Learning for Adaptive and Reactive Robot Control
Instructions for exercises of lecture 10

Professor: Aude Billard
Contact: aude.billard@epfl.ch

Spring Semester

1 Theoretical exercises [1h]

1.1

Book correspondence: Ex10.10
Let the nominal task model $f(x)$ be composed of conservative and non-conservative parts:

$$f(x) = f_c(x) + f_r(x)$$

Let the system $M(x)\ddot{x} + C(x,\dot{x})\dot{x} + g(x) = \tau_c + \tau_e$ be controlled by the following:

$$\tau_c = g(x) - D\dot{x} + \lambda_1 f_c(x) + \beta_R(z,s)\lambda_1 f_r(x)$$

where $z = \dot{x}^T f_r(x)$
The storage variable s has the following dynamics

$$\dot{s} = \alpha(s)\dot{x}^T D(x)\dot{x} - \beta_s(s,z)\lambda_1 z$$

and the following properties are satisfied,

$$\begin{aligned}
0 \leq \alpha(s) \leq 1 \quad & s < \bar{s} \\
\alpha(s) = 0 \quad & s > \bar{s} \\
\beta_s(z,s) = 0 \quad & s \leq 0 \ \textit{and} \ z \geq 0 \\
\beta_s(z,s) = 0 \quad & s \geq \bar{s} \ \textit{and} \ z \leq 0 \\
0 \leq \beta_s(z,s) \leq 1 \quad & \textit{elsewhere} \\
\beta_R(z,s) = \beta_s(z,s) \quad & z \geq 0 \\
\beta_R(z,s) \geq \beta_s(z,s) \quad & z < 0
\end{aligned}$$

Consider the storage function $W(x,\dot{x},s) = \frac{1}{2}\dot{x}^T \mathrm{M}\dot{x} + \lambda_1 V_{\mathbf{c}(x)} + s$, where $V_c(x)$ is the potential function associated with $\mathrm{f}_c(x)$

Prove that if $0 < s(0) \leq s$, the resulting closed loop system is passive with respect to the input-output pair τ_e, x.

1.2

Book correspondence: Ex11.1, p302

Consider the control law :

$$\tau_c = D(x)(\dot{x}_d - \dot{x}) = \lambda_1 \dot{x}_d - D(x)\dot{x} \tag{1}$$

This control law is passive for conservative DSs. However, equation 2

$$\dot{x}_d = f(x) + f_n(x) \tag{2}$$

is not a conservative DS. By using the energy tanks approach, modify equation 2 such that the system stays passive.
Hint 1: define a dynamics for the storage variable s. $\dot{s} = \alpha(s)\dot{x}^T D(x)\dot{x} - \beta_s(s,z)\lambda_1 z$
Hint 2: $\dot{M}(x) - 2C(x,\dot{x})$ is a skew symmetric matrix
As mentioned in Exercise 1.1, following properties are satisfied,

$$\begin{aligned}
0 \leq \alpha(s) \leq 1 \quad & s < \bar{s} \\
\alpha(s) = 0 \quad & s > \bar{s} \\
\beta_s(z,s) = 0 \quad & s \leq 0 \ \textit{and} \ z \geq 0 \\
\beta_s(z,s) = 0 \quad & s \geq \bar{s} \ \textit{and} \ z \leq 0 \\
0 \leq \beta_s(z,s) \leq 1 \quad & \textit{elsewhere} \\
\beta_R(z,s) = \beta_s(z,s) \quad & z \geq 0 \\
\beta_R(z,s) \geq \beta_s(z,s) \quad & z < 0
\end{aligned}$$

1.3 (Optional)

Book correspondence: Ex11.2, p302

In section 11.1.2, the presented control law 1

$$\tau_c = D(x)(\dot{x}_d - \dot{x}) = \lambda_1 \dot{x}_d - D(x)\dot{x}$$

is used to accomplish a dual-arm scenario. Modify the DS 2

$$\dot{x}_d = f(x) + f_n(x)$$

for each robotic arm, such that the robots reach the object, apply a specific amount of force on the object, and move it.

