# Multivariable Control

# Laboratory Session 3

# LQG Control of a Rotating Crane[1]

Department of Automatic Control
Lund University

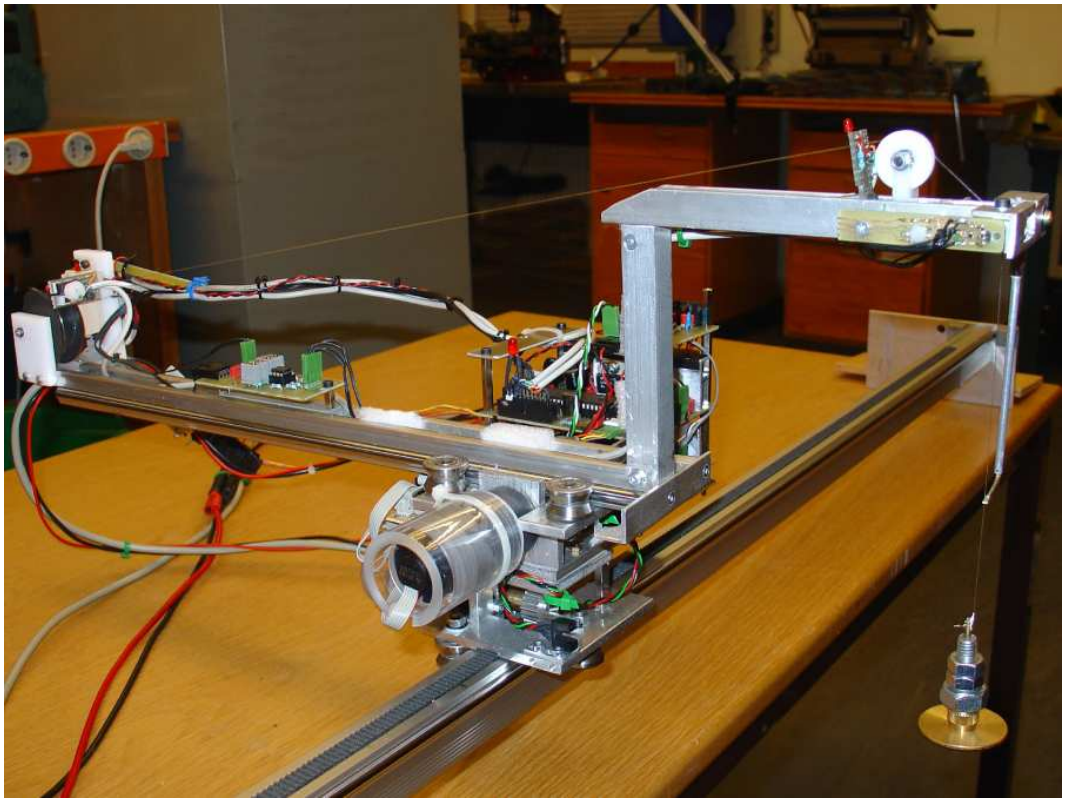

**Figure 1** The crane used in the laboratory session

## Preparations

- Read Chapters 9.1–9.4 in Glad & Ljung, *Control Theory — Multivariable and Nonlinear Methods*.
- Read through this lab manual carefully.
- Do Pre-lab assignments 1–6 in the lab manual.

## 1. Introduction

In this laboratory session we will design both LQ and LQG controllers for a crane with a load. The control objective is to make the load follow a circular orbit and at the same time keep the cart close to a reference position. Different design properties will be evaluated using Matlab/Simulink and a satisfying design will be tried on a real crane.

[1] Written by Per-Ola Larsson, latest update October 31, 2016.

