

Laboratory Exercise 3

LQG Control of Track Following in a DVD Player¹



Figure 3.1 The DVD servo.

Note: There are two **home assignments** in the lab. These must be prepared before the lab!

Optional: Bring your own music CD to play at the lab! CD-R (burned) CDs do not work as the tracks are invisible to the red DVD laser.

3.1 Introduction

A DVD player is, as you now know, highly dependent on automatic control to be able to data from the disk. In this laboratory exercise *you* will design a radial

¹Written by Bo Lincoln



The DVD servo was kindly donated by AudioDev AB in Malmö. <http://www.audiodev.com>

controller for our DVD servo which follows the track on a DVD or your favorite CD.

The task of your radial controller is to make the laser follow the track on the disk by moving the lens sideways. This is done by measuring the relative error between the track and the laser lens sideways, called the Radial Error (RE), and moving the lens using the control signal u .

There are several disturbances acting on the system. One is of course physical vibrations or shocks to the DVD player. Another very important disturbance is due to the nature of the DVD or CD discs – they are almost always *eccentric*. This means that the rotation center is not exactly the track center, see Figure 3.2. This leads to a very strong sinus-like disturbance sideways, which may be *up to 100 tracks sideways in one rotation!* Your controller must be able to compensate for these disturbances.

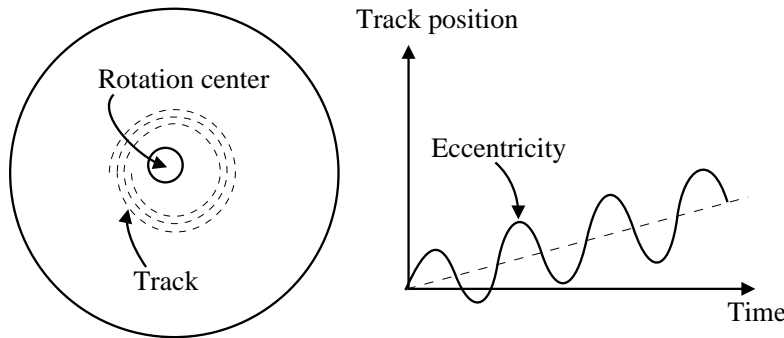


Figure 3.2 The disc is often a bit eccentric (i.e. not rotating around the track center). The resulting track position, which the Pick-Up-Head has to follow, is a sinus-like.

3.2 Process model

A linear model from the control signal u to the radial error RE has been estimated by inserting sinus signals at discrete frequencies to the servo at a low disk rotation speed. By measuring the RE signal and using an identification method called N4SID, a second order linear model has been found:

$$\dot{x}(t) = \underbrace{\begin{pmatrix} 13.4753 & -613.3032 \\ 160.4315 & -221.6478 \end{pmatrix}}_{A_G} x(t) + \underbrace{\begin{pmatrix} -9.5702 \\ -1045.8 \end{pmatrix}}_{B_G} u(t) \quad (3.1)$$

$$RE(t) = \underbrace{\begin{pmatrix} 3353.6 & 5.3953 \end{pmatrix}}_{C_G} x(t) \quad (3.2)$$

On transfer function form, this can be written as

$$RE(s) = G_{\text{radial}}(s)U(s) \quad (3.3)$$

The resulting bode diagram can be seen in Figure 3.3. This model is in the file `radialmodel.mat` on the web page.

Noise model

The above linear model describes how the output RE is affected by the input u . To be able to build an optimal Kalman filter, it is also necessary tell the Kalman filter what kind of noise to expect. We will call this the *noise model*.

