ISSN 0280-5316
ISRN LUTFD2/TFRT--5595--SE

Controlling a Ball and Plate Process Using Computer Vision

Martin Rentsch

Department of Automatic Control
Lund Institute of Technology
March 1998

Department of Automatic Control
Lund Institute of Technology
Box 118
S-221 00 Lund Sweden

Document name
Master Thesis

Date of issue
March 1998

Document Number
ISRN LUTFD2/TFRT--5595--SE

Author(s)
Martin Rentsch

Supervisor
Bo Bernhardsson
Karl Åström (Department of Mathematics)
Anders Robertsson

Sponsoring organisation

Title and subtitle
Controlling a Ball and Plate Process using Computer Vision

Abstract

This thesis describes the setup of a Ball and Plate process. The aim is to make the ball roll along a predefined trajectory by tilting the plate appropriately. The position of the ball is sensed by means of a video camera and the plate is actuated by an industrial robot.

An analog video camera acquires pictures to a host computer, where they are digitized. The visual information is processed to give as a result the location of the ball on the plate. Control signals for the two degrees of freedom of the ball are derived and transmitted to the robot system.

The exact mathematical model of the process is linearized and discretized for the design of an observer-based digital controller.

Position control of the robot is implemented on a detached computer system.

Key words
Ball and Plate, Ball and Beam, Computer Vision, Image Processing, Robot

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280–5316

ISBN

Language
English

Number of pages
115

Recipient's notes

Security classification

The report may be ordered from the Department of Automatic Control or borrowed through:
University Library 2, Box 3, S-221 00 Lund, Sweden
Fax +46 46 222 44 22 E-mail ub2@ub2.lu.se

Preface

Towards the end of my education at the Swiss Federal Institute of Technology (ETH) in Zurich, I decided that I wanted to write my Master Thesis in another country. Professor Manfred Morari provided me with valuable contacts and I applied to Professor Karl Johan Åström, who gave me the opportunity to conduct my thesis at his department in Lund, Sweden.

I would like to thank my supervisors Karl Åström, Bo Bernhardsson, Anders Robertsson and Klas Nilsson for their valuable help in overcoming various difficulties, as well as all the staff of the Department of Automatic Control, who made me feel comfortable throughout my stay. Special thanks go to Mario Clerici for his considerable contributions to the physical model.

This report was typeset in LaTeX 2_ε.

Contents

List of Figures

Chapter 1

Introduction

This project is about a ball on a plate, which is slanted in such a way that the ball follows a predefined trajectory. Among the numerous available sensor and actor systems, a video camera was chosen as the sole input device and an industrial robot to handle the plate in two degrees of freedom. A standard Personal Computer served as a platform for the implementation of the controller.

Outline of the Thesis

Chapter 2 presents the acquisition of the image and the analysis of the visual information.

Chapter 3 describes the plant itself. A linearized model is derived from the exact physical model.

Chapter 4 is about the controller. Design aspects are discussed as well as the implementation.

Chapter 5 introduces the robot, how it is programmed and how it is excited by the controller.

Chapter 6 concludes the report and shows possible improvements for future development.

Appendix A contains the derivation of the moment of inertia for both a solid sphere and a spherical shell.

Appendix B defines the unary cross product operator, which allows convenient handling of vector cross products.

Appendix C discusses various smoothened step functions with a steady derivative.

Appendix D is a collection of MATLAB files written at various stages of the project.

Bibliography and Index sections are provided for quick reference.

Chapter 2

Image Processing

2.1 Image Acquisition Equipment

A standard VHS video camera is used to acquire pictures to the host, which is a Personal Computer running the Microsoft Windows NT operating system. It is equipped with a Frame Grabber Interface Card [5]. A device-independent software library [6] makes the features of the dedicated hardware available to the C programming language environment.

An algorithm to find the ball on the plate was developed off-line using MATLAB before it was implemented in C for on-line usage.

2.2 Off-Line Image Analysis

Two fundamentally different approaches have been made to compute the location of the ball on the plate:

Line Search: Based on the knowledge about the position of the ball on previous images, we draw a line in the direction where we assume the ball to be on the current image. An exceptional change in colour along this line indicates the edge of the ball.

Global Search: Provided the colours of the ball and the plate are sufficiently different, we can classify the pixels with regard to their colour. The colour determines whether a particular pixel belongs to the ball or to the plate.

Line Search

An earlier project has dealt with the one-dimensional version of the Ball and Plate system, namely the Ball and Beam system [7]. The Computer Vision methods developed therein have been extended in such a way that they could be applied to a ball with two degrees of freedom instead of just one.

The underlying mathematical algorithm is only briefly summarized here, see [3] for an in-depth discussion. A general reference about Image Processing is [9].

Along a line in the picture, an abrupt change in colour is sought, mathematically speaking, the edge of the ball is at the point of the maximum derivative of the colour along the line. The following operations are applied to the real signal:

- The optical system of the camera blurs the image. It is additionally perturbed by noise.
- In order to be stored in a computer, the continuous signal has to be sampled.
- The now discrete signal is interpolated to produce a continuous function.
- As a preparation for the next step, the function is smoothed, i.e. it is convoluted with a Gaussian function (nonlinear low-pass filter). Depending on the grade of smoothing, more or less computation time is needed. Some tuning is required to determine a suitable grade of smoothing: noise peaks must be removed whereas the actual step of the original signal must be preserved.
- The maximum derivative of the above function indicates the edge of the ball. Higher order derivatives are computed to find this maximum iteratively using the Newton-Raphson method.
- The results allow an estimation of the accuracy of the edge location.

Figure 2.1 shows an example of such an edge detection.

Thanks to the above algorithm, the accuracy with which the location of the ball can be specified is, in fact, higher than the resolution of the image! The key to this result is interpolation.

Our line intersects the circle twice to deliver a chord. The bisector of the latter is a superset of a diameter. The computation of the center of the circle is then straightforward.

Initially, the ball can be anywhere on the plate. Starting in the center of the image, we draw horizontal lines until we hit the ball. On subsequent images, we can reduce the search area to a single line, set by the two previous positions of the ball. Figures 2.2 and 2.3 illustrate the concept. A red ball on a white background was used.

See Appendix D for the MATLAB files. Note that the camera is assumed to focus solely on the plate and not the surroundings, i.e. the algorithm does not attempt to detect the border of the plate.

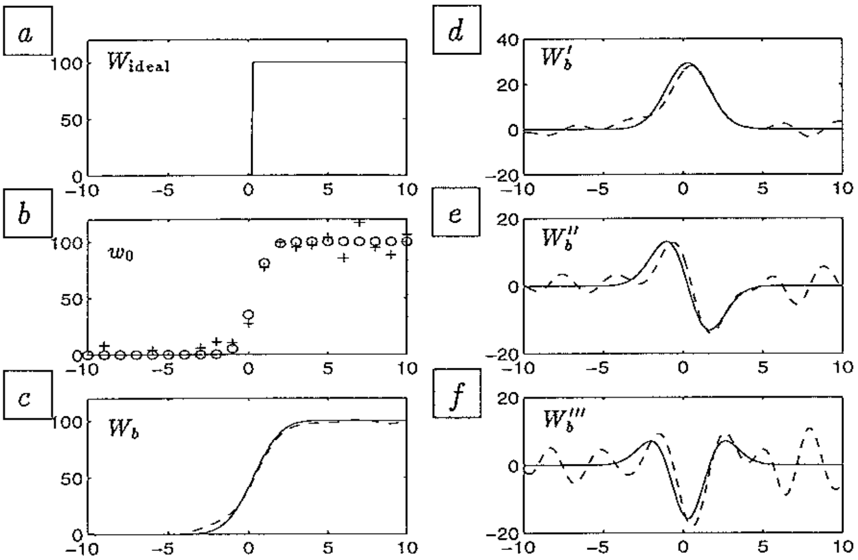
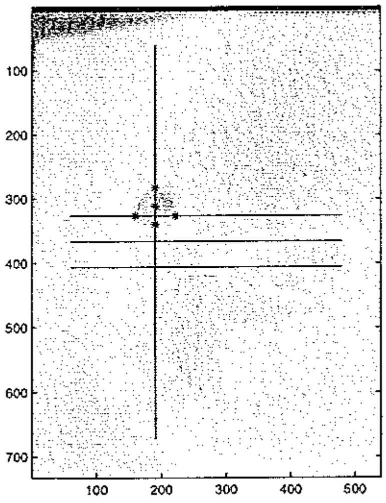


Figure 2.1: Illustration of one-dimensional sub-pixel edge detection using scale-space smoothing. (a) The ideal step edge. (b) The discretized 'image' without noise (o) and with noise (+). (c) The scale-space interpolations. The deterministic signal is shown as a solid line, whereas the perturbed signal is shown as a dashed line. (d–f) The first three derivatives. The sub-pixel edge position is defined as the position of the maximum of the first derivative.



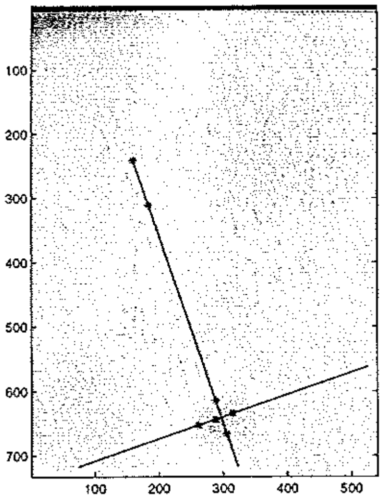


Figure 2.2: Finding the ball on the plate using Line Search.

Figure 2.3: Tracking the ball on the plate using Line Search.

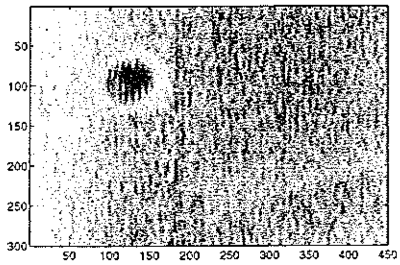


Figure 2.4: An image before the colour thresholding.

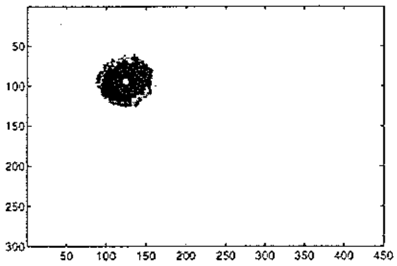


Figure 2.5: Finding the ball on the plate using Global Search.

Global Search

A threshold is chosen experimentally to divide the pixels of the images into two groups: foreground and background, i.e. ball and plate. Taking into account all the pixels which belong to the ball, the following equation yields the center of mass.

$$\vec{v}_{cm} = \frac{1}{N}\sum_{i=1}^{N}\vec{v}_i$$

Figures 2.4 and 2.5 show an example for a red ball on a white background. Later, a white ball on a black plate was chosen to eliminate visible shadows.

While the accuracy of this algorithm is strictly limited to the resolution of the image, it is significantly faster than the Line Search variant.

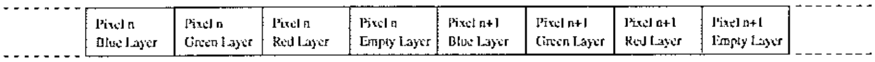
See Appendix D for the MATLAB file.

2.3 On-Line Image Analysis

The sampling rate is a critical issue in real-time systems. It is evident that the sophisticated Line Search algorithm is more time-consuming than Global Search. Moreover, Line Search is slowed down by poor image quality. Unfortunately, our images are so blurred that its application is out of the question. The computation time could be cut down by improving the image quality using other hardware devices, but for the time being, we decided to confine ourselves to Global Search.

The images from the video camera consist of three Layers: red, green and blue. Figure 2.6 shows how the data is stored. For the image analysis, one layer is sufficient. The choice is made depending on the colours of the ball and the plate.

Figure 2.7 shows an extract of the corresponding C code.



Pixel n Blue Layer	Pixel n Green Layer	Pixel n Red Layer	Pixel n Empty Layer	Pixel n+1 Blue Layer	Pixel n+1 Green Layer	Pixel n+1 Red Layer	Pixel n+1 Empty Layer

Figure 2.6: The data representation of an image.

```
const int indentrow=12, indentcol=16;
   // The image is slightly reduced due to a hardware defect.

layer=Green;
xsum=ysum=N=0;
data=hpAcquireBuf;   // pointer to the buffer
data+=indentrow*width*4;
for( row=indentrow; row<height; row++ )
{
    data+=indentcol*4;
    for( col=indentcol; col<width; col++ )
    {
        if( *(data+layer) < threshold )
        {
            xsum+=col; ysum+=row; ++N;
        }
        data+=4;
    }
}
assert(N);
x=(float)xsum/N-indentcol;   // coordinates of the center of gravity ...
y=(float)ysum/N-indentrow;   // ...  of the ball in the reduced image
```

Figure 2.7: Extract of the C code to compute the center of the ball .

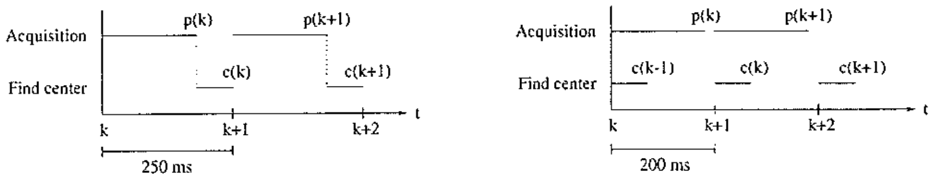


Figure 2.8: Synchronous (left) and asynchronous (right) sampling. An acquired picture is denoted by p, the position of the center of mass by c.

Synchronous versus Asynchronous Acquisition

The task for each sampling period can be roughly divided into two parts: the acquisition of the image and the computation of the center of mass of the ball. The average time for a single image acquisition is 180 ms. The computation of the center takes about 70 ms. Synchronous acquisition means that the location of the center is computed immediately after the picture was acquired. This results in a total sampling period of 250 ms or a sampling frequency of 4 Hz. With asynchronous acquisition, on the other hand, the two jobs are handled concurrently: thanks to the dedicated microprocessor on the Frame Grabber Card and Direct Memory Access (DMA), we can acquire the picture and compute the center at the same time. After the computation of the center, the program waits for the acquisition of the image to complete. The two processes synchronize every 200 ms, which corresponds to a sampling frequency of 5 Hz. The 20 ms the sampling period is longer than the time for the image acquisition alone is needed by the operating system for the management of the concurrent processes. Naturally, we can only compute the center of the ball on the *previous* image during each sampling period. This results in a time delay of approximately 70 ms with regard to the sampling times kT_s, where T_s is the sampling period.

For open-loop experiments, where we only need to record the trajectory of the ball, the asynchronous method benefits from a higher sampling rate with no drawbacks. In the closed-loop case, however, the time delay will result in a state increase of the discrete model (one extra state).

Chapter 3

Model of the Plant

In order to gain some fundamental information about the behaviour of the Ball and Plate system, the simpler case of a Ball and Beam system is introduced first. While the ball has two degrees of freedom on a plate, it only has one degree of freedom on a beam.

3.1 The Ball and Beam System

Consider the experiment depicted in Figure 3.1. The beam is made to rotate in a vertical plane and the ball is free to roll along the beam without rolling friction or air resistance. We require that the ball remain in contact with the beam and that the rolling occur without slipping, which imposes a constraint on the rotational acceleration of the beam.

It is assumed that the actuators are strong enough to apply any allowable torque to the beam and that the motion of the ball has no effect on the beam.

The position of the ball's center of mass C can be expressed in global coordinates $\{x, y\}$, but the ball has only one degree of freedom, so the local

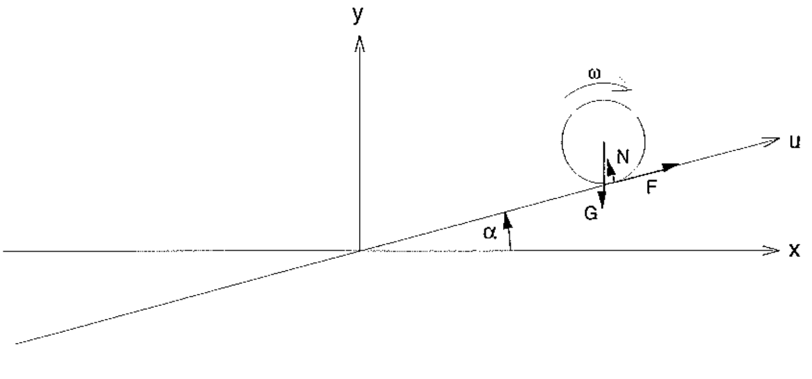


Figure 3.1: The Ball and Beam system.

coordinate $\{u\}$ will be used for the equation of motion. The radius of the ball is denoted by r.

$$\begin{aligned} x &= u\cos\alpha - r\sin\alpha \\ y &= u\sin\alpha + r\cos\alpha \end{aligned}$$

The first and second derivatives of the position represent the velocity and acceleration, respectively.

$$\begin{aligned} \dot{x} &= \dot{u}\cos\alpha - u\dot{\alpha}\sin\alpha - r\dot{\alpha}\cos\alpha \\ \dot{y} &= \dot{u}\sin\alpha + u\dot{\alpha}\cos\alpha - r\dot{\alpha}\sin\alpha \end{aligned} \tag{3.1}$$

$$\begin{aligned} \ddot{x} &= \ddot{u}\cos\alpha - 2\dot{u}\dot{\alpha}\sin\alpha - u\ddot{\alpha}\sin\alpha - u\dot{\alpha}^2\cos\alpha - r\ddot{\alpha}\cos\alpha + r\dot{\alpha}^2\sin\alpha \\ \ddot{y} &= \ddot{u}\sin\alpha + 2\dot{u}\dot{\alpha}\cos\alpha + u\ddot{\alpha}\cos\alpha - u\dot{\alpha}^2\sin\alpha - r\ddot{\alpha}\sin\alpha - r\dot{\alpha}^2\cos\alpha \end{aligned} \tag{3.2}$$

The equation of motion can be derived in different ways: using the Newton-Euler or the Lagrange formalism. Both methods ought to eventually yield the same results.

Newton-Euler Formalism

The conservation law for momentum is valid for both directions x and y. The mass of the ball is denoted by m.

$$\begin{aligned} m\ddot{x} &= F\cos\alpha - N\sin\alpha \\ m\ddot{y} &= F\sin\alpha + N\cos\alpha - G \end{aligned} \tag{3.3}$$

In Equations 3.3, the normal force N can be eliminated.

$$m\ddot{x}\cos\alpha + m\ddot{y}\sin\alpha = F - G\sin\alpha \tag{3.4}$$

From Equations 3.2 we gather

$$\ddot{x}\cos\alpha + \ddot{y}\sin\alpha = \ddot{u} - u\dot{\alpha}^2 - r\ddot{\alpha}. \tag{3.5}$$

The conservation of angular momentum is

$$-J\dot{\omega} = rF, \tag{3.6}$$

where r denotes the radius of the ball and ω its angular velocity in the inertial system $\{x, y\}$. See Appendix A for the moment of inertia J with respect to a rotational axis through the center of the ball.

The fact that the ball does not slide, but roll on the beam, is expressed in a constraint equation, which relates the angular velocity ω with the velocity $\dot{u}$. If we denote by ω_{rel} the angular velocity of the ball relative to the beam, the velocity of the ball's center of mass C can be expressed in two ways to find an expression for ω_{rel}.

$$\left.\begin{aligned} v_C &= \omega_{\text{rel}} r \\ v_C &= \dot{u} \end{aligned}\right\} \Longrightarrow \omega_{\text{rel}} = \frac{\dot{u}}{r} \tag{3.7}$$

The orientation of angles in Figure 3.1 gives rise to the relation between the absolute angular velocity ω and the relative angular velocity ω_{rel}.

$$\omega = \omega_{\text{rel}} - \dot{\alpha} \tag{3.8}$$

We get the constraint equation of the ball on the beam by substituting Equation 3.7 into Equation 3.8.

$$\omega = \frac{\dot{u}}{r} - \dot{\alpha} \tag{3.9}$$

Substitution of Equation 3.9 into Equation 3.6 gives

$$F = -\frac{J}{r^2}(\ddot{u} - r\ddot{\alpha}) \tag{3.10}$$

and the gravitational force G can be expanded to

$$G = mg. \tag{3.11}$$

Factoring out m in Equation 3.4 and substituting Equations 3.5, 3.10 and 3.11 yields an equation that only contains the coordinate u.

$$m(\ddot{u} - u\dot{\alpha}^2 - r\ddot{\alpha}) = -\frac{J}{r^2}(\ddot{u} - r\ddot{\alpha}) - mg\sin\alpha$$

After a division by m, we have the desired equation of motion for a ball on a beam.

$$\left(1 + \frac{J}{mr^2}\right)(\ddot{u} - r\ddot{\alpha}) - u\dot{\alpha}^2 + g\sin\alpha = 0 \tag{3.12}$$

For the design of a controller, the differential equation is linearized.

$$\left(1 + \frac{J}{mr^2}\right)(\ddot{u} - r\ddot{\alpha}) + g\alpha = 0$$

Solving for $\ddot{u}$, we see that for sufficiently small radii r, angles α, angular velocities $\dot{\alpha}$ and angular accelerations $\ddot{\alpha}$, the plant is in principle a double integrator.

$$\ddot{u} = -\frac{mr^2}{mr^2 + J}g\alpha + r\ddot{\alpha} \tag{3.13}$$

Lagrange Formalism

The so-called *Lagrangian* is the difference between the kinetic and the potential energy of a rigid body.

$$L = E_{\text{kin}} - E_{\text{pot}} \tag{3.14}$$

The kinetic energy is the sum of translational and rotational energy.

$$E_{\text{kin}} = E_{\text{trans}} + E_{\text{rot}}$$

With the magnitude of the velocity of the center of mass denoted by v, the translational energy is defined as

$$E_{\text{trans}} = \frac{m}{2}v^2,$$

where we make use of Equation 3.1 to compute v^2.

$$v^2 = \left|\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix}\right|^2 = \dot{x}^2 + \dot{y}^2 = (\dot{u} - r\dot{\alpha})^2 + u^2\dot{\alpha}^2$$

With the magnitude of the angular velocity denoted by ω, the rotational energy is defined as

$$E_{\text{rot}} = \frac{J}{2}\omega^2,$$

where J is the moment of inertia. See Appendix A for the definition of J. For ω^2 we use Equation 3.9.

$$\omega^2 = \frac{1}{r^2}(\dot{u} - r\dot{\alpha})^2$$

The potential energy of the ball is

$$E_{\text{pot}} = mgy = mg(u \sin\alpha + r\cos\alpha).$$

Inserting everything into Equation 3.14 yields an expression for L that only contains the coordinate u.

$$L = \frac{1}{2}\left(m + \frac{J}{r^2}\right)(\dot{u} - r\dot{\alpha})^2 + \frac{m}{2}u^2\dot{\alpha}^2 - mg(u\sin\alpha + r\cos\alpha)$$

The condition

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{u}} - \frac{\partial L}{\partial u} \overset{!}{=} 0$$

produces the desired equation of motion for a ball on a beam.

$$(1 + \frac{J}{mr^2})(\ddot{u} - r\ddot{\alpha}) - u\dot{\alpha}^2 + g\sin\alpha = 0 \tag{3.15}$$

Note that Equation 3.12 is the same as Equation 3.15.

3.2 Full Model of the Ball and Plate System

The assumptions made for the Ball and Beam system—the ball does not take off, it does not slip and rolling occurs with neither friction nor air resistance—equivalently hold for the Ball and Plate system.

The derivation of the reduced model presented in Section 3.3 is not directly based on the full model. Apart from the definition of the transformation between the frames of reference, the two sections are independent.

Transformation between the Frames of Reference

Figure 3.2 illustrates the motion of the plate. Our first goal is to find the transformation between the two frames of reference $\{x, y, z\}$ and $\{u, v, w\}$. This is accomplished in two consecutive steps: the rotation by α and the rotation by β, which can be described individually. Note, however, that the two operations do not commute.

If the construction of the system should ever be changed, i.e. if the plate is handled in a different way, only the transformation must be adopted and all other derivations can be reused.

In Figure 3.3, the global frame of reference $\{x, y, z\}$, which is presumed to be inertial, is mapped onto the auxiliary frame of reference $\{u^*, v^*, w^*\}$.

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}}_{T_\alpha} \begin{bmatrix} u^* \\ v^* \\ w^* \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ \cos\alpha \\ \sin\alpha \end{bmatrix}}_{q_\alpha} d \tag{3.16}$$

In Figure 3.4, the auxiliary frame of reference $\{u^*, v^*, w^*\}$ is mapped onto the local frame of reference $\{u, v, w\}$, which is attached to the plate.

$$\begin{bmatrix} u^* \\ v^* \\ w^* \end{bmatrix} = \underbrace{\begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}}_{T_\beta} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{3.17}$$

Substitute Equation 3.17 into Equation 3.16 to get the direct transformation from the local frame of reference $\{u, v, w\}$ to the global frame of reference $\{x, y, z\}$.

$$[x\ y\ z]' = T_\alpha T_\beta\,[u\ v\ w]' + q_\alpha d$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \underbrace{\begin{bmatrix} \cos\beta & 0 & \sin\beta \\ \sin\alpha\sin\beta & \cos\alpha & -\sin\alpha\cos\beta \\ -\cos\alpha\sin\beta & \sin\alpha & \cos\alpha\cos\beta \end{bmatrix}}_{T_{\alpha\beta}} \begin{bmatrix} u \\ v \\ w \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ \cos\alpha \\ \sin\alpha \end{bmatrix}}_{q_\alpha} d \tag{3.18}$$

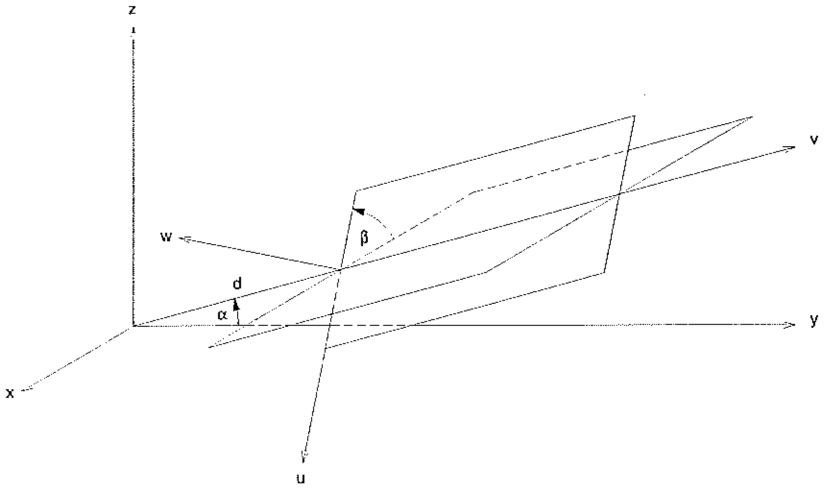


Figure 3.2: The plate and its descriptive coordinates.

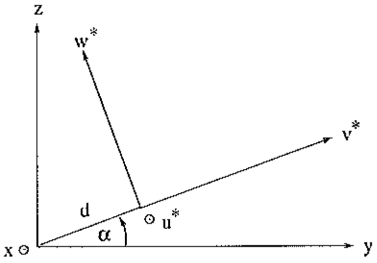


Figure 3.3: Rotate the plate by α.

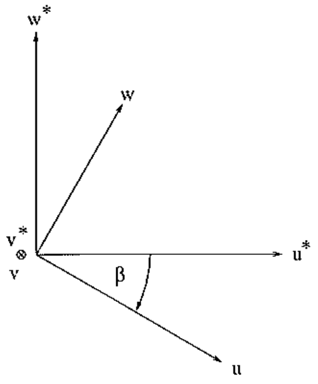


Figure 3.4: Rotate the plate by β.

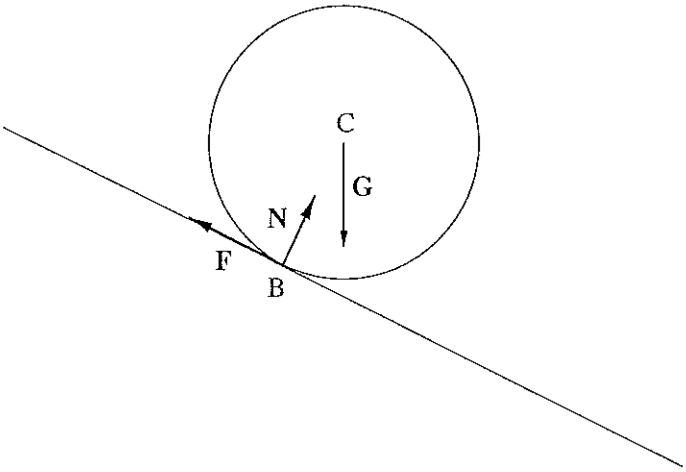


Figure 3.5: Momentary view of the ball on the plate.

Definitions

The distance d in Figure 3.2 is set to zero since its value does not change the dynamic properties of the model.

Figure 3.5 shows a momentary view of the ball on the plate. F is the force of adhesive friction, N is the normal force and G the gravitational force. Motion parameters and frames of reference are defined as follows:

$\boldsymbol{\Omega}$	angular velocity of the plate (excitation)
$\boldsymbol{\omega}$	angular velocity of the ball
$\boldsymbol{r}$	position vector
$\boldsymbol{e}_x$, $\boldsymbol{e}_y$, $\boldsymbol{e}_z$	unit vectors of the inertial frame of reference
$\boldsymbol{e}_u$, $\boldsymbol{e}_v$, $\boldsymbol{e}_w$	unit vectors of the plate's frame of reference

Indices i and p on the lower left-hand side of a variable indicate whether the variable is expressed in coordinates of the inertial frame of reference or the plate's frame of reference, respectively. Take the position vector $\boldsymbol{r}$ as an example.

$$\begin{aligned} {}_i\boldsymbol{r}(t) &= \begin{bmatrix} x(t) \\ y(t) \\ z(t) \end{bmatrix} = x(t)\,{}_i\boldsymbol{e}_x + y(t)\,{}_i\boldsymbol{e}_y + z(t)\,{}_i\boldsymbol{e}_z \\ {}_p\boldsymbol{r}(t) &= \begin{bmatrix} u(t) \\ v(t) \\ w(t) \end{bmatrix} = u(t)\,{}_p\boldsymbol{e}_u + v(t)\,{}_p\boldsymbol{e}_v + w(t)\,{}_p\boldsymbol{e}_w \end{aligned}$$

Define the reference motion $\boldsymbol{T}(t)$, such that

$${}_i\boldsymbol{e}_k(t) = \boldsymbol{T}(t)\,{}_i\boldsymbol{e}_l \quad \text{where } k \in \{u, v, w\} \text{ and } l \in \{x, y, z\}.$$

$\boldsymbol{T}$ is the transformation matrix $\boldsymbol{T}_{\alpha\beta}$ defined in Equation 3.18.

$$\boldsymbol{T} = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ \sin\alpha\sin\beta & \cos\alpha & -\sin\alpha\cos\beta \\ -\cos\alpha\sin\beta & \sin\alpha & \cos\alpha\cos\beta \end{bmatrix} \tag{3.19}$$

It can easily be verified that $\boldsymbol{T}$ is orthogonal.