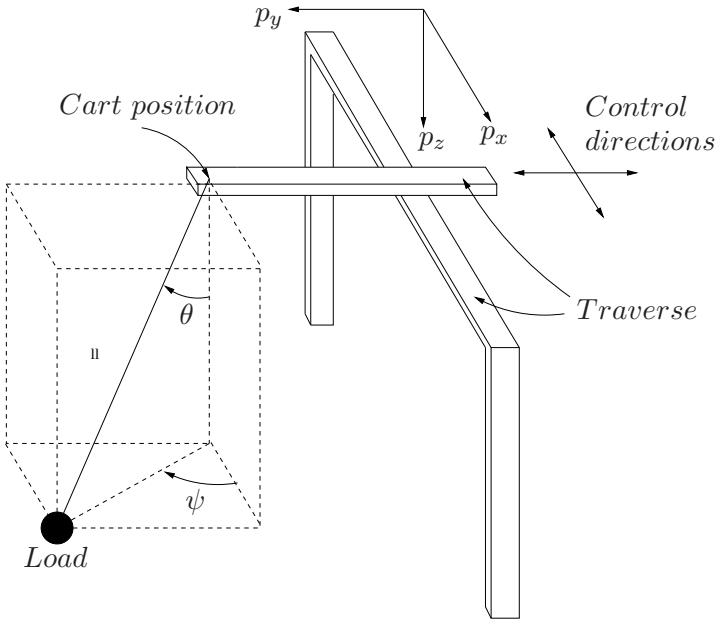


**Figure 2** Crane layout and coordinates. The cart position, i.e., the pivot point of the crane load, can be moved in the $(p_x, p_y)$-plane. The control objective is to keep the load rotating in a circular orbit while having the cart at a certain position.

## 2. Physical modeling of the crane

In the use of linear-quadratic-Gaussian (LQG) control, we design the state feedback and Kalman filter using a model of the process. The more accurate model, the better control we get.

Modelling of the crane and load can be done using Lagrange mechanics. Introduce the following system coordinates, as in Figure 2,

- cart position $p_x(t)$, $p_y(t)$, $p_z(t)$
- load angles $\theta(t)$ and $\psi(t)$.

Assuming that the load movements do not affect the position of the cart[1], gives the following non-linear equations of motions

$$
\begin{aligned}
&2l\dot{\theta}(t)\dot{\psi}(t)\cos\theta(t) + \ddot{p}_y(t)\cos\psi(t) - \ddot{p}_x(t)\sin\psi(t) + l\ddot{\psi}(t)\sin\theta(t) = 0 \\
&l\ddot{\theta}(t) - \frac{1}{2}l\dot{\psi}^2(t)\sin 2\theta(t) + g\sin\theta(t) + \ddot{p}_y\sin\psi(t)\cos\theta(t) \\
&\qquad\qquad + \ddot{p}_x(t)\cos\psi(t)\cos\theta(t) = 0.
\end{aligned}
$$

The LQG control framework assumes a linear time-invariant model of the system, therefore we need to linearize our equations. Assuming that we want the cart to be positioned at the origin of our coordinate system, the specified trajectory can be

[1] When is this almost true? What approximations have we done?

expressed using the system coordinates and its derivatives as

$$\begin{pmatrix} p_y(t) \\ \dot{p}_y(t) \\ p_x(t) \\ \dot{p}_x(t) \\ \theta(t) \\ \dot{\theta}(t) \\ \psi(t) \\ \dot{\psi}(t) \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \theta_o \\ 0 \\ \omega_o t \\ \omega_o \end{pmatrix}.$$

For a certain load length $l$ and angle $\theta_o$ we must have a certain rotational velocity[2], i.e., $\omega_o$. This constraint can be derived using simple physics to

$$\omega_o = \sqrt{\frac{g}{l \cos \theta_o}}, \tag{1}$$

where $g$ is the gravitational constant. Thus, the linearization trajectory depends on this constraint, and *can not be chosen arbitrarily.*

Linearizing the equations of motions around the trajectory and letting the accelerations in $p_x$- and $p_y$-directions be control inputs, gives a time-varying system. The states are the deviations from the nominal trajectory, i.e.,

$$\begin{pmatrix} \Delta\dot{p}_y \\ \Delta\ddot{p}_y \\ \Delta\dot{p}_x \\ \Delta\ddot{p}_x \\ \Delta\dot{\theta} \\ \Delta\ddot{\theta} \\ \Delta\ddot{\psi} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & s_1 & 0 & s_2 \\ 0 & 0 & 0 & 0 & 0 & s_3 & 0 \end{pmatrix} \begin{pmatrix} \Delta p_y \\ \Delta\dot{p}_y \\ \Delta p_x \\ \Delta\dot{p}_x \\ \Delta\theta \\ \Delta\dot{\theta} \\ \Delta\dot{\psi} \end{pmatrix}$$

$$+ \begin{pmatrix} 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ -b\cos\psi(t) & -b\sin\psi(t) \\ a\sin\psi(t) & -a\cos\psi(t) \end{pmatrix} \begin{pmatrix} \Delta u_x \\ \Delta u_y \end{pmatrix}, \tag{2}$$

