output feedback is, that it needs less information about the system than the full state feedback. It is able to control a system that does not undergo any static perturbations (see figures 5, 6 and 7, upper part for x), but produces an offset if such a perturbation, like gravity, is present (see figures 5, 6 and 7, under part for z).

The offset can be reduced by increasing the gains (by choosing $A = A + a*I$, $a \geqslant 0$), but for real systems with limited actuator power, there are limits to do this.

5.3 Observer based output feedback vs. static output feedback

A solution to this static offset problem is to introduce an observer, that computes the estimated states $\hat{\boldsymbol{x}}$. The control part of the system however becomes more complex and more difficult to adjust (so did the LMI approach not work for the system in question).

5.3 Observer-based output feedback vs. state feedback

Using an observer-based output feedback instead of state-feedback, reduces the number of captors necessary while the performances of the two controllers are comparable (Figure 9 and 10). In our example this means that the integral of the positions has to be given to the controller, but is computed in the observer part of the controller itself.
The cost of the additional computation power needed is nowadays smaller than the cost saved by the captor reduction. Further on with every eliminated captor, also the error risk due to failure of physical parts is reduced.

6. Conclusion

The gains of the static output feedback control where successfully computed using the linear matrix inequality theorem. The approach however can not overcome the main weakness of the static output feedback, to have permanent offsets if the affected states are not stable in open loop condition. For the pick and place robot such an offset is unacceptable and a observer-based output feedback has to be used in case that not all states are accessible.
For the observer-based output feedback, gains that stabilize the system could be found by scientific guessing, while the LMI approach was not able to find them.

References

[1] Bracher Stefan,“State feedback controller and observer design for a pick and place robot”, Homeworks MEC6313, Ecole Polytechnique de Montréal, 2007

[2] Bracher Stefan,“State feedback controller design for a system with uncertainties using linear matrix inequalities”, Homeworks MEC6313, Ecole Polytechnique de Montréal, 2007

[3] Boukas El-Kebir*, “Class Notes MEC6313“*, Ecole Polytechnique de Montréal, 2007

[4] Löfberg J.*, “YALMIP : A Toolbox for Modeling and Optimization in MATLAB.*” In Proceedings of the CACSD Conference, Taipei, Taiwan, 2004.

[5] Sedumi: http://sedumi.mcmaster.ca

Appendix

static_output_feedback_nominal.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Script:       MEC 6313 Homework 3, static_output_feedback_nominal.m
% Description:  Finds the gains K to stabilize the system dx=(A+BKC)x+Bu
% Author:       Stefan Bracher
% Requirements: Yalmip (http://control.ee.ethz.ch/~joloef/yalmip.php)
%               Sedumi (http://sedumi.mcmaster.ca)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% Some cleaning up %%%
clear all;
clc;
yalmip('clear');

%%% Parameters %%%
m1=1;       % Mass of the first moving member, including motor
m2=1;       % Mass of the second moving member
k1=0.1;     % Friction coefficient member 1
k2=0.1;     % Friction coefficient member 2
speed=-0.0001; % Additional(>0)/Reduction(<0) Reacion Speed (Deplacement of A matrix)
%speed=0; % Additional(>0)/Reduction(<0) Reacion Speed (Deplacement of A matrix)

%%% System %%%
A=[ 0   1    0           0    0    0;
    0   0    1           0    0    0;
    0   0   -k1/(m1+m2)  0    0    0;
    0   0    0           0    1    0;
    0   0    0           0    0    1;
    0   0    0           0    0   -k2/m2];
Almi=A+speed*eye(size(A));
B=[0           0;
   0           0;
   1/(m1+m2)   0;
   0           0;
   0           0;
   0           1/m2];
C=[0 1 0 0 0 0;
   0 0 1 0 0 0;
   0 0 0 0 1 0;
   0 0 0 0 0 1];

%%% LMI Variables %%%
n=size(C, 1);
m=size(C, 2);
l=size(B, 2);
X=sdpvar(m, m, 'symmetric');
Y=sdpvar(n, n, 'symmetric');
Z=sdpvar(l, n, 'full');

%%% LMI %%%
 F=set(X>0);            % X positive definit
 F=F+set(Y>0);          % Y positive definit
 F=F+set(Y*C==C*X);     % YC=CX
 F=F+set(X*Almi'+Almi*X+B*Z*C+C'*Z'*B'<0);
 Sol=solvesdp(F);       % Solving it

%%% Extract Data %%%
X=double(X);
Y=double(Y);
Z=double(Z);
K=Z*inv(Y)
[primal, dual]=checkset(F);   % Get primary and dual constraint residuals