$\boldsymbol{T}$ transforms any vector from the plate's frame of reference to the inertial frame of reference.

$$\begin{aligned} {}_i\boldsymbol{\Omega} &= \boldsymbol{T}\,{}_p\boldsymbol{\Omega} \\ {}_i\boldsymbol{\omega} &= \boldsymbol{T}\,{}_p\boldsymbol{\omega} \\ {}_i\boldsymbol{r} &= \boldsymbol{T}\,{}_p\boldsymbol{r} \end{aligned}$$

Angular Velocity of the Plate

Consider the velocity of an arbitrary point A on the plate.

$$ {}_i\dot{\boldsymbol{r}}_A = {}_i\boldsymbol{\Omega} \times {}_i\boldsymbol{r}_A \tag{3.20}$$

With the unary cross product operator $(\cdot)^{\sim}$ defined in Appendix B, Equation 3.20 becomes

$$ {}_i\dot{\boldsymbol{r}}_A = {}_i\tilde{\boldsymbol{\Omega}}\,{}_i\boldsymbol{r}_A, \tag{3.21}$$

where ${}_i\tilde{\boldsymbol{\Omega}}$ is a skew-symmetric matrix.

Alternatively, ${}_i\dot{\boldsymbol{r}}_A$ can be written as

$$ {}_i\dot{\boldsymbol{r}}_A = \frac{\mathrm{d}}{\mathrm{d}t}(\boldsymbol{T}\,{}_p\boldsymbol{r}_A) = \dot{\boldsymbol{T}}\,{}_p\boldsymbol{r}_A + \boldsymbol{T}\,{}_p\dot{\boldsymbol{r}}_A.$$

Since ${}_p\boldsymbol{r}_A$ is constant, we have ${}_p\dot{\boldsymbol{r}}_A = 0$ and therefore

$$ {}_i\dot{\boldsymbol{r}}_A = \dot{\boldsymbol{T}}\,{}_p\boldsymbol{r}_A = \dot{\boldsymbol{T}}\boldsymbol{T}^{-1}\,{}_i\boldsymbol{r}_A. \tag{3.22}$$

Comparing Equations 3.21 and 3.22 we gather

$$ {}_i\tilde{\boldsymbol{\Omega}} = \dot{\boldsymbol{T}}\boldsymbol{T}^{-1}.$$

Lemma Let $\boldsymbol{T}$ an arbitrary orthogonal matrix and $\boldsymbol{a}, \boldsymbol{b}$ vectors. Then

$$\boldsymbol{T}(\boldsymbol{a} \times \boldsymbol{b}) = (\boldsymbol{T}\boldsymbol{a}) \times (\boldsymbol{T}\boldsymbol{b}).$$

Multiply Equation 3.20 with $\boldsymbol{T}^{-1}$ and use Lemma to expand.

$$\boldsymbol{T}^{-1}\,{}_i\dot{\boldsymbol{r}}_A = \boldsymbol{T}^{-1}({}_i\boldsymbol{\Omega} \times {}_i\boldsymbol{r}_A) = (\boldsymbol{T}^{-1}\,{}_i\boldsymbol{\Omega}) \times (\boldsymbol{T}^{-1}\,{}_i\boldsymbol{r}_A) = {}_p\boldsymbol{\Omega} \times {}_p\boldsymbol{r}_A = {}_p\tilde{\boldsymbol{\Omega}}\,{}_p\boldsymbol{r}_A \tag{3.23}$$

Now, if we multiply Equation 3.22 with $\boldsymbol{T}^{-1}$ and compare with Equation 3.23, we gather

$$ {}_p\tilde{\boldsymbol{\Omega}} = \boldsymbol{T}^{-1}\dot{\boldsymbol{T}}.$$

When computing the inverse of T , remember that $T^{-1} = T'$ because T is orthogonal.[1]

$$T^{-1} = \begin{bmatrix} \cos\beta & \sin\alpha\sin\beta & -\cos\alpha\sin\beta \\ 0 & \cos\alpha & \sin\alpha \\ \sin\beta & -\sin\alpha\cos\beta & \cos\alpha\cos\beta \end{bmatrix}$$

$$\dot{T} = \begin{bmatrix} -\dot\beta\sin\beta & 0 & \dot\beta\cos\beta \\ \dot\alpha\cos\alpha\sin\beta + \dot\beta\sin\alpha\cos\beta & -\dot\alpha\sin\alpha & -\dot\alpha\cos\alpha\cos\beta + \dot\beta\sin\alpha\sin\beta \\ \dot\alpha\sin\alpha\sin\beta - \dot\beta\cos\alpha\cos\beta & \dot\alpha\cos\alpha & -\dot\alpha\sin\alpha\cos\beta - \dot\beta\cos\alpha\sin\beta \end{bmatrix}$$

$${}_p\tilde{\Omega} = \begin{bmatrix} 0 & -\dot\alpha\sin\beta & -\dot\beta \\ \dot\alpha\sin\beta & 0 & -\dot\alpha\cos\beta \\ -\dot\beta & \dot\alpha\cos\beta & 0 \end{bmatrix} \tag{3.24}$$

Since the transformation denoted by the operator $(\cdot)^{\sim}$ is a linear isomorphism, the vector ${}_p\Omega$, such that ${}_p\Omega \times {}_p r = {}_p\tilde{\Omega}\,{}_p r$, is unique. See Theorem in Appendix B for a proof.

$${}_p\Omega = \begin{bmatrix} \dot\alpha\cos\beta \\ \dot\beta \\ \dot\alpha\sin\beta \end{bmatrix} = \begin{bmatrix} \cos\beta & 0 \\ 0 & 1 \\ \sin\beta & 0 \end{bmatrix} \begin{bmatrix} \dot\alpha \\ \dot\beta \end{bmatrix} \tag{3.25}$$

$${}_p\dot{\Omega} = \begin{bmatrix} \cos\beta & 0 \\ 0 & 1 \\ \sin\beta & 0 \end{bmatrix} \begin{bmatrix} \ddot\alpha \\ \ddot\beta \end{bmatrix} + \begin{bmatrix} -\sin\beta \\ 0 \\ \cos\beta \end{bmatrix} \dot\alpha\dot\beta \tag{3.26}$$

Preliminary Computations

We aim at equations of motion expressed in coordinates which are local to the plate. Hence it is necessary to transform all relevant variables from the inertial frame of reference to the plate's frame of reference, namely position r_C, velocity $\dot{r}_C$ and acceleration $\ddot{r}_C$ of the center of mass C as well as angular velocity ω and angular acceleration $\dot\omega$ of the ball.

$$T^{-1}\,{}_i r_C = {}_p r_C$$

$$\begin{aligned} T^{-1}\,{}_i\dot{r}_C &= T^{-1}\tfrac{\mathrm{d}}{\mathrm{d}t}(T\,{}_p r_C) = T^{-1}(\dot{T}\,{}_p r_C + T\,{}_p\dot{r}_C) = T^{-1}\dot{T}\,{}_p r_C + {}_p\dot{r}_C \\ &= {}_p\tilde{\Omega}\,{}_p r_C + {}_p\dot{r}_C = {}_p\Omega \times {}_p r_C + {}_p\dot{r}_C \end{aligned} \tag{3.27}$$

This is the sum of the velocity caused by the reference motion and the velocity relative to the plate.

[1] Transposition of a matrix will be denoted by a prime.

$$\begin{aligned}
\boldsymbol{T}^{-1}\,{}_i\ddot{\boldsymbol{r}}_C &= \boldsymbol{T}^{-1}\tfrac{\mathrm{d}}{\mathrm{d}t}\,{}_i\dot{\boldsymbol{r}}_C = \boldsymbol{T}^{-1}\tfrac{\mathrm{d}}{\mathrm{d}t}(\boldsymbol{T}\boldsymbol{T}^{-1}\,{}_i\dot{\boldsymbol{r}}_C) \\
&= \boldsymbol{T}^{-1}\tfrac{\mathrm{d}}{\mathrm{d}t}(\boldsymbol{T}(\,{}_p\tilde{\boldsymbol{\Omega}}\,{}_p\boldsymbol{r}_C + {}_p\dot{\boldsymbol{r}}_C)) \\
&= \boldsymbol{T}^{-1}(\dot{\boldsymbol{T}}(\,{}_p\tilde{\boldsymbol{\Omega}}\,{}_p\boldsymbol{r}_C + {}_p\dot{\boldsymbol{r}}_C) + \boldsymbol{T}(\,{}_p\dot{\tilde{\boldsymbol{\Omega}}}\,{}_p\boldsymbol{r}_C + {}_p\tilde{\boldsymbol{\Omega}}\,{}_p\dot{\boldsymbol{r}}_C + {}_p\ddot{\boldsymbol{r}}_C)) \\
&= {}_p\tilde{\boldsymbol{\Omega}}^2\,{}_p\boldsymbol{r}_C + 2\,{}_p\tilde{\boldsymbol{\Omega}}\,{}_p\dot{\boldsymbol{r}}_C + {}_p\dot{\tilde{\boldsymbol{\Omega}}}\,{}_p\boldsymbol{r}_C + {}_p\ddot{\boldsymbol{r}}_C \\
&= {}_p\dot{\boldsymbol{\Omega}} \times {}_p\boldsymbol{r}_C + {}_p\boldsymbol{\Omega} \times (\,{}_p\boldsymbol{\Omega} \times {}_p\boldsymbol{r}_C) + {}_p\ddot{\boldsymbol{r}}_C + 2\,{}_p\boldsymbol{\Omega} \times {}_p\dot{\boldsymbol{r}}_C
\end{aligned} \tag{3.28}$$

We recognize the acceleration caused by the reference motion in the direction of the momentary orbit, ${}_p\dot{\boldsymbol{\Omega}} \times {}_p\boldsymbol{r}_C$, and towards the momentary center of rotation, ${}_p\boldsymbol{\Omega} \times (\,{}_p\boldsymbol{\Omega} \times {}_p\boldsymbol{r}_C)$. Furthermore, the acceleration relative to the plate, ${}_p\ddot{\boldsymbol{r}}_C$, and the Coriolis acceleration, $2\,{}_p\boldsymbol{\Omega} \times {}_p\dot{\boldsymbol{r}}_C$.

$$\boldsymbol{T}^{-1}\,{}_i\boldsymbol{\omega} = {}_p\boldsymbol{\omega}$$

$$\begin{aligned}
\boldsymbol{T}^{-1}\,{}_i\dot{\boldsymbol{\omega}} &= \boldsymbol{T}^{-1}\tfrac{\mathrm{d}}{\mathrm{d}t}(\boldsymbol{T}\,{}_p\boldsymbol{\omega}) = \boldsymbol{T}^{-1}(\dot{\boldsymbol{T}}\,{}_p\boldsymbol{\omega} + \boldsymbol{T}\,{}_p\dot{\boldsymbol{\omega}}) \\
&= {}_p\tilde{\boldsymbol{\Omega}}\,{}_p\boldsymbol{\omega} + {}_p\dot{\boldsymbol{\omega}} = {}_p\boldsymbol{\Omega} \times {}_p\boldsymbol{\omega} + {}_p\dot{\boldsymbol{\omega}}
\end{aligned} \tag{3.29}$$

We have the sum of the angular acceleration caused by the reference motion and the angular acceleration relative to the plate.

Newton-Euler Equations

The conservation law of momentum is

$$m\,{}_i\ddot{\boldsymbol{r}}_C = {}_i\vec{G} + {}_i\vec{R}, \tag{3.30}$$

where m denotes the mass of the ball and $\boldsymbol{r}_C$ the position of the center of mass. $\vec{G}$ is the gravitational force and $\vec{R}$ the reaction force.

$$\begin{aligned}
{}_i\vec{G} &= -mg\,{}_i\boldsymbol{e}_z \\
{}_i\vec{R} &= {}_i\vec{F} + {}_i\vec{N}
\end{aligned}$$

$\vec{F}$ and $\vec{N}$ are adhesive friction and normal forces, respectively, as depicted in Figure 3.5.

Transform Equation 3.30 to the p frame and substitute Equation 3.28.

$$m\boldsymbol{T}^{-1}\,{}_i\ddot{\boldsymbol{r}}_C = \boldsymbol{T}^{-1}\,{}_i\vec{G} + \boldsymbol{T}^{-1}\,{}_i\vec{R}$$

$$m(\,{}_p\dot{\boldsymbol{\Omega}} \times {}_p\boldsymbol{r}_C + {}_p\boldsymbol{\Omega} \times (\,{}_p\boldsymbol{\Omega} \times {}_p\boldsymbol{r}_C) + {}_p\ddot{\boldsymbol{r}}_C + 2\,{}_p\boldsymbol{\Omega} \times {}_p\dot{\boldsymbol{r}}_C) = m\boldsymbol{T}^{-1}\begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + {}_p\vec{R} \tag{3.31}$$

The conservation law of angular momentum is

$$\boldsymbol{J}\,{}_i\dot{\boldsymbol{\omega}} + {}_i\boldsymbol{\omega} \times \boldsymbol{J}\,{}_i\boldsymbol{\omega} = {}_i(\overrightarrow{CB}) \times {}_i\vec{R}. \tag{3.32}$$

For symmetry reasons, we have the tensor

$$\boldsymbol{J} = J\boldsymbol{I}_3,$$

where J is the scalar moment of inertia defined in Appendix A and $\boldsymbol{I}_3$ the 3×3 identity matrix. Consequently,

$${}_i\boldsymbol{\omega} \times \boldsymbol{J}\,{}_i\boldsymbol{\omega} = J\,{}_i\boldsymbol{\omega} \times {}_i\boldsymbol{\omega} = 0.$$

With r denoting the radius of the ball,[2] the vector from the center of mass C to the point of contact B is

$${}_i(\overrightarrow{CB}) = -r\,{}_i\boldsymbol{e}_w.$$

Equation 3.32 can thus be written as

$$J\,{}_i\dot{\boldsymbol{\omega}} = -r\,{}_i\boldsymbol{e}_w \times {}_i\vec{R}. \tag{3.33}$$

Transform Equation 3.33 to the p frame, substitute Equation 3.29 and use Lemma to expand the right-hand side of the equation.

$$J\boldsymbol{T}^{-1}\,{}_i\dot{\boldsymbol{\omega}} = -r\boldsymbol{T}^{-1}({}_i\boldsymbol{e}_w \times {}_i\vec{R})$$

$$J({}_p\boldsymbol{\Omega} \times {}_p\boldsymbol{\omega} + {}_p\dot{\boldsymbol{\omega}}) = -r({}_p\boldsymbol{e}_w \times {}_p\vec{R}) \tag{3.34}$$

Equation 3.34 expands to

$${}_p\dot{\boldsymbol{\omega}} = -\frac{r}{J}({}_p\boldsymbol{e}_w \times {}_p\vec{R}) - {}_p\boldsymbol{\Omega} \times {}_p\boldsymbol{\omega}$$

$$\begin{bmatrix} \dot{\omega}_u \\ \dot{\omega}_v \\ \dot{\omega}_w \end{bmatrix} = \frac{r}{J}\begin{bmatrix} F_v \\ -F_u \\ 0 \end{bmatrix} + \begin{bmatrix} \Omega_w\omega_v - \Omega_v\omega_w \\ \Omega_u\omega_w - \Omega_w\omega_u \\ \Omega_v\omega_u - \Omega_u\omega_v \end{bmatrix}$$

and in particular

$$\dot{\omega}_w = \Omega_v\omega_u - \Omega_u\omega_v. \tag{3.35}$$

We go one step further and substitute Equation 3.25 into Equation 3.35.

$$\dot{\omega}_w = \dot{\beta}\omega_u - \dot{\alpha}\cos\beta\omega_v \tag{3.36}$$

Equation 3.36 clearly shows that the ball has a spin normal to the plate.

For convenient further manipulations, we replace the binary cross product operator $(\cdot) \times (\cdot)$ by the unary cross product operator $(\cdot)^{\sim}$ in Equations 3.31 and 3.34.

$$\begin{aligned} {}_p\dot{\tilde{\boldsymbol{\Omega}}}\,{}_p\boldsymbol{r}_C + {}_p\tilde{\boldsymbol{\Omega}}^2\,{}_p\boldsymbol{r}_C + {}_p\ddot{\boldsymbol{r}}_C + 2\,{}_p\tilde{\boldsymbol{\Omega}}\,{}_p\dot{\boldsymbol{r}}_C &= \boldsymbol{T}^{-1}\begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \tfrac{1}{m}\,{}_p\vec{R} \\ {}_p\tilde{\boldsymbol{\Omega}}\,{}_p\boldsymbol{\omega} + {}_p\dot{\boldsymbol{\omega}} &= \tfrac{1}{J}(-r\,{}_p\tilde{\boldsymbol{e}}_w)\,{}_p\vec{R} \end{aligned} \tag{3.37}$$

[2] Mind the difference between the radius r and the position $\boldsymbol{r}$.

We combine Equations 3.37 to a single equation in block matrix form while at the same time making use of the identity $-{}_p\tilde{e}_w = {}_p\tilde{e}'_w$, which holds for all skew-symmetric matrices. 0_3 is the 3×3 zero matrix and $\boldsymbol{I}_3$ the 3×3 identity matrix.

$$\begin{bmatrix} {}_p\ddot{\boldsymbol{r}}_C \\ {}_p\dot{\boldsymbol{\omega}} \end{bmatrix} + \begin{bmatrix} {}_p\dot{\tilde{\boldsymbol{\Omega}}} + {}_p\tilde{\boldsymbol{\Omega}}^2 & 2\,{}_p\tilde{\boldsymbol{\Omega}} & 0_3 \\ 0_3 & 0_3 & {}_p\tilde{\boldsymbol{\Omega}} \end{bmatrix} \begin{bmatrix} {}_p\boldsymbol{r}_C \\ {}_p\dot{\boldsymbol{r}}_C \\ {}_p\boldsymbol{\omega} \end{bmatrix}$$

$$= \begin{bmatrix} \boldsymbol{T}^{-1} \\ 0_3 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} + \begin{bmatrix} \frac{1}{m}\boldsymbol{I}_3 & 0_3 \\ 0_3 & \frac{1}{J}\boldsymbol{I}_3 \end{bmatrix} \begin{bmatrix} \boldsymbol{I}_3 \\ r\,{}_p\tilde{\boldsymbol{e}}'_w \end{bmatrix} {}_p\vec{R} \qquad (3.38)$$

Constraint Equations

Fundamental to the Ball and Plate experiment is the provision that the ball remains in contact with the plate at all times. This imposes three scalar constraint equations, one for each degree of freedom of a rigid body in space.

Consider the point of contact B. Its velocity $\dot{\boldsymbol{r}}_B$ can be expressed in two different ways.

$$\begin{aligned} {}_i\dot{\boldsymbol{r}}_B &= {}_i\dot{\boldsymbol{r}}_C + {}_i\boldsymbol{\omega} \times {}_i(\overrightarrow{CB}) = {}_i\dot{\boldsymbol{r}}_C - r\,{}_i\boldsymbol{\omega} \times {}_i\boldsymbol{e}_w \\ {}_i\dot{\boldsymbol{r}}_B &= {}_i\boldsymbol{\Omega} \times {}_i\boldsymbol{r}_B = {}_i\boldsymbol{\Omega} \times ({}_i\boldsymbol{r}_C - r\,{}_i\boldsymbol{e}_w) = {}_i\boldsymbol{\Omega} \times {}_i\boldsymbol{r}_C - r\,{}_i\boldsymbol{\Omega} \times {}_i\boldsymbol{e}_w \end{aligned}$$

With the identity $\boldsymbol{a} \times \boldsymbol{c} + \boldsymbol{b} \times \boldsymbol{c} = (\boldsymbol{a} + \boldsymbol{b}) \times \boldsymbol{c}$, where $\boldsymbol{a}$, $\boldsymbol{b}$ and $\boldsymbol{c}$ are vectors, it follows immediately that

$${}_i\dot{\boldsymbol{r}}_C - {}_i\boldsymbol{\Omega} \times {}_i\boldsymbol{r}_C = r({}_i\boldsymbol{\omega} - {}_i\boldsymbol{\Omega}) \times {}_i\boldsymbol{e}_w. \qquad (3.39)$$

Transform Equation 3.39 to the p frame, substitute Equation 3.27 and use Lemma to expand the cross products.

$$\boldsymbol{T}^{-1}\,{}_i\dot{\boldsymbol{r}}_C - \boldsymbol{T}^{-1}({}_i\boldsymbol{\Omega} \times {}_i\boldsymbol{r}_C) = r\boldsymbol{T}^{-1}(({}_i\boldsymbol{\omega} - {}_i\boldsymbol{\Omega}) \times {}_i\boldsymbol{e}_w)$$

$${}_p\dot{\boldsymbol{r}}_C = r({}_p\boldsymbol{\omega} - {}_p\boldsymbol{\Omega}) \times {}_p\boldsymbol{e}_w \qquad (3.40)$$

Equation 3.40 expands to

$$\begin{bmatrix} \dot{u}_C \\ \dot{v}_C \\ \dot{w}_C \end{bmatrix} = r \begin{bmatrix} \omega_v - \Omega_v \\ \omega_u - \Omega_u \\ 0 \end{bmatrix}$$

and in particular

$$\dot{w}_C = 0, \qquad (3.41)$$

which is exactly as expected, since we requested that the ball should never loose contact with the plate.

Using the unary cross product operator $(\cdot)^{\sim}$, we can write Equation 3.40 in block matrix form. $\mathbf{0}$ denotes the 3×1 zero vector.

$$
\begin{gathered}
{}_p\dot{\boldsymbol{r}}_C = r\,{}_p\boldsymbol{e}_w \times \left({}_p\boldsymbol{\Omega} - {}_p\boldsymbol{\omega}\right) = r\,{}_p\tilde{\boldsymbol{e}}_w\,{}_p\boldsymbol{\Omega} - r\,{}_p\tilde{\boldsymbol{e}}_w\,{}_p\boldsymbol{\omega} \\
\begin{bmatrix} \boldsymbol{I}_3 & r\,{}_p\tilde{\boldsymbol{e}}_w \end{bmatrix} \begin{bmatrix} {}_p\dot{\boldsymbol{r}}_C \\ {}_p\boldsymbol{\omega} \end{bmatrix} = \begin{bmatrix} \boldsymbol{I}_3 & r\,{}_p\tilde{\boldsymbol{e}}_w \end{bmatrix} \begin{bmatrix} \mathbf{0} \\ {}_p\boldsymbol{\Omega} \end{bmatrix}
\end{gathered} \tag{3.42}
$$

Combining Newton-Euler and Constraint Equations

In order to gain equations of motion with the only time-varying variables ${}_p\boldsymbol{r}_C$, ${}_p\boldsymbol{\omega}$, ${}_p\boldsymbol{\Omega}$ and their derivatives, we must find an explicit solution for ${}_p\vec{R}$ in Equation 3.38.

To make handling of equations easier, we introduce a few shortcut notations.

$$
\begin{gathered}
\boldsymbol{S} = \begin{bmatrix} {}_p\dot{\tilde{\boldsymbol{\Omega}}} + {}_p\tilde{\boldsymbol{\Omega}}^2 & 2\,{}_p\tilde{\boldsymbol{\Omega}} & \boldsymbol{0}_3 \\ \boldsymbol{0}_3 & \boldsymbol{0}_3 & {}_p\tilde{\boldsymbol{\Omega}} \end{bmatrix} \\
\boldsymbol{g} = \begin{bmatrix} \boldsymbol{T}^{-1} \\ \boldsymbol{0}_3 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} \\
\boldsymbol{M} = \begin{bmatrix} m\boldsymbol{I}_3 & \boldsymbol{0}_3 \\ \boldsymbol{0}_3 & J\boldsymbol{I}_3 \end{bmatrix} \\
\boldsymbol{E} = \begin{bmatrix} \boldsymbol{I}_3 & r\,{}_p\tilde{\boldsymbol{e}}_w \end{bmatrix}
\end{gathered} \tag{3.43}
$$

From now on, all differential equations will be expressed in coordinates of the plate's frame of reference and $\boldsymbol{r}$ will always denote the position vector of the ball's center of mass, so that we can henceforth leave aside the indices p and C.

$$
\begin{aligned}
\boldsymbol{r} &= {}_p\boldsymbol{r}_C \\
\boldsymbol{\omega} &= {}_p\boldsymbol{\omega} \\
\boldsymbol{\Omega} &= {}_p\boldsymbol{\Omega} \\
\vec{R} &= {}_p\vec{R}
\end{aligned} \tag{3.44}
$$

Substitute Equations 3.43 and 3.44 into Equations 3.38 and 3.42 to write them in a more compact way.

$$
\begin{bmatrix} \ddot{\boldsymbol{r}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix} + \boldsymbol{S} \begin{bmatrix} \boldsymbol{r} \\ \dot{\boldsymbol{r}} \\ \boldsymbol{\omega} \end{bmatrix} = \boldsymbol{g} + \boldsymbol{M}^{-1}\boldsymbol{E}'\vec{R} \tag{3.45}
$$

$$
\boldsymbol{E} \begin{bmatrix} \dot{\boldsymbol{r}} \\ \boldsymbol{\omega} \end{bmatrix} = \boldsymbol{E} \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\Omega} \end{bmatrix} \tag{3.46}
$$

Equation 3.45 cannot be solved for $\vec{R}$ by itself because $\boldsymbol{M}^{-1}\boldsymbol{E}'$ is not square. The key to the solution is as follows: Multiply Equation 3.45 by $\boldsymbol{E}$ and differentiate Equation 3.46 with respect to time.

$$\boldsymbol{E}\begin{bmatrix} \ddot{\boldsymbol{r}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix} + \boldsymbol{ES}\begin{bmatrix} \boldsymbol{r} \\ \dot{\boldsymbol{r}} \\ \boldsymbol{\omega} \end{bmatrix} = \boldsymbol{E}\boldsymbol{g} + \underbrace{\boldsymbol{EM}^{-1}\boldsymbol{E}'}_{\boldsymbol{G}}\vec{R} \tag{3.47}$$

$$\boldsymbol{E}\begin{bmatrix} \ddot{\boldsymbol{r}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix} = \boldsymbol{E}\begin{bmatrix} 0 \\ \dot{\boldsymbol{\Omega}} \end{bmatrix} \tag{3.48}$$

Substitute Equation 3.48 into Equation 3.47 and solve for $\vec{R}$. This is possible because $\boldsymbol{G}$ is nonsingular.

$$\vec{R} = \boldsymbol{G}^{-1}\boldsymbol{E}\begin{bmatrix} 0 \\ \dot{\boldsymbol{\Omega}} \end{bmatrix} + \boldsymbol{G}^{-1}\boldsymbol{ES}\begin{bmatrix} \boldsymbol{r} \\ \dot{\boldsymbol{r}} \\ \boldsymbol{\omega} \end{bmatrix} - \boldsymbol{G}^{-1}\boldsymbol{E}\boldsymbol{g} \tag{3.49}$$

Equation 3.49 is now substituted into Equation 3.45 to produce the complete equations of motion in a compact form.

$$\begin{bmatrix} \ddot{\boldsymbol{r}} \\ \dot{\boldsymbol{\omega}} \end{bmatrix} = \left(\boldsymbol{I}_6 - \boldsymbol{M}^{-1}\boldsymbol{E}'\boldsymbol{G}^{-1}\boldsymbol{E}\right)\left(\boldsymbol{g} - \boldsymbol{S}\begin{bmatrix} \boldsymbol{r} \\ \dot{\boldsymbol{r}} \\ \boldsymbol{\omega} \end{bmatrix}\right) + \boldsymbol{M}^{-1}\boldsymbol{E}'\boldsymbol{G}^{-1}\boldsymbol{E}\begin{bmatrix} 0 \\ \dot{\boldsymbol{\Omega}} \end{bmatrix} \tag{3.50}$$

Explicit Equations of Motion

The matrices that constitute Equation 3.50 remain to be evaluated for a formulation of the equations of motion in terms of the position of the center of mass $\boldsymbol{r}$, the angular velocity $\boldsymbol{\omega}$ and the inclination angles α and β. The results of these evaluations were verified in MAPLE.