where $\psi(t) = \omega_o t$ and

$$s_1 = \omega_o^2 \cos(2\theta_o) - \frac{g}{l}\cos\theta_o \qquad a = \frac{1}{l \sin\theta_o}$$

$$s_2 = \omega_o \sin 2\theta_o \qquad b = \frac{\cos\theta_o}{l}$$

$$s_3 = -2\omega_o \cot\theta_o.$$

[2]You might want to try this on your own. Try to make a pendulum go in a circular orbit with a constant angle $\theta_o$ and at the same time increase the rotational velocity.

You can recognize e.g., the double integrators for the positions and velocities in $p_x$- and $p_y$-directions, and the integrator for $\theta(t)$ in the system matrix.

Since we need a linear *time-invariant* system to design a standard LQ controller, we introduce a coordinate system that rotates with the load. The model can be transformed into these coordinates using the state-dependent input transformation matrix $P(\psi(t))$ and state transformation matrix $T(\psi(t))$,

$$P(\psi(t)) = \begin{pmatrix} \cos\psi(t) & -\sin\psi(t) \\ \sin\psi(t) & \cos\psi(t) \end{pmatrix},$$

$$T(\psi(t)) = \text{blockdiag}\left( \begin{pmatrix} 0 & -\sin\psi(t) & 0 & -\cos\psi(t) \\ \cos\psi(t) & 0 & -\sin\psi(t) & 0 \\ \sin\psi(t) & 0 & \cos\psi(t) & 0 \\ 0 & \cos\psi(t) & 0 & -\sin\psi(t) \end{pmatrix}, I_3 \right).$$

Applying these transformations gives a time-invariant system

$$\dot{x}(t) = Ax(t) + Bu(t), \tag{3}$$

with

$$A = \begin{pmatrix} 0 & 0 & 0 & -\omega_o & 0 & 0 & 0 \\ 0 & 0 & -\omega_o & 1 & 0 & 0 & 0 \\ -1 & \omega_o & 0 & 0 & 0 & 0 & 0 \\ \omega_o & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & s_1 & 0 & s_2 \\ 0 & 0 & 0 & 0 & 0 & s_3 & 0 \end{pmatrix}, \qquad B = \begin{pmatrix} -1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ -b & 0 \\ 0 & -a \end{pmatrix}.$$

This system will be used in the design of an LQ/LQG controller in the lab.

**Pre-lab assignment 1: State and input transformations**
Show by, for example, using Matlab symbolic toolbox, that applying $T(\psi(t))$ and $P(\psi(t))$ on the time-varying linearized system (2), we get the time invariant system in Eq. (3).

*Hint:* If we have the system

$$\dot{x}(t) = A(t)x(t) + B(t)u(t)$$

and apply $z(t) = T(t)x(t)$ and $u(t) = P(t)\tilde{u}(t)$ then we get

$$\dot{z}(t) = \left(\dot{T}(t) + T(t)A(t)\right) T^{-1}(t)z(t) + T(t)B(t)P(t)\tilde{u}(t).$$

**Matlab example:** Consider the system

$$\dot{x}(t) = \begin{pmatrix} \cos 2\omega t & -\sin 2\omega t \\ -\sin 2\omega t & -\cos 2\omega t \end{pmatrix} x(t) + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} u(t)$$

with the transformation matrices

$$T(t) = \begin{pmatrix} \cos \omega t & -\sin \omega t \\ \sin \omega t & \cos \omega t \end{pmatrix}, \qquad P(t) = \begin{pmatrix} \cos \omega t & \sin \omega t \\ -\sin \omega t & \cos \omega t \end{pmatrix}.$$