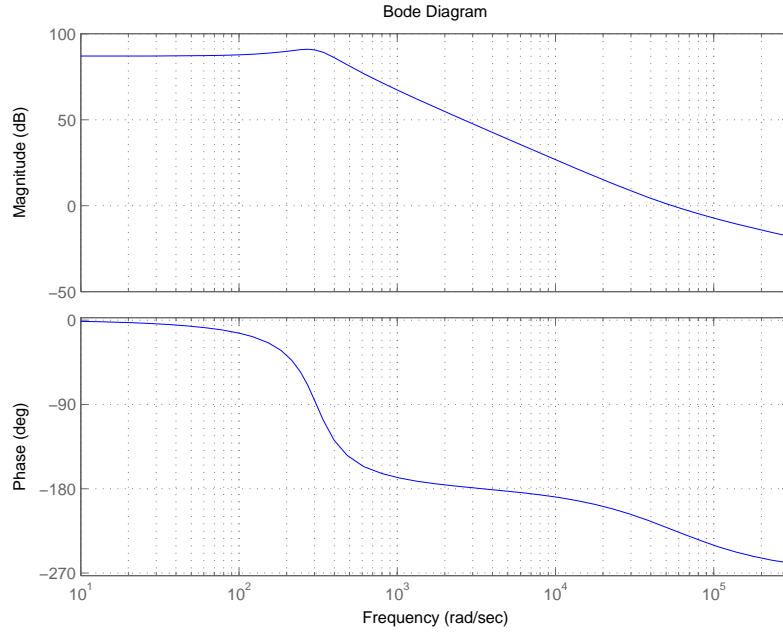


Figure 3.3 The estimated transfer function for the radial servo.

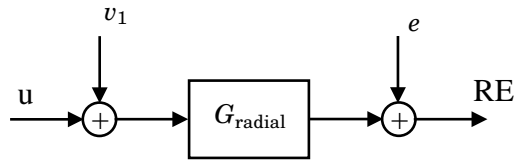


Figure 3.4 The noise model of the radial servo. The process noise v_1 and the measurement noise e are white and Gaussian.

In the first step of the lab, we will assume that white Gaussian noise affect the input and the output – see Figure 3.4.

- The noise v_1 is called the *process noise*, and describes the fact that the process will move a bit differently than the control signal ordered (or that something else changed in the system).
- The noise e is called the *measurement noise*, and models the noisy and otherwise non-perfect sensors.

Since the noise is Gaussian, all we need to describe it is its variance. We will let R_{v_1} denote the variance of v_1 and similarly R_e for e .

Note: The noise model tells the Kalman filter what to *expect*. Therefore, the variances R_{v_1} and R_e can be seen as design parameters, which we can tune to get the behavior we want from the Kalman filter!

3.3 Controller design

The main task in this lab is to LQG-design a controller for the radial servo (which follows the track sideways).

Design specifications

In Figure 3.5 a copy of the DVD specification is shown, from the section on radial

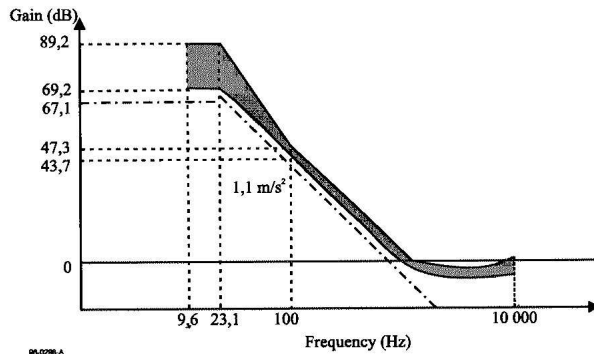


Figure 4 - Reference Servo for Radial Tracking

Bandwidth from 100 Hz to 10 kHz

$|1 + H|$ shall be within 20 % of $|1 + H_s|$.

The crossover frequency $f_0 = \omega_0 / 2\pi$ shall be specified by equation (III), where α_{\max} shall be 1,5 times larger than the expected maximum radial acceleration of 1,1 m/s². The tracking error e_{\max} shall not exceed 0,022 μm . Thus the crossover frequency f_0 shall be

$$f_0 = \frac{1}{2\pi} \sqrt{\frac{3 \alpha_{\max}}{e_{\max}}} = \frac{1}{2\pi} \sqrt{\frac{1,1 \times 1,5 \times 3}{0,022 \times 10^{-6}}} = 2,4 \text{ kHz} \quad (\text{III})$$