Remember ${}_p\dot{\boldsymbol{\Omega}}$ from Equation 3.26 with $\boldsymbol{\Omega} = {}_p\boldsymbol{\Omega}$ from Equation 3.44.

$$\boldsymbol{r} = \begin{bmatrix} u \\ v \\ w \end{bmatrix} \qquad \boldsymbol{\omega} = \begin{bmatrix} \omega_u \\ \omega_v \\ \omega_w \end{bmatrix} \qquad \dot{\boldsymbol{\Omega}} = \begin{bmatrix} \ddot{\alpha}\cos\beta - \dot{\alpha}\dot{\beta}\sin\beta \\ \ddot{\beta} \\ \ddot{\alpha}\sin\beta + \dot{\alpha}\dot{\beta}\cos\beta \end{bmatrix}$$

We start with the matrices defined in Equation 3.43. Recall ${}_p\tilde{\boldsymbol{\Omega}}$ from Equation 3.24 and $\boldsymbol{T}$ from Equation 3.19.

$$\boldsymbol{S} = \begin{bmatrix} \boldsymbol{S}_{11} & \boldsymbol{S}_{12} & 0_3 \\ \boldsymbol{0}_3 & \boldsymbol{0}_3 & \boldsymbol{S}_{23} \end{bmatrix}$$

$$\boldsymbol{S}_{11} = \begin{bmatrix} -\dot{\alpha}^2\sin^2\beta - \dot{\beta}^2 & -\ddot{\alpha}\sin\beta & \dot{\alpha}^2\sin\beta\cos\beta + \ddot{\beta} \\ \ddot{\alpha}\sin\beta + 2\dot{\alpha}\dot{\beta}\cos\beta & -\dot{\alpha}^2 & -\ddot{\alpha}\cos\beta + 2\dot{\alpha}\dot{\beta}\sin\beta \\ \dot{\alpha}^2\sin\beta\cos\beta - \ddot{\beta} & \ddot{\alpha}\cos\beta & -\dot{\alpha}^2\cos\beta - \dot{\beta}^2 \end{bmatrix}$$

$$S_{12} = \begin{bmatrix} 0 & -2\dot{\alpha}\sin\beta & 2\dot{\beta} \\ 2\dot{\alpha}\sin\beta & 0 & -2\dot{\alpha}\cos\beta \\ -2\dot{\beta} & 2\dot{\alpha}\cos\beta & 0 \end{bmatrix}$$

$$S_{23} = \begin{bmatrix} 0 & -\dot{\alpha}\sin\beta & \dot{\beta} \\ \dot{\alpha}\sin\beta & 0 & -\dot{\alpha}\cos\beta \\ -\dot{\beta} & \dot{\alpha} & 0 \end{bmatrix}$$

$$g = \begin{bmatrix} g\cos\alpha\sin\beta \\ -g\sin\alpha \\ -g\cos\alpha\cos\beta \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$M = \begin{bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & J & 0 & 0 \\ 0 & 0 & 0 & 0 & J & 0 \\ 0 & 0 & 0 & 0 & 0 & J \end{bmatrix}$$

$$E = \begin{bmatrix} 1 & 0 & 0 & 0 & -r & 0 \\ 0 & 1 & 0 & r & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

G was defined in Equation 3.47.

$$G = EM^{-1}E' = \begin{bmatrix} \frac{1}{m} + \frac{r^2}{J} & 0 & 0 \\ 0 & \frac{1}{m} + \frac{r^2}{J} & 0 \\ 0 & 0 & \frac{1}{m} \end{bmatrix}$$

The distance w from the center of mass C to the point of contact B in Figure 3.5 is constant and equal to the radius of the ball r.

$$w = r$$

This is in accordance with Equation 3.41.

$$\dot{w} = 0$$

It turns out that the third line of Equations 3.50 is

$$\ddot{w} = 0$$

and clearly, these trivial differential equations are of no interest for the description of the two-dimensional motion of the ball on the plate.

If we introduce the state vector

$$\boldsymbol{x} = [u\ v\ \dot{u}\ \dot{v}\ \omega_u\ \omega_v\ \omega_w]',$$

Equations 3.50 can be written in the standard form

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{A}(t)\boldsymbol{x}(t) + \boldsymbol{b}(t), \tag{3.51}$$

where

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{bmatrix} \qquad \boldsymbol{b} = \boldsymbol{b}_1 + \boldsymbol{b}_2$$

and

$$\boldsymbol{A}_{11} = \frac{mr^2}{mr^2+J}\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \dot\alpha^2\sin^2\beta + \dot\beta^2 & \ddot\alpha\sin\beta & 0 & 2\dot\alpha\sin\beta \\ -\ddot\alpha\sin\beta - 2\dot\alpha\dot\beta\cos\beta & \dot\alpha^2 & -2\dot\alpha\sin\beta & 0 \end{bmatrix}$$

$$\boldsymbol{A}_{12} = \frac{Jr}{mr^2+J}\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ -\dot\alpha\sin\beta & 0 & \dot\alpha\cos\beta \\ 0 & -\dot\alpha\sin\beta & \dot\beta \end{bmatrix}$$

$$\boldsymbol{A}_{21} = \frac{mr}{mr^2+J}\begin{bmatrix} \ddot\alpha\sin\beta + 2\dot\alpha\dot\beta\cos\beta & -\dot\alpha^2 & 2\dot\alpha\sin\beta & 0 \\ \dot\alpha^2\sin^2\beta + \dot\beta^2 & \ddot\alpha\sin\beta & 0 & 2\dot\alpha\sin\beta \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{A}_{22} = \frac{J}{mr^2+J}\begin{bmatrix} 0 & \dot\alpha\sin\beta & -\dot\beta \\ -\dot\alpha\sin\beta & 0 & \dot\alpha\cos\beta \\ (1+\frac{mr^2}{J})\dot\beta & -(1+\frac{mr^2}{J})\dot\alpha\cos\beta & 0 \end{bmatrix}$$

$$\boldsymbol{b}_1 = \begin{bmatrix} 0 \\ 0 \\ -r\ddot\beta \\ r\ddot\alpha\cos\beta \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\boldsymbol{b}_2 = \frac{1}{mr^2+J}\begin{bmatrix} 0 \\ 0 \\ mr^2(-r\dot\alpha^2\sin\beta\cos\beta + g\cos\alpha\sin\beta) \\ mr^2(-2r\dot\alpha\dot\beta\sin\beta - g\sin\alpha) - Jr\dot\alpha\dot\beta\sin\beta \\ mr(r\dot\alpha\dot\beta\sin\beta + g\sin\alpha) \\ mr(-r\dot\alpha^2\sin\beta\cos\beta + g\cos\alpha\sin\beta) \\ 0 \end{bmatrix}.$$

If we set $\beta \equiv 0$, the fourth line of Equation 3.51 reduces to

$$\left(1+\frac{J}{mr^2}\right)(\ddot{v}-r\ddot{\alpha})-v\dot{\alpha}^2+g\sin\alpha=0. \tag{3.52}$$

Except for the name of the position variable, this differential equation is identical to Equation 3.12, the equation of the Ball and Beam system.

Likewise, if we set $\alpha \equiv 0$, the third line of Equation 3.51 reduces to

$$\left(1+\frac{J}{mr^2}\right)(\ddot{u}+r\ddot{\beta})-u\dot{\beta}^2-g\sin\beta=0.$$

Due to the definition of β in Fig 3.2, the sign changes in all but the quadratic terms of β compared to Equation 3.52.

For small angles α and β, the approximations $\sin\xi \approx \xi$ and $\cos\xi \approx 1$ for $\xi \in \{\alpha, \beta\}$ may be substituted into the matrices $\boldsymbol{A}$ and $\boldsymbol{b}$ in Equation 3.51.

$$\boldsymbol{A}_{11}=\frac{mr^2}{mr^2+J}\begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \dot{\alpha}^2\beta^2+\dot{\beta}^2 & \ddot{\alpha}\beta & 0 & 2\dot{\alpha}\beta \\ -\ddot{\alpha}\beta-2\dot{\alpha}\dot{\beta} & \dot{\alpha}^2 & -2\dot{\alpha}\beta & 0 \end{bmatrix}$$

$$\boldsymbol{A}_{12}=\frac{Jr}{mr^2+J}\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ -\dot{\alpha}\beta & 0 & \dot{\alpha} \\ 0 & -\dot{\alpha}\beta & \dot{\beta} \end{bmatrix}$$

$$\boldsymbol{A}_{21}=\frac{mr}{mr^2+J}\begin{bmatrix} \ddot{\alpha}\beta+2\dot{\alpha}\dot{\beta} & -\dot{\alpha}^2 & 2\dot{\alpha}\beta & 0 \\ \dot{\alpha}^2\beta^2+\dot{\beta}^2 & \ddot{\alpha}\beta & 0 & 2\dot{\alpha}\beta \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\boldsymbol{A}_{22}=\frac{J}{mr^2+J}\begin{bmatrix} 0 & \dot{\alpha}\beta & -\dot{\beta} \\ -\dot{\alpha}\beta & 0 & \dot{\alpha} \\ (1+\frac{mr^2}{J})\dot{\beta} & -(1+\frac{mr^2}{J})\dot{\alpha} & 0 \end{bmatrix}$$

$$\boldsymbol{b}_1=\begin{bmatrix} 0 \\ 0 \\ -r\ddot{\beta} \\ r\ddot{\alpha} \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\boldsymbol{b}_2=\frac{1}{mr^2+J}\begin{bmatrix} 0 \\ 0 \\ mr^2(-r\dot{\alpha}^2\beta+g\beta) \\ mr^2(-2r\dot{\alpha}\dot{\beta}\beta-g\alpha)-Jr\dot{\alpha}\dot{\beta}\beta \\ mr(r\dot{\alpha}\dot{\beta}\beta+g\alpha) \\ mr(-r\dot{\alpha}^2\beta+g\beta) \\ 0 \end{bmatrix}$$

As a last step, we linearize Equations 3.51.

$$\begin{aligned}
\ddot{u} &= \frac{mr^2}{mr^2 + J} g\beta - r\ddot{\beta} && (3.53)\\
\ddot{v} &= -\frac{mr^2}{mr^2 + J} g\alpha + r\ddot{\alpha} && (3.54)\\
\dot{\omega}_u &= \frac{mr}{mr^2 + J} g\alpha && (3.55)\\
\dot{\omega}_v &= \frac{mr}{mr^2 + J} g\beta && (3.56)\\
\dot{\omega}_w &= 0 && (3.57)
\end{aligned}$$

Equations 3.53 and 3.54 constitute the linearized equations of motion for a ball on a plate. Since they do not depend on $\dot{\omega}$, Equations 3.55–3.57 are not necessary for a linear controller design. It will be sufficient to design a controller for either dimension and then take two of these to control the motion in both dimensions. We have seen in Equation 3.36 that the normal spin $\dot{\omega}_w$ is non-zero, but there is no linear term and hence the linearization yields Equation 3.57.

3.3 Reduced Model of the Ball and Plate System

Assumptions

The transformation between the two frames of reference depicted in Figure 3.2 is the same as in the full model. Since the ball is required to remain in contact with the plate at all times, centrifugal forces normal to the plate are of no interest. Thus, the distance from the origin to the ball's center of mass can safely be approximated by the distance from the origin to the point of contact of the ball and the plate. This means that we can use the same position vector $\boldsymbol{r}$ for the center of mass and the point of contact.

$$\boldsymbol{r}_C \approx \boldsymbol{r}_B \tag{3.58}$$

Consequently, w remains zero in Equation 3.18.

$$\begin{aligned}
x &= u\cos\beta\\
y &= u\sin\alpha\sin\beta + (v+d)\cos\alpha\\
z &= -u\cos\alpha\sin\beta + (v+d)\sin\alpha
\end{aligned}$$

The first and second derivatives of the position represent the velocity and acceleration, respectively.

$$\begin{aligned}
\dot{x} &= \dot{u}\cos\beta - u\dot{\beta}\sin\beta\\
\dot{y} &= \dot{u}\sin\alpha\sin\beta + u\dot{\alpha}\cos\alpha\sin\beta + u\dot{\beta}\sin\alpha\cos\beta\\
&\quad + \dot{v}\cos\alpha - (v+d)\dot{\alpha}\sin\alpha\\
\dot{z} &= -\dot{u}\cos\alpha\sin\beta + u\dot{\alpha}\sin\alpha\sin\beta - u\dot{\beta}\cos\alpha\cos\beta\\
&\quad + \dot{v}\sin\alpha + (v+d)\dot{\alpha}\cos\alpha
\end{aligned} \tag{3.59}$$

$$\begin{aligned}
\ddot{x} &= \ddot{u}\cos\beta - 2\dot{u}\dot{\beta}\sin\beta - u\ddot{\beta}\sin\beta - u\dot{\beta}^2\cos\beta \\
\ddot{y} &= \ddot{u}\sin\alpha\sin\beta + 2\dot{u}\dot{\alpha}\cos\alpha\sin\beta + u\ddot{\alpha}\cos\alpha\sin\beta - u\dot{\alpha}^2\sin\alpha\sin\beta \\
&\quad + 2\dot{u}\dot{\beta}\sin\alpha\cos\beta + u\ddot{\beta}\sin\alpha\cos\beta - u\dot{\beta}^2\sin\alpha\sin\beta + 2u\dot{\alpha}\dot{\beta}\cos\alpha\cos\beta \\
&\quad + \ddot{v}\cos\alpha - 2\dot{v}\dot{\alpha}\sin\alpha - (v+d)\ddot{\alpha}\sin\alpha - (v+d)\dot{\alpha}^2\cos\alpha \\
\ddot{z} &= -\ddot{u}\cos\alpha\sin\beta + 2\dot{u}\dot{\alpha}\sin\alpha\sin\beta + u\ddot{\alpha}\sin\alpha\sin\beta + u\dot{\alpha}^2\cos\alpha\sin\beta \\
&\quad - 2\dot{u}\dot{\beta}\cos\alpha\cos\beta - u\ddot{\beta}\cos\alpha\cos\beta + u\dot{\beta}^2\cos\alpha sin\beta + 2u\dot{\alpha}\dot{\beta}\sin\alpha\cos\beta \\
&\quad + \ddot{v}\sin\alpha + 2\dot{v}\dot{\alpha}\cos\alpha + (v+d)\ddot{\alpha}\cos\alpha - (v+d)\dot{\alpha}^2\sin\alpha
\end{aligned} \tag{3.60}$$

As a further simplification, we disregard the coupling between the three components of the angular velocity ω. This is equivalent to neglecting the spin normal to the plate.

$$\omega_w \approx 0 \tag{3.61}$$

Approximation 3.61 allows us to apply the conservation law for angular momentum in the moving frame of reference, although it actually only holds in the inertial frame of reference. The Lagrange equations only hold for holonomic constraints.[3] The ball on the plate is subject to a non-holonomic constraint because the rotation around the axis normal to the plate is independent of the position parameters, i.e. the location of the center of mass. Hence we should actually use the Boltzmann-Hamel equations instead of the Lagrange equations. However, Approximation 3.61 permits to nevertheless use the Lagrange equations.

Newton-Euler Formalism

The unit vectors of the local frame of reference can be expressed in global coordinates as follows:

$$e_u = T_{\alpha\beta}\begin{bmatrix}1\\0\\0\end{bmatrix} = \begin{bmatrix}\cos\beta\\ \sin\alpha\sin\beta\\ -\cos\alpha\sin\beta\end{bmatrix}$$

$$e_v = T_{\alpha\beta}\begin{bmatrix}0\\1\\0\end{bmatrix} = \begin{bmatrix}0\\ \cos\alpha\\ \sin\alpha\end{bmatrix}$$

$$e_w = T_{\alpha\beta}\begin{bmatrix}0\\0\\1\end{bmatrix} = \begin{bmatrix}\sin\beta\\ -\sin\alpha\cos\beta\\ \cos\alpha\cos\beta\end{bmatrix}$$

[3] Holonomic constraints are expressed in position parameters, i.e. $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0}$. Differentiation with respect to time leads to constraints that are expressed in position and velocity parameters: $D\boldsymbol{f}(\boldsymbol{x})\cdot\dot{\boldsymbol{x}} = \boldsymbol{0}$. Non-holonomic constraints are of the form $\boldsymbol{g}(\boldsymbol{x},\dot{\boldsymbol{x}}) = \boldsymbol{0}$, where no $\boldsymbol{f}$ exists, such that $\boldsymbol{g} = D\boldsymbol{f}$.

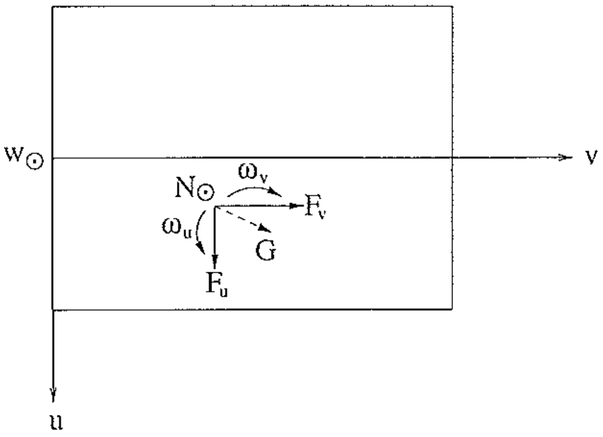


Figure 3.6: Top view of the plate.

The conservation law for momentum is valid in all three directions in space. Forces and local coordinates are defined according to Figure 3.6 and m denotes the mass of the ball.

$$m \begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = F_u e_u + F_v e_v + N e_w$$

$$\begin{aligned} m\ddot{x} &= F_u \cos\beta + N \sin\beta \\ m\ddot{y} &= F_u \sin\alpha \sin\beta + F_v \cos\alpha - N \sin\alpha \cos\beta \\ m\ddot{z} &= -F_u \cos\alpha \sin\beta + F_v \sin\alpha + N \cos\alpha \cos\beta - G \end{aligned} \tag{3.62}$$

If we assume $\omega_w \equiv 0$, the conservation of angular momentum holds for both degrees of freedom of the ball. We denote by J the moment of inertia of the ball which is derived in Appendix A and by r the radius of the ball.

$$\begin{aligned} -J\dot{\omega_u} &= rF_u \\ -J\dot{\omega_v} &= rF_v \end{aligned} \tag{3.63}$$

In analogy to Equation 3.7 we have the following relations between angular velocities and velocities.

$$\begin{aligned} \omega_u &= \tfrac{\dot{u}}{r} \\ \omega_v &= \tfrac{\dot{v}}{r} \end{aligned} \tag{3.64}$$

Substitute Equations 3.64 into Equations 3.63 and solve for F_u and F_v.

$$\begin{aligned} F_u &= -\tfrac{J}{r^2}\ddot{u} \\ F_v &= -\tfrac{J}{r^2}\ddot{v} \end{aligned} \tag{3.65}$$

Combining the second and third of Equations 3.62, either F_v or F_u can be eliminated.

$$\begin{aligned} m\ddot{y}\sin\alpha - m\ddot{z}\cos\alpha &= F_u\sin\beta - N\cos\beta + G\cos\alpha \\ m\ddot{y}\cos\alpha + m\ddot{z}\sin\alpha &= F_v - G\sin\alpha \end{aligned} \tag{3.66}$$

Then, combine the first lines of Equations 3.62 and Equations 3.66, respectively, to eliminate N in Equations 3.66.

$$\begin{aligned} m\ddot{x}\cos\beta + (m\ddot{y}\sin\alpha - m\ddot{z}\cos\alpha)\sin\beta &= F_u + G\cos\alpha\sin\beta \\ m\ddot{y}\cos\alpha + m\ddot{z}\sin\alpha &= F_v - G\sin\alpha \end{aligned} \tag{3.67}$$

Next, substitute Equations 3.11 and 3.65 into Equations 3.67 and divide by m.

$$\begin{aligned} \ddot{x}\cos\beta + \ddot{y}\sin\alpha\sin\beta - \ddot{z}\cos\alpha\sin\beta &= -\tfrac{J}{mr^2}\ddot{u} + g\cos\alpha\sin\beta \\ \ddot{y}\cos\alpha + \ddot{z}\sin\alpha &= -\tfrac{J}{mr^2}\ddot{v} - g\sin\alpha \end{aligned} \tag{3.68}$$

Finally, Equations 3.60 are substituted into Equations 3.68 to produce the desired equations of motion for the reduced model of a ball on a plate.

$$\begin{aligned} &(1+\tfrac{J}{mr^2})\ddot{u} - u\dot{\beta}^2 - g\cos\alpha\sin\beta - u\dot{\alpha}^2\sin^2\beta - (v+d)\ddot{\alpha}\sin\beta - 2\dot{v}\dot{\alpha}\sin\beta = 0 \\ &(1+\tfrac{J}{mr^2})\ddot{v} - (v+d)\dot{\alpha}^2 + g\sin\alpha + u\ddot{\alpha}\sin\beta + 2u\dot{\alpha}\dot{\beta}\cos\beta + 2\dot{u}\dot{\alpha}\sin\beta = 0 \end{aligned} \tag{3.69}$$

Keeping α zero in the first line of Equations 3.69 and β in the second results in two differential equations of equal structure.[4]

$$\begin{aligned} \left(1+\frac{J}{mr^2}\right)\ddot{u} - u\dot{\beta}^2 - g\sin\beta &= 0 \\ \left(1+\frac{J}{mr^2}\right)\ddot{v} - (v+d)\dot{\alpha}^2 + g\sin\alpha &= 0 \end{aligned} \tag{3.70}$$

When Equations 3.69 are linearized for small values of α and β, they are at the same time decoupled.

$$\begin{aligned} (1+\tfrac{J}{mr^2})\ddot{u} - g\beta &= 0 \\ (1+\tfrac{J}{mr^2})\ddot{v} + g\alpha &= 0 \end{aligned}$$

Solving for $\ddot{u}$ we get two separate double integrators.

$$\begin{aligned} \ddot{u} &= \tfrac{mgr^2}{mr^2+J}\beta \\ \ddot{v} &= -\tfrac{mgr^2}{mr^2+J}\alpha \end{aligned} \tag{3.71}$$

[4] The minus sign in $-g\sin\beta$ is due to the definition of β in Fig 3.2.

Lagrange Formalism

Again, we assume $\omega_w \equiv 0$. Recall that the *Lagrangian* is the difference between kinetic and potential energy of a rigid body.

$$L = E_{\text{kin}} - E_{\text{pot}} \tag{3.72}$$

The kinetic energy is the sum of translational and rotational energy.

$$E_{\text{kin}} = E_{\text{trans}} + E_{\text{rot}} = \frac{m}{2}v^2 + \frac{J}{2}\omega^2$$

We denote by v the magnitude of the velocity of the center of mass and make use of Equations 3.59 to compute v.

$$v^2 = \left| \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{pmatrix} \right|^2 = \dot{x}^2 + \dot{y}^2 + \dot{z}^2$$

$$\begin{aligned} \dot{x}^2 + \dot{y}^2 + \dot{z}^2 = {} & \dot{u}^2 + u^2\dot{\alpha}^2 \sin^2\beta + u^2\dot{\beta}^2 + \dot{v}^2 + (v+d)^2\dot{\alpha}^2 \\ & - 2\dot{u}(v+d)\dot{\alpha}\sin\beta + 2u\dot{v}\dot{\alpha}\sin\beta - 2u(v+d)\dot{\alpha}\dot{\beta}\cos\beta \end{aligned}$$

We denote by ω the magnitude of the angular velocity of the body and substitute Equations 3.64 into the definition.

$$\omega^2 = \left| \begin{pmatrix} \omega_u \\ \omega_v \end{pmatrix} \right|^2 = \omega_u^2 + \omega_v^2 = \frac{1}{r^2}(\dot{u}^2 + \dot{v}^2)$$

Because we approximate the distance from the origin of the $\{x, y, z\}$ frame to the ball's center of mass by the distance to the point where the ball touches the plate, the potential energy of the ball becomes

$$E_{\text{pot}} = mgz = -mgu\cos\alpha\sin\beta + mg(v+d)\sin\alpha.$$

Inserting everything into Equation 3.72 yields an expression for L that only contains the coordinates u and v.

$$\begin{aligned} L = {} & \tfrac{m}{2}\dot{u}^2 + \tfrac{m}{2}u^2\dot{\alpha}^2\sin^2\beta + \tfrac{m}{2}u^2\dot{\beta}^2 + \tfrac{m}{2}\dot{v}^2 + \tfrac{m}{2}(v+d)^2\dot{\alpha}^2 \\ & - m\dot{u}(v+d)\dot{\alpha}\sin\beta + mu\dot{v}\dot{\alpha}\sin\beta - mu(v+d)\dot{\alpha}\dot{\beta}\cos\beta \\ & + \tfrac{J}{2r^2}\dot{u}^2 + \tfrac{J}{2r^2}\dot{v}^2 + mgu\cos\alpha\sin\beta - mg(v+d)\sin\alpha \end{aligned}$$

The conditions

$$\begin{aligned} \tfrac{d}{dt}\tfrac{\partial L}{\partial \dot{u}} - \tfrac{\partial L}{\partial u} \overset{!}{=} 0 \\ \tfrac{d}{dt}\tfrac{\partial L}{\partial \dot{v}} - \tfrac{\partial L}{\partial v} \overset{!}{=} 0 \end{aligned}$$

lead to the desired equations of motion for the reduced model of a ball on a plate.

$$\begin{aligned} (1+\tfrac{J}{mr^2})\ddot{u} - u\dot{\beta}^2 - g\cos\alpha\sin\beta - u\dot{\alpha}^2\sin^2\beta - (v+d)\ddot{\alpha}\sin\beta - 2\dot{v}\dot{\alpha}\sin\beta = 0 \\ (1+\tfrac{J}{mr^2})\ddot{v} - (v+d)\dot{\alpha}^2 + g\sin\alpha + u\ddot{\alpha}\sin\beta + 2u\dot{\alpha}\dot{\beta}\cos\beta + 2\dot{u}\dot{\alpha}\sin\beta = 0 \end{aligned} \tag{3.73}$$

Note that Equation 3.69 is the same as Equation 3.73.

3.4 Comments on the Results

From the non-linear models of the Ball and Plate system derived in Sections 3.2 and 3.3, we know that the plant is not symmetric with respect to the inputs. The reason for this lies in the set-up of the experiment depicted in Figure 3.2. While the first rotational axis (the x-axis) is motionless, the second rotational axis (the v-axis) is rotated around the first one. Consequently, there is no duality between the two non-linear equations of motion. However, when we linearize these equations for small inclination angles α and β, they become mathematically equivalent.

It is interesting to see how the assumptions which the reduced model is based on, effect the resulting differential equations.

Compare Equation 3.52 to Equation 3.70 with $d = 0$.

$$(1 + \tfrac{J}{mr^2})\ddot{v} - v\dot{\alpha}^2 + g\sin\alpha - r\ddot{\alpha} - \tfrac{J}{mr^2}r\ddot{\alpha} = 0$$

$$(1 + \tfrac{J}{mr^2})\ddot{v} - v\dot{\alpha}^2 + g\sin\alpha = 0$$

Approximation 3.58 is responsible for the absence of the term $-r\ddot{\alpha}$ and Approximation 3.61 for the absence of the term $-\frac{J}{mr^2}r\ddot{\alpha}$ in the reduced equation of motion.

It remains to be checked whether the assumptions which led to the reduction, are permissible. The smaller the ball, the better the approximation $r_C \approx r_B$ and the approximation $\omega_w \approx 0$ is justified by Equation 3.57. The exact value of $-\frac{J}{mr^2}$ depends on the ball, but is significantly smaller than 1. Both approximations naturally rely on relatively small angular accelerations $\ddot{\alpha}$.

Chapter 4

Controller Design

Because of the duality between the two differential equations that describe the linearized Ball and Plate system, we can start by designing a controller for the Ball and Beam system and then apply two of these controllers to the Ball and Plate system.

4.1 Time Response of the Plant

Before we can design a controller, we need to understand the behaviour of the plant itself. In our case, analysis of the time response is more appropriate than the frequency response because the latter is only applicable to linear systems.

Comparison of the Non-Linear and the Linearized System

Since our techniques for regulator design rely on linear models, comparing the non-linear and the linear model of the plant is of particular interest.

With the introduction of the constant

$$c = \frac{mr^2}{mr^2 + J}$$

we can write the equations of motion 3.12 and 3.13 for the non-linear and the linearized model, respectively, in a more convenient way.

$$\ddot{u} = c(u\dot{\alpha}^2 - g\sin\alpha) + r\ddot{\alpha} \tag{4.1}$$

$$\ddot{u} = -cg\alpha + r\ddot{\alpha} \tag{4.2}$$

For the following simulations with the plant alone, we assume a hollow ball of radius $r = 2$ cm and mass $m = 10$ g, which leads to the constant $c = 0.6$. Note that c only depends on whether the ball is solid or hollow. The initial velocity of the ball will always be zero.

Figure 4.1 shows a SIMULINK representation of both the non-linear and the linearized system. Because the inclination angle α is differentiated, we cannot excite the system with an ordinary step function. Instead we take the smoothened step function depicted in Figure 4.2, which has a steady first derivative and a bounded second derivative. T is the rise time.

$$u(t) = \begin{cases} 0 & (t \leq 0) \\ \frac{1}{2}(1 - \cos\frac{\pi}{T}t) & (0 \leq t \leq T) \\ 1 & (t \geq T) \end{cases}$$

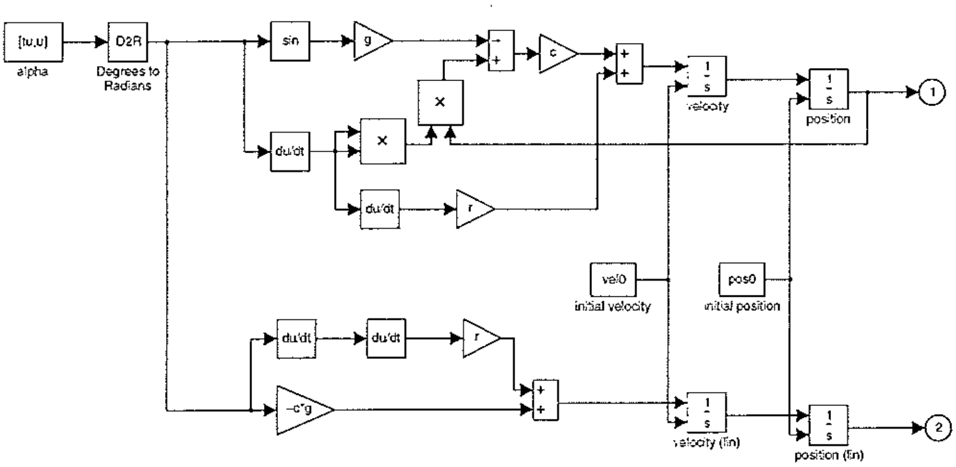


Figure 4.1: The non-linear and the linearized plant.

The exact differentiation was approximated by a high-pass filter.

$$s \quad \longrightarrow \quad \frac{Ns}{s+N} \quad \text{with } N = 50\ldots100$$

If the initial position of the ball is non-zero, there is a considerable difference between the response of the non-linear and the linearized system. In Figure 4.5, we recognize the effect of the centrifugal term $u\dot{\alpha}^2$ in Equation 4.1, which has vanished in Equation 4.2.

Next, we excite the plant with the smoothened pulse shown in Figure 4.3. Unlike in the previous case, the inclination angle is reset to zero, so that the ball is not accelerated further. Figure 4.6 shows the response.

Our last excitation function is one period of a smoothened square wave, depicted in Figure 4.4. Since the plant is time-invariant, all the energy which is transferred to the linear system by the positive pulse, is removed again by its negative counterpart, so that the position u remains constant after the transient phase. In reality, in the non-linear system, the ball keeps rolling beyond the end of the excitation. Figure 4.7 illustrates this difference.