The transformations are done i Matlab as follows,

```
>syms w t real; %define w t as symbolic variables, real.
>A = [cos(2*w*t) -sin(2*w*t); -sin(2*w*t) -cos(2*w*t)];
>B = eye(2);
>T = [cos(w*t) -sin(w*t);sin(w*t) cos(w*t)];
>P = [cos(w*t) sin(w*t);-sin(w*t) cos(w*t)];
>Ahat = simplify((diff(T,t)+T*A)/T)
>Bhat = simplify(T*B*P)
```

which gives

```
>Ahat = [1 -w; w -1]
>Bhat = [1 0; 0 1]
```

### Interpretation of states and control signal

Now that we have a linear time-invariant system, it is a good idea to interpret the states. Let us first concentrate on the four states, $x_1(t), \ldots, x_4(t)$. Using the structure of $T(\psi(t))$ we can write

$$\begin{pmatrix} x_2(t) \\ x_3(t) \end{pmatrix} = \begin{pmatrix} \cos \psi(t) & -\sin \psi(t) \\ \sin \psi(t) & \cos \psi(t) \end{pmatrix} \begin{pmatrix} \Delta p_y(t) \\ \Delta p_x(t) \end{pmatrix}$$

$$\begin{pmatrix} x_1(t) \\ x_4(t) \end{pmatrix} = \begin{pmatrix} -\sin \psi(t) & -\cos \psi(t) \\ \cos \psi(t) & -\sin \psi(t) \end{pmatrix} \begin{pmatrix} \Delta \dot{p}_y(t) \\ \Delta \dot{p}_x(t) \end{pmatrix}.$$

These expressions are linear transformations using rotational matrices. Thus, $x_2(t)$ and $x_3(t)$ are interpreted as deviations from the nominal trajectory in the tangential and normal direction, respectively, and $x_1(t)$ and $x_4(t)$ are velocities in opposite normal and tangential direction, respectively, see Figure 3 for a graphical illustration.

Since $T(\psi(t))$ has the identity matrix in the bottom right corner the last three states

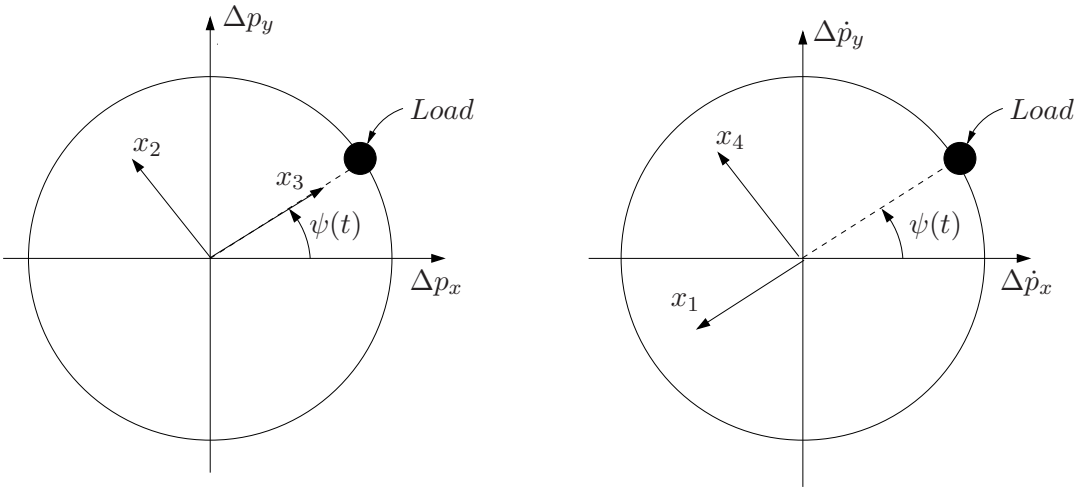


**Figure 3** Graphical interpretation of coordinate transformations for positions and velocities in $p_x$- and $p_y$-directions.

remain the same

$$x_5(t) = \Delta\theta(t)$$
$$x_6(t) = \Delta\dot{\theta}(t)$$
$$x_7(t) = \Delta\dot{\psi}(t).$$

**Summary of states**

From the discussion above, the states of the linear time-invariant system that we are to control are

$$\begin{pmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \\ x_4(t) \\ x_5(t) \\ x_6(t) \\ x_7(t) \end{pmatrix} = \begin{pmatrix} \text{velocity deviation} \\ \text{position deviation} \\ \text{position deviation} \\ \text{velocity deviation} \\ \theta\text{-deviation} \\ \dot{\theta}\text{-deviation} \\ \dot{\psi}\text{-deviation} \end{pmatrix}$$

These states will be available both in the simulation model and in the hardware implementation.

