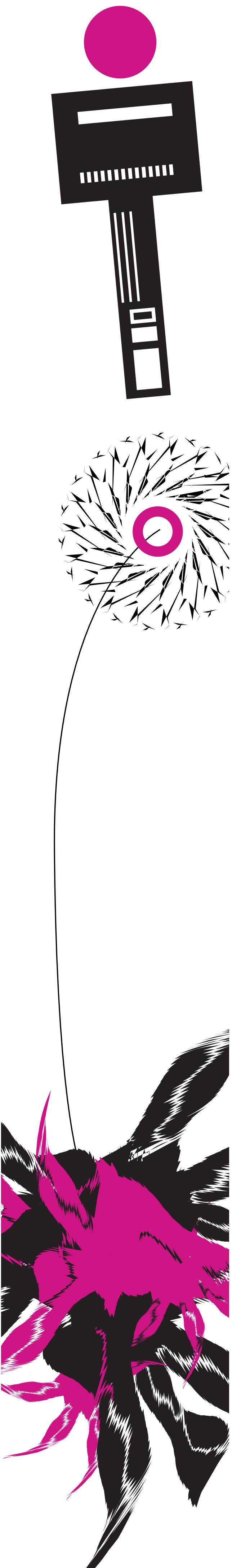


ROBOTICS
AND
MECHATRONICS

# ENERGY-AWARE CONFIGURATION AND COORDINATION OF CONTROL-SOFTWARE FOR ROBOTICS

X. (Xinwen) Zhang

MSC ASSIGNMENT

**Committee:**
dr. ir. J.F. Broenink
R. Cobos Mendez, MSc
dr. ing. G. Englebienne

September, 2021

065RaM2021
Robotics and Mechatronics
EEMathCS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

UNIVERSITY OF TWENTE. | TECHMED CENTRE
UNIVERSITY OF TWENTE. | DIGITAL SOCIETY INSTITUTE

# Summary

Configuration action gives concrete values to the provided configuration parameters, such as changing the stiffness value in impedance controller. Coordination action provides the *discrete behaviour* of a component or a system, such as changing the entire control law from PID control to impedance control. These two actions are common in robotics applications. However, they may lead to undesired effect, such as the sudden interaction energy changing, making the system unstable. By applying the *energy-based control* approach, it is possible to acknowledge the consequences in robot behaviour from an energy point of view, allowing the system to have much more information available to make a better decision.

RaM group is working towards fully enabling energy awareness in control systems. For 5Cs of control software, the energy-aware composition, communication and computation of the control software has been addressed in previous work, but few has been explored for coordination and configuration concerning energy-awareness, which is the aim of the project discussed in this thesis. This thesis presents the design pattern of energy-aware configuration and coordination, conceptual implementation of the energy-aware coordination, and the simulation has been done to test the conceptual implementation.

First, existing projects related to configuration and/or coordination are analysed to determine requirement for designing and implementing energy-aware coordination and configuration. From this analysis, it is concluded that the integration of energy awareness for configuration and coordination actions can lead to the improvement of system-level properties, but does require both *effort* and *flow*. Next, the results are used to analyse control interface requirements, which is applied to communicate the four physical quantities (*effort*, *flow*, *generalised momenta* and *generalised displacement*). Then, description of the composition of the energy in the system and the energy-communication standard for the system are presented. To demonstrate energy awareness of coordination and configuration actions, use cases are needed. However, a form of energy-aware configuration has been addressed in van Teeffelen (2018)'s work. Therefore, the demonstration only focuses on the energy-aware coordination, i.e. integrating energy awareness when changing the control law. Based on the analyses, a list of prioritised requirements is composed to develop the design pattern.

Fulfilling all design requirements, the design pattern for the energy-aware configuration and coordination are presented respectively. They both applies the control interface and energy-communication standard mentioned above, and consists of 4 components, i.e. agent, sequence controller, loop controller and energy-aware component. The agent handles the configuration and coordination actions by altering the behaviour of the other 3 components. The sequence controller sends the setpoints to the loop controller directly. The loop controller runs the control law that steers the physical system to follow the setpoint value. Last, the energy-aware component integrates energy awareness into the control software, and tries to reduce the undesired effect of configuration and coordination actions.

The energy-aware coordination design pattern is assessed by simulation. The use case, changing the control law is implemented. By conducting and evaluating 4 experiments of different conditional settings, it is shown that energy awareness is successfully integrated in the coordination action and enables decreasing the sudden energy changing if a suitable maximum energy level of energy tank is applied.

Although the work presented in this thesis has developed the design pattern for energy-aware configuration and coordination, there is work to be done in the future. Future work should focus on expanding energy assessment to cover a broader range of physical applications, providing a more appropriate upper limit of energy tank and testing the system on a physical robot.