A MATLAB script for the above simulations can be found in Appendix D.

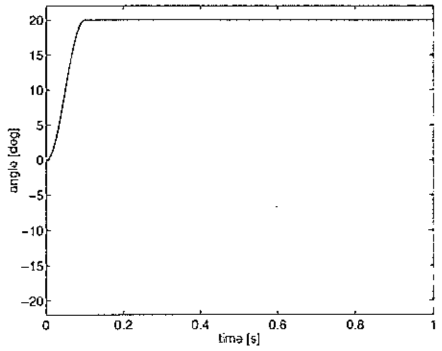


Figure 4.2: Smoothened step.

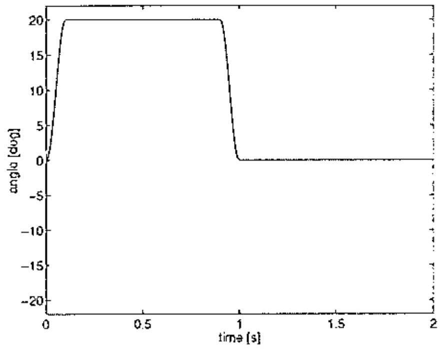


Figure 4.3: Smoothened square pulse.

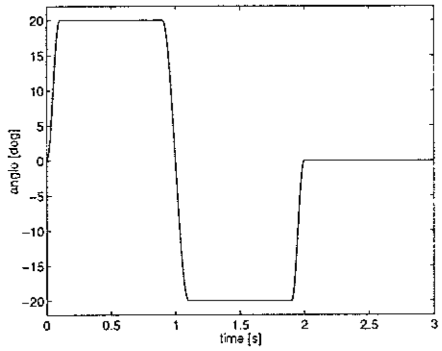


Figure 4.4: Smoothened two-pulse square wave.

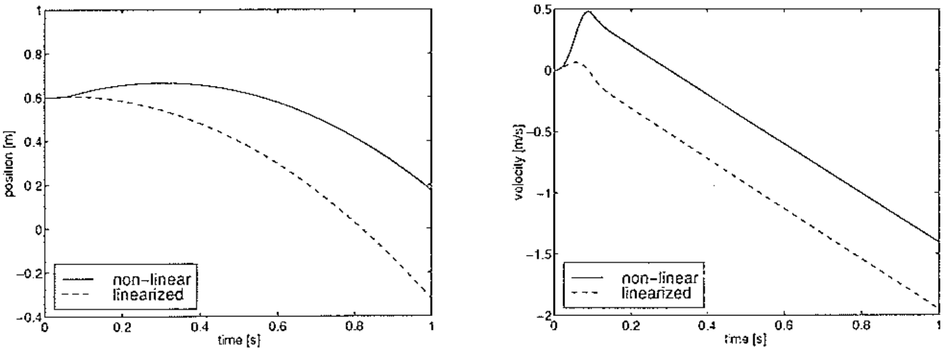


Figure 4.5: Step responses of the non-linear and the linearized system.

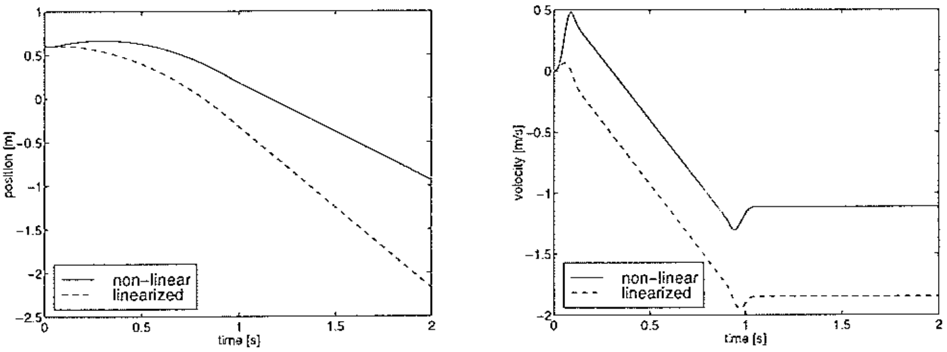


Figure 4.6: Pulse responses of the non-linear and the linearized system.

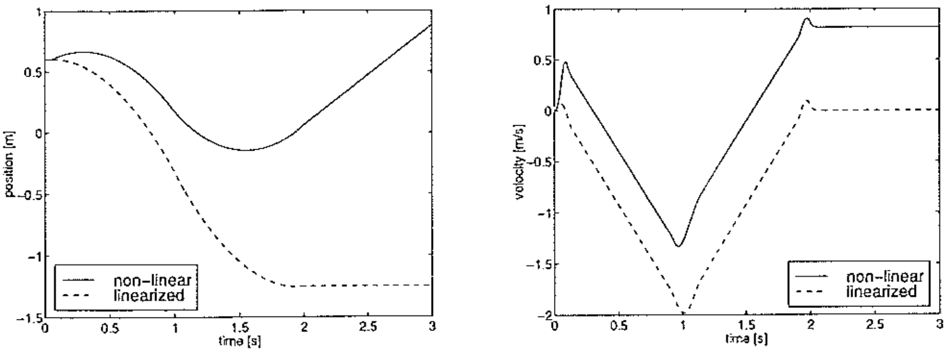


Figure 4.7: Wave responses of the non-linear and the linearized system.

Comparing wave responses which result from different simulation parameters gives additional insight into the non-linear nature of the plant. We reduce the amplitude of the inclination angle from 20° to 10°. The results are depicted in Figure 4.8.

An interesting effect can be observed in Figure 4.9, where the initial position of the ball was decreased from 60 cm to 40 cm: the ball will roll in the opposite direction after the transient phase.

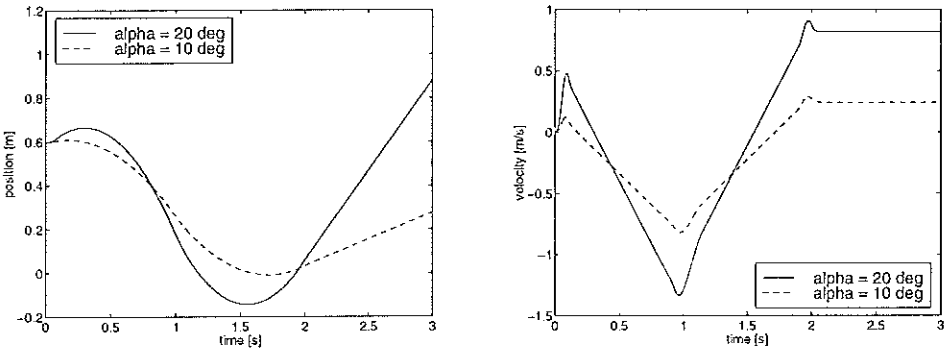


Figure 4.8: Wave response of the nonlinear system for $\hat{\alpha} = 20^\circ$ and $\hat{\alpha} = 10^\circ$.

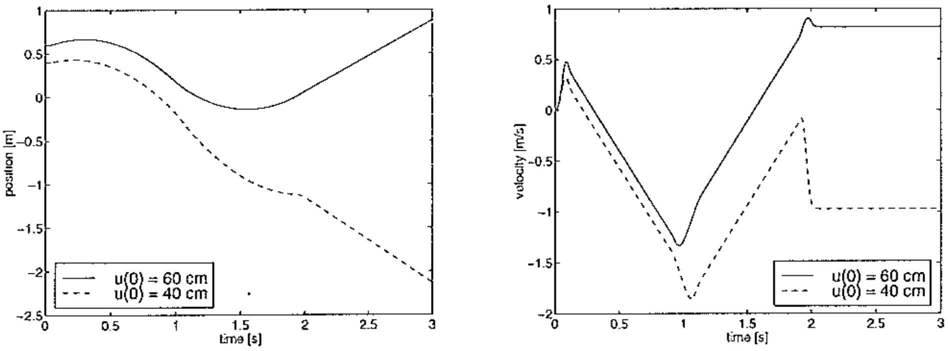


Figure 4.9: Wave response of the nonlinear system for $u(0) = 60$ cm and $u(0) = 40$ cm.

Remember that the primary cause for non-linear effects are centrifugal forces. Neglecting them is equivalent to considering a system with very limited angular velocities $\dot{\alpha}$. The approximation $\sin\alpha \approx \alpha$ is less ticklish for small values of α.

4.2 Design of a Continuous-Time Controller

In Section 3.4 the differences between the full and the reduced model of the Ball and Beam system were discussed. We shall base the design of all our regulators on the reduced linearized model, i.e. we neglect the term $r\ddot{\alpha}$ in Equation 4.2. Hence, the transfer function of our plant becomes a pure double integrator.

$$P(s) = -cg\frac{1}{s^2} \tag{4.3}$$

If we ran the simulations in Section 4.1 again with the reduced model, we would indeed notice merely minor differences in the transient responses.[1]

P Controller

The root locus curve with positive feedback in Figure 4.10 shows that it is impossible to stabilize the plant with a proportional controller.

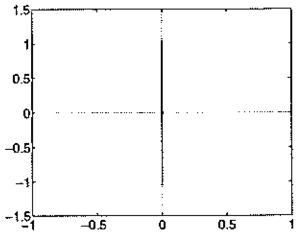


Figure 4.10: Root locus curve for the linearized plant.

PD Controller

With the PD controller in the feedback path depicted in Figure 4.11, the two pole locations of the closed-loop system can be selected arbitrarily by choosing the proper values of k_P and k_D.

$$\frac{Y(s)}{R(s)} = \frac{-cg\frac{1}{s^2}}{1 + (k_P + k_D s)(-cg\frac{1}{s^2})} = \frac{-cg}{s^2 - k_D cgs - k_P cg} \tag{4.4}$$

State Feedback

When all the elements of the state vector are at our disposal, the poles can be set likewise by the state feedback controller in Figure 4.12.

$$\frac{Y(s)}{R(s)} = \frac{-cg\frac{1}{s^2}}{1 + (-k_2 cg\frac{1}{s}) + (-k_1 cg\frac{1}{s^2})} = \frac{-cg}{s^2 - k_2 cgs - k_1 cg} \tag{4.5}$$

[1] Equation 4.2 describes a non-minimum-phase system with a zero at $\sqrt{cg/r}$. This right-half plane zero causes the small under- and overshoots in the velocity responses of the linear system on page 34.

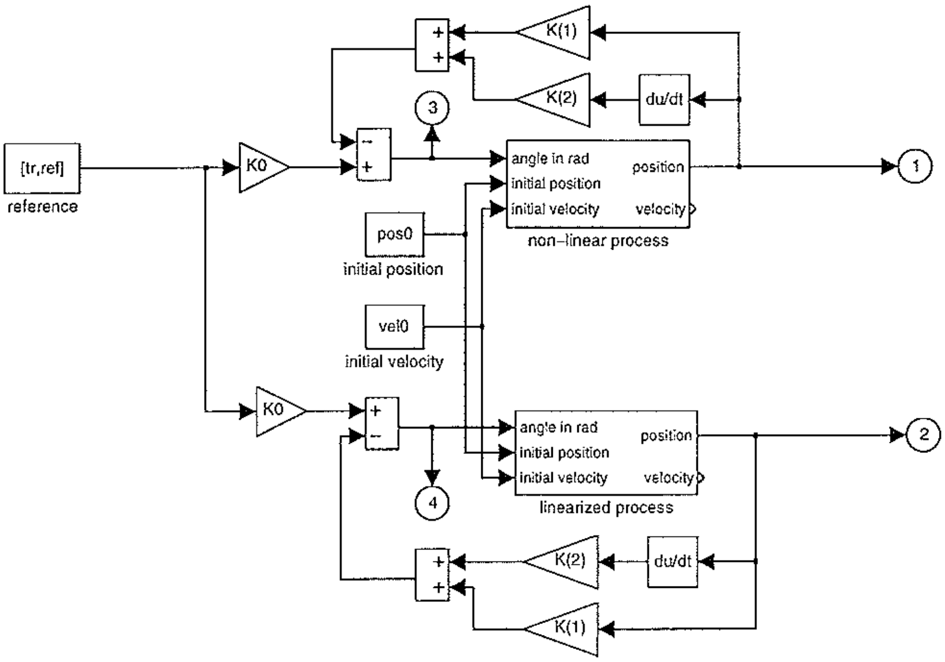


Figure 4.11: The non-linear and the linearized process with a PD controller.

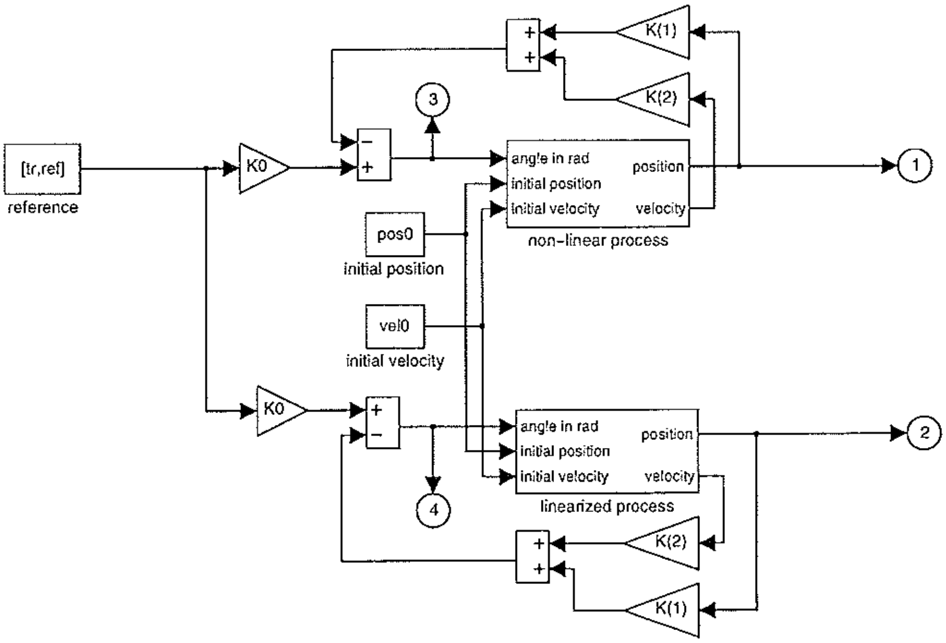


Figure 4.12: The non-linear and the linearized process with state feedback.

Comparison of Equations 4.4 and 4.5 shows that PD control and state feedback are equivalent for our system.

$$\begin{aligned} k_1 &= k_P \\ k_2 &= k_D \end{aligned}$$

Both gains must be negative for stability in the closed-loop system.

Since the velocity of the ball is not sensed directly, we will have to use an estimator for the state vector. The estimator can be designed separately, though, and we continue with the control law design for full state feedback.

By defining the two states

$$\begin{aligned} x_1 &= y \\ x_2 &= \dot{y} \end{aligned} \tag{4.6}$$

we gather a state-space representation of the transfer function in Equation 4.3.

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \tag{4.7}$$

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 0 \\ -cg \end{bmatrix} \qquad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{4.8}$$

With the control law

$$u = -Kx + k_0 r \tag{4.9}$$

the closed-loop system becomes

$$\dot{x} = (A - BK)x + Bk_0 r.$$

The static prefilter k_0 is computed so that the static gain of the closed-loop system becomes 1.

$$\frac{Y(s)}{R(s)} = C(sI - A + BK)^{-1}Bk_0 \xrightarrow{!} 1 \quad (s \to 0)$$

$$k_0 = \frac{1}{C(BK - A)^{-1}B} \tag{4.10}$$

The first step is to find the feedback vector K in Equation 4.9. Different design methods exist for this problem. Some of them are discussed below.

Time-Domain Specifications

The operating range of actuators is often limited. It is thus possible to iterate the design of K until we have satisfactory response behaviour without the actuators going into saturation. In our case, the inclination angle, which is the input value of the plant, is not a priori limited. This method is therefore not particularly advantageous.

A more expressive quantity is the rise time of the output value in response to a step excitation of the system. A difficulty is imposed by the differentiations in the non-linear model in Figure 4.1 which will be used for the simulations.[2] They prevent us from applying an unsteady function like a step. Smoother excitation functions circumvent the problem of infinite derivatives. Such functions are exhaustively covered in Appendix C. We choose as our excitation function the step response of a second-order system with damping $\zeta_u = 0.9$ and rise time $T_u = 0.1$ s, where T_u is the time when the tangent through the inflection point reaches the step height. See Figure 4.13 for the looks of it. The reason why we chose a reference step from 40 to 50 cm rather than from 0 to 10 cm is the distance d in Figure 3.2, which has a value of 23 cm, and the desire for a worst-case analysis with regard to non-linear effects, which are greater far away from the rotation center.

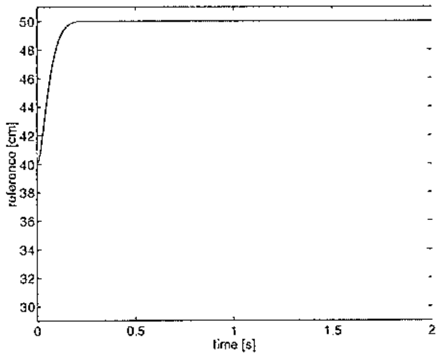


Figure 4.13: Smooth step-like excitation function.

For the closed-loop system, we specify the rise time T and the damping ratio ζ and select the natural frequency ω_n accordingly using Equation C.13.

$$\left.\begin{array}{r} T = 0.5 \text{ s} \\ \zeta = 0.7 \end{array}\right\} \Longrightarrow \omega_n = 5 \text{ rad/s}$$

The closed-loop characteristic polynomial $s^2 + 2\zeta\omega_n s + \omega_n^2$ has the roots $-3.52 \pm 3.59j$. We use Ackermann's formula to compute the corresponding state feedback gains $k_1 = -4.29$ and $k_2 = -1.20$, and Equation 4.10 to compute the static prefilter $k_0 = -4.29$. The resulting bandwidth is 5.1 rad/s. Figure 4.14 shows the transient response of the closed-loop system.

[2] The differentiations in the linear model were discarded when we formulated the transfer function $P(s)$ in Equation 4.3.

Prototype Design

For a given system, a step response can be worked out to minimize the integral of the time multiplied by the absolute value of the error (ITAE criterion).

$$J = \int_0^\infty t|e(t)|\,\mathrm{d}t$$

Depending on the order of the system, a set of pole locations is evaluated such that the loss function J is minimized. For a second-order system, the poles must be placed at

$$-\frac{\omega_0}{\sqrt{2}} \pm \frac{\omega_0}{\sqrt{2}}j,$$

where ω_0 is the desired cut-off frequency. We choose

$$\omega_0 = 4\ \mathrm{rad/s}.$$

Then the state feedback gains are $k_1 = -2.72$ and $k_2 = -0.96$, the static prefilter $k_0 = -2.72$ and the bandwidth 4.0 rad/s. When we compare the transient responses of this system in Figure 4.15 with the responses in Figure 4.14, we see that there is significantly less overshoot and a much better matching of the behaviour of the linearized model with the non-linear model. On the other hand, the settling time is prolonged.

When overshoot must be avoided altogether, we can choose the characteristic closed-loop polynomial to be equal to the nth-degree Bessel polynomial. For a second-order system, the roots of the polynomial are

$$(-0.8660 \pm 0.5000j)\omega_0.$$

For the cut-off frequency $\omega_0 = 4$ rad/s we get the state feedback gains $k_1 = -2.72$ and $k_2 = -1.18$, and the static prefilter $k_0 = -2.72$. The bandwidth of this system is only 3.1 rad/s, whereas it was 4.0 rad/s in the ITAE criterion design. This means that, given the same value of ω_0, the ITAE prototype has a higher bandwidth for the same attenuation at higher frequencies. Therefore, by selecting ω_0 to give the same bandwidths, we achieve reduced sensitivity to sensor noise at high frequencies with the ITAE prototype. Notice in Figure 4.16 that, as requested, there is no overshoot in the output signal, and that the settling time is reduced.

Frequency-Domain Specifications

We could also specify the parameters of the characteristic polynomial directly. We would e.g. specify the damping ratio and vary the natural frequency in order to achieve a certain bandwidth.

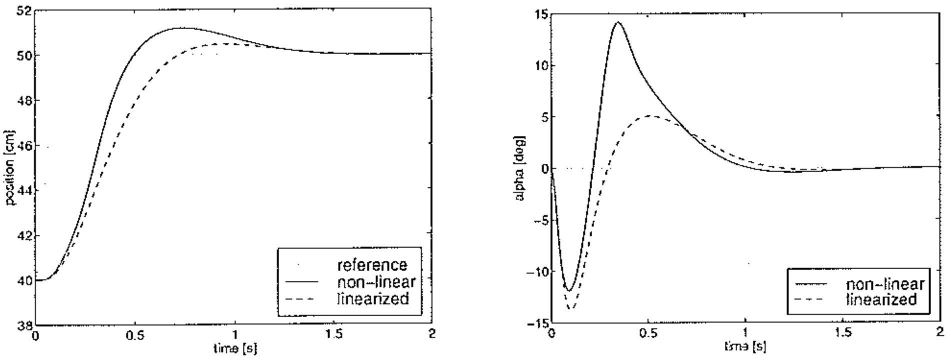


Figure 4.14: Input and output signals of the regulated plant with specified rise time $T = 0.5$ s and damping ratio $\zeta = 0.7$.

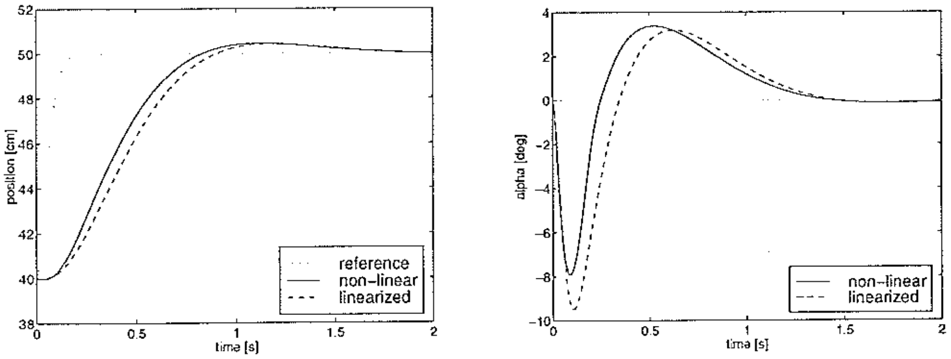


Figure 4.15: Input and output signals of the regulated plant with cut-off frequency $\omega_0 = 4$ rad/s and ITAE criterion.

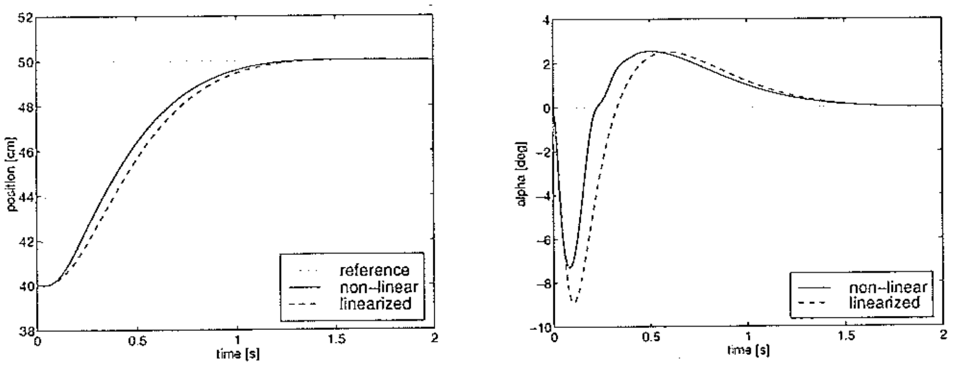


Figure 4.16: Input and output signals of the regulated plant with cut-off frequency $\omega_0 = 4$ rad/s and Bessel polynomial design.

Optimal Linear Quadratic Regulator

The LQR problem is to find the state feedback vector K such that

$$J = \int_0^\infty \Big(\boldsymbol{x}'(t)\boldsymbol{Q}\boldsymbol{x}(t) + \boldsymbol{u}(t)\boldsymbol{R}\boldsymbol{u}(t)\Big)\,\mathrm{d}t \tag{4.11}$$

is minimized for the system in Equation 4.7. $\boldsymbol{Q}$ and $\boldsymbol{R}$ are matrices which weight the tracking error $\boldsymbol{x}$ with respect to the control effort $\boldsymbol{u}$. To reduce the number of design parameters, we choose

$$Q = C'C.$$

In the single-input single-output case, this choice of Q together with

$$\boldsymbol{R} = 1$$

turns Equation 4.11 into

$$J = \int_0^\infty \Big(Qy^2(t) + u^2(t)\Big)\,\mathrm{d}t,$$

where the scalar Q is the only design parameter. We wish to find a compromise between a fast response (small values of $\boldsymbol{x}$ for a zero reference) and a low control effort (small values of u). The larger Q, the faster the system and the higher the bandwidth. Consider

$$Q = 10$$

and solve the LQR problem. The resulting state feedback gains are $k_1 = -3.16$ and $k_2 = -1.04$. The static prefilter is $k_0 = -3.16$ and the bandwidth becomes 4.3 rad/s. See Figure 4.17 for the transient responses of the system.

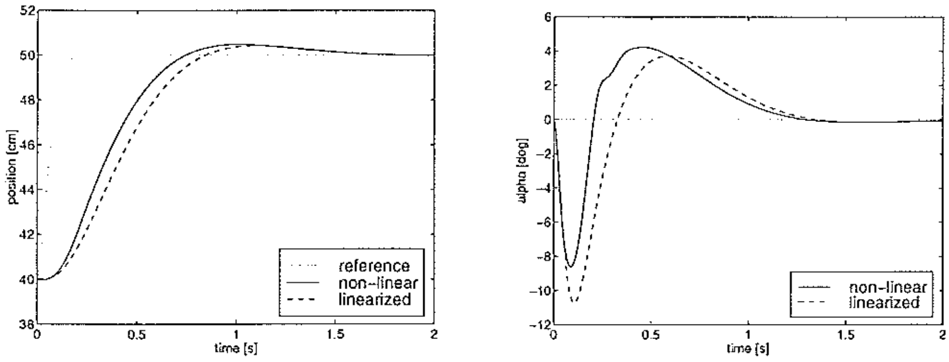


Figure 4.17: Input and output signals of the regulated plant with output weighting factor Q=10 and LQR design in response to a step excitation.

Instead of a step-like excitation, we might as well have a sinusoid input. indexexcitation function!sinusoid We take a magnitude of 10 cm and a frequency of 0.5 Hz. The position response in Figure 4.18 has a phase lag and an attenuated amplitude of 8.8 cm, which corresponds directly to the magnitude gain of 0.88 at the frequency π rad/s.

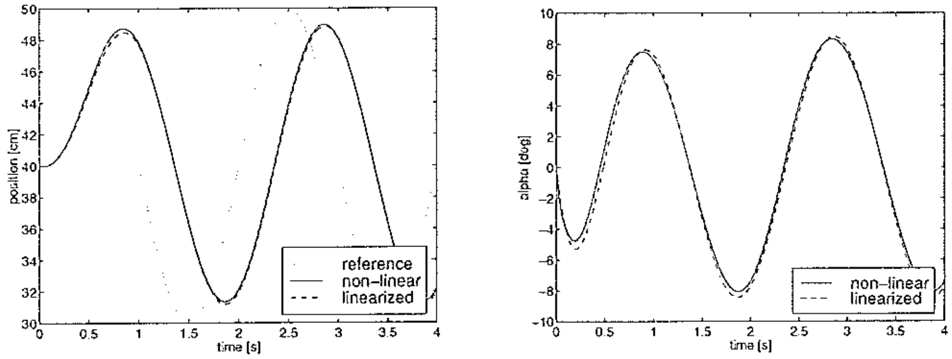


Figure 4.18: Input and output signals of the regulated plant with output weighting factor Q=10 and LQR design in response to a sinusoidal excitation of 0.5 Hz.

Choosing $Q = 5$ in the LQR design reduces the bandwidth of the closed-loop system to 3.6 rad/s.

Appendix D contains a MATLAB script for all the simulations.

State Feedback with Additional Integrator and Feedforward

Consider the perturbed plant in Figure 4.19. There are three disturbance inputs of different nature:

Load disturbance d: Within this category fall rolling friction and external disturbances such as somebody blowing at the ball.

Actuator error v: Although the actuator is calibrated, a minor error will always remain. Typically, this error is constant.

Measurement noise w: The optical system of the camera as well as the transmission link add noise to the actual signal.

We investigate the transfer functions from the disturbances to the ball position. For a principal result, it is sufficient to use

$$P(s) = \frac{1}{s^2} \quad \text{instead of the true} \quad -cg\frac{1}{s^2}.$$

In reality, the compensator in Figure 4.19 is either a PD controller with a low-pass prefilter for the D part or a state estimator with state feedback, but we assume that the plant is regulated by a full state feedback or, equivalently, a PD controller.

$$C(s) = 1 + sT_D$$

$$\frac{Y(s)}{D(s)} = \frac{1}{1+P(s)C(s)} = \frac{1}{1+\frac{1}{s^2}(1+sT_D)} = \frac{s^2}{s^2+sT_D+1} \longrightarrow 0 \quad (s \to 0)$$

$$\frac{Y(s)}{V(s)} = \frac{P(s)}{1+P(s)C(s)} = \frac{\frac{1}{s^2}}{1+\frac{1}{s^2}(1+sT_D)} = \frac{1}{s^2+sT_D+1} \longrightarrow 1 \quad (s \to 0)$$

$$\frac{Y(s)}{W(s)} = \frac{P(s)C(s)}{1+P(s)C(s)} = \frac{\frac{1}{s^2}(1+sT_D)}{1+\frac{1}{s^2}(1+sT_D)} = \frac{1+sT_D}{s^2+sT_D+1} \longrightarrow 1 \quad (s \to 0)$$

We observe that, due to the integral character of the plant, we need not be concerned about tracking errors resulting from a constant output disturbance, but the typical constant input disturbance on the actuator causes a steady-state regulation error. This error can be compensated by an additional integrator.

$$C(s) = 1 + \frac{1}{sT_I} + sT_D$$

$$\begin{aligned}\frac{Y(s)}{D(s)} &= \frac{1}{1+P(s)C(s)} = \frac{1}{1+\frac{1}{s^2}(1+\frac{1}{sT_I}+sT_D)} \\ &= \frac{s^3T_I}{s^3T_I+s^2T_IT_D+sT_I+1} \longrightarrow 0 \quad (s \to 0)\end{aligned}$$

$$\begin{aligned}\frac{Y(s)}{V(s)} &= \frac{P(s)}{1+P(s)C(s)} = \frac{\frac{1}{s^2}}{1+\frac{1}{s^2}(1+\frac{1}{sT_I}+sT_D)} \\ &= \frac{sT_I}{s^3T_I+s^2T_IT_D+sT_I+1} \longrightarrow 0 \quad (s \to 0)\end{aligned}$$

$$\begin{aligned}\frac{Y(s)}{W(s)} &= \frac{P(s)C(s)}{1+P(s)C(s)} = \frac{\frac{1}{s^2}(1+\frac{1}{sT_I}+sT_D)}{1+\frac{1}{s^2}(1+\frac{1}{sT_I}+sT_D)} \\ &= \frac{s^2T_IT_D+sT_I+1}{s^3T_I+s^2T_IT_D+sT_I+1} \longrightarrow 1 \quad (s \to 0)\end{aligned}$$

Constant measurement offset is still not compensated, but this should not be a problem because the sensor noise w typically consists of higher frequencies.