**Pre-lab assignment 2: Interpretation of control signals**

Express in words and a simple picture, analogous to Figure 3, an interpretation of the control signal transformation

$$\begin{pmatrix} \Delta u_x \\ \Delta u_y \end{pmatrix} = P(\psi(t)) \begin{pmatrix} u_1(t) \\ u_2(t) \end{pmatrix}.$$

*Hint: Consider e.g.* $(\, u_1(t) \quad u_2(t) \,)^T = (\, 1 \quad 0 \,)^T$. *What kind of accelerations* $(\, \Delta u_x(t) \quad \Delta u_y(t) \,)^T$ *will this correspond to?*

## 3. Process characteristics

In essentially all control problems, one must get familiar with the process dynamics before attempting to control it. In this section, we will build a model of the process, and investigate its characteristics by simulation.

**Assignment 1: Constructing and getting to know the system**

1. Download lab files `lab3_files.zip` from the course homepage, `http://www.control.lth.se/course/FRTN10/Multivariable-Control-Labs.html`.
2. Start Matlab R2015a by typing `VERSION=R2015a matlab` in a terminal.
3. Run `start_lab`.
4. In the opened Simulink model `LQGlab.mdl` we have access to the time-invariant state $x(t)$. Make sure that you understand the different variables in the `init_crane`-file. In this file, you will set initial values for the crane, set linearization trajectory, and design LQ and LQG controllers.
5. Try different initial values on the crane by setting them in the `init_crane`-file. Let the length of the load be 0.3 m, which is the same as on the real crane, and use $\theta_o(t) = 0.3$ rad to specify a desired trajectory. What happens to the $x(t)$-coordinates? Use the scopes to view the states. *Note: We can never have the pendulum hanging straight down since this gives a singular model, thus we must always have* `theta_0_init` *and* `psi_dot_init` $> 0$
   - What happens if we initialize the crane to follow the specified trajectory of $\theta_o(t) = 0.3$?
   - Compare the load angle $\theta(t)$ to $x_5(t)$. What is the relation? Can $\theta(t)$ and $x_5(t)$ be negative?
   - Verify the physical constraint in Eq. (1).
6. The linear time-invariant system in Eq. (3) can be loaded to workspace by using the command `[A, B] = getAandB(l_0_init, theta_0, psi_dot_0)`, see the `init_crane`-file. Compute the poles of the system. What is the process characteristics? Can you relate any of the poles to physical parameters?

## 4. Linear-quadratic (LQ) control

Now that we are familiar with the process, it is time to control it. We want it to follow the specified trajectory regardless of what the initial conditions are. That is, all states $x(t)$ should be asymptotically stable. The control strategy will of course be LQ control!

We will assume that all states of the system are measured and equal to the outputs, that is, we have the model

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

where we assume that the full state $x(t)$ can be measured. The system matrices were defined in Eq. (3). The cost function we are minimizing is the familiar

$$J = \int_0^\infty \left( x^T(t) Q_1 x(t) + u^T(t) Q_2 u(t) \right) dt. \tag{4}$$

**Pre-lab assignment 3: Control signal weighting**
If we have ( $u_1^{max}$, $u_2^{max}$ ) as maximum allowed values of the control signals, formulate a weighting matrix $Q_2$ such that the control signal part of the LQ cost function is

$$\|u(t)\|_{Q_2} = \int_0^\infty \left( \left( \frac{u_1(t)}{u_1^{max}} \right)^2 + \left( \frac{u_2(t)}{u_2^{max}} \right)^2 \right) dt.$$

**Assignment 2: LQ control of crane in simulation**

Throughout this assignment we want the crane to follow a trajectory with $\theta_o = 0.3$ rad and $\omega_o = \sqrt{\frac{g}{l \cos \theta_o}}$. The length of the load should be 0.3 m, i.e., the same as for the real crane.

1. Design a state-feedback controller using LQ technique by use of the Matlab function `lqr`, see `init_crane`-file. Use equal weight on outputs and control signals, i.e., $Q_1 = I$ and $Q_2 = I$.

   *Hint: Use* `help lqr` *in Matlab to figure out how the function works. Consider also the command* `diag`*.*

   - Is the closed loop stable? What guarantees do we have?
   - Plot the amplitude of the closed loop, `bodemag(ss(A-B*L, B, eye(7), 0))`. Do we have any resonance peaks in the closed loop?

2. Implement the state feedback using the Simulink model in Assignment 1, use the gain block in the Math library. Set the gain block to "matrix times vector"-mode. Try different initial values of the crane. Use scopes in Matlab to display different signals.