Figure 3.5 A copy from the DVD specification, standard ECMA-267. The plot shows the specified $|1 + G_{\text{radial}} \cdot C_{\text{radial}}|$, which is the inverse of the sensitivity function. You do not have to meet the specifications, but it is a good guideline of how the controller should behave.

control. This specifies the gain of the open loop transfer function plus 1, i.e

$$1 + G_{\text{radial}} \cdot C_{\text{radial}}$$

Note that this is simply the inverse of the sensitivity function S

$$S = \frac{1}{1 + GC}$$

When $|G_{\text{radial}} \cdot C_{\text{radial}}| \gg 1$, the “1” can simply be ignored, and the curve corresponds to the *open-loop transfer function*. In clear text, the specification requires the following:

- A low-frequency (< 23 Hz) gain of 70 dB or more for the open-loop system.
- A cut-off frequency of $\omega_c = 2.4 \text{ kHz} = 15 \text{ krad/s}$.

For your control designs, you do not have to meet the specifications exactly. They give a good indication on how the controller should behave, though.

LQG Controller

The LQG controller can as usual be seen as a state feedback

$$u(k) = -L\hat{x}(k|k)$$

together with a Kalman filter

$$\hat{x}(k+1|k) = \Phi\hat{x}(k|k-1) + \Gamma u(k) + K(y(k) - C\hat{x}(k|k-1))$$

$$\hat{x}(k|k) = \hat{x}(k|k-1) + K_f(y(k) - C\hat{x}(k|k-1)),$$

where Φ and Γ correspond to the sampled process model.

3.4 Part 1: A simple LQG design

Your first assignment is to create an LQG controller directly from the noise model in Section 3.2.

We will use the Matlab command `designlqg` to design our controller. This is a simple script included with the lab files, which combines the Matlab design commands for the optimal feedback gain L and the Kalman filter.

```
ctrl = designlqg(sys,Q1,Q2,R1,R2)
```

The noise and process model is given to the function as

$$\begin{aligned} x(k+1) &= \Phi x(k) + \Gamma u(k) + v(k) \\ y(k) &= Cx(k) + e(k) \end{aligned} \tag{3.4}$$

where v has variance R_1 and e has variance R_2 . In this case, since $v_1(k)$ goes the same way as the input signal u , we have $v(k) = \Gamma v_1(k)$. Note that v is a vector, and R_1 is therefore a matrix. Use `help designlqg` to get more information on `designlqg`.

HOME ASSIGNMENT 3.1—NOISE MODEL

The noise model in Section 3.2 is just a *model* of reality. What happens if we change this model? Try to figure out, without calculations, the answers to the following questions:

- Explain why the actual control loop changes when we alter the noise model (for example change the variance of some noise), even though the *real* process remains the same.
- What happens to the Kalman filter when we *increase* the measurement noise in the noise model. Does the Kalman filter put more trust in the model or in the measurements?
- If the Kalman filter is changed so that it trusts the measurements less, how is then the cut-off frequency (or bandwidth) of the controller changed?

□

ASSIGNMENT 3.1—SIMPLE LQG CONTROL

- Download the lab files from the web. Unzip using `unzip lab3_files.zip`. Run `initlab` in Matlab. The process model is now in the variable named `G`.
- Choose noise variance matrices R_1 and R_2 . As a first try, you can use $R_{v_1} = R_e = 1$.
- Choose feedback weight matrices Q for the design of the feedback gain L . *Hint*: Usually we have a lot of power to move the lens quickly, so there is no practical limit on u_{radial} .
- Create an LQG controller by using the command `designlqg` and the plant sampled at $h = 1/40000$.
- Check your designs open-loop Bode diagram (`margin` in Matlab is useful). Try to meet the bandwidth approximately. You will have problems meeting the low-frequency gain specification.
- Iterate the above until the specifications are met.
- Create a controller in the Simulink model of the DVD.
 - Open the model by running `lab_model` in Matlab.
 - Start Simulink by typing `simulink` in Matlab.
 - Create your controller by dragging blocks from the Simulink library to the model.

Repeat the steps above until it works in Simulink.

- Try it on the real DVD servo (see Appendix B).

