EECS C106B / 206B Robotic Manipulation and Interaction **Spring 2020**

Lecture 5: Non-Linear Control

Scribes: Artun Dalyan, Léo Toulet

Topics covered

This lecture introduces non-linear control methods. We start by defining the necessary mathematical tools, and develop two control methods: control Lyapunov functions, and feedback linearization. Model predictive control was not covered due to lack of time.

Note: Here we talk about stability problems, but any control problem can be transformed into a stability problem by stabilizing the error $e(t) := x(t) - x_d(t)$.

5.1 Mathematical tools

5.1.1 Optimization

There exists four main types of optimization problems: two "easy" ones, and two "hard" ones.

- *Linear* and *quadratic* optimization problems are easy problems. There exists a global optimum. They are convex problems that can be solved efficiently (and in finite time!) by computers, and it is easy to know if a solution exists.
- *Quadratically constrained* and *non-linear* optimization are hard problems. Here, there exists no general algorithms to find a solution in finite time (NP problems). There is no global optimum for these problems. Plus, there is no way to know if a solution to the problem exists.

Any controller that relies on real-time optimization needs to operate in such a way that it only has to solve the first two types of optimization, otherwise we can offer no guarantees about the stability and responsiveness of the controlled system.

5.1.2 Lyapunov direct method

The Lyapunov direct method is used to analyse the stability of non-linear systems. Let $\dot{x} = F(x)$ be a general non-linear system and x_e be an equilibrium point. Let D be an open neighborhood around x_e. If there exists a function $V : D \longrightarrow R$ that is positive for all $x \neq x_e$, then:

- If $\dot{V} \leq 0$, $\forall x \in D$, the system is stable in the sense of Lyapunov.
- If $\dot{V} < 0$, $\forall x \neq x_e \in D$, the system is asymptotically stable.
- If $\dot{V} < -\gamma V$, $\forall x \neq x_e \in D$, the system is exponentially stable.

5.2 Lyapunov control

5.2.1 First attempt

This optimization method makes use of Lyapunov's direct method to find a suitable controller for a non-linear system. In this section, we will use the asymptotic stability criteria, but any criteria may be used with the appropriate modifications.

Let's define a controlled system $S : \dot{x} = f(x) + g(x)u$. Let x_e be an equilibrium of S and D an open neighborhood of x_e. Our objective is to find a controller $u(x)$ that stabilizes S. Let $u(x)$ be such a controller and $V(x)$ an adequate Lyapunov function, as defined in the previous section.

Differentiating V yields:

$$\dot{V} = \frac{\partial V}{\partial x}\frac{\partial x}{\partial t}$$
$$\dot{V} = \frac{\partial V}{\partial x}(f(x) + g(x)u(x))$$

or, using Lie derivatives:

$$\boxed{\dot{V} = L_f V + L_g V u}$$

A first attempt at finding a controller would be to solve the optimization problem of finding both u and V such that $\forall x \in D$, $\dot{V}(x) < 0$. This problem is a "hard" problem. It is a global non-linear optimization problem and we cannot expect to find an optimal solution in a finite amount of time. In fact, we cannot even prove that such a solution exists. **We need a more efficient method.**

5.2.2 Control Lyapunov Functions

Let's assume for a moment that we don't want to find a controller u, but simply to prove that one exists. Doing this requires us to prove that $\forall x \in D, \exists u \in R$ s.t. $\dot{V} < 0$, where $\dot{V} = L_f V + L_g V u$.

If we assume that u can be as large as we want (no limits on the actuators of our system), this is equivalent to proving that:

$$\exists V : D \longrightarrow R \text{ s.t } \forall x \in D, L_g V(x) = 0 \implies L_f V(x) < 0 \tag{5.1}$$

This proves that if we can't affect the system ($L_g V = 0$), then it naturally moves towards its equilibrium point ($L_f V < 0$). If we can prove it, we can, at each position x, find an optimal control input u by solving the following **quadratic** optimization problem:

$$\boxed{u = \text{argmin}(u^T R u) \text{ s.t. } L_f V(x) + L_g V(x) u < 0} \tag{5.2}$$

This problem is an easy one, it can be solved in real time, and if eq. 5.1 is satisfied, we know that a solution exists. On modern robotic systems, it can be solved several thousand times per second.