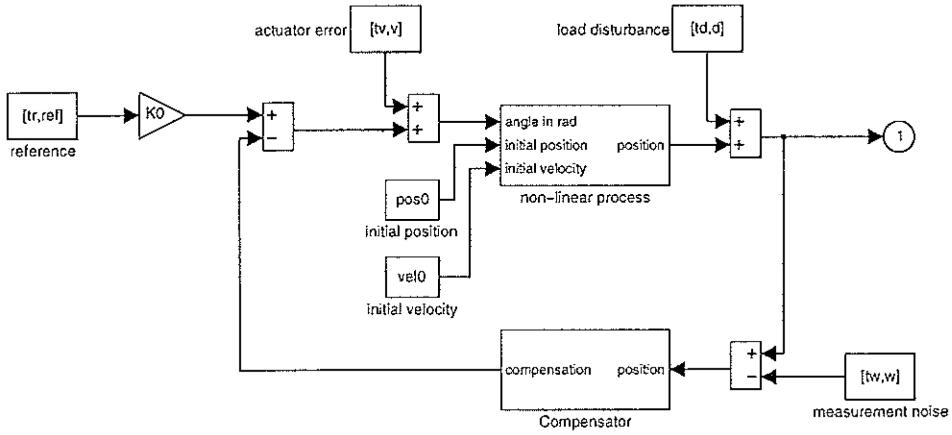


Figure 4.19: The perturbed non-linear process with a compensator.

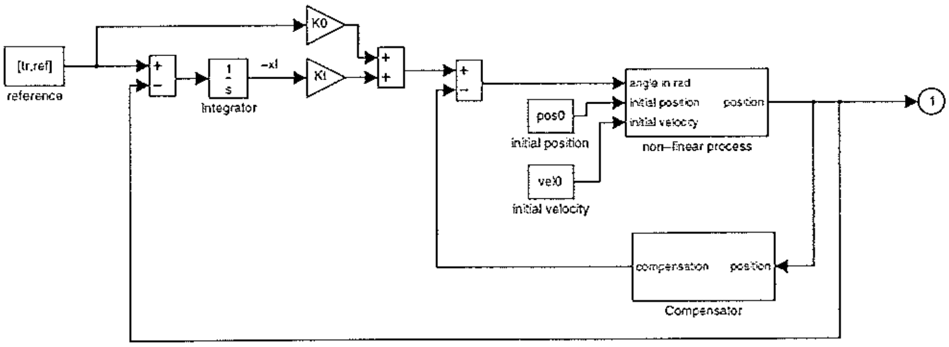


Figure 4.20: The non-linear process with a compensator and an additional integrator with reference feedforward.

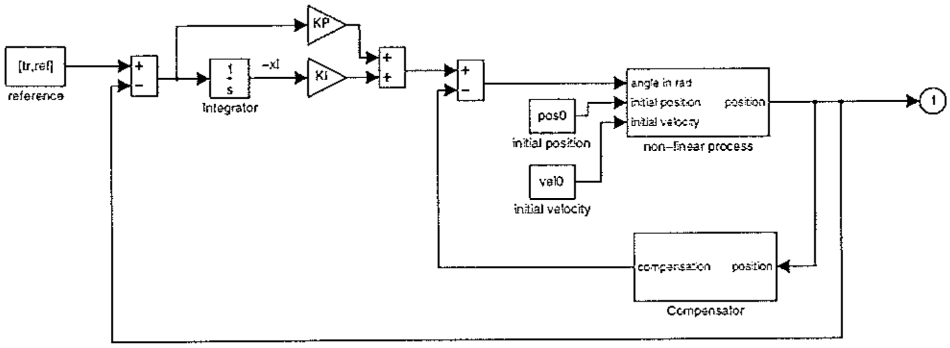


Figure 4.21: The non-linear process with a compensator and an additional PI controller.

In addition to the compensator, the system in Figure 4.20 contains a feedback loop for the tracking error. Instead of a PD controller, we use the notation for full state feedback, although the velocity of the ball cannot be measured directly.

$$\begin{aligned} \dot{x} &= Ax + Bu \\ \dot{x}_I &= y - r = Cx - r \\ u &= -Kx - k_I x_I + k_0 r \end{aligned} \tag{4.12}$$

The integrator adds another state to the plant. The state-space description of the open-loop system is written with an extended state vector.

$$\begin{bmatrix} \dot{x} \\ \dot{x}_I \end{bmatrix} = \underbrace{\begin{bmatrix} A & 0 \\ C & 0 \end{bmatrix}}_{A_I} \begin{bmatrix} x \\ x_I \end{bmatrix} + \underbrace{\begin{bmatrix} B \\ 0 \end{bmatrix}}_{B_I} u \tag{4.13}$$

$$y = \underbrace{\begin{bmatrix} C & 0 \end{bmatrix}}_{C_I} \begin{bmatrix} x \\ x_I \end{bmatrix}$$

The closed-loop system becomes

$$\begin{bmatrix} \dot{x} \\ \dot{x}_I \end{bmatrix} = \begin{bmatrix} A - BK & -Bk_I \\ C & 0 \end{bmatrix} \begin{bmatrix} x \\ x_I \end{bmatrix} + \begin{bmatrix} Bk_0 \\ -1 \end{bmatrix} r.$$

A pole-placement function in MATLAB such as acker will return the feedback vector $[K \; k_I]$.

The feedforward from the reference to the process input improves the tracking performance: the direct influence of the reference is much faster than the reaction of the integrator to the tracking error. The static prefilter k_0 is the same as in Equation 4.10.

State Feedback with Additional PI Controller

Faster response to the tracking error is achieved by adding a proportional part as in Figure 4.21.

$$\begin{aligned} \dot{x} &= Ax + Bu \\ \dot{x}_I &= y - r = Cx - r \\ u &= -Kx - k_I x_I - k_P(Cx - r) \end{aligned} \tag{4.14}$$

The state-space representation of the open-loop system remains the same as in Equation 4.13, only the feedback law changes. Compare Equations 4.12 and 4.14. The closed-loop system becomes

$$\begin{bmatrix} \dot{x} \\ \dot{x}_I \end{bmatrix} = \begin{bmatrix} A - B\tilde{K} & -Bk_I \\ C & 0 \end{bmatrix} \begin{bmatrix} x \\ x_I \end{bmatrix} + \begin{bmatrix} Bk_P \\ -1 \end{bmatrix} r,$$

where

$$\tilde{K} = K + k_P C.$$

Note that pole-placement functions return the vector $\tilde{K}$ when passed A_I and B_I as arguments. We choose k_P to be similar to k_0 in Equation 4.10

$$k_P = \frac{1}{C(B\tilde{K} - A)^{-1}B}$$

and compute K from $\tilde{K}$ and k_P.

$$K = \tilde{K} - k_P C$$

4.3 The Discretized Plant

State-Space Description

The controller for our process will be implemented on a computer system. While it would be possible to derive a digital controller from a continuous controller by emulation (e.g. Euler or Tustin approximation), we expect better results from a *direct digital design* based on a discrete model of the plant. By a rule of thumb [8], emulation design yields reasonable results at sample rates on the order of 20 times the bandwidth of the plant. Our plant was defined in Equation 4.3 and has a bandwidth of about 3 rad/s. The sample rate is limited to 25 rad/s (4 Hz) by the image processing architecture. Hence the extra effort to design a discrete-time controller from scratch is justified. The results from Section 4.2 will still be helpful to choose the parameters of the discrete-time controller.

The digital controller holds the analog output signal (inclination angle) until a new value is commanded. This is called zero-order-hold. It is then natural to choose the sampling instants of the analog signal (the position of the ball) at the times when the control changes. We introduce the zero-order-hold equivalent of Equation 4.7.

$$\begin{aligned} x(k+1) &= \Phi x(k) + \Gamma u(k) \\ y(k) &= Cx(k) \end{aligned} \tag{4.15}$$

$$\Phi = e^{AT} \qquad \Gamma = \int_0^T e^{At} B \, \mathrm{d}t \qquad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{4.16}$$

The evaluation of a matrix exponential can be done in different ways, for instance by defining the function

$$\phi(t) = e^{At},$$

which is subject to the differential equation

$$\dot{\phi}(t) = A\phi(t) \quad \text{with} \quad \phi(0) = I. \tag{4.17}$$

Compute the Laplace transform of the equation with initial condition 4.17,

$$s\Phi(s) - \phi(0) = A\Phi(s)$$

solve for $\Phi(s)$

$$\Phi(s) = (sI - A)^{-1}$$

and find the corresponding time function.

$$\phi(t) = \mathcal{L}^{-1}\{\, (sI - A)^{-1}\, \}$$

For the parameters in Equations 4.8, we get the following results.

$$\Phi(s) = \begin{bmatrix} \frac{1}{s} & \frac{1}{s^2} \\ 0 & \frac{1}{s} \end{bmatrix}$$

$$\phi(t) = \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix}$$

Note that $\phi(T) = \Phi$ and evaluate Equations 4.16.

$$\Phi = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \qquad \Gamma = -cg \begin{bmatrix} T^2/2 \\ T \end{bmatrix} \qquad C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{4.18}$$

Transfer Function

For a plant described by $P(s)$ and preceded by a zero-order-hold, the discrete transfer function is

$$P(z) = (1 - z^{-1})\,\zeta\{\,\frac{P(s)}{s}\,\},$$

where $\zeta\{\, F(s)\,\}$ is the z-transform of the sampled time series whose Laplace transform is $F(s)$. Extensive tables [1] exist for ζ-transforms.

We define the transfer functions P_1 and P_2 from the input to the first and the second state, respectively. Recall from Equations 4.6 that x_1 is the position and x_2 the velocity of the ball.

$$P_1(s) = -cg\frac{1}{s^2} \xrightarrow{\zeta} P_1(z) = -cg\frac{T^2}{2}\frac{z+1}{(z-1)^2} \tag{4.19}$$

$$P_2(s) = -cg\frac{1}{s} \xrightarrow{\zeta} P_2(z) = -cg\frac{T}{z-1} \tag{4.20}$$

$P(z) = P_1(z)$ can also be expressed by the parameters in Equation 4.18.

$$P(z) = C(zI - \Phi)^{-1}\Gamma$$

Compare Equations 4.19 and 4.20 with the discrete-time model in Figure 4.22. When the two models are excited with the wave in Figure 4.23, they produce the responses in Figure 4.24. The initial position is 60 cm, the initial velocity 0 m/s.

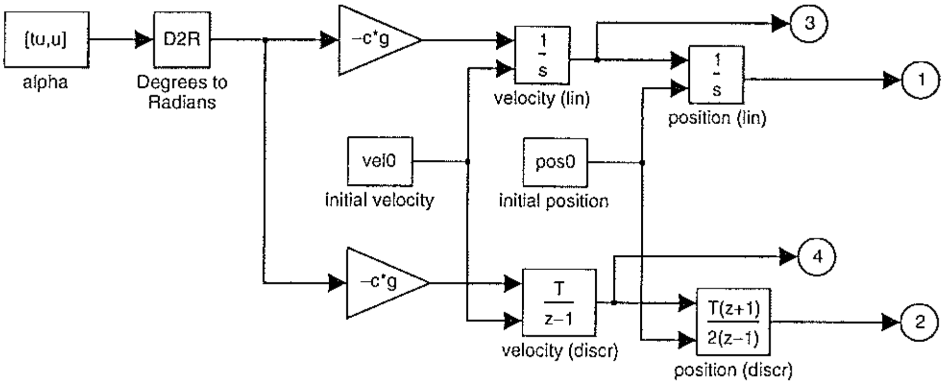


Figure 4.22: Linearized continuous- and discrete-time models of the plant.

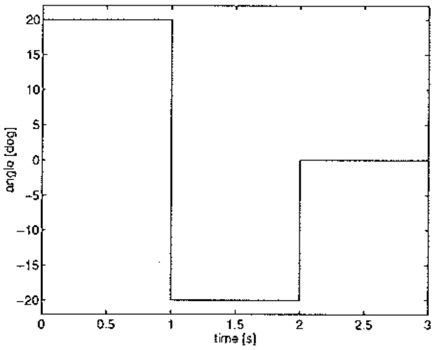


Figure 4.23: Wave excitation function for the linearized continuous and sampled systems in Figure 4.22.

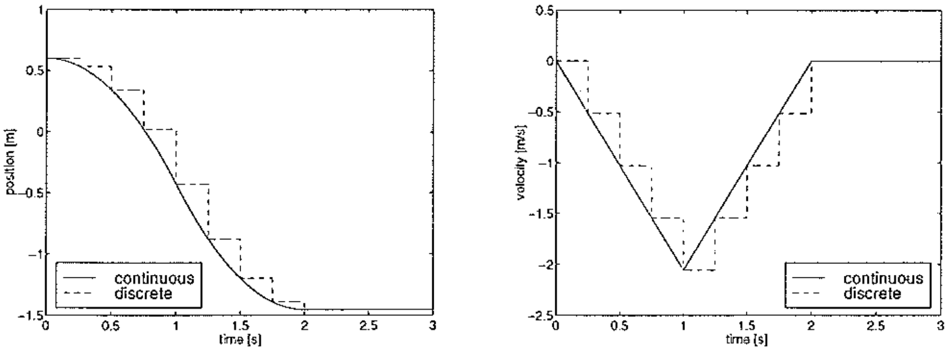


Figure 4.24: The wave responses illustrate the effect of the zero-order-hold.

4.4 Design of a Discrete-Time Controller

The computation of the feedback gain K is the same as in continuous time. An additional strategy for discrete-time systems is deadbeat control: the desired poles are all chosen to be at the origin. After an impulse disturbance, all states will be driven to zero in at most n steps, where n is the order of the system. There is no corresponding feature for continuous-time systems.

Time-Shift Operator

For convenient manipulation of linear difference equations with constant coefficients, we define the *forward-shift* operator q. It hast the property

$$q\, f(k) = f(k+1),$$

where f is a doubly infinite sequence. The inverse of the forward-shift operator is called *backward-shift* operator and is denoted by q^{-1}.

$$q^{-1}\, f(k) = f(k-1)$$

State Feedback with Integrator

The integrator in Figure 4.20 was introduced to compensate for an actuator error.[3] This concept works just as well in a discrete-time system, where the integrator is replaced by its zero-order-hold equivalent. Notice that we do not sample the system in Equation 4.13 because the integrator contained therein is discretized separately. Distinguish between the digitally implemented integrator and the continuous physical process.

$$\begin{aligned} qx &= \Phi x + \Gamma u \\ qx_I &= x_I + T(y-r) \;=\; x_I + TCx - Tr \\ u &= -Kx - k_I x_I + k_0 r \end{aligned} \tag{4.21}$$

The integrator adds another state to the plant. The state-space description of the open-loop system is written with an extended state vector.

$$q\begin{bmatrix} x \\ x_I \end{bmatrix} = \underbrace{\begin{bmatrix} \Phi & 0 \\ TC & 1 \end{bmatrix}}_{\Phi_I} \begin{bmatrix} x \\ x_I \end{bmatrix} + \underbrace{\begin{bmatrix} \Gamma \\ 0 \end{bmatrix}}_{\Gamma_I} u$$

$$y = \underbrace{\begin{bmatrix} C & 0 \end{bmatrix}}_{C_I} \begin{bmatrix} x \\ x_I \end{bmatrix}$$

[3] The direct path from y to u in Figure 4.21 implies that $u(k)$ depends on $y(k)$, which would request that the time it takes to measure y and compute u is very small compared to the sampling interval. This is not the case with our process, where the measurement of y by means of Computer Vision actually takes up most of the time in a sampling period.

The control law in Equation 4.21 is the same as the continuous-time control law in Equation 4.12 and the closed-loop system becomes

$$q\begin{bmatrix} x \\ x_I \end{bmatrix} = \begin{bmatrix} \Phi - \Gamma K & -\Gamma k_I \\ TC & 1 \end{bmatrix} \begin{bmatrix} x \\ x_I \end{bmatrix} + \begin{bmatrix} \Gamma k_0 \\ -T \end{bmatrix} r.$$

A pole-placement function in MATLAB such as acker will return the feedback vector $[K\ k_I]$.

The static prefilter k_0 is computed so that the static gain of the closed-loop system becomes 1. We take the case where $x_I = 0$.

$$\frac{Y(s)}{R(s)} = C(qI - \Phi + \Gamma K)^{-1}\Gamma k_0 \overset{!}{=} 1 \quad (qx = x)$$

$$k_0 = \frac{1}{C(I - \Phi + \Gamma K)^{-1}\Gamma} \tag{4.22}$$

Estimator

Since the velocity of the ball is unavailable for measurement, we have to estimate the states of the plant in order to be able to use full state feedback. The notation $\hat{x}(k+1|k)$ is used to indicate that the estimate of $x(k+1)$ is based on measurements up to time k, that is, a one-step prediction.

$$\begin{aligned} \hat{x}(k+1|k) &= \Phi\,\hat{x}(k|k-1) + \Gamma u(k) + L\Big(y(k) - C\,\hat{x}(k|k-1)\Big) \\ &= (\Phi - LC)\,\hat{x}(k|k-1) + \Gamma u(k) + Ly(k) \qquad (4.23) \end{aligned}$$

To determine L we introduce the reconstruction error

$$e_x = x - \hat{x}.$$

Subtraction of Equation 4.23 from the first line of Equations 4.15 gives

$$\begin{aligned} e_x(k+1|k) &= \Phi\Big(x(k) - \hat{x}(k|k-1)\Big) - LC\Big(x(k) - \hat{x}(k|k-1)\Big) \\ &= (\Phi - LC)\,e_x(k|k-1). \qquad (4.24) \end{aligned}$$

Hence if L is chosen so that the system in Equation 4.24 is asymptotically stable, the error e_x will always converge to zero.

The determination of the estimator gain L is the same mathematical problem as determining the feedback gain K. If the estimator gain is chosen so that all eigenvalues of the matrix $\Phi - LC$ are zero, we get a deadbeat observer and the estimation error goes to zero in at most n steps after an impulse disturbance.

Notice that we do not need to estimate the state x_I of the additional integrator because it is part of the controller and therefore well-known.

Compensator: Combined Control Law and Estimator

When using an estimator, we replace the true state x in the control law given by Equation 4.21 with the estimates $\hat{x}$.

$$\begin{aligned} qx &= \Phi x + \Gamma u \\ q\hat{x} &= \Phi\hat{x} + \Gamma u + L(y - \hat{y}) \\ qx_I &= x_I + T(y - r) \;=\; x_I + TCx - Tr \\ u &= -K\hat{x} - k_I x_I + k_0 r \end{aligned}$$

The closed-loop system has order five.

$$q \begin{bmatrix} x \\ e_x \\ x_I \end{bmatrix} = \begin{bmatrix} \Phi - \Gamma K & \Gamma K & -\Gamma k_I \\ 0 & \Phi - LC & 0 \\ TC & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ e_x \\ x_I \end{bmatrix} + \begin{bmatrix} \Gamma k_0 \\ 0 \\ -T \end{bmatrix} r$$

The compensator can be viewed as a black box that generates the control signal u from the process output y. It is an nth-order system.

$$q \begin{bmatrix} \hat{x} \\ x_I \end{bmatrix} = \underbrace{\begin{bmatrix} \Phi - \Gamma K - LC & -\Gamma k_I \\ 0 & 1 \end{bmatrix}}_{\Phi_c} \begin{bmatrix} \hat{x} \\ x_I \end{bmatrix} + \underbrace{\begin{bmatrix} L \\ T \end{bmatrix}}_{\Gamma_c} y$$

$$-u = \underbrace{\begin{bmatrix} K & k_I \end{bmatrix}}_{C_c} \begin{bmatrix} \hat{x} \\ x_I \end{bmatrix}$$

Notice that the estimator contains a model of the process internally. This is a special case of the internal-model principle, which says that a good controller contains a model of the controlled system.

The transfer function of the compensator is

$$C(z) = C_c(zI - \Phi_c)^{-1}\Gamma_c.$$

The transfer function of the plant is

$$P(z) = C(zI - \Phi)^{-1}\Gamma.$$

Besides the analysis of time responses we investigate frequency responses in order to ensure good performance with respect to both disturbance rejection and sensor noise alleviation. The frequency response of a discrete-time system is given by the map

$$F(e^{j\omega T}) \text{ for } 0 \le \frac{\omega}{2\pi} \le \frac{1}{2T},$$

that is, up to the Nyquist frequency.

We define the transfer functions from the disturbance d and the sensor noise w to the output y in Figure 4.19.

$$\begin{aligned} S(z) &= \frac{Y(z)}{D(z)} = \frac{1}{1+P(z)C(z)} \\ T(z) &= \frac{Y(z)}{W(z)} = \frac{P(z)C(z)}{1+P(z)C(z)} \end{aligned}$$

$S(z)$ is the sensitivity and $T(z)$ the complementary sensitivity. From the definition of $S(z)$ and $T(z)$ it follows that

$$S(z) + T(z) = 1.$$

S is the primary measure of performance as far as it relates to disturbance rejection. Thus it is important to make the value of S small. For physically realizable systems, the loop gain $|PC|$ becomes small for high frequencies and S approaches unity. Therefore, it is only possible to make the sensitivity function small over low and midrange frequencies. At the same time, we wish to make T small to minimize the impact of measurement noise. This brings out a trade-off between disturbance rejection (S small) and attenuation of the effects of sensor noise (T small).

The transfer function of the closed-loop system is

$$G(z) = \frac{P(z)}{1+P(z)C(z)}.$$

If we denote by $P_0(z)$ the nominal plant and by $\Delta P(z)$ the plant uncertainty, we can define a corresponding nominal closed-loop transfer function $G_0(z)$ and an uncertainty $\Delta G(z)$.

$$\begin{aligned} G &= G_0 + \Delta G \\ P &= P_0 + \Delta P \end{aligned}$$

From

$$\Delta G = G - G_0 = \frac{P_0 + \Delta P}{1 + P_0 C + \Delta P\, C} - \frac{P_0}{1 + P_0 C}$$

if follows that

$$\frac{\Delta G}{G_0} = \frac{1}{1 + P_0 C + \Delta P\, C} \cdot \frac{\Delta P}{P_0}$$

and

$$\lim_{\Delta P \to 0} \frac{\Delta G / G_0}{\Delta P / P_0} = S. \tag{4.25}$$

Equation 4.25 gives a new interpretation to the sensitivity function S: it describes how relative changes in the plant P affect the reference tracking G.

Typically, $|P|$ is large for low frequencies and small for high frequencies, which leads to the following approximations.

$$\begin{aligned} PC &\approx S^{-1} \quad (\omega \text{ small}) \\ PC &\approx T \quad (\omega \text{ large}) \end{aligned}$$

Pole Selection

The magnitude of the control signal increases with increasing natural frequency of the closed-loop system. Hence, an increase in the speed of the response of the system will require an increase in the control signals. A consequence of large control signals are noticeable non-linear effects and unmodelled dynamics, which affect stability robustness. In other words, model uncertainty dictates an upper bound on the feedback gain. The sample rate imposes another limitation on the speed of the system.

The selection of the estimator poles is a compromise between sensitivity to measurement errors and rapid recovery from estimation errors. A fast estimator will converge quickly, but it will also be sensitive to measurement errors. The important consequence of increasing the speed of response of an estimator is that the bandwidth of the estimator becomes higher, thus causing more sensor noise to pass on to the control actuator.

The Pole placement was based on the results of Section 4.2; the poles were selected in the s-plane and mapped to the z-plane. In particular, the left-half plane is mapped to the unit circle.

$$z = e^{sT}$$

Different methods were used to specify the pole locations. One set of parameters that yielded particularly good results not only in simulations but also when applied to the real plant, is presented in the following. For the state feedback, apart from the two poles of the plant, the pole of the additional integrator had to be chosen adequately. Moving a mode a long distance results in excessive control signals, which is not desirable. But since the integrator was only introduced to compensate for a constant actuator error, a slow integrator gain suffices. This is achieved by moving the third pole only slightly from its initial position into the left-half plane.

$$\begin{aligned} \omega_n &= 4\,\text{rad/s} \\ \zeta &= 0.8 \\ \text{integrator pole} &= -0.2\,\text{rad/s} \end{aligned}$$

For the full state estimator, the natural frequency was increased to make the recovery from an estimation error faster than the dynamics of the plant.

$$\begin{aligned} \omega_n &= 6\,\text{rad/s} \\ \zeta &= 0.8 \end{aligned}$$

Figures 4.25 through 4.27 present simulated time and frequency responses, and Figures 4.28 and 4.29 are made from experimental data. A billiard ball was chosen because of its weight and smooth surface. Other experiments with e.g. a table tennis ball were less successful in the sense that the reference tracking was inferior.

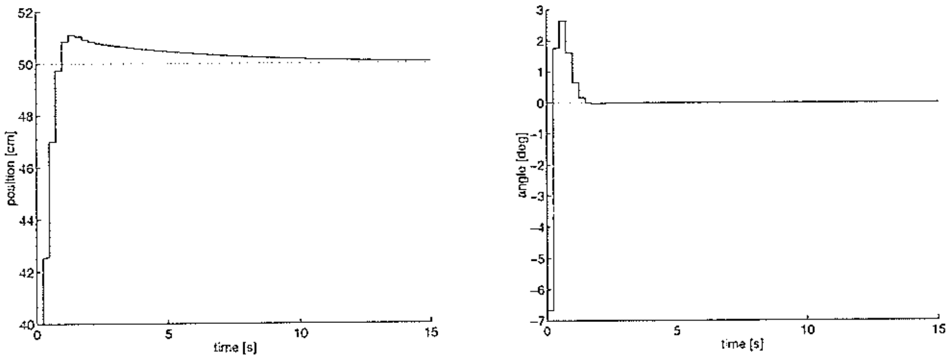


Figure 4.25: Step responses with full state estimator, state feedback and additional integrator.

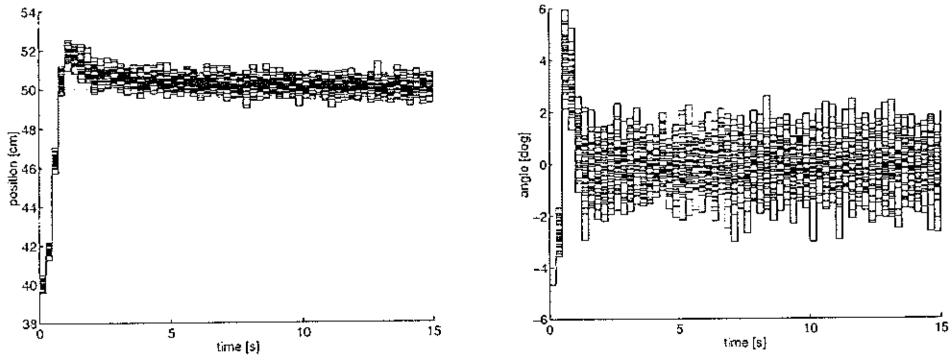


Figure 4.26: Step responses with position measurement noise (normal distribution, 2 mm standard deviation).

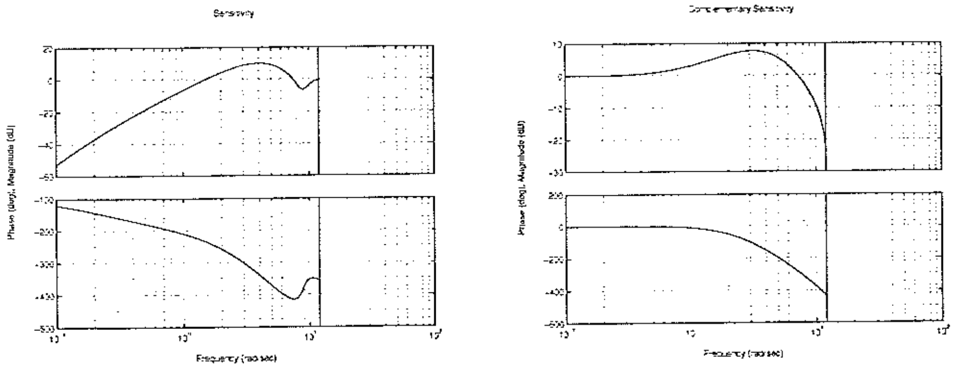


Figure 4.27: Sensitivity and complementary sensitivity.

See Appendix D for a MATLAB script of the simulations.

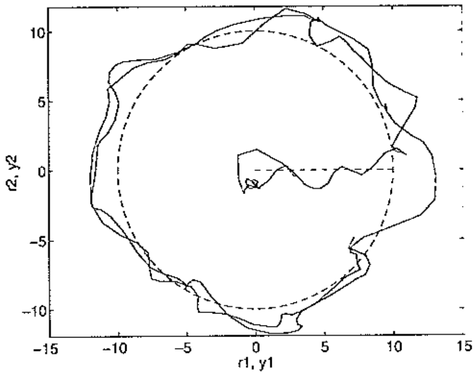


Figure 4.28: Attempt to follow a circle trajectory.