□

3.5 Part 2: LQG controller with noise model

The controller in the first assignment is not able to remove the disk eccentricity error around the rotational frequency. The disk rotates at around 10-20 Hz, and oscillates up to 100 tracks sideways. Figure 3.6 shows how this oscillation can be included as noise in the model. The oscillation can not be modeled exactly, since

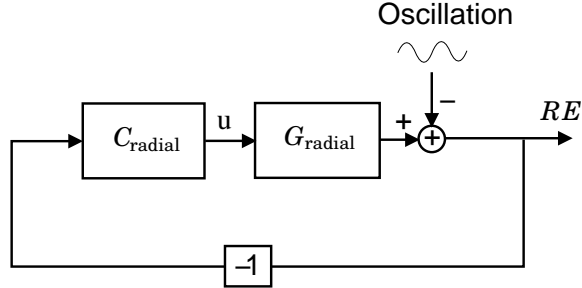


Figure 3.6 A model of how the disk oscillation affects the system. For example, if the oscillation offset at some point in time is +6.2 tracks, the DVD radial servo has to be at +6.2 tracks too to have zero RE .

we do not know the exact eccentricity of the disk. By modeling as track offset as noise, the Kalman filter can estimate the current offset.

We can give the Kalman filter a “hint” of what to expect, by modeling the eccentricity as white noise through a filter H as shown in Figure 3.7. The filter H should have a high gain in the frequency range where the oscillation acts. This noise model is now included in the Kalman filter design. In this way, the noise shape filter will be included in the controller; an integrator as a noise filter gives an integrator in the controller, and so on.

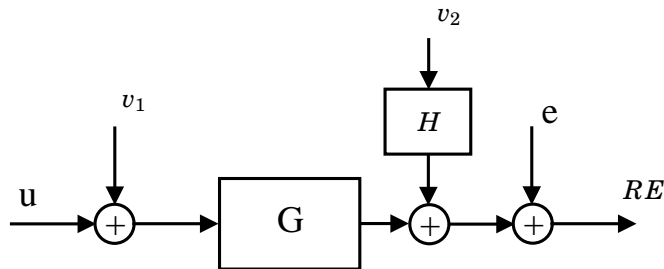


Figure 3.7 Noise model: There is both the old v_1 white process noise, and a track-offset which is modeled as the white noise v_2 through a filter H .

The Kalman filter now has three parameters; The v_1 (process noise) variance R_{v_1} , the v_2 (track offset noise) variance R_{v_2} and the e (measurement noise) variance R_2 . As these are relative weights, we effectively have two individual parameters to tune in the Kalman filter.

HOME ASSIGNMENT 3.2—PROCESS AND NOISE MODEL

The process model is given in Section 3.2 on state-space form. In discrete-time, it can be written as

$$\begin{aligned}x(k+1) &= \Phi_G x(k) + \Gamma_G u(k) \\ y(k) &= C_G x(k).\end{aligned}$$

Assume that the noise filter H is represented similarly with Φ_H , Γ_H , and C_H , and that the noise model (see Figure 3.7) has noise variances R_{v_1} , R_{v_2} and R_e .

To be able to use `designlqg`, we must construct an extended model which consists of *both* the process model *and* the noise filter:

$$\begin{aligned}x_e(k+1) &= \Phi x_e(k) + \Gamma u(k) + v(k) \\ y(k) &= C x_e(k) + e(k)\end{aligned}$$

where the process noise $v(t)$ has variance R_1 and the measurement noise $e(t)$ has variance R_2 .

What are the extended model matrices Φ , Γ , C , R_1 and R_2 expressed in terms of Φ_G , Γ_G , C_G , Φ_H , Γ_H , C_H , R_{v_1} , R_{v_2} and R_e ? The extended model should correspond to the noise model in Figure 3.7. \square

ASSIGNMENT 3.2—NOISE MODELING AND BETTER LQG

- What is the open-loop gain at 20 Hz for your control setup in assignment 1? What does that imply for the sensitivity function, and how much is the oscillation damped?
- What should the open-loop gain be at 20 Hz to dampen the oscillation by a factor of 3000?
- Create a noise filter H which shapes white noise to look more like the disturbance oscillation caused by the eccentricity.
- Use the result in home assignment 2 and `designlqg` to design a *with* the noise model H . Tune the variances R_1 and R_2 until you obtain a good controller.
- What is the order of this controller? Why?
- Simulate the control system in Simulink. Does the sinus-shaped error in the radial loop go away? If not, change H or R_1 or R_2 .
- When it works, try it on the real system.