Note 1: $\text{argmin}(u^T R u)$ is used to ensure that the control input (or combination of inputs) found by the optimizer are minimized in some sense (amount of fuel/energy used for example). The R matrix, which should be positive and diagonal, allows to prioritize some inputs over others. For example, a plane has two controls when trying to change altitude: gas x and inclination α. The R matrix can prioritize the inclination α over the gas command to safe fuel, by having a higher coefficient applied to it.

Note 2: In the real world, actuators do not have infinite ranges (motors have a max torque for ex.). Accounting for that requires adding the following condition in equation 5.2 : $u < u_{max}$. This limits the possible range of inputs, but removes the guarantee that a solution exists.

5.2.3 Deriving a controller

Using the previous equations, stabilizing a system S around an equilibrium x_e is a three-steps process:

- Find a Lyapunov function $V : D \longrightarrow R$ that satisfies the criteria laid out in section 5.1.2. In the case of a physical system, the total energy of the system can be used a a Lyapunov function.
- If V satisfies equation 5.1, we know that our system is controllable everywhere around the equilibrium point, assuming no restrictions on the inputs.
- In real time, solve the optimization problem 5.2, and use the solution u as input to the system.

5.3 Feedback linearization

A second method of non-linear control consists in transforming the mathematical model of the system into a linear model by using intermediary functions, and using known linear controllers on that linear model. Mathematically, we have a system $S : \dot{x} = f(x) + g(x)u$, and we want to hide the non linear dynamics from our controller by designing an equivalent system $S_2 : \dot{y} = Ay + Bv$, where A and B are matrices.

5.3.1 Single input, single output systems

In this section, we consider SISO systems. The input command u is therefore a scalar. To transform our system into a single output system, we first define an output $y = h(x)$ where h can be any function (linear or non-linear) of the system states. (Heuristics for finding h will be discussed later by prof. Sastry)

Our first objective is to establish a relationship between y and u. To do that, we differentiate y with respect to x as many times as necessary to see u.

$$\begin{aligned} y &= h(x) \\ \dot{y} &= \frac{\partial h}{\partial x}\dot{x} = L_f h + L_g h u \end{aligned}$$

Here, we have two cases, either u affects the first derivative of y, and we are done, or it does not and $L_g h = 0$. In this case we need to differentiate again and again until we find k such that $y^{(k)}$ that depends on u. **If a system has n states, we know that k exists and that $k < n$.**

$$\begin{aligned} y &= h(x) \\ \dot{y} &= \frac{\partial h}{\partial x}\dot{x} = L_f h \\ &\dots \\ y^{(k)} &= L_f^k h + L_g L_f^{k-1} h u \end{aligned}$$

The minimal number of differentiations needed to control y is called the **relative degree** k of the system S. In general, this relative degree k **is not constant throughout the state space**. However, we will assume that it is during this lecture.

We now define z to be the following modified state vector:

$$z = \begin{bmatrix} y \\ \dot{y} \\ \dots \\ y^{(k-1)} \end{bmatrix}$$

Which yields:

$$\dot{z} = \begin{bmatrix} \dot{y} \\ \ddot{y} \\ ... \\ y^{(k)} \end{bmatrix} = \begin{bmatrix} L_f h \\ L_f^2 h \\ ... \\ L_f^k \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ ... \\ L_g L_f^{k-1} h \end{bmatrix} u$$

This is really close to a linear system. The first $k-1$ coordinates of $\dot{z}$ are just state variables. If we could find a way to make $y^{(k)}$ depend linearly from the other variables, we would have a linear system. If we set:

$$u(x) = \frac{1}{L_g L_f^{k-1} h(x)} (v - L_f^k h(x))$$

The system becomes:

$$\dot{z} = \begin{bmatrix} 0 & 1 & 0 & ... & 0 \\ 0 & 0 & 1 & ... & 0 \\ & & ... & & \\ 0 & 0 & 0 & ... & 1 \\ 0 & 0 & 0 & ... & 0 \end{bmatrix} z + \begin{bmatrix} 0 \\ 0 \\ 0 \\ ... \\ 1 \end{bmatrix} v$$

By defining u as a non-linear function of a new command input v, we now have a new system S_2 that behaves exactly like a linear system. We can use linear theory to control it.

Note 1: Contrary to linearization close to an equilibrium, we lose no information by doing feedback linearization, we just "hide" the non-linear dynamics from our controller.

Note 2: This methods is a **global** optimization method, but it requires a precise model of the system S. Otherwise, non-linear dynamics that were not well captured by the model would be impossible to predict and the controlled system could be unstable.