3. Try changing the weights on the control signals, try e.g. $Q_2 = 0.001I$ and $Q_2 = 1000I$.

   - What are the implications on the outputs?
   - What happens to the closed loop poles?
   - Any practical aspects to consider? *Hint: Check control signal.*

4. Tuning an LQ controller involves weighting the outputs and control signals *relative* to each other. Try weighting the outputs with $Q_1 = 10I$ and the control signals with $Q_2 = 0.1I$ and compare it with $Q_1 = 1000I$ and $Q_2 = 10I$, respectively. What is the difference? Why?

5. Turn on the measurement noise by setting the parameter `noise_variance` to e.g., $10^{-5}$ in the `crane_init`-file. What happens when you change the weights on the control signals? Check the control signals.

6. As you can see on the real crane, we can not move the cart a great distance, and the motors have limited acceleration capacity. With the initial conditions that the load is in a circular orbit with angle $\theta = 0.2$ rad and $\omega = \sqrt{\frac{g}{l \cos \theta}}$ rad/s, design a controller that fulfills the following specifications when the desired trajectory has $\theta_o = 0.3$ rad and $\omega_o = \sqrt{\frac{g}{l \cos \theta_o}}$ rad/s.

   (a) The cart should not accelerate faster than 0.3 m/s$^2$ due to physical limitations. Remember that the control signals are accelerations.

   (b) The cart should not move more than 2 cm in either direction, i.e., $p_x(t)$ and $p_y(t)$ should stay within $\pm 0.02$.

(c) We should reach the desired trajectory in approximately 6 s.

*To think about at the design*

(a) Pre-lab assignment 3.

(b) Remember what the different states $x(t)$ are.

(c) We are not so interested in the load having correct rotational velocity. It will come naturally from physical constraint if we follow rest of the trajectory. Thus, we should use a very small weight on $x_7$.

**Assignment 3: LQ control of real crane**

Once the LQ design gives acceptable control of the simulated crane, it will be tried on the real process. Running the crane requires using a computer to measure signals, compute control signals, and actuate the motors. This will be done in a sampled fashion. Hence our controller must be a sampled version at this stage, i.e., the discrete time equivalent of the continuous time version taught in the course. Discretizing a linear system is a straightforward procedure, and is a part of the course Real Time Systems[3] (FRTN01) given at the department. As the design is tried on the real crane, we change to the discrete time version. This will not affect the result of the control, nor the interpretation of signals.

1. Change the design to discrete time by using `lqrd` instead. Use sampling period 0.010 s. See `init_crane`.
2. Change to the discrete time controller in the Simulink model.
3. Switch to the crane hardware-block from the library `crane_lib` and run your controller on the real crane by help from the lab assistant. Do not forget to run `init_hardware.m` before running the real crane.

   **Note:** The hardware-block has four additional outputs, `y_volt_no_offset`, `y_volt_with_offset`, `x_volt_no_offset` and `x_volt_with_offset`. These are only used when calibrating the angle measurements of the crane, and should otherwise be connected to terminator-blocks.

   **Do not turn on the power supply if the Simulink model is running. The control signal might be large which can damage the crane. Always reset the crane by switching the power supply on and off and then pressing the reset button on the crane before a simulation is begun.**

## 5. Linear-quadratic-Gaussian (LQG) control

In many control problems, you do not have access to all the states of the system. This can be due to e.g., hard to position sensors, not physically possible to measure the state or the state has no physical interpretation as in the case of using an experimentally estimated model. In any of these cases, the Kalman filter might be the solution and the control structure will be as shown in Figure 4. In this part of the lab we assume that the measured states might not be the same as the ones we want to control. Thus, the process is described as

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) + v_1(t) \\ y(t) &= Cx(t) + v_2(t), \end{aligned}$$