\square

- **Radial Control On** This block is enabled by the **Focus Control On** block. As the disk is eccentric, the rotation of the disk will cause a number of tracks to pass back and forth. This block waits until the the tracks pass at a relatively slow rate (at the endpoints of the eccentricity), before outputting “On”. In this way, the radial controller gets a smooth start.
- **Radial Controller** This is where you put your LQG-designed controller.
- **Sledge Controller** As the track is a spiral from the center of the disk and out, the radial controller will have to move the lens further and further out. The sledge controller tries to avoid this by slowly following. The controller is a simple PI controller which moves the sledge (on which the laser and lens system sits) to keep the radial control signal around zero. The sledge suffers from a lot of friction, and therefore in practice, it moves in a “jumpy” fashion.
- **DVD Servo Model** This is a model of how the DVD player works in reality. As you design your controller, you will work with this model. When the controller works well, we replace this block by real analog in- and outputs connected to the real DVD player.

The inputs are u_{focus} and u_{radial} , controlling the lens in focus and radial direction, respectively, u_{sledge} for the sledge and u_{rotation} for the rotation motor. The outputs are FE (focus error) and RE (radial error).

Appendix B: The real DVD process and dSPACE

The real DVD process is controlled by a dSPACE card, which contains a 250 MHz PowerPC processor for control and has analog I/O.

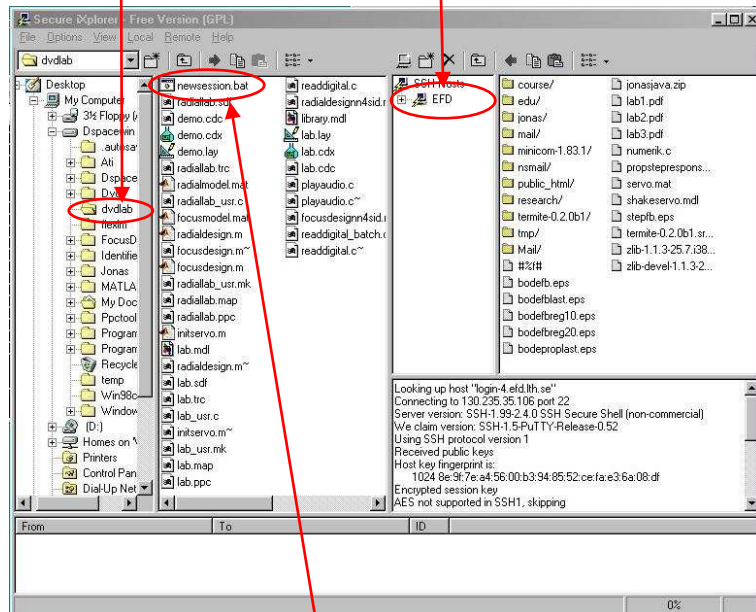
The dSPACE-card can be programmed directly from a Simulink Model. The real-time operation of the controller is then handled by a program called “dSPACE ControlDesk”.

How to run an experiment:

1. Copy fresh lab-files to the computer by double-clicking on “newsession.bat”.
2. Transfer your controller files (design “.m”-function and Simulink block) from your EFD account to the computer using “Secure iXplorer”.
3. If Matlab is not running, start it. Type “cd c:/dvdlab”.
4. Run “initservo” in Matlab. This will create a focus controller and other filters.
4. Run “lab”. This will open the Simulink block which is connected to the real process. Replace the (empty) blue controller block with your controller. Run your controller design file.
5. Type <ctrl-B>, or choose “Tools->Real-time Workshop->Build Model” from the menu. This will compile your Simulink model and run it on the dSPACE card.
6. If it is not running, start “dSPACE ControlDesk” from the desktop. Choose “File->Open Experiment” and open “lab.cdx”. This will bring up the user interface shown in Figure 3.10. Click the “Animation mode” icon to start plotting (see the figure).
7. Make sure the DVD process is powered (a lot of small green LEDs are on). Put a CD or DVD in the player.
8. Turn on the servos and the laser by turning the switch on the DVD board to “on”. Turn this switch off whenever anything goes wrong or you want to stop the DVD. **DO NOT TURN OFF POWER TO THE BOARD.** This will destroy the laser. Always keep the servo/laser-switch “off” when turning on and off power.

Transfer controller to c:\dvdlab

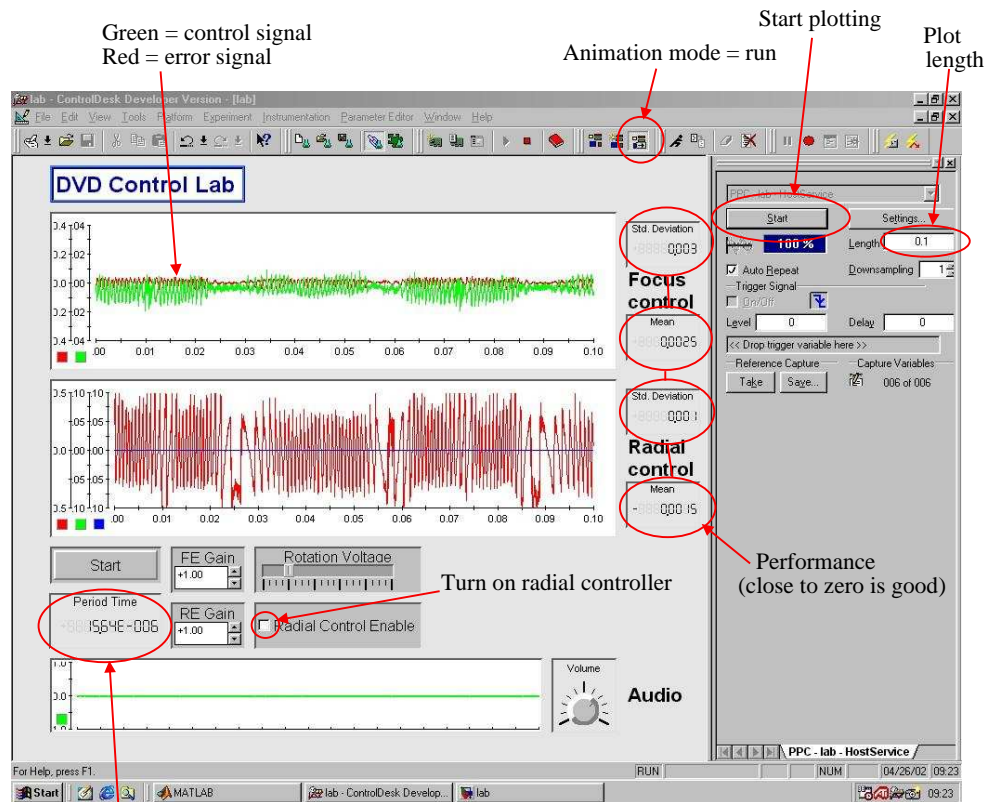
Click here to log in to an EFD account



Double-click "newsession.bat" to copy fresh template files to c:\dvdlab

Figure 3.9 The “Secure iXplore” program used to transfer your controller to the DVD PC. Click “EFD”, log in, and transfer your controller Simulink block and design “.m”-function to c:/dvdlab.

9. Push start in ControlDesk. This should start the focus servo, and you will (hopefully) be able to see the tracks pass by in the “Radial Error” signal.
10. Turn on your radial controller by checking “Radial Control Enable”. Good luck!



Execution time per period (should be less than $25e-6!$)

Figure 3.10 The “ControlDesk” user interface. To run an experiment, the program must be in “animation” mode, and the experiment is started by pressing the “Start” button below the plots.