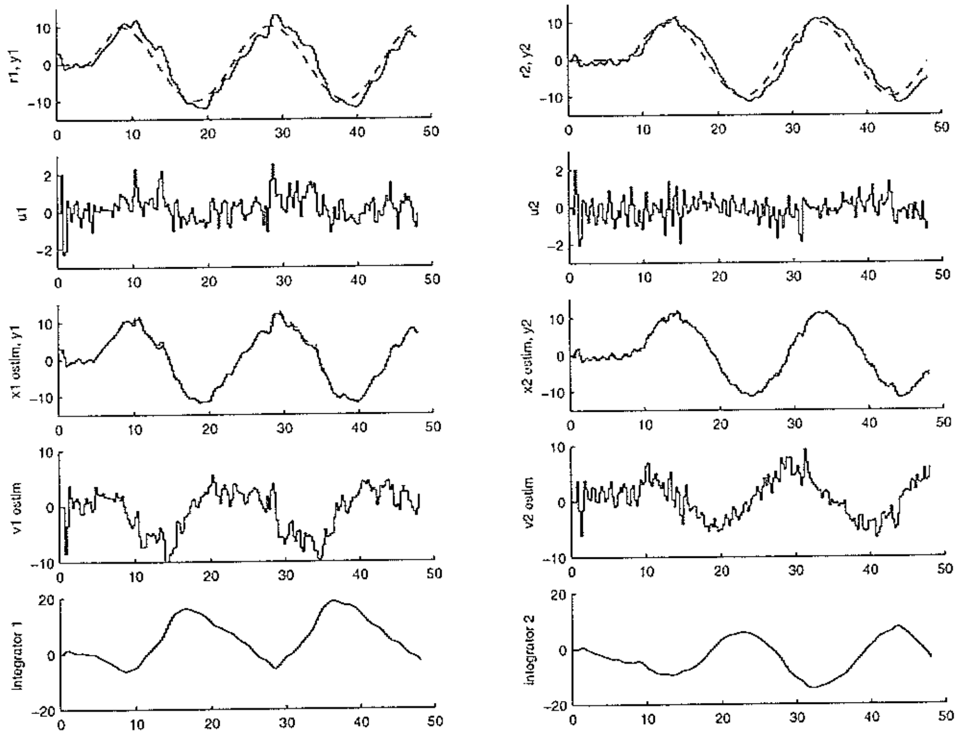


Figure 4.29: Experimental data from a Ball and Plate experiment with a circular reference trajectory. The two columns contain the data for the two degrees of freedom of the ball. 1st row: reference trajectory (dashed line) and measured position (solid line). 2nd row: inclination angle. 3rd row: estimated (solid line) and measured (dashed line) position. 4th row: estimated velocity. 5th row: additional integrator.

Implementation

The implementation of an observer-based controller with state feedback is straightforward. Figure 4.30 shows a pseudo code example. F stands for Φ, G for Γ and x for $\hat{x}$. The mathematical operators +, −, *, = are overloaded for matrix and vector arguments.

```
initialization of x[0], u and r
i=1
loop
{
   wait for trigger
   AnalogOut(u)
   AnalogIn(y)
   x[1]=(F-L*C)*x[0]+G*u+L*y
   u=-K*x[1]+k0*r[i]
   x[0]=x[1]
   i=i+1
}
```

Figure 4.30: Pseudo code for a controller with an estimator $x(k+1|k)$.

4.5 Alternative Discrete-Time Controller

The controller designed in Section 4.4 uses the measured variables read at time kT to compute the control signal applied at time $(k+1)T$. Another possibility is to read the measured variables at time kT and to apply the new control signal as soon as possible. Figure 4.31 illustrates the difference.

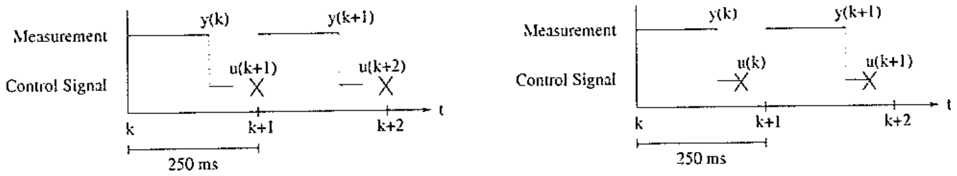


Figure 4.31: Two different ways to synchronize inputs and outputs.

The computational delay may be neglected if it is sufficiently small compared to the sample period. We can then design a controller that uses the variables measured at time kT to compute the control signal to be applied at time kT. In our case it is however necessary to take the computational delay into account. The control signal due at time kT will in fact only reach the plant at time $kT + \tau$, where τ is the computational delay. The trick is to pretend the *plant* has an input delay τ and not the controller. The effect will be the same.

Discretized Plant with Input Time Delay

A continuous-time system with a time delay is described as follows.

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t - \tau) \\ y(t) &= Cx(t) \end{aligned} \tag{4.26}$$

It is assumed that the time delay is less than the sample period.

$$\tau < T$$

Then we compute the zero-order-hold equivalent of Equation 4.26. Notice the introduction of the new state variable $u(k-1)$.

$$\begin{bmatrix} x(k+1) \\ u(k) \end{bmatrix} = \underbrace{\begin{bmatrix} \Phi & \Gamma_1 \\ 0 & 0 \end{bmatrix}}_{\Phi_d} \begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix} + \underbrace{\begin{bmatrix} \Gamma_0 \\ 1 \end{bmatrix}}_{\Gamma_d} u(k) \tag{4.27}$$

$$y(k) = \underbrace{\begin{bmatrix} C & 0 \end{bmatrix}}_{C_d} \underbrace{\begin{bmatrix} x(k) \\ u(k-1) \end{bmatrix}}_{x_d}$$

$$\Phi = e^{AT} \qquad \Gamma_0 = \int_0^{T-\tau} e^{At} B \, dt \qquad \Gamma_1 = e^{A(T-\tau)} \int_0^{\tau} e^{At} B \, dt \tag{4.28}$$

Evaluation of Equations 4.28 for the parameters in Equations 4.8 leads to the matrices which make up the state-space model with delay τ.

$$\Phi_d = \begin{bmatrix} 1 & T & -cg\tau(T - \frac{\tau}{2}) \\ 0 & 1 & -cg\tau \\ 0 & 0 & 0 \end{bmatrix} \qquad \Gamma_d = \begin{bmatrix} -\frac{1}{2}cg(T-\tau)^2 \\ -cg(T-\tau) \\ 1 \end{bmatrix}$$

$$C_d = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

State Feedback with Integrator and Direct Output Feedback

Since we explicitly take the computational delay into account, we can allow the input of the process to depend directly on its output and use the control structure in Figure 4.21.

$$\begin{aligned} qx_d &= \Phi_d x_d + \Gamma_d u \\ qx_I &= x_I + T(y - r) \ = \ x_I + TC_d x_d - Tr \\ u &= -K_d x_d - k_I x_I - k_P(C_d x_d - r) \end{aligned} \tag{4.29}$$

Compare the control laws in Equations 4.29 and 4.21.

The integrator adds another state to the plant. The state-space description of the open-loop system is written with an extended state vector.

$$q\begin{bmatrix}\boldsymbol{x}_d\\ x_I\end{bmatrix} = \underbrace{\begin{bmatrix}\boldsymbol{\Phi}_d & 0\\ T\boldsymbol{C}_d & 1\end{bmatrix}}_{\boldsymbol{\Phi}_{Id}}\begin{bmatrix}\boldsymbol{x}_d\\ x_I\end{bmatrix} + \underbrace{\begin{bmatrix}\boldsymbol{\Gamma}_d\\ 0\end{bmatrix}}_{\boldsymbol{\Gamma}_{Id}} u$$

$$y = \underbrace{\begin{bmatrix}\boldsymbol{C}_d & 0\end{bmatrix}}_{\boldsymbol{C}_{Id}}\begin{bmatrix}\boldsymbol{x}_d\\ x_I\end{bmatrix}$$

The closed-loop system becomes

$$q\begin{bmatrix}\boldsymbol{x}_d\\ x_I\end{bmatrix} = \begin{bmatrix}\boldsymbol{\Phi}_d - \boldsymbol{\Gamma}_d\tilde{\boldsymbol{K}}_d & -\boldsymbol{\Gamma}_d k_I\\ T\boldsymbol{C}_d & 1\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_d\\ x_I\end{bmatrix} + \begin{bmatrix}\boldsymbol{\Gamma}_d k_p\\ -T\end{bmatrix} r,$$

where

$$\tilde{\boldsymbol{K}}_d = \boldsymbol{K}_d + k_P\boldsymbol{C}_d.$$

Note that pole-placement functions return the vector $\tilde{\boldsymbol{K}}_d$. We choose k_P to be similar to k_0 in Equation 4.22

$$k_P = \frac{1}{\boldsymbol{C}_d(\boldsymbol{I} - \boldsymbol{\Phi}_d + \boldsymbol{\Gamma}_d\tilde{\boldsymbol{K}}_d)^{-1}\boldsymbol{\Gamma}_d}$$

and compute $\boldsymbol{K}_d$ from $\tilde{\boldsymbol{K}}_d$ and k_P.

$$\boldsymbol{K}_d = \tilde{\boldsymbol{K}}_d - k_P\boldsymbol{C}_d$$

Estimator without Delay

The estimator in Equation 4.23 has a delay, because $\hat{\boldsymbol{x}}(k|k-1)$ depends on measurements up to time $k-1$. The following estimator can be used to avoid the delay. Note the $\boldsymbol{y}(k)$ is a vector that contains all measured or otherwise known[4] states.

$$\hat{\boldsymbol{x}}(k|k) = \boldsymbol{\Phi}\,\hat{\boldsymbol{x}}(k-1|k-1) + \boldsymbol{\Gamma}\,u(k-1) + \boldsymbol{L}\Big(\boldsymbol{y}(k) - \bar{\boldsymbol{y}}(k|k-1)\Big) \quad (4.30)$$

$\bar{\boldsymbol{y}}(k|k-1)$ is the output of an auxiliary estimator.

$$\begin{aligned}\bar{\boldsymbol{x}}(k|k-1) &= \boldsymbol{\Phi}\,\hat{\boldsymbol{x}}(k-1|k-1) + \boldsymbol{\Gamma}\,u(k-1)\\ \bar{\boldsymbol{y}}(k|k-1) &= \boldsymbol{C}_e\,\bar{\boldsymbol{x}}(k|k-1)\end{aligned}$$

Substitution of the auxiliary estimator into Equation 4.30 yields

$$\hat{\boldsymbol{x}}(k|k) = (\boldsymbol{I} - \boldsymbol{L}\boldsymbol{C}_e)\Big(\boldsymbol{\Phi}\,\hat{\boldsymbol{x}}(k-1|k-1) + \boldsymbol{\Gamma}\,u(k-1)\Big) + \boldsymbol{L}\,\boldsymbol{y}(k).$$

[4] Otherwise known states are e.g. earlier output values, in case the model has a delay.

We define the state estimation error

$$e_x(k|k) = x(k) - \hat{x}(k|k) = (I - LC_e)\Phi\, e_x(k-1|k-1) \qquad (4.31)$$

and the output estimation error

$$\begin{aligned} e_y(k|k) &= y(k) - \hat{y}(k|k) = C_e\, e_x(k|k) \\ &= C_e(I - LC_e)\Phi\, e_x(k-1|k-1) \\ &= (C_e - C_e L C_e)\Phi\, e_x(k-1|k-1) \\ &= (I - C_e L)C_e \Phi\, e_x(k-1|k-1). \end{aligned}$$

If we choose L such that

$$C_e L = I, \qquad (4.32)$$

which is possible if $\operatorname{rank}(C_e)$ equals the number of outputs p, then

$$e_y(k|k) = 0. \qquad (4.33)$$

This means that the outputs of the system is estimated without error; in fact they need not be estimated at all. Hence it is possible to reduce the estimator by p states. Estimation is only implemented for the states that are not measured.

The system in Equation 4.27 has the states $x_1(k)$, $x_2(k)$ and $u(k-1)$. Since we wish to make use of all measurable and otherwise known states, we define the input of the estimator $y(k)$ to contain the states $x_1(k)$ and $u(k-1)$.

$$C_e = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Consequently, the estimator gain is a 3 × 2 matrix.

$$L = \begin{bmatrix} l_{11} & l_{12} \\ l_{21} & l_{22} \\ l_{31} & l_{32} \end{bmatrix}$$

The Condition in Equation 4.32 is

$$C_e L = \begin{bmatrix} l_{11} & l_{12} \\ l_{31} & l_{32} \end{bmatrix} \overset{!}{=} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

so that the number of parameters in L is reduced to two.

$$L = \begin{bmatrix} 1 & 0 \\ l_{21} & l_{22} \\ 0 & 1 \end{bmatrix}$$

We denote by φ_{ij} an element of the matrix Φ_d and evaluate $(I - LC_e)\Phi$ in Equation 4.31.

$$\begin{bmatrix} 0 & 0 & 0 \\ -l_{21}\varphi_{11} + \varphi_{21} - l_{22}\varphi_{31} & -l_{21}\varphi_{12} + \varphi_{22} - l_{22}\varphi_{32} & -l_{21}\varphi_{13} + \varphi_{23} - l_{22}\varphi_{33} \\ 0 & 0 & 0 \end{bmatrix}$$

According to Equation 4.33, the outputs are estimated without error. It follows immediately that

$$e_{x_2}(k|k) = (-l_{21}\varphi_{12} + \varphi_{22} - l_{22}\varphi_{32})\, e_{x_2}(k-1|k-1). \qquad (4.34)$$

With the definition

$$l = l_{21}$$

Equation 4.34 evaluates to

$$e_{x_2}(k|k) = (1 - lT)\, e_{x_2}(k-1|k-1).$$

Compensator with Reduced Estimator

The reduced estimator provides an estimate of the unsensed velocity x_2.

$$\begin{aligned} q\boldsymbol{x}_d &= \Phi_d \boldsymbol{x}_d + \Gamma_d u \\ q e_{x_2} &= (1 - lT) e_{x_2} \\ q x_I &= x_I + T(y - r) \;=\; x_I + T C_d \boldsymbol{x}_d - Tr \\ u &= -k_1 x_1 - k_2 \hat{x}_2 - k_3 q^{-1} u - k_I x_I - k_P (C_d \boldsymbol{x}_d - r) \end{aligned}$$

Although the position x_1 is not estimated any more, the closed-loop system has the same order as with a full estimator because the input delay adds another state.

$$q \begin{bmatrix} \boldsymbol{x}_d \\ e_{x_2} \\ x_I \end{bmatrix} = \begin{bmatrix} \Phi_d - \Gamma_d \tilde{K}_d & \Gamma_d k_2 & -\Gamma_d k_I \\ 0 & 1 - lT & 0 \\ TC_d & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_d \\ e_{x_2} \\ x_I \end{bmatrix} + \begin{bmatrix} \Gamma_d k_P \\ 0 \\ -T \end{bmatrix} r$$

Pole Selection

The pole specifications were chosen to be the same as in Section 4.4, where applicable. The additional pole for the input delay τ of the plant was left at the origin of the z-plane.

State feedback:

$$\begin{aligned} \omega_n &= 4\ \text{rad/s} \\ \zeta &= 0.8 \\ \text{integrator pole} &= -0.2\ \text{rad/s} \\ \text{discrete-time pole} &= 0 \end{aligned}$$

Reduced estimator:

$$-6\ \text{rad/s}$$

See Figures 4.32 and 4.33 for the simulated time responses if a reduced estimator is used. A comparison with the simulations in Section 4.4 shows only minor differences between the controllers based on the full and the reduced estimator. This is because of the relatively long computational delay τ.

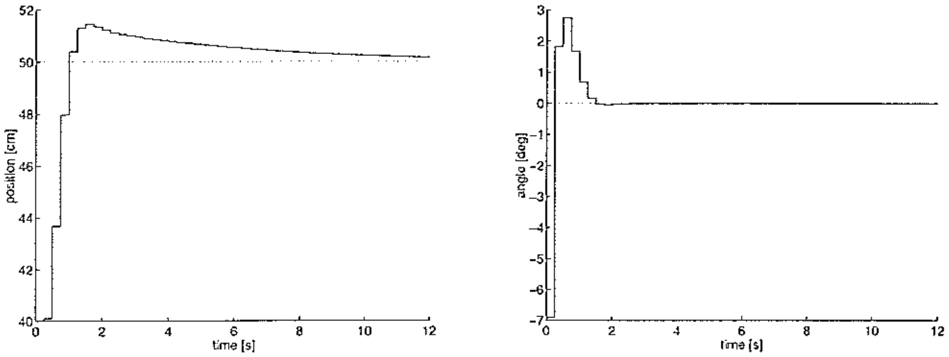


Figure 4.32: Step responses with reduced estimator, state feedback and additional PI controller.

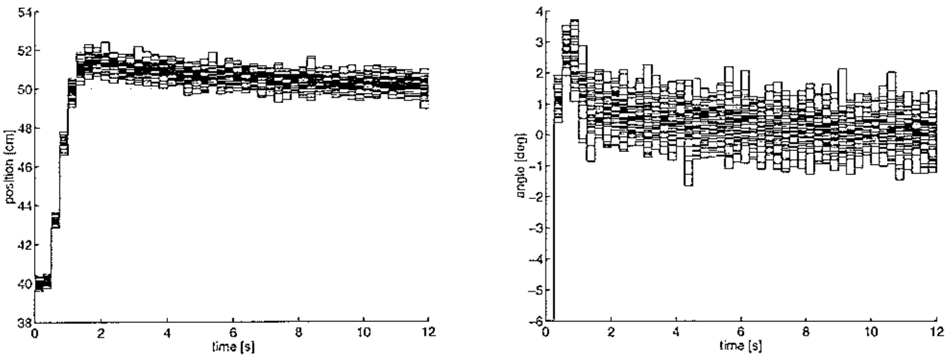


Figure 4.33: Step responses with position measurement noise (normal distribution, 2 mm standard deviation).

Implementation

For the implementation, it is essential to keep the delay between the measurement of the output y and the application of the new control signal u as short as possible. If part of the the new signal is precomputed before the output is read, some time can be saved. Figure 4.34 illustrates the concept.

In our Ball and Plate process, the measurement of the output signal is very time-consuming and the model is rather simple, so that the improvements are hardly relevant.

```
initialization of xpre, upre and r
i=1
loop
{
   wait for trigger
   AnalogIn(y)
   u=upre-K*L*y
   AnalogOut(u)
   x=xpre+L*y
   xpre=(I-L*C)*(F*x+G*u)
   upre=-K*xpre+k0*r[i]
   i=i+1
}
```

Figure 4.34: Pseudo code for a controller with an estimator $x(k|k)$.

4.6 Identification

System identification deals with the problem of building mathematical models of dynamical systems, based on observed system data. The two degrees of freedom of the Ball and Plate system were considered separately, i.e. two Ball and Beam systems were identified. One of them is presented here. Since our plant is known to be roughly a double integrator, only the gain was estimated.

$$P(s) = \underbrace{-cg}_{\text{gain}} \frac{1}{s^2}$$

Because the plant is unstable, merely very short input-output sequences can be recorded when the system is operating in open-loop. Therefore, the data was collected in a closed-loop experiment.

To ensure a persistent excitation, the parameters of the controller were set such that the input signal oscillated. The amplitude should not be too large since the linearized model is only valid for small inclination angles. On the other hand, one can expect a higher accuracy of the estimation for increased input amplitudes because the signal-to-noise ratio will increase and disturbances will play a less important role. See Figure 4.35 for an extract of the input signal.

Unfortunately, analog anti-aliasing filters cannot be applied to the video signal before it is sampled, but a simple open-loop experiment with a fixed white object instead of the ball showed that the noise amplitudes are so small that they are hardly relevant. See Figure 4.37 for a noise sequence and its spectrum.

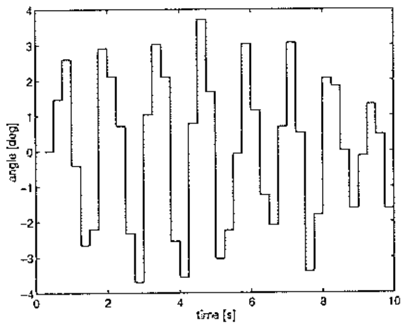


Figure 4.35: Input signal for the identification experiment.

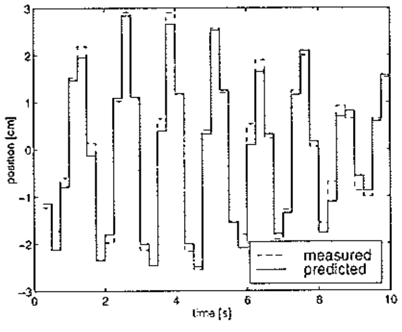


Figure 4.36: Measured and predicted signal.

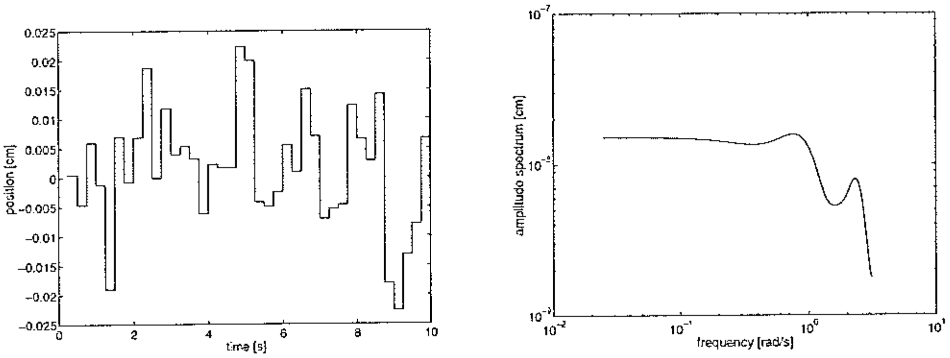


Figure 4.37: Position noise and its spectrum.

As a precomputation it is common practice to filter the recorded data with a discrete-time filter. This is done to narrow the frequency band in which the model should be a good approximation of the plant. Both the input and the output data are low-pass filtered with a cut-off frequency of half the Nyquist frequency.

One way of fixing parameters in a model structure is filtering the input-output data appropriately before the identification. A naive approach to estimate the gain would be to run the measured input through a process model, i.e. a predictor, and then compare the result with the measured output. In the case of a double integrator, the predictor would however be unstable and produce unusable results.

Recall the transfer function of the discrete-time model in Equation 4.19. With $P_1(z) = \frac{Y(z)}{U(z)}$ Equation 4.19 can be written as

$$(z-1)^2 Y(z) = -cg\frac{T^2}{2}(z+1)U(z). \tag{4.35}$$

The definition of the two signals

$$\begin{aligned} y_{\text{filt}}(k) &= (q-1)^2 y(k) \\ u_{\text{filt}}(k) &= \frac{T^2}{2}(q+1)u(k) \end{aligned}$$

turns the difference equation that corresponds to Equation 4.35 into

$$y_{\text{filt}}(k) = -cg\, u_{\text{filt}}(k).$$

Since y_{filt} is supposed to be proportional to u_{filt}, their mean values are subtracted before the gain is estimated with a linear regression.

$$\begin{aligned} -c_{\text{estim}}\, g &= -0.85 \quad \text{(standard deviation 0.08)} \\ -c_{\text{theory}}\, g &= -0.71 \end{aligned}$$

Possible reasons for the 20% difference between the estimated and the theoretical gain include friction between the ball and the plate, slipping of the ball, unmodelled actuator dynamics and resonances in the camera suspension. An extract of the signals y_{filt} and $-c_{\text{estim}}\, g\, u_{\text{filt}}$ is depicted in Figure 4.36.

Appendix D contains the MATLAB script for the identification.

Chapter 5

Robot Programming

For experimental purposes, the plate for the rolling ball was screwed to the gripper of an industrial robot [2] like the one in Figure 5.1.

Figure 5.2 shows the robot part of the laboratory setup. The master computer of the system is based on a Motorola 68040 microprocessor. Supervision and safety functions are implemented on a Motorola 68030. Digital signal processors (DSP) are used for low-level control and filtering of analog sensor signals. The embedded computers are connected by a VME bus.[1] Software cross-development in Modula-2 is hosted by a Sun workstation, from where the object code is downloaded to the target via Ethernet.

The robot software for the Ball and Plate project was built on an existing framework [10]. Position control for each joint of the robot was realized with an inner velocity control. Figure 5.3 shows the cascade of two PID controllers. Given a maximum rotational speed of 300°/s and a sampling frequency of 4 kHz, the robot is so much faster than the dynamics of the Ball and Plate system, that the parameters of the PID controllers for the robot motion could be tuned independently of the control design for the trajectory tracking of the ball.

An abstract model of the program in the master computer of the robot is the finite state machine in Figure 5.4. Special attention was paid to guarantee a smooth transition from standby to run mode before the control loop is entered. An emergency stop may be triggered either by the supervision program or the user. Once back to standby mode, the program does not need to be manually rerun if more experiments are to be carried out; the system recovers automatically when the safety pad is pressed.

The two control signals for the two degrees of freedom of the plate are computed in the PC. They are written to an analog interface [4] and transmitted to the VME system. The D/A and A/D conversion has a resolution of 12 bits on both sides. See Figure 5.5 for a sketch of the closed loop.

[1] VME is an acronym for Versatile Module Eurocard. The VME bus is widely used in industrial, commercial and military applications.

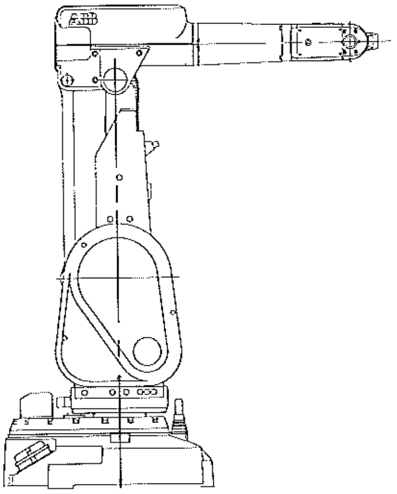


Figure 5.1: The industrial robot that serves as an actuator.

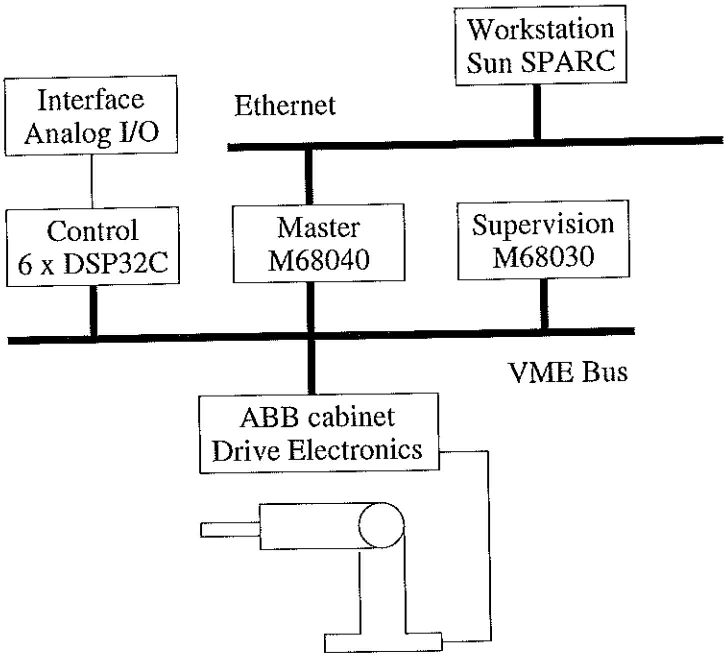


Figure 5.2: Overview of the experimental robot system.

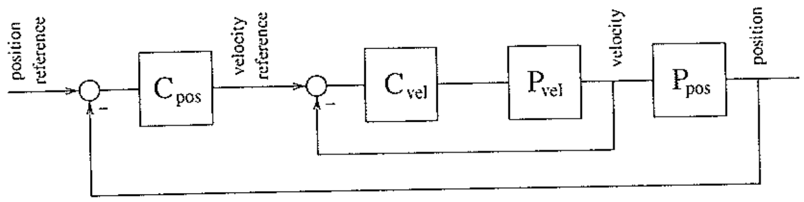


Figure 5.3: Robot motion control with two cascaded PID controllers.

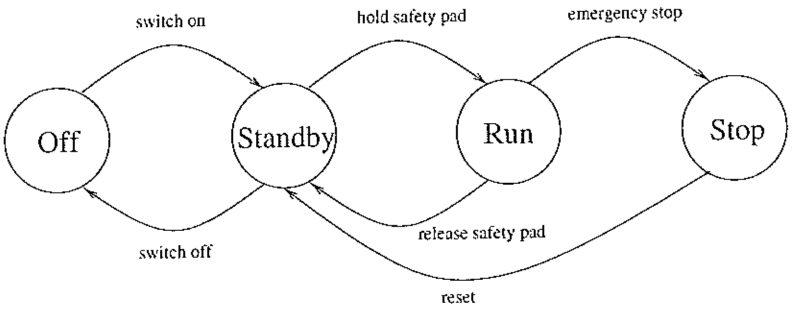


Figure 5.4: Finite state machine representation of the robot.

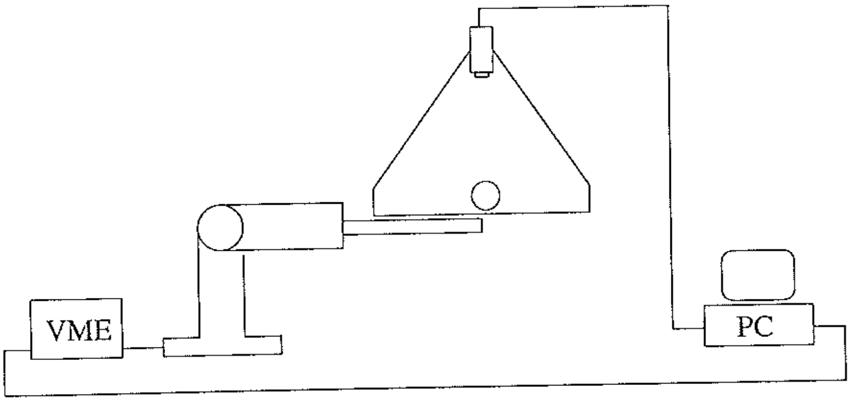


Figure 5.5: Sketch of the control loop for the Ball and Plate system.

Chapter 6

Conclusions

This chapter sums up the contributions of the thesis and propounds ideas for future work.