[3] `http://www.control.lth.se/course/FRTN01`

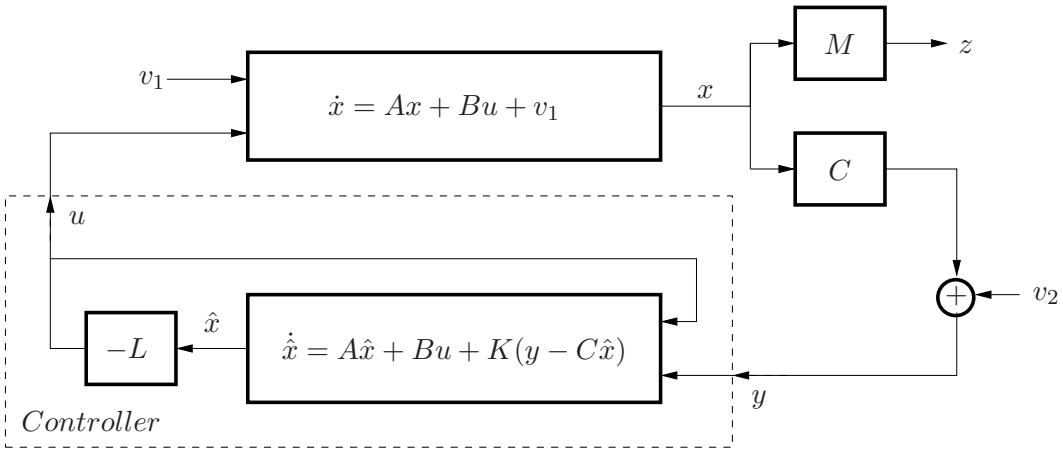


**Figure 4** Control structure when using an LQG controller on a process. The variable $y$ is the measurements, $z$ controlled variables, $x$ states of the process, $\hat{x}$ estimated states, $u$ control signal, $v_1$ process noise and $v_2$ measurement noise.

where $y(t)$ are the measurements. We also have process noise $v_1(t)$ acting on the states and measurement noise $v_2(t)$ acting on the measurements. The cost function we are trying to minimize is the same as in the LQ case, see Eq. (4), but now we must extend the controller to include a Kalman filter, i.e., the controller is

$$\begin{aligned}\dot{\hat{x}} &= A\hat{x} + Bu + K(y - C\hat{x})\\ u &= -L\hat{x}.\end{aligned}$$

**Pre-lab assignment 4: Kalman filter error dynamics**
Answer the following questions.

1. Explain in words what a Kalman filter does. What are its inputs and outputs?
2. Assume that we have the system

$$\begin{aligned}\dot{x}(t) &= Ax(t) + Bu(t) + v_1(t)\\ y(t) &= Cx + v_2(t).\end{aligned}$$

   What is the state estimation error dynamics for the corresponding Kalman filter?

**Pre-lab assignment 5: Kalman filter noise dependence**
Try to figure out, without calculations, the answers to the following questions,

1. What happens to the Kalman filter when we **increase** the intensity of the measurement noise? Does the Kalman filter put more trust in the model or in the measurements?
2. If the Kalman filter is changed so that it trusts the measurements less, how is then the cut-off frequency (bandwidth) of the controller changed?

**Pre-lab assignment 6: System structure and noise interpretation**
In the LQG design, we are assuming that only three states of the system are measured, $x_2(t)$, $x_3(t)$, and $x_5(t)$. Because of limitations on the physical process, we must be able to have weights on all states in the state-feedback design.

1. Write the model of the process in state-space form

$$\dot{x}(t) = Ax(t) + Bu(t) + v_1(t)$$
$$y(t) = Cx(t) + v_2(t),$$

   assuming that each state and measured output has unique noise signals acting on them.

2. Interpret the noise signals $v_i(t),\ i = 1, 2$. What do they model?

3. Assuming all noise signals are white and uncorrelated, what are the structures of the intensity matrices $R_1$ and $R_2$?

**Assignment 4: LQG control of crane in simulation**

Throughout this assignment we want the crane to follow a trajectory with $\theta_o = 0.3$ rad and $\omega_o = \sqrt{\frac{g}{l\cos\theta_o}}$. The length of the load should be 0.3 m, i.e., the same as for the real crane. It is now assumed that only $x_2(t)$, $x_3(t)$ and $x_5(t)$ are measured. It might be helpful to use Figure 4 as support for discussion.

1. Design a Kalman filter using the function `kalman` on the system in Pre-lab assignment 6, see `init_crane`. Begin with setting the simulated measurement noise intensity (`noise_variance`) to 0, and the estimated noise intensities to 1, i.e., $R_1 = I$ and $R_2 = I$. An implementation of the Kalman filter and state feedback can be found in `crane_lib`. Remember that the control signal input to the Kalman filter should be the same as the control signal input to the process.