2 Programming exercises [1h]

2.1 Programming Exercise : Passive dynamical system

Book correspondence: Programming Ex10.3, p292

The aim of this exercise is to help readers to get a better understanding of the control law [equation (10.50) in the book] and how it can be implemented. Open the following MATLAB file:

```
1 lecture10-force-control/exercises/lect10_ex1.m
```

This package provides a graphical user interface (GUI) that allows you to learn a first-order, LPV-based DS and control the robot using passive interaction control such that it follows the generated motion by the DS. To achieve this, we use a feedback controller consisting solely of a damping term and a gravity cancellation term :

$$\tau_c = g(x) - D\dot{x} \qquad \text{with} \quad D(x) = Q(x)\mathbf{\Lambda}Q(x)^T$$

where $\mathbf{\Lambda}$ is a diagonal damping matrix with nonnegative eigenvalues.

Warning: The GUI records any click in the entire figure as a point in a demonstration until you have learned the DS. Be careful to stay in the workspace when recording demonstrations, otherwise you will need to start over. The steps to follow to use the GUI :

- Draw demonstrations inside the workspace by clicking and moving the mouse
- Click on 'Store Data'
- Once the data is stored, you can choose the number of Gaussians using the slider (from 2 to 10) and click on 'Learn DS'.
- Once the DS is learned, you can start the simulation where you can perturb the robot by clicking and dragging in the desired direction.

Answer the following questions:

1. Do the eigenvalues of the damping matrix matter? What would happen if the eigenvalues were very small or big? What if one of them was zero? In which of these situations can the robot reach the target while following the DS?
   Hint : You can set the eigenvalues directly in your function call in the command window: `lect10_ex1([0.1 0;0 0.1])`
2. How robust is the robot in face of perturbations if the perturbations are (1) aligned and (2) not aligned with the learned DS? You can perturb the robot, in the execution phase, by dragging it in any direction.
3. What would happen if the target were defined outside the robot-reachable space?

2.2 Programming Exercise 11.1: Motion force generation

Book correspondence: Programming Ex11.1, p299

The aim of this exercise is to help readers to get a better understanding of the control law [equation (10.50) in chapter 10] and the effect of the open parameters on the generated motion. Open the following MATLAB file:

```
lecture10-force-control/exercises/lect10_ex2.m
```

This package provides a graphical user interface (GUI). You can draw a surface and define an attractor for the DS on the surface. The system will learn the distance function Γ to the surface and generate a linear DS, given by equation (11.5) in the book, that converges to the surface, moves along the surface while applying a fixed force, and stops at the attractor.

Answer the following questions:

1. Does the shape of the surface have an effect on the generated motion/force? Would the robot reach the target in each of the following circumstances?
   (a) The surface is convex.
   (b) The surface is concave.
   (c) The surface has multiple local minima.
2. What happens if the target was not located on the surface? Can the robot reach it?
3. What happens if the target was located under the surface? Can the robot get close to it?

2.3 Programming Exercise 11.3: Learning force adaptation

Book correspondence: Programming Ex11.1.5, p299

Open the following MATLAB file:

```
lecture10-force-control/exercises/lect9_ex3.m
```

Run the file `lect9_ex3.m`. This code allows to drive a robot to generate contact forces, and learn a compensation term to adapt to poorly modeled or unknown forces at contact.

1. Modify the array `centers` (row-vector) to have the RBF functions placed randomly. What is the effect on speed of convergence?
2. Try grouping the centers closer to the trajectory. Does this improve the speed of convergence ?
   Hint: use polar coordinates to place the center points.
3. Modify the surface force `Fs` to be time varying. Under which conditions can the learning adapt to the time varying force ? You can adapt the learning rate `alpha` so that the adaptation can keep up with the time varying force.

References

[1] Aude Billard, Sina Mirrazavi, and Nadia Figueroa. *Learning for Adaptive and Reactive Robot Control: A Dynamical Systems Approach.* MIT press, 2022.