Motivation

The Ball and Beam system had been used in several laboratory experiments at the department, and it was natural to extend the motion of the ball to two dimensions. A major goal of the project was to attain an eye-catching demonstration object. Unlike the beam, a specially built plate process with sensors and actors was not at hand. For the purpose of this project, a plate was attached to an industrial robot. Instead of electrical measurement, a video camera should provide information about the position of the ball. Experience was to be gained from using computer vision in control processes.

Summary

An analog video camera was connected to a host computer with a dedicated interface. After a first analysis of single pictures, a Windows application was written to continuously acquire and process images of the ball on the plate. Because of limitations in the frame grabber hardware, the feasible sample rate turned out to be no more than 4 to 5 Hz.

Although a usable model of the Ball and Plate system was known, an *exact* physical model was derived to investigate the effects of simplifications. The modelling of the Ball and Plate system showed that the equations of motion for the two degrees of freedom are coupled and not mathematically equivalent. Only when linearized are they decoupled and, not surprisingly, equal to the equation of motion for a ball on a beam.

A comparison of various simulations with continuous-time controllers suggested that a bandwidth of about 4 rad/s could be achieved for the closed-loop system. The design of the digital controller included sampled-data theory and was based on the discretized model of the plant rather than an emulation of an analog controller. Full state feedback with an estimator was chosen as the control structure. An additional integrator needed to be added to compensate for an actuator bias. The controller was implemented on the same computer as the image acquisition. Hard restrictions resulting from the inevitably low sampling frequency made the tuning a challenge. Finally, the physical model was compared to the result of an identification experiment.

Position control of the robot, and therefore the plate, was implemented on a separate computer system. Thanks to the high performance of the robot, its transient behaviour could be disregarded for the position control of the ball.

The entire system was successfully demonstrated in diverse experiments.

Outlook

The limiting component in the present system is clearly the digitization hardware for the video signals, i.e. the slow sample rate. While a faster frame grabber would certainly bring substantial improvements, a digital video camera would be a preferable solution.

Faster trajectories of the ball would on the one hand be fascinating, but a constantly rolling ball also eliminates the problem of static friction and thus enhances the tracking accuracy. Faster sampling would however be a prerequisite.

A larger platform together with a higher bandwidth would expose the inherent non-linearities and multiple-input multiple-output effects that the mathematical model evinced. This would allow more advanced control design techniques to be tested for their aptitude.

Perhaps the most challenging endeavour would be to automate the famous labyrinth game, where a ball rolls on a plate with obstacles and holes. By tilting the plate, the ball must be directed along a line without falling into a hole. A commercially available toy set could be equipped with two motors and a video camera mounted above.

The contributions of this thesis will hopefully benefit further development of precious control systems.

Appendix A

Moment of Inertia

The ball that is used in the experiment can be either homogeneously filled or hollow. The moment of inertia is different for the two types of ball.

A.1 Solid Sphere

It is natural to use spherical coordinates as defined in Figure A.1.

$$\begin{aligned} x &= r\cos\phi\cos\theta \\ y &= r\sin\phi\cos\theta \\ z &= r\sin\theta \end{aligned}$$

We compute the Jacobian determinant

$$\left|\frac{\partial(x,y,z)}{\partial(r,\phi,\theta)}\right| = \begin{vmatrix} \cos\phi\cos\theta & -r\sin\phi\cos\theta & -r\cos\phi\sin\theta \\ \sin\phi\cos\theta & r\cos\phi\cos\theta & -r\sin\phi\sin\theta \\ \sin\theta & 0 & r\cos\theta \end{vmatrix} = r^2\cos\theta$$

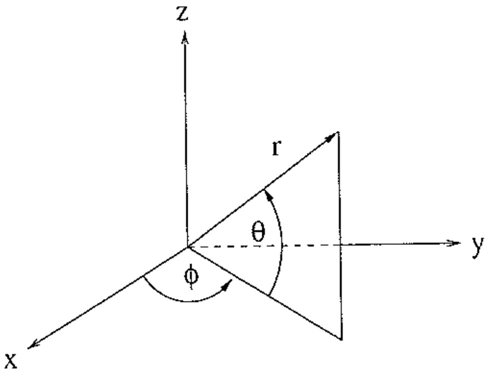


Figure A.1: Definition of spherical coordinates.

to define the infinitesimal volume element

$$\mathrm{d}\mu = r^2 \cos\theta \,\mathrm{d}\theta \,\mathrm{d}\phi \,\mathrm{d}r. \tag{A.1}$$

The volume of a solid sphere with radius R is computed by integrating over all three coordinates.

$$V = \int_V \mathrm{d}\mu = \int_0^R \int_0^{2\pi} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} r^2 \cos\theta \,\mathrm{d}\theta \,\mathrm{d}\phi \,\mathrm{d}r = \frac{4}{3}\pi R^3 \tag{A.2}$$

If the sphere is rotated around the z-axis, then the distance of an arbitrary point in space from the rotational axis is

$$a = r\cos\theta. \tag{A.3}$$

Let m denote the mass and ρ the constant density of a rigid body. Then the moment of inertia is defined as follows:

$$J = \int_m a^2 \,\mathrm{d}m = \rho \int_V a^2 \,\mathrm{d}\mu. \tag{A.4}$$

Substituting Equations A.1 and A.3 into Equation A.4 yields

$$J = \rho \int_0^R \int_0^{2\pi} \int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} r^4 \cos^3\theta \,\mathrm{d}\theta \,\mathrm{d}\phi \,\mathrm{d}r = \rho\frac{8\pi}{15}R^5.$$

Then we use Equation A.2 to express J.

$$J = \rho\frac{8\pi}{15}R^5 = \rho\frac{4\pi}{3}R^3\frac{2}{5}R^2 = \rho V\frac{2}{5}R^2$$

This is the moment of inertia of a solid sphere with mass m and radius R.

$$J = \frac{2}{5}mR^2 \tag{A.5}$$

A.2 Spherical Shell

Let J denote the moment of inertia and m the mass of a spherical shell with inner radius R_1 and outer radius R_2. Then from the definition of the moment of inertia we gather

$$\begin{aligned} J &= J_2 - J_1 \\ m &= m_2 - m_1 \end{aligned} \tag{A.6}$$

where J_1 and J_2 are the moments of inertia and m_1 and m_2 the masses of *solid* spheres with radii R_1 and R_2, respectively. Substituting Equation A.5 for J_1 and J_2 in Equation A.6, we compute the ratio

$$\frac{J}{m} = \frac{J_2 - J_1}{m_2 - m_1} = \frac{\frac{2}{5}m_2R_2^2 - \frac{2}{5}m_1R_1^2}{m_2 - m_1}. \tag{A.7}$$

Using equation A.2, we substitute $m_i = \rho V_i = \rho \frac{4}{3}\pi R_i^3$ into equation A.7.

$$\frac{J}{m} = \frac{2}{5}\frac{R_2^5 - R_1^5}{R_2^3 - R_1^3}$$

Provided the shell is sufficiently thin compared to its radius, it is legitimate to use the limit $R_1 \to R_2$.

$$\lim_{R_1 \to R_2} \frac{J}{m} = \frac{2}{5} \lim_{R_1 \to R_2} \frac{R_2^5 - R_1^5}{R_2^3 - R_1^3}$$

With the rule of Bernoulli-de l'Hôpital, this evaluates to

$$\frac{2}{5} \lim_{R_1 \to R_2} \frac{\frac{\partial}{\partial R_1}(R_2^5 - R_1^5)}{\frac{\partial}{\partial R_1}(R_2^3 - R_1^3)} = \frac{2}{5} \lim_{R_1 \to R_2} \frac{-5R_1^4}{-3R_1^2} = \frac{2}{3} \lim_{R_1 \to R_2} R_1^2 = \frac{2}{3}R_2^2.$$

We have found

$$\frac{J}{m} \to \frac{2}{3}R_2^2 \quad (R_1 \to R_2).$$

Because of $R_1 \to R_2$, we can define $R = R_2$ and solve the above limit for J to get the moment of inertia of a thin spherical shell with mass m and radius R.

$$J = \frac{2}{3}mR^2 \tag{A.8}$$

Appendix B

Unary Cross Product Operator

The purpose of this appendix is to define a unary operator which turns the left-hand side argument of the binary vector cross product operator into a matrix, such that the vector cross product can be written as a product of a matrix and a vector.

Definition 1 A matrix $\boldsymbol{A}$ is *skew-symmetric*, if $\boldsymbol{A}' = -\boldsymbol{A}$.

The subspace of all skew-symmetric $n \times n$ matrices is the Lie-Algebra $so(n)$ of the special orthogonal group $SO(n)$, which is also called n-dimensional rotation group. Its dimension follows immediately from the definition.

$$dim(so(n)) = \frac{n(n-1)}{2}$$

For the case $n = 3$, we have $dim(so(3)) = 3$.

Definition 2 We define the unary cross product operator $(\cdot)\tilde{}$, such that for two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathrm{R}^3$,

$$\tilde{\boldsymbol{x}} \cdot \boldsymbol{y} = \boldsymbol{x} \times \boldsymbol{y}.$$

Theorem The mapping

$$\tilde{}: \mathrm{R}^3 \mapsto so(3),\ \boldsymbol{x} \mapsto \tilde{\boldsymbol{x}}$$

is a linear isomorphism between two three-dimensional vector spaces. In particular, this means that for every matrix $\boldsymbol{A} \in so(3)$, we can find a unique vector $\boldsymbol{a}$, such that $\tilde{\boldsymbol{a}} = \boldsymbol{A}$.

Proof: Consider the skew-symmetric matrix $\boldsymbol{A}$ and the vectors $\boldsymbol{a}$ and $\boldsymbol{b}$.

$$\boldsymbol{A} = \begin{bmatrix} 0 & a_{12} & a_{13} \\ -a_{12} & 0 & a_{23} \\ -a_{13} & -a_{23} & 0 \end{bmatrix} \quad \boldsymbol{a} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \quad \boldsymbol{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

We request that

$$\boldsymbol{A} \cdot \boldsymbol{b} = \boldsymbol{a} \times \boldsymbol{b}.$$

It follows that

$$\begin{aligned} a_{12} &= -a_3 \\ a_{13} &= a_2 \\ a_{23} &= -a_1. \end{aligned}$$

This is a bijective mapping from $\boldsymbol{a}$ to $\boldsymbol{A}$, which gives rise to the following explicit definition of the binary cross product operator $(\cdot)^{\tilde{}}$:

$$\tilde{}: \mathsf{R}^3 \mapsto so(3), \quad \boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \mapsto \tilde{\boldsymbol{x}} = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}$$

Appendix C

Smooth step functions

The function usually referred to as *step function* is the Heaviside function.

$$y(t) = \begin{cases} 0 & (t < 0) \\ 1 & (t \geq 0) \end{cases}$$

The derivative of this function is not steady. To circumvent numerical difficulties when differentiating in SIMULINK, smoother step-like excitation functions must be found.

C.1 Reach the Step Height in Finite Time

The first approach for the transient phase of the step was a cosine wave, depicted in Figure C.1. T is the rise time.

$$y(t) = \begin{cases} 0 & (t \leq 0) \\ \frac{1}{2}(1 - \cos \frac{\pi}{T} t) & (0 \leq t \leq T) \\ 1 & (t \geq T) \end{cases}$$

Another possibility is a hyperbolic cosine, as in Figure C.2.

$$y(t) = \begin{cases} 0 & (t \leq 0) \\ \frac{1}{2(\sqrt{2}-1)} \left(\cosh \frac{2 \operatorname{arsinh}(1)}{T} t - 1 \right) & (0 \leq t \leq \frac{T}{2}) \\ -\frac{1}{2(\sqrt{2}-1)} \left(\cosh \left(2 \operatorname{arsinh}(1) \left(\frac{t}{T} - 1 \right) \right) - 2\sqrt{2} + 1 \right) & (\frac{T}{2} \leq t \leq T) \\ 1 & (t \geq T) \end{cases}$$

C.2 Reach the Step Height Asymptotically

The problem with step-like functions that reach the step height in a finite time T is the vertex at $t = T$ which makes the second derivative unsteady. The solution are functions that only reach the step height asymptotically.

First-Order System

In the simplest case, we take the step response of a linear time-invariant system with a single pole $-1/T$ in the left half-plane. T is the time constant of the system.

$$y(t) = 1 - e^{-t/T}$$

This function is analytic. However, from Figure C.3 we can see that the first derivative is not steady at $t = 0$.

Second-Order System with High Damping

To get a steady first derivative, we need a second-order system. We ask for a characteristic equation with two roots $-a$ and $-b$ in the left half-plane.

$$(s + a)(s + b) = 0 \tag{C.1}$$

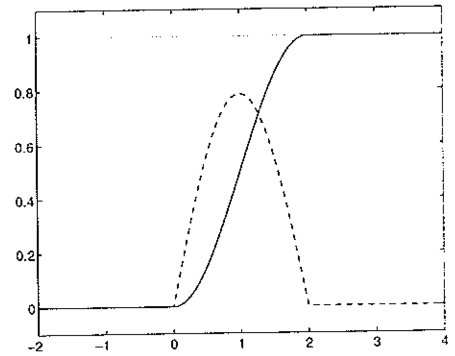


Figure C.1: Smooth step made of cos(·), and its first derivative.

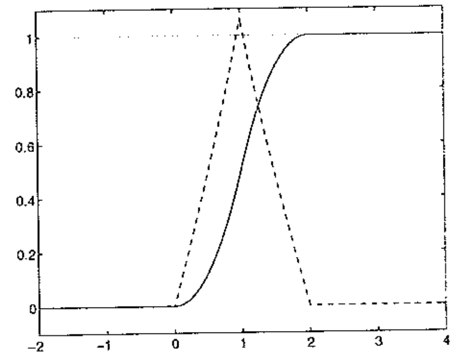


Figure C.2: Smooth step made of cosh(·), and its first derivative.

This equation is commonly written with the damping ratio ζ and the undamped natural frequency ω_n.

$$s^2 + 2\zeta\omega_n s + \omega_n^2 = 0 \tag{C.2}$$

By comparing the coefficients of Equations C.1 and C.2, we can express a and b by ζ and ω_n.

$$\begin{aligned} a &= \omega_n(\zeta + \sqrt{\zeta^2 - 1}) \\ b &= \omega_n(\zeta - \sqrt{\zeta^2 - 1}) \end{aligned} \tag{C.3}$$

For high damping ($\zeta > 1$), we get two different real negative poles. The step response

$$y(t) = 1 - \frac{ae^{-bt} - be^{-at}}{a - b} \tag{C.4}$$

is depicted in Figure C.4. Note that the first derivative is steady as opposed to the first-order system.

If we wish to define a *single* time constant T for a second-order system, similar to the first-order case, we need to compute the time when the tangent

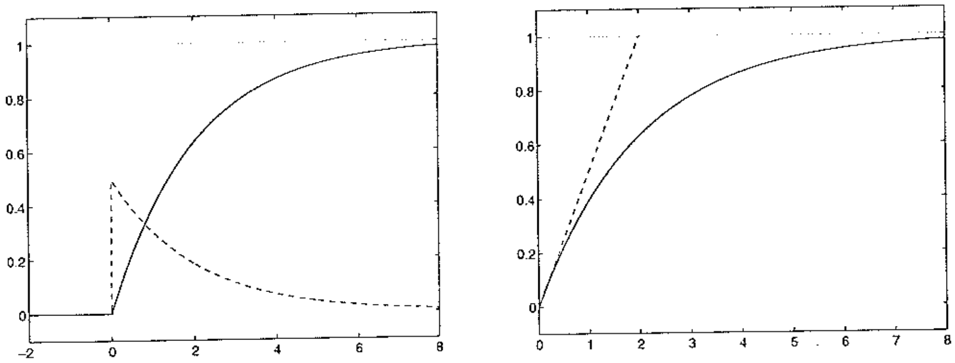


Figure C.3: Step response of an asymptotically stable first-order system.

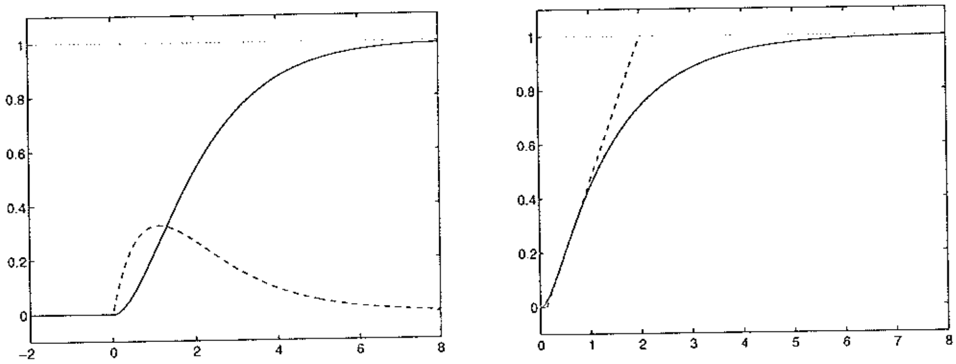


Figure C.4: Step response of an asymptotically stable second-order system with high damping ($\zeta = 1.4$).

through the inflection point reaches the step height. In order to do this, we derive Equation C.4 twice.

$$\dot{y}(t) = \tfrac{ab}{a-b}(e^{-bt} - e^{-at})$$
$$\ddot{y}(t) = \tfrac{ab}{a-b}(ae^{-at} - be^{-bt})$$

The condition for the inflection point is $\ddot{y}(t_0) = 0$. Solving for t_0 yields [1]

$$t_0 = \frac{\log a - \log b}{a - b}.$$

The tangent through the inflection point is defined by

$$x(t) = \dot{y}(t_0)\, t + d(t_0).$$

At the inflection point we have $t = t_0$ and $x(t) = y(t_0)$, which leads to

$$d(t_0) = y(t_0) - \dot{y}(t_0)\, t_0.$$

The condition for our time constant T is $x(T) = 1$. Solving for T yields

$$T = \frac{1 - y(t_0)}{\dot{y}(t_0)} + t_0 \tag{C.5}$$

and expands to

$$T = \frac{a+b}{ab} + \frac{\log a - \log b}{a - b}. \tag{C.6}$$

Now we substitute Equations C.3 into Equation C.6 and solve for ω_n to get

$$\omega_n = \frac{1}{T}\left(2\zeta + \frac{1}{2\sqrt{\zeta^2 - 1}} \log \frac{\zeta + \sqrt{\zeta^2 - 1}}{\zeta - \sqrt{\zeta^2 - 1}}\right). \tag{C.7}$$

Given a time constant T and a damping ratio ζ, we compute the appropriate natural frequency ω_n as defined in Equation C.7 and then the real poles $-a$ and $-b$ using Equations C.3, as long as $\zeta > 1$ (high damping).

Second-Order System with Critical Damping

In the case of critical damping ($\zeta = 1$), Equations C.3 show that both poles are identical; they are located at $-a$ in the left half-plane. The corresponding step response

$$y(t) = 1 - e^{-at}(1 + at)$$

is depicted in Figure C.5.

[1] log denotes the natural logarithm.

Just like we did for high damping, we define the time constant T of the system to be the time when the tangent through the inflection point reaches the step height.

$$\begin{aligned}\dot{y}(t) &= a^2te^{-at}\\ \ddot{y}(t) &= a^2e^{-at}(1-at)\end{aligned}$$

The zero of the second derivative is

$$t_0 = \frac{1}{a}$$

and with Equation C.5, the time constant eventually becomes $T = 3/a$, which is equivalent to

$$a = \frac{3}{T}. \tag{C.8}$$

Equation C.8 specifies the double pole at $-a$, given a time constant T.

Second-Order System with Low Damping

According to Equations C.3, low damping ($\zeta < 1$) gives rise to a pair of conjugate complex poles. They are typically written in terms of the common real part $-\sigma$ and the damped frequency ω_d.

$$\begin{aligned}a &= \sigma + j\omega_d\\ b &= \sigma - j\omega_d\end{aligned} \tag{C.9}$$

The characteristic equation of the system becomes

$$(s+\sigma)^2 + \omega_d^2 = 0. \tag{C.10}$$

By comparing the coefficients of Equations C.2 and C.10, we can express σ and ω_d by ζ and ω_n.

$$\begin{aligned}\sigma &= \zeta\omega_n\\ \omega_d &= \omega_n\sqrt{1-\zeta^2}\end{aligned} \tag{C.11}$$

The step response

$$y(t) = 1 - e^{-\sigma t}(\cos\omega_d t + \frac{\sigma}{\omega_d}\sin\omega_d t)$$

is depicted in Figure C.6.

The derivation of the time constant T follows the well-known pattern: Find the zero of the second derivative and compute the time when the tangent through the inflection point reaches the step height.

$$\begin{aligned}\dot{y}(t) &= (\omega_d + \tfrac{\sigma^2}{\omega_d})e^{-\sigma t}\sin\omega_d t\\ \ddot{y}(t) &= (\omega_d + \tfrac{\sigma^2}{\omega_d})(\omega_d e^{-\sigma t}\cos\omega_d t - \sigma e^{-\sigma t}\sin\omega_d t)\end{aligned}$$

The inflection point is at

$$t_0 = \frac{1}{\omega_d} \arctan \frac{\omega_d}{\sigma}$$

and Equation C.5 produces

$$T = \frac{2\sigma}{\sigma^2 + \omega_d^2} + \frac{1}{\omega_d} \arctan \frac{\omega_d}{\sigma}. \tag{C.12}$$

Finally, we substitute Equations C.11 into Equation C.12 and solve for ω_n.

$$\omega_n = \frac{1}{T}\left(2\zeta + \frac{1}{\sqrt{1-\zeta^2}} \arctan \frac{\sqrt{1-\zeta^2}}{\zeta}\right) \tag{C.13}$$

Given a time constant T and a damping ratio ζ, we compute the appropriate natural frequency ω_n as defined in Equation C.13 and then the complex poles $-\sigma \mp j\omega_d$ using Equations C.11, as long as $\zeta < 1$ (low damping).

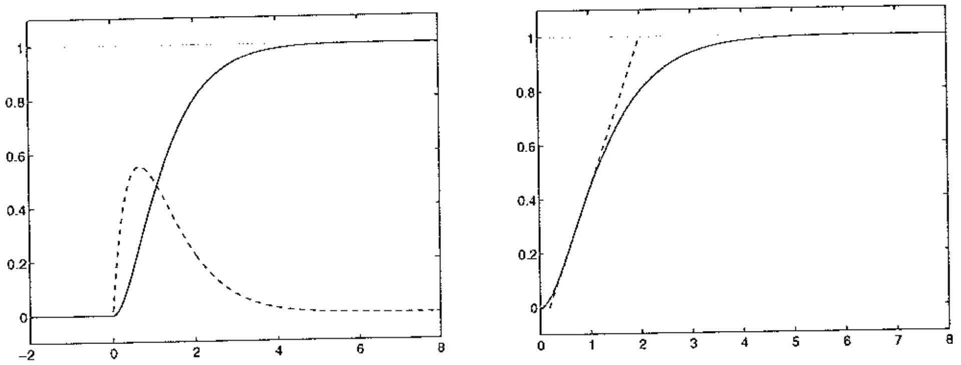


Figure C.5: Step response of an asymptotically stable second-order system with critical damping ($\zeta = 1$).

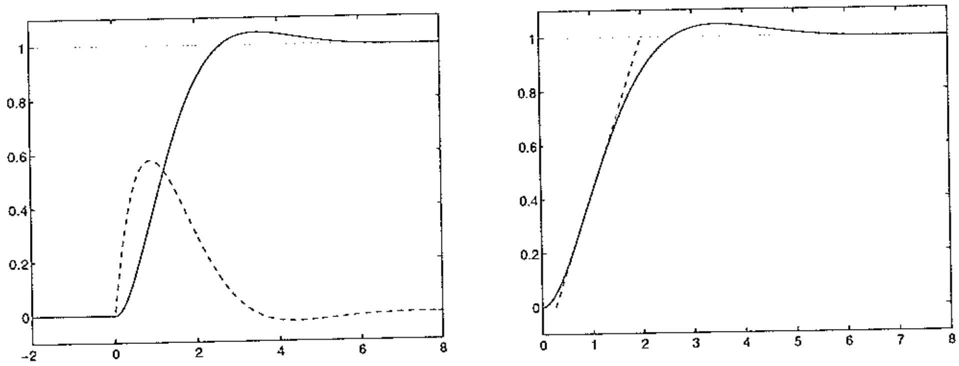


Figure C.6: Step response of an asymptotically stable second-order system with low damping ($\zeta = 0.7$).

Root Loci

The results for low, critical and high damping can be combined to express the poles $-a$ and $-b$ in terms of the positive time constant T and all nonnegative damping ratios ζ. Note that the poles are complex for low damping and real for critical or high damping.

$$-a, -b = \begin{cases} \mp\frac{1}{T}\frac{\pi}{2}j & (\zeta \to 0) \\ -\frac{1}{T}\left(2\zeta + \frac{1}{\sqrt{1-\zeta^2}} \arctan \frac{\sqrt{1-\zeta^2}}{\zeta}\right)(\zeta \pm j\sqrt{1-\zeta^2}\,) & (0 < \zeta < 1) \\ -\frac{3}{T} & (\zeta = 1) \\ -\frac{1}{T}\left(2\zeta + \frac{1}{2\sqrt{\zeta^2-1}} \log \frac{\zeta+\sqrt{\zeta^2-1}}{\zeta-\sqrt{\zeta^2-1}}\right)(\zeta \pm \sqrt{\zeta^2-1}\,) & (\zeta > 1) \\ -\infty, 0 & (\zeta \to \infty) \end{cases}$$

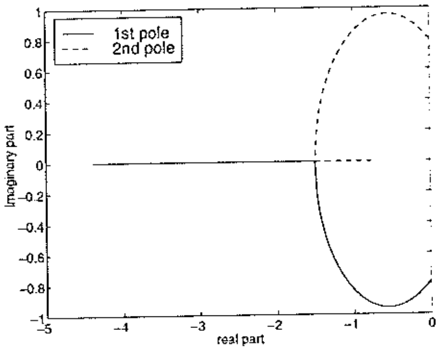


Figure C.7: Poles of a second-order system for $T = 2$ s and $0 \leq \zeta \leq 1.4$.

Although the second derivative of the step response of a second-order system is still not steady at $t = 0$, it is bounded. This is sufficient for excitation functions of mechanical systems, where only the natural derivatives *velocity* and *acceleration* occur.