if ((primal(1)>0)&&(primal(2)>0)&&(abs(dual(1))<1)&&(abs(dual(2))<1))
% The system is stable if the primal residuals are positive and the dual
% residuals are small
disp('The system is stable');
else
disp('No information about the stability of the system can be given');
end
```

static_output_feedback_uncertain..m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Script:      MEC 6313 Homework 3, static_output_feedback_uncertain.m
% Description: Finds the gains K to stabilize the uncertain system
%              dx=(A+BKC)x+Bu
% Author:      Stefan Bracher
% Requirements: Yalmip (http://control.ee.ethz.ch/~joloef/yalmip.php)
%              Sedumi (http://sedumi.mcmaster.ca)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% Some cleaning up %%%
clear all;
clc;
yalmip('clear');

%%% Parameters %%%
m1=1;      % Mass of the first moving member, including motor
m2=1;      % Mass of the second moving member
m3max=0.5; % Maximal Load Mass
k1=0.1;    % Friction coefficient member 1
k2=0.1;    % Friction coefficient member 2
speed=-0.0001;   % Additional(>0)/Reduction(<0) Reaction Speed (Deplacement of A matrix)

%%% System %%%
A=[ 0  1   0           0    0    0;
   0  0   1           0    0    0;
   0  0  -k1/(m1+m2)  0    0    0;
   0  0   0           0    1    0;
   0  0   0           0    0    1;
   0  0   0           0    0   -k2/m2];
Almi=A+speed*eye(size(A));
EA=[0 0 -k1 0 0 0;
   0 0 0  0 0 -k2];
DA=[0                           0;
   0                           0;
   1/(m1+m2+m3max)-1/(m1+m2)      0;
   0                           0;
   0                           0;
   0                           1/(m2+m3max)-1/m2];

B=[0          0;
  0          0;
  1/(m1+m2)   0;
  0          0;
  0          0;
  0          1/m2];
EB=eye(2, 2);
DB=DA;

C=[0 1 0 0 0 0;
  0 0 1 0 0 0;
  0 0 0 0 1 0;
  0 0 0 0 0 1];

%%% LMI Variables %%%
n=size(C, 1);
m=size(C, 2);
l=size(B, 2);
X=sdpvar(m, m, 'symmetric');
Y=sdpvar(n, n, 'symmetric');
Z=sdpvar(l, n, 'full');
epA=sdpvar(1, 1);
epB=sdpvar(1,1);

%%% LMI %%%
 F=set(X>0);          % X positive definit
 F=F+set(Y>0);          % Y positive definit
 F=F+set(Y*C==C*X);       % YC=CX

 F=F+set([
        X*Almi'+Almi*X+B*Z*C+C'*Z'*B'+epA*DA*DA'+epB*DB*DB'  X*EA'   C'*Z'*EB';
        EA*X  -epA*eye(2,2)   zeros(2, 2);
```

```
        EB*Z*C  zeros(2, 2)     -epB*eye(2,2)
                              ]<0);

Sol=solvesdp(F);       % Solving it

%%% Extract Data %%%
X=double(X);
Y=double(Y);
Z=double(Z);
K=Z*inv(Y)
[primal, dual]=checkset(F);  % Get primary and dual constraint residuals

if ((primal(1)>0)&&(primal(2)>0)&&(abs(dual(1))<1)&&(abs(dual(2))<1))
% The system is stable if the primal residuals are positive and the dual
% residuals are small
disp('The system is stable');
else
disp('No information about the stability of the system can be given');
end
```

observer_output_feedback_nominal.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Script:      MEC 6313 Homework 3, observer_output_feedback_nominal.m
% Description: Finds the gains K and L to stabilize the system
% Author:      Stefan Bracher
% Requirements: Yalmip (http://control.ee.ethz.ch/~joloef/yalmip.php)
%              Sedumi (http://sedumi.mcmaster.ca)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% Some cleaning up %%%
clear all;
clc;
yalmip('clear');

%%% Parameters %%%
m1=1;      % Mass of the first moving member, including motor
m2=1;      % Mass of the second moving member
k1=0.1;    % Friction coefficient member 1
k2=0.1;    % Friction coefficient member 2
speed=3;   % Additional Reacion Speed (Deplacement of A matrix)

%%% System %%%
A=[ 0  1   0          0    0    0;
    0  0   1          0    0    0;
    0  0  -k1/(m1+m2)  0    0    0;
    0  0   0          0    1    0;
    0  0   0          0    0    1;
    0  0   0          0    0   -k2/m2];
Almi=A+speed*eye(size(A));

B=[0          0;
   0          0;
   1/(m1+m2)   0;
   0          0;
   0          0;
   0          1/m2];

C=[0 1 0 0 0 0;
   0 0 1 0 0 0;
   0 0 0 0 1 0;
   0 0 0 0 0 1];

%%% LMI Variables%%%
lineA=size(A, 1);  % Lines of A
colB=size(B, 2);   % Colums of B
X=sdpvar(lineA, lineA, 'symmetric');
Yc=sdpvar(colB, lineA,'full');

%%% LMI Controller %%%
 F=set(X>0);           % X positive definit
 F=F+set(Almi*X+X*Almi'+B*Yc+Yc'*B'+B*B'<0); % Controller
 Sol=solvesdp(F);      % Solving it
```

```
 %%% Extract Data %%%
 X=double(X);
 Yc=double(Yc);
 K=Yc*inv(X);
 [primal, dual]=checkset(F);   % Get primary and dual constraint residuals
if ((primal(1)>0)&&(primal(2)>0)&&(abs(dual(1))<1)&&(abs(dual(2))<1))
% The system is stable if the primal residuals are positive and the dual
% residuals are small
disp('The controller part of the system is stable');
else
disp('No information about the stability of the system can be given');
end

 %%% LMI Variables Observer %%%
 lineC=size(C, 1);    % Colums of C
 Q=sdpvar(lineA, lineA, 'symmetric');
 Yo=sdpvar(lineA, lineC , 'full');

 %% LMI Observer %%
 G=set(Q>0);           % Q positive definit
 G=G+set([Almi'*Q+Q*Almi+Yo*C+C'*Yo'    K';
           K -eye(2, 2)]<0); % Observer
 Sol=solvesdp(G);       % Solving it

 %%% Extract Data %%%
 Q=double(Q)
 Yo=double(Yo)

 %%% Show result %%%
K
L=inv(Q)*Yo
[primal2, dual2]=checkset(G);   % Get primary and dual constraint residuals

if ((primal2(1)>0)&&(primal2(2)>0)&&(abs(dual2(1))<1)&&(abs(dual2(2))<1))
% The system is stable if the primal residuals are positive and the dual
% residuals are small
disp('The system is stable');
else
disp('No information about the stability of the system can be given');
end

checkset(G)
```