   Implement the Kalman filter in the Simulink model. Simulate the system with different initial values *without using any feedback*, e.g. let the load go in a circular orbit not equal to the desired trajectory with control signals equal to 0. Does the Kalman filter converge to correct state-values?

   *Hint: Use the subsystem "Plot system" found in the library* `crane_lib`*. This subsystem takes the true state vector $x$ and estimated state vector $\hat{x}$ and outputs the pairs $(x_i(t), \hat{x}_i(t))$, $i = 1, \ldots, 7$ in scopes.*

2. Now try to change the measurement noise intensity $R_2$, try e.g., $R_2 = 0.01I$ and $R_2 = 100I$.

   - Does the convergence property change? Explain!
   - What happens to the error dynamics? Any similarities to the process characteristics? Explain! Look at the Kalman filter gain. Any conclusion about desired gain?

3. Try adding noise to the measurements by setting `noise_variance` to $10^{-5}$ in `init_crane`. Investigate the property of changing $R_2$ as above. Try different initial values of the crane.

   - Does the Kalman filter converge to correct values?
   - Any conclusion about desired gain? Compare to above. Trade-off?

4. Plot the Bode diagram (only amplitude) of the transfer functions from measurements to their filtered equivalents from the Kalman filter. How do they depend on the measurement noise? Can you relate the shape to the process characteristics?

   *Hint:* The plots can be generated from the `Kest`-object created when running `kalman` as follows

   ```
   bodemag(Kest(1:3,3:5));grid
   ```

5. In reality, the model and process do not behave exactly the same due to e.g. parameter estimation errors in the model. Change the length of the load by setting `l_error` to e.g. 0.03. This will set the crane load length to $l + 0.03$, that is, our model will have an error of 10% in the length parameter.

   - How do you model this uncertainty in $l$? What linearized states depends most on $l$?
   - What happens to the estimations? Do they converge to correct values? What is the Kalman filter trying to do?

6. Connect the control signal from the controller to the process and the Kalman filter, that is, now we have feedback. Set the error in load length to 0. Simulate the *closed loop* system.

   - Does the controller stabilize the system?
   - What is the order of the controller designed above? What are its states?
   - What is the controller transfer function $F_{uy}(s)$, $(y(t) \rightarrow u(t))$ expressed in Kalman filter gain $K$ and state-feedback gain $L$?

7. Plot the magnitude in a Bode diagram from measurements to control signal. How does the bandwidth of the controller depend on measurement noise? Try again e.g., $R_2 = 0.01I$ and $R_2 = 100I$. Compare to the above results.

   *Hint:* Use `bodemag(lqgreg(Kest, L));grid`

**Assignment 5: LQG control of real crane**

1. Figure out good settings of the weight matrices $Q_1$ and $Q_2$ and intensity matrices $R_1$ and $R_2$ by experience from the LQ experiment and the LQG control of the simulated crane.

2. Check with Simulink model that we have reasonable control when we start the simulation with the load in a circular orbit with $\theta = 0.2$ rad and $\omega_o = \sqrt{\frac{g}{l \cos\theta}}$. You do not have to fulfill the specifications from the LQ control problem.

3. Change the design to discrete time by using `lqrd` and `kalmd` instead, see `init_crane`. Change the Simulink model so that discrete time Kalman filter and state feedback are used, these are found in `crane_lib`.

4. Switch to the crane hardware-block from the library `crane_lib`. Run your controller on the real crane by help from the lab assistant. Do not forget to run `init_hardware.m`.

   **Do not turn on the power supply if the Simulink model is running. The control signal might be large which can damage the crane. Always reset the crane by switching the power supply on and off and then pressing the reset button on the crane before a simulation is begun.**

# A. Files for simulation/experiments

**crane_lib.mdl** Contains the simulation model, hardware model, input signal transformation and state transformation block.

**LQGlab.mdl** Contains a simulation model of the linearized time-invariant system, created by interconnecting the non-linear model, linearization, and the two transformation blocks. In this model, the LQ/LQG controllers are implemented.

**init_crane.m** Initialization and design file for the crane and controllers. All parameters will be set in this file.

**init_hardware.m** Sets up hardware parameters for experiments on real crane. Initializes velocity controllers.

**getAandB.m** Gives the system and input matrix of the linearized time-invariant system for a certain crane initialization.

**start_lab.m** Starts Simulink, opens `LQGlab.mdl` and `crane_init.m`.