Appendix D

MATLAB Source Code

D.1 Image Processing

findball.m

Corresponding code to the illustration in Figure 2.2, where we have no previous knowledge about the position of the ball.

```
function pos=findball(image)

% Find the ball on the image.
% The ball is preferably placed near the center of the plate.
%
% This function uses Line Search.
%
% See also trackball.

step=40;       % distance between two parallel lines
border=60;     % area near the edge of the plate
threshold=1.5; % parameter for the linesearch algorithm (ball&beam)
color=90;      % average value of a pixel that belongs to the ball
margin=20;     % pixels with a value between color-margin and
               %   color+margin are considered to be part of the ball

[height,width]=size(image)

figure(1); hold off; clf; imagesc(image); axis image; hold on;

step=round(step);
border=round(border);
halfstep=round(step/2);
halfheight=round(height/2);


% Search from the left-hand side
% ------------------------------
horizontal=1:width-2*border;   % search interval
sig=1;   % signature of offset
offset=0;
max_offset=halfheight-border;
while offset<max_offset
  vertical=halfheight+sig*offset
  distance=linesearch(image, [vertical;border], 0, horizontal, threshold)
  if (distance>0) & abs(image(vertical,border+distance+halfstep)-color)<margin
```

```
    left=[vertical;border+distance]
    break;
  end
  if sig==-1, offset=offset+step; end
  sig=-sig;
end
if offset>max_offset, error('Did not find the ball.'); end

figure(1); hold on; plot(left(2),left(1),'*');


% Search from the right-hand side
% -------------------------------
distance=linesearch(image, [vertical;width-border], pi, horizontal, threshold)
if (distance>0) & abs(image(vertical,width-border-distance-halfstep)-color)<margin
  right=[vertical;width-border-distance]
else
  error('Did not find the ball.');
end

figure(1); hold on; plot(right(2),right(1),'*');


% Search from the upper-hand side
% -------------------------------
vertical=1:height-2*border;
horizontal=round( (left(2)+right(2))/2 )
distance=linesearch(image, [border;horizontal], pi/2, vertical, threshold);
if (distance>0) & abs(image(border+distance+halfstep,horizontal)-color)<margin
  up=[border+distance;horizontal]
else
  error('Did not find the ball.');
end

figure(1); hold on; plot(up(2),up(1),'*');


% Search from the lower-hand side
% -------------------------------
distance=linesearch(image, [height-border;horizontal], -pi/2, vertical, threshold)
if (distance>0) & abs(image(height-border-distance-halfstep,horizontal)-color)<margin
  low=[height-border-distance;horizontal]
else
  error('Did not find the ball.');
end

figure(1); hold on; plot(low(2),low(1),'*');


% Compute the center of the circle
% --------------------------------
pos=round((low+up)/2)

figure(1); hold on; plot(pos(2),pos(1),'*r');
```

trackball.m

The rolling ball in Figure 2.3 is tracked on its moving path, which is defined by its last two positions.

```
function pos=trackball(image, lastpos)

% Find the ball on the image, provided its positions on the last two images.
% positions are stored in the columns of the matrix 'lastpos', with the last
% position in the first column, the second last position in the second column.
%
% This function uses Line Search.
%
% See also findball.

border=15;      % area near the edge of the plate
threshold=1.5; % parameter for the linesearch algorithm (ball&beam)

[height,width]=size(image)

figure(1); hold off; clf; imagesc(image); axis image; hold on;
plot(lastpos(2,:),lastpos(1,:),'*r');


% Search in the direction the ball is supposed to roll
% ----------------------------------------------------
startpoint=[lastpos(1,2);lastpos(2,2)]
angle=atan2(lastpos(1,1)-lastpos(1,2),lastpos(2,1)-lastpos(2,2))

interval=1:maxdist(image,startpoint,angle,border);
distance=linesearch(image, startpoint, angle, interval, threshold);
if distance>0
  low=startpoint+round(distance*[sin(angle);cos(angle)])
else
  error('Did not find the ball.');
end

figure(1); hold on; plot(low(2),low(1),'*');


% Search opposite to the ball's rolling direction
% -----------------------------------------------
startpoint=startpoint+round(length(interval)*[sin(angle);cos(angle)])
if angle<=0
  angle=angle+pi;
else
  angle=angle-pi;
end
distance=linesearch(image, startpoint, angle, interval, threshold);
if distance>0
  up=startpoint+round(distance*[sin(angle);cos(angle)])
else
  error('Did not find the ball.');
end

figure(1); hold on; plot(up(2),up(1),'*');


% Search from the left-hand side when looking in the rolling direction
% --------------------------------------------------------------------
mean=round((low+up)/2);
if angle<=pi/2
  angle=angle+pi/2;
```

```
else
  angle=angle-3*pi/2;
end
distance=maxdist(image,mean,angle,border);
startpoint=mean+round(distance*[sin(angle);cos(angle)])
if angle<=0
  angle=angle+pi
else
  angle=angle-pi
end
interval=1:maxdist(image,startpoint,angle,border);
distance=linesearch(image, startpoint, angle, interval, threshold)
if distance>0
  left=startpoint+round(distance*[sin(angle);cos(angle)])
else
  error('Did not find the ball.');
end

figure(1); hold on; plot(left(2),left(1),'*');


% Search from the right-hand side when looking in the rolling direction
% ---------------------------------------------------------------------
startpoint=startpoint+round(length(interval)*[sin(angle);cos(angle)])
if angle<=0
  angle=angle+pi;
else
  angle=angle-pi;
end
distance=linesearch(image, startpoint, angle, interval, threshold);
if distance>0
  right=startpoint+round(distance*[sin(angle);cos(angle)])
else
  error('Did not find the ball.');
end

figure(1); hold on; plot(right(2),right(1),'*');


% Compute the center of the circle
% --------------------------------
pos=round((left+right)/2)

figure(1); hold on; plot(pos(2),pos(1),'*r');




% sub-function:

% Compute the distance to the border from a given point in a given direction
% --------------------------------------------------------------------------

function dist=maxdist(image,pos,angle,border)

[height,width]=size(image);
sine=sin(angle);
cosine=cos(angle);

if angle==0
  dist=width-pos(2)-border;
elseif (angle==pi)|(angle==-pi)
```

```
  dist=pos(2)-border;
elseif angle==pi/2
  dist=height-pos(1)-border;
elseif angle==-pi/2
  dist=pos(1)-border;
end
if sine>0
  vertical=height-pos(1)-border;
elseif sine<0
  vertical=pos(1)-border;
end
if cosine>0
  horizontal=width-pos(2)-border;
elseif cosine<0
  horizontal=pos(2)-border;
end
dist=min( vertical/abs(sine), horizontal/abs(cosine) );
```

quickfind.m

All pixels are thresholded to compute the center of the ball in Figure 2.5.

```
% Find the ball on the image.
% The algorithm is written in C style as a preparation for the online
% implementation, and then in Matlab style to check the results.
%
% This script uses Global Search.


% Store the image in a one-dimensional array
% -----------------------------------------
layer=imageG; % Green layer of RGB image
[height,width]=size(layer);
data=zeros(1,height*width);
i=1;
for row=1:height
  for col=1:width
    data(i)=layer(row,col);
    i=i+1;
  end
end


% Find the center of the ball using pseudo C code
% -----------------------------------------------
indentrow=12; indentcol=16; % the data is partly invalid
                            %   due to a hardware defect
threshold=120; % used to determine whether or not
               % a particular pixel belongs to the ball
xsum=0; ysum=0; N=0;
i=1+indentrow*width;
for row=1+indentrow:height
  i=i+indentcol;
  for col=1+indentcol:width
    if data(i)<threshold
      xsum=xsum+col;
      ysum=ysum+row;
      N=N+1;
    end
    i=i+1;
  end
end
x=xsum/N-indentcol   % local coordinates
y=ysum/N-indentrow   %

   % draw
extract=layer(1+indentrow:height,1+indentcol:width);
ball=extract<threshold;
figure; imagesc(ball);
hold on; plot(x,y,'*y');


% Find the center of the ball using Matlab commands
% -------------------------------------------------
[xsamp,ysamp]=meshgrid(1:size(extract,2),1:size(extract,1));
N_=sum(sum(ball));
x_=sum(sum(ball.*xsamp))/N_
y_=sum(sum(ball.*ysamp))/N_

   % draw
plot(x_,y_,'*w');
```

D.2 Controller Design

beamsim.m

The non-linear and the linearized continuous-time models of the Ball and Beam system in Figure 4.1 are excited by smoothened rectangular and a sine function.

```
m=0.01;         % mass (irrelevant for the model)
r=0.02;         % radius
J=2/3*m*r^2;    % moment of inertia of a hollow ball
g=9.81;         % gravitational constant

c=m*r^2/(m*r^2+J);    % system constant

N=100;        % cut-off frequency of approx. derivative (rad/s)


alpha=20;     % step of inclination angle

pos0=0.6;     % initial position
vel0=0;       % initial velocity

rise=0.1;     % time for u to rise from 0 to alpha
sdur=1;       % duration of one step

fsize=15;     % font size for figures


inputchoice = 3;     % Choose the input here!     <--  <--  <--

switch inputchoice

case 1   %   ----
%-----     _|
%
tu1=linspace(0,rise,101);
tu2=(rise+0.01):0.01:(sdur);
tu=[tu1 tu2]';
u=alpha*[(1-cos(pi*tu1/rise))/2 ones(size(tu2))]';

case 2   %   --
%-----     _|  |____
%
tu1=linspace(0,rise,101);
tu2=(rise+0.01):0.01:(sdur-rise-0.01);
tu3=linspace(sdur-rise,sdur,101);
tu4=(sdur+0.01):0.01:(2*sdur);
tu=[tu1 tu2 tu3 tu4]';
u=alpha*[(1-cos(pi*tu1/rise))/2 ones(size(tu2)) ...
         (1+cos(pi*(tu3-sdur+rise)/rise))/2 -zeros(size(tu4))]';

case 3   %   --
%-----     _|  |   ----
%              |__|
tu1=linspace(0,rise,101);
tu2=(rise+0.01):0.01:(sdur-rise-0.01);
tu3=linspace(sdur-rise,sdur+rise,201);
tu4=(sdur+rise+0.01):0.01:(2*sdur-rise-0.01);
tu5=linspace(2*sdur-rise,2*sdur,101);
tu6=(2*sdur+0.01):0.01:(3*sdur);
tu=[tu1 tu2 tu3 tu4 tu5 tu6]';
```

```
u=alpha*[(1-cos(pi*tu1/rise))/2 ones(size(tu2)) ...
         cos(pi*(tu3-sdur+rise)/rise/2) -ones(size(tu4)) ...
         -(1+cos(pi*(tu5-2*sdur+rise)/rise))/2 zeros(size(tu6))]';

case 4   % sine
%-----
%
omega=0.4;   % frequency (rad/s)
tu=0:0.1:20*sdur;
u=alpha*sin(omega*tu+asin(pos0/alpha));

end %switch


figure(1); clf;
plot(tu,u,'k');
set(gca,'FontSize',fsize);
axis([0 max(tu) -1.1*abs(alpha) 1.1*abs(alpha)]);
xlabel('time [s]'); ylabel('angle [deg]');

[t,x,y]=sim('beamN',tu);

figure(2); clf;
plot(t,y(:,1),'k-',t,y(:,2),'k--');
set(gca,'FontSize',fsize);
xlabel('time [s]'); ylabel('position [m]');
legend('non-linear','linearized',3);

figure(3); clf;
plot(t,x(:,7),'k-',t,x(:,8),'k--');
set(gca,'FontSize',fsize);
xlabel('time [s]'); ylabel('velocity [m/s]');
legend('non-linear','linearized',4);
```

beamcontrsim.m

Different design methods for the continuous-time state feedback controller in Figure 4.12 are presented. Time responses of both the non-linear and the linearized plant are computed for different position reference trajectories. The closed-loop bandwidth is computed as well.

```
m=0.01;          % mass
r=0.02;          % radius
J=2/3*m*r^2;     % moment of inertia for a spherical shell
g=9.81;          % gravitational constant

c=m*r^2/(m*r^2+J);   % system constant

N=50;           % cut-off frequency of approx. derivative (rad/s)


A=[0 1; 0 0];
B=[0; -c*g];
C=[1 0];
D=0;


refmax=0.1;   % step or amplitude of position reference
reffreq=0.5*2*pi; % frequency of sinusoid reference (rad/s)

pos0=0.4;      % initial position
vel0=0.0;      % initial velocity

rise=0.1;      % time for ref to rise from 0 to refmax
sdur=2;        % duration of one step

fsize=15;      % font size for figures


% Input
%------

inputchoice = 6;   % Choose the input here!   <--  <--  <--

switch inputchoice

case 0   % Heaviside
%-----
tr=[0:0.01:sdur]';
ref=pos0+refmax*ones(size(tr));

case 1   % cos step
%-----
tr1=linspace(0,rise,101);
tr2=(rise+0.01):0.01:(sdur);
tr=[tr1 tr2]';
ref=pos0+refmax*[(1-cos(pi*tr1/rise))/2 ones(size(tr2))]';

case 2   % cosh step
%-----
c1=1/2/(sqrt(2)-1);
c2=2*asinh(1);
c3=c2/rise;
tr1=0:0.01:1/2*rise;
tr2=1/2*rise+0.01:0.01:rise;
tr3=rise+0.01:0.01:sdur;
```

```
tr=[tr1 tr2 tr3]';
ref=pos0+refmax*[c1*(cosh(c3*tr1)-1) -c1*(cosh(c3*tr2-c2)-2*sqrt(2)+1) ones(size(tr3))]';

case 3   % exp step (low damping)
%-----
zr=0.9;
wnr=(2*zr+atan(sqrt(1-zr^2)/zr)/sqrt(1-zr^2))/rise;
sr=zr*wnr;
wdr=wnr*sqrt(1-zr^2);
tr=[0:0.01:sdur]';
ref=pos0+refmax*(1-exp(-sr*tr).*(cos(wdr*tr)+sr/wdr*sin(wdr*tr)));

case 4   % exp step (critical damping)
% ----
ar=3/rise;
tr=[0:0.01:sdur]';
ref=pos0+refmax*(1-exp(-ar*tr).*(1+ar*tr));

case 5   % exp step (high damping)
% ----
zr=1.7;
wnr=(2*zr+log((zr+sqrt(zr^2-1))/(zr-sqrt(zr^2-1)))/2/sqrt(zr^2-1))/rise;
ar=wnr*(zr+sqrt(zr^2-1));
br=wnr*(zr-sqrt(zr^2-1));
tr=[0:0.01:sdur]';
ref=pos0+refmax*(1-(ar*exp(-br*tr)-br*exp(-ar*tr))/(ar-br));

case 6   % sin
%-----
tr=[0:0.01:sdur*2]';
ref=pos0+refmax*(sin(reffreq*tr));

end %switch inputchoice



% State Feedback
%---------------

methodchoice = 4;   % Choose the method here!   <--  <--  <--

switch methodchoice

case 1   % natural frequency (bandwidth) specification
%-----
wn=5.0276;     % natural frequency (rad/s)
z=0.7;   % damping ratio

if z<=1
  if z<0
    display('Damping must be non-negative.')
    return;
  else % 0<=z<=1
    a=z*wn;
    b=wn*sqrt(1-z^2);
    P=[-a+b*i, -a-b*i]   % poles
  end %if
else % z>1
  a=z*wn;
  b=wn*sqrt(z^2-1);
  P=[-a+b, -a-b]   % poles
end %if
```

```
K=acker(A,B,P)        % feedback vector

case 2   % ITAE criterion
%-----
w0=4;    % cutoff frequency
a=1/sqrt(2);
b=a;

P=[-a+b*i, -a-b*i]*w0   % poles
K=acker(A,B,P)        % feedback vector

case 3   % Bessel polynomial
%-----
w0=4;
a=0.8660;
b=0.5000;

P=[-a+b*i, -a-b*i]*w0   % poles
K=acker(A,B,P)        % feedback vector

case 4   % LQR
%-----
Q=10;   % weighting factor for y
R=1;    % weighting factor for u

K=lqry(A,B,C,D,Q,R)   % feedback vector

end %switch methodchoice


K0=1/dcgain(A-B*K,B,C,D)   % static input filter


figure(1); clf;   % reference
plot(tr,100*ref,'k-');
set(gca,'FontSize',fsize);
axis([0 max(tr) 100*pos0-1.1*abs(100*refmax) 100*pos0+1.1*abs(100*refmax)]);
xlabel('time [s]'); ylabel('reference [cm]');

[t,x,y]=sim('beamcontrN',max(tr));

figure(2); clf;   % position
plot(t,100*y(:,1),'k-', t,100*y(:,2),'k--');
set(gca,'FontSize',fsize);
xlabel('time [s]'); ylabel('position [cm]');
legend('non-linear','linearized',4);

figure(3); clf;   % alpha
if inputchoice <= 5
  plot(t,(180/pi)*y(:,3),'k-', t,(180/pi)*y(:,4),'k--', t,zeros(size(t)),'k:');
else % inputchoice >=6
  plot(t,(180/pi)*y(:,3),'k-', t,(180/pi)*y(:,4),'k--');
end %if
set(gca,'FontSize',fsize);
xlabel('time [s]'); ylabel('alpha [deg]');
legend('non-linear','linearized',4);

figure(4); clf;   % alpha dot
y3dot=zeros(length(t)-1,1);
y4dot=zeros(length(t)-1,1);
for index=1:length(t)-1
```

```
  y3dot(index)=( y(index+1,3)-y(index,3) )/( t(index+1)-t(index) );
  y4dot(index)=( y(index+1,4)-y(index,4) )/( t(index+1)-t(index) );
end %for
plot(t(1:length(t)-1),y3dot,'k-', t(1:length(t)-1),y4dot,'k--');
set(gca,'FontSize',fsize);
xlabel('time [s]'); ylabel('alpha dot [rad/s]');
legend('non-linear','linearized',4);

figure(5); clf;   % reference and position
plot(tr,100*ref,'k:', t,100*y(:,1),'k-', t,100*y(:,2),'k--');
set(gca,'FontSize',fsize);
xlabel('time [s]'); ylabel('position [cm]');
legend('reference','non-linear','linearized',4);

Ac=A-B*K;
Bc=B*K0;

figure(6); clf; bode(Ac,Bc,C,D);
set(gca,'FontSize',fsize);
[mag,phase,omega0]=bode(Ac,Bc,C,D);
[mag,phase,omega]=bode(Ac,Bc,C,D,1, ...
                     logspace(log10(min(omega0)),log10(max(omega0)),1000));
bwpoint=10^(-3/20);   % -3dB (bandwidth)
aux=mag-bwpoint;
[minimum,index]=min(abs(aux));

bandwidth=omega(index)   % rad/s (without estimator)
```

beamDcontrdesign.m

A digital full state feedback controller with an estimator and an additional integrator is designed based on the discretized linear model of the plant. Sensitivity and complementary sensitivity functions are considered as well as the crossover frequency.

```
m=0.1;          % mass
r=0.027;        % radius
J=2/5*m*r^2;    % moment of inertia of a solid ball
g=9.81;         % gravitational constant

c=m*r^2/(m*r^2+J);   % system constant

A=[0 1; 0 0];
B=[0; -c*g];
C=[1 0];
D=0;


height = 0.1;     % height of the step (m)

hw=0.002;         % measurement noise (m)

pos0=0.4;         % initial position (m)
vel0=0.0;         % initial velocity (m/s)
poserr0=-0.01;    % initial position error (m)
velerr0=-0.01;    % initial velocity error (m/s)

fsize=15;   % font size for figures


% Discretization
% --------------

Ts=0.25;
[Phi,Gamma]=c2d(A,B,Ts);
sys=ss(Phi,Gamma,C,D,Ts);

PhiI=[Phi zeros(size(Gamma)); Ts*C 1];
GammaI=[Gamma; 0];
CI=[C 0];
DI=D;
sysI=ss(PhiI,GammaI,CI,DI,Ts);


% State feedback
% --------------

sfmethod = 1;   % change method here

switch sfmethod
case 1   % dominant second order poles
  wn=4;    % equivalent s-plane natural frequency (rad/s)
  z=0.8;   % equivalent s-plane damping ratio
  wi=0.2;   % integrator pole

  if z<=1
    if z<0
      display('Damping must be non-negative.')
      return;
    else % 0<=z<=1
```

```
   a=z*wn;
   b=wn*sqrt(1-z^2);
   P=exp(Ts*[-a+b*i, -a-b*i, -wi]);   % poles
  end %if
 else % z>1
  a=z*wn;
  b=wn*sqrt(z^2-1);
  P=exp(Ts*[-a+b, -a-b, -wi]);   % poles
 end %if

 K=acker(PhiI,GammaI,P)       % feedback vector

case 2   % ITAE criterion
 w0=4;   % equivalent s-plane cut-off frequency (rad/s)
 a=0.5210;
 b=1.068;
 wi=0.7081;

 P=exp(Ts*[-a+b*i, -a-b*i, -wi]*w0);   % poles
 K=acker(PhiI,GammaI,P)              % feedback vector

case 3   % Bessel polynomial
 w0=4;   % equivalent s-plane cut-off frequency (rad/s)
 a=0.7455;
 b=0.7112;
 wi=0.9420;

 P=exp(Ts*[-a+b*i, -a-b*i, -wi]*w0);   % poles
 K=acker(PhiI,GammaI,P)          % feedback vector

case 4   % LQR
 Q=10;   % weighting factor for y
 R=1;    % weighting factor for u

 K=lqry(sysI,Q,R)   % feedback vector

case 5   % deadbeat
 P=[0 0 0];
 K=acker(PhiI,GammaI,P)
end %switch


K0 = 1/dcgain(ss(Phi-Gamma*K(1:2),Gamma,C,D,Ts));   % static input filter

PhiCI=[Phi-Gamma*K(1:2) -Gamma*K(3); Ts*C 1];
GammaCI=[Gamma*K0; -Ts];
sysC=ss(PhiCI,GammaCI,CI,DI,Ts);

damp(sysC)
wnat=damp(sysC); wnat=wnat(2);  % equivalent s-plane natural frequency

[out,time,state]=step(sysC);
[out,time,state]=lsim(sysC,pos0+height*ones(size(time)),[],[pos0 vel0 0]');
in=K0*(pos0+height)-K*state';

figure(1); clf;
hold on;
plot(time,100*(pos0+height)*ones(size(time)),'k:');
stairs(time,100*out,'r');
hold off;
set(gca,'FontSize',fsize);
xlabel('time [s]'); ylabel('position [cm]');
```

```
figure(2); clf;
hold on;
plot(time,zeros(size(time)),'k:');
stairs(time,in*(180/pi),'b');
hold off;
set(gca,'FontSize',fsize);
xlabel('time [s]'); ylabel('angle [deg]');


% Bode diagram, using transformation z = exp(j*omega*Ts)
[mag,phase,omega0]=bode(sysC);

figure(3); clf;
bode(sysC,{min(omega0),floor(pi/Ts)});
set(gca,'FontSize',fsize);

[mag,phase,omega]=bode(sysC, ...
                  logspace(log10(min(omega0)),log10(max(omega0)),1000));
bwpoint=10^(-3/20);   % -3dB (bandwidth)
aux=mag-bwpoint;
[minimum,index]=min(abs(aux));

bandwidth=omega(index)   % rad/s (without estimator)


% Full Estimator
% --------------

emethod=1;   % change method here

switch emethod
case 1
  faster=1.5;       % speed factor compared to state feedback
  wnE=faster*wnat;   % equivalent s-plane natural frequency of estimator (rad/s)
  zE=0.8;            % equivalent s-plane damping ratio

  if zE<=1
    if zE<0
      display('Damping must be non-negative.')
      return;
    else % 0<=zE<=1
      aE=zE*wnE;
      bE=wnE*sqrt(1-zE^2);
      PE=exp(Ts*[-aE+bE*i, -aE-bE*i]);   % poles
    end %if
  else % zE>1
    aE=zE*wnE;
    bE=wnE*sqrt(zE^2-1);
    PE=exp(Ts*[-aE+bE, -aE-bE]);   % poles
  end %if

case 2
  PE=exp(Ts*[-0.5*wnat -2*wnat]);  % independent of state feedback

case 3
  PE=[0 0];   % deadbeat
end %switch

L=acker(Phi',C',PE)'              % estimator gain vector
```

```
% Regulated Loop with Full Estimator
% ----------------------------------

% states: position, velocity,
%         position estimation error, velocity estimation error

PhiR=[Phi-Gamma*K(1:2) Gamma*K(1:2) -Gamma*K(3);
      zeros(size(Phi)) Phi-L*C zeros(size(Gamma))
      Ts*C zeros(1,length(Phi)) 1];
GammaR=[Gamma*K0; zeros(size(Gamma)); -Ts];
CR=[C zeros(size(C)) 0];
DR=D;
sysR=ss(PhiR,GammaR,CR,DR,Ts);

[outR,time,stateR]=lsim(sysR,pos0+height*ones(size(time)),[], ...
                        [pos0 vel0 poserr0 velerr0 0]');
inR=K0*(pos0+height)-K(1:2)*( stateR(:,1:2)-stateR(:,3:4) )'-K(3)*stateR(:,5)';
                % state estimation = state - estimation error

figure(4); clf;
hold on;
plot(time,100*(pos0+height)*ones(size(time)),'k:');
stairs(time,100*out,'r-');
stairs(time,100*outR,'r--');
hold off;
set(gca,'FontSize',fsize);
xlabel('time [s]'); ylabel('position [cm]');
legend('reference','without estimator','with full estimator',4);

figure(5); clf;
hold on;
stairs(time,in*(180/pi),'b-');
stairs(time,inR*(180/pi),'b--');
% plot(time,zeros(size(time)),'k:');
hold off;
set(gca,'FontSize',fsize);
xlabel('time [s]'); ylabel('angle [deg]');
legend('without estimator','with full estimator',4);


% Noise    y=Cx+w
% -----

% closed-loop transfer function w->u (with full estimator)
PhiU=PhiR;
GammaU=[zeros(size(Gamma)); L; -Ts];
CU=[-K(1:2) K(1:2) -K(3)];
DU=0;
sysU=ss(PhiU,GammaU,CU,DU,Ts);

% closed-loop transfer function w->y (with full estimator)
PhiY=PhiU;
GammaY=GammaU;
CY=[C zeros(size(C)) 0];
DY=-1;
sysY=ss(PhiY,GammaY,CY,DY,Ts);

noiseN=50;   % number of iterations for noise
w=hw*randn(length(time),noiseN);
inRw=zeros(length(time),noiseN);
outRw=zeros(length(time),noiseN);
```

```
for index=1:noiseN
  [inRw(:,index),time]=lsim(sysU,w(:,index),[], ...
                         [0 0 poserr0 velerr0 0]');   % initial states zero!
end %for

for index=1:noiseN
  [outRw(:,index),time]=lsim(sysY,w(:,index),[], ...
                         [0 0 poserr0 velerr0 0]');   % initial states zero!
end %for

figure(6); clf;
hold on;
% plot(time,zeros(size(time)),'k:');
for index=1:noiseN
  stairs(time,(inR'+inRw(:,index))*(180/pi),'b-');
end %for
hold off;
set(gca,'FontSize',fsize);
xlabel('time [s]'); ylabel('angle [deg]');

figure(7); clf;
hold on;
plot(time,100*(pos0+height)*ones(size(time)),'k:');
for index=1:noiseN
  stairs(time,100*(outR+outRw(:,index)),'r-');
end %for
hold off;
set(gca,'FontSize',fsize);
xlabel('time [s]'); ylabel('position [cm]');


% Full Compensator
% ----------------

PhiE=Phi-L*C;

figure(8); clf;   % settling time of estimation error is `faster'
                  %  than settling time of closed loop step response
initial(ss(PhiE,Gamma,C,D,Ts),[height 0]);
set(gca,'FontSize',fsize);

PhiEC=[Phi-Gamma*K(1:2)-L*C -Gamma*K(3); zeros(1,length(Phi)) 1];
GammaEC=[L; Ts];
CEC=K;
DEC=0;

sysEC=ss(PhiEC,GammaEC,CEC,DEC,Ts);

damp(PhiEC)   % Compensator dynamics


% Sensitivity
% -----------

PC=series(sys,sysEC);
S=feedback(1,PC);   % sensitivity
T=feedback(PC,1);   % complementary sensitivity

nyqfreq=floor( (1/Ts/2)*(2*pi) );

figure(9); clf;
bode(S,{0.1 nyqfreq});
```

```
title('Sensitivity');

figure(10); clf;
bode(T,{0.1 nyqfreq});
title('Complementary Sensitivity');


int_PC={0.1 nyqfreq};
int_S={0.1 0.8};
int_T={7 nyqfreq};

[mag_PC,phase_PC,w_PC]=bode(PC,int_PC);
[mag_invS,phase_invS,w_invS]=bode(inv(S),int_S);
[mag_T,phase_T,w_T]=bode(T,int_T);

mag_PC=reshape(20*log10(mag_PC),length(mag_PC),1);
mag_invS=reshape(20*log10(mag_invS),length(mag_invS),1);
mag_T=reshape(20*log10(mag_T),length(mag_T),1);

w_PC=reshape(w_PC,length(w_PC),1);
w_invS=reshape(w_invS,length(w_invS),1);
w_T=reshape(w_T,length(w_T),1);

figure(11); clf;   % loop shaping
semilogx(w_PC,mag_PC,'k-', w_invS,mag_invS,'k--', w_T,mag_T,'k-.', ...
         w_PC,zeros(size(w_PC)),'k:');
xlabel('frequency [rad/s]'); ylabel('magnitude [dB]');

[mag_PC_cross,phase_PC_cross,w_PC_cross] = ...
 bode(PC,logspace(log10(max(w_invS)),log10(min(w_T)),1000));
[minimum,index]=min(abs(log10(mag_PC_cross)));
crossover=w_PC_cross(index)   % rad/s
```

identify.m

Input-output data for the identification was collected in a closed-loop experiment. The model is presumed to be a double integrator and only the gain is identified. Indices 1 and 2 denote the data for the two degrees of freedom of the ball.

```
% y1, u1, y2, u2 must be loaded into workspace

T=0.25;
sys=c2d( tf(1,[1 0 0]), T );
[num,den]=tfdata(sys,'v');

% pre-filter
cutoff=0.5;   % cutoff frequency for Butterworth filter,
              % measured as fractions of the Nyquist frequency
order=5;

c0=1/(1+2/5)   % for a solid ball
g=9.81;
t=T*cumsum(ones(1,length(y1)-2));
n1=1; n2=40;   % displayed data extract

fsize=15;    % font size for figures


y1b=idfilt(y1',order,cutoff);
u1b=idfilt(u1',order,cutoff);

y1f=filter(den,1,y1b); y1f=y1f(3:length(y1f));
u1f=filter(num,1,u1b); u1f=u1f(3:length(u1f));

y1d=dtrend(y1f);
u1d=dtrend(u1f);

z1=[y1d u1d];

th1=arx(z1,[0 1 0],[],T);
present(th1)
[A1,B1]=th2poly(th1)
c1=B1/g

figure(1); clf;
hold on;
stairs(t(n1:n2),A1*y1d(n1:n2)*100,'k--');
stairs(t(n1:n2),B1*u1d(n1:n2)*100,'k-');
set(gca,'FontSize',fsize);
xlabel('time [s]'); ylabel('position [cm]');
legend('measured','predicted',4);
hold off;
box on;


y2b=idfilt(y2',order,cutoff);
u2b=idfilt(u2',order,cutoff);

y2f=filter(den,1,y2b); y2f=y2f(3:length(y2f));
u2f=filter(num,1,u2b); u2f=u2f(3:length(u2f));

y2d=dtrend(y2f);
u2d=dtrend(u2f);
```

```
z2=[y2d u2d];

th2=arx(z2,[0 1 0],[],T);
present(th2)
[A2,B2]=th2poly(th2)
c2=B2/g

figure(2); clf;
hold on;
stairs(t(n1:n2),A2*y2d(n1:n2)*100,'k--');
stairs(t(n1:n2),B2*u2d(n1:n2)*100,'k-');
set(gca,'FontSize',fsize);
xlabel('time [s]'); ylabel('position [cm]');
legend('measured','predicted',4);
hold off;
box on;


figure(3); clf;
stairs(t(n1:n2),(180/pi)*u1(n1:n2),'k');
set(gca,'FontSize',fsize);
xlabel('time [s]'); ylabel('angle [deg]');
box on;

figure(4); clf;
stairs(t(n1:n2),(180/pi)*u2(n1:n2),'k');
set(gca,'FontSize',fsize);
xlabel('time [s]'); ylabel('angle [deg]');
box on;
```

Bibliography

[1] Karl J. Åström and Björn Wittenmark.
Computer-Controlled Systems.
Prentice Hall, third edition, 1997.

[2] ABB Robotics AB, 721 68 Västerås, Sweden.
Product Manual IRB 2000, Jan 1991.

[3] K. Åström and A. Heyden.
Stochastic analysis of image acquisition and scale-space smoothing.
In J. Sporring, M. Nielsen, L. Florack, and P. Johansen, editors, *Gaussian Scale-Space Theory.* Kluwer Academic Publishers, 1997.

[4] Computer Boards, Inc., 125 High Street, #6, Mansfield, MA, U.S.A.
CIO-DAS1600 User's Manual, third edition, Jan 1994.

[5] Data Translation, Inc., 100 Locke Drive, Marlboro, MA, U.S.A.
MACH Series DT3153 User Manual, first edition, Oct 1996.

[6] Data Translation, Inc., 100 Locke Drive, Marlboro, MA, U.S.A.
Frame Grabber Software Development Kit, sixth edition, Feb 1997.

[7] Fredrik Emilsson.
Controlling of the ball and beam process using a video camera.
Master's thesis, Lund Institute of Technology, Sweden, 1997.
Department of Automatic Control.

[8] Gene F. Franklin, J. David Powell, and Abbas Emami-Naeini.
Feedback Control of Dynamic Systems.
Addison-Wesley, third edition, 1994.

[9] R. C. Gonzalez and P. Wintz.
Digital Image Processing.
Addison-Wesley, 1987.

[10] Klas Nilsson.
Industrial Robot Programming.
PhD thesis, Lund Institute of Technology, Sweden, 1996.
Department of Automatic Control.

Index