Note 3: If the relative degree of the system is equal to the number of states ($k = n$), then the system if fully controllable. However, if $k < n$, the linearization can't capture all the dynamics of the system, and if any of these dynamics are unstable, feedback linearization will fail. The uncaptured dynamics are called **zero dynamics**.

Note 4: We have assumed that our system was feedback controllable. However, this property strongly depends on the choice of y, and therefore h. The conditions for the existence of a good h are outside the scope of the lecture, and are not trivial.

5.3.2 MIMO systems

Feedback linearization can also be done for multiple-inputs, multiple-outputs systems, with more complicated math. In this case the input command becomes a vector $u = [u_1 u_2 ... u_p]^T$ and the chosen output y becomes a vector:

$$y = [y_1 ... y_p]^T = [h_1(x) ... hp(x)]^T$$

Note that the number of outputs must be equal to the number of inputs. The relative degree becomes a vector relative degree $k = [k_1 ... k_p]$. If the $\Sigma_1^p k_i = n$, the system will not have any zero dynamics.

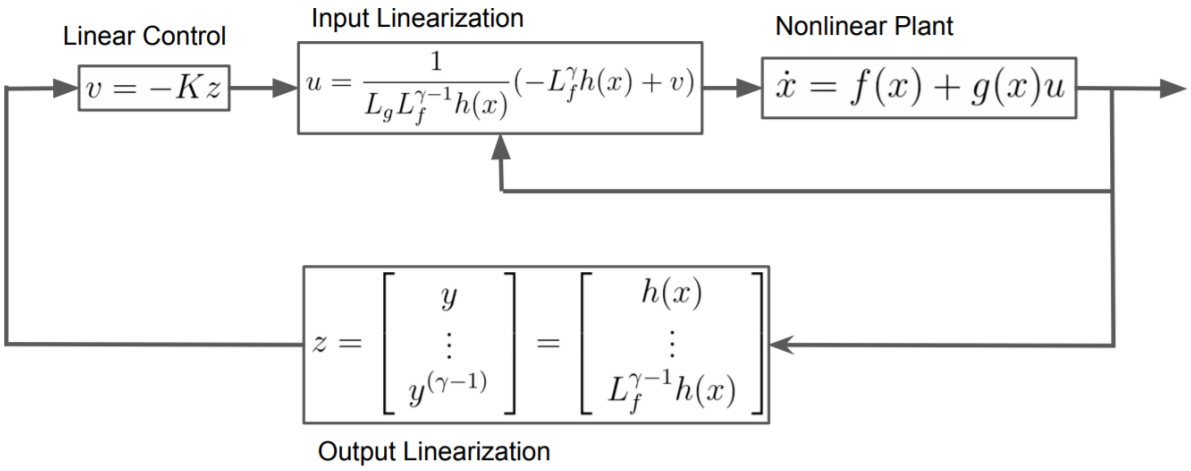

Figure 5.1: Feedback linearization, from Valmik's slides

5.3.3 Professor Sastry and Professor Bajcsy' Side Notes

Linear Feedback Blocks are used in different nonlinear systems such as tornadoes and helicopters. According to Professor Sastry's notes, for Black Hawk helicopters, researchers computed derivatives of the helicopter's dynamics at 90 different flight points in 1975-1976 in order to design a controller.

Also, Prof. Sastry made a very interesting tangent on Superbowl ads.

5.3.4 Special case of Lagrangian systems

Lagrangian systems, and **all fully actuated robots in general** have feedback linearizations that are **easy to find and work with**. In the case of open-chain manipulators, the inputs are the joint torques $\tau = [\tau_1 ... \tau_p]^T$, and the state is the set of joint angles $q = [q_1 ... q_p]^T$. The dynamics of open chain manipulators (such as Baxter) are as follows:

$$\ddot{q} = M^{-1}(\tau - C\dot{q} - G)$$
$$\tau = M\ddot{q} + C\dot{q} + G$$

We choose the outputs to be the states of the system: $y_i = q_i$. It follows that $\dot{y} = \dot{q}$. Hence:

$$\dot{y} = \dot{q}$$
$$\ddot{y} = M^{-1}(\tau - C\dot{q} - G)$$

With $\boxed{\tau = Mv + C\dot{q} + G}$, we find ourselves with an easily controllable linear system. v is called **Linear central input**.