# Contents

# 1 Introduction

## 1.1 Context

Compared with the traditional industrial robots, which are only applied in a fixed environment, advanced robots with new abilities, such as decisional autonomy and Human-Robot Interaction (HRI), will face more challenges on safety (Guiochet et al., 2017). For example, in the self-driving Tesla car, the Autopilot Assist feature is only an assist that does not work all the time and requires the driver to be ready to take over at any time. This is because the self-driving car faces two major challenges which are same for all autonomous robots: the adequate perception of the environment in spite of sensing uncertainties, and the adequate reactions to unexpected situations (Guiochet et al., 2017). However, switching the driving mode from Autopilot to Manual has caused a number of accidents. By applying the *energy-based control* approach, it is possible to acknowledge the consequences in robot behaviour from an energy point of view when switching control law, allowing the system to have much more information available to make a better decision.

Under the *energy-based control* approach, the physical interactions between the control software and the physical system – e.g., a robot – can be almost exclusively characterised by energy exchange (Folkertsma and Stramigioli, 2017) when generalised forces (efforts, $\boldsymbol{e}$) and generalised velocities (flows, $\boldsymbol{f}$) are used as steering and feedback signals (Paynter et al., 1961). Therefore, there will be a communication of energy-data between the control software and the physical system. Energy-awareness in robotics implies enabling *energy* as a system property and using it to enhance system behaviour and make better decisions to accomplish tasks effectively and safely (Brodskiy, 2014).

To prevents this work from having to spend time on developing software components that may already exist (reinventing the wheel), *Component-Based Software Development* (CBSD) is applied as was done in the work of Bezemer (2013); Brodskiy (2014); Tadele (2014); Hobert (2020), and has been analysed in detail, see Chapter 3. The control software can be divided into multiple control layers. The layered approach that is used in this thesis is depicted in Figure 1.1 , and as shown in the figure, the *loop control* layer is directly responsible for steering the physical system in the cyber domain (Embedded Control Software). Therefore, this work focuses on the *loop control* layer rather than other layers in cyber domain.

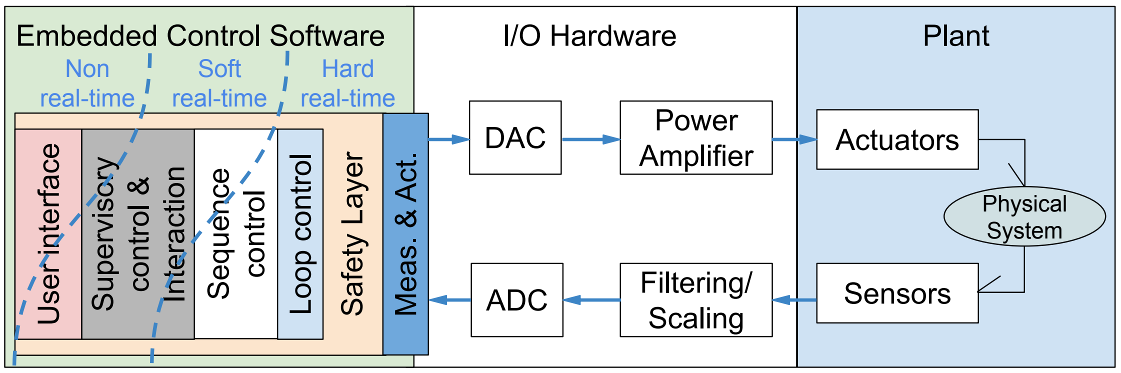


**Figure 1.1:** Detailed system architecture of a cyber-physical system (Bezemer, 2013)

An important topic in CBSD is separation of concerns, in which a system is separated on the basis of specific functional distinctions (concerns). In this thesis, the *BRICS Component Model* (BCM) has been applied, which provides a set of five concerns (5Cs), shown in Figure 1.2, for the robotics domain (Bruyninckx et al., 2013). A more recent realisation of the separation of concerns includes safety as another functional distinction to be taken into account (Bezemer, 2013). In this work, 2 concerns are addressed, i.e. configuration and coordination.

Awareness of the system's energy during configuration and coordination actions is not being taken care of yet. Although a form of energy-aware configuration has been achieved in van Teeffelen (2018)'s work, it did not emphasize the configuration behaviour, which has been done in this work, to generalize energy-aware configuration. Furthermore, energy-aware coordination in this work pioneers the use of port-Hamiltonian energy tank to solve the problem in coordination, such as bumpy transfer, which is a big jump in the plant input, caused by a significant difference in the outputs between different controllers, leading to a poor tracking performance (Peng et al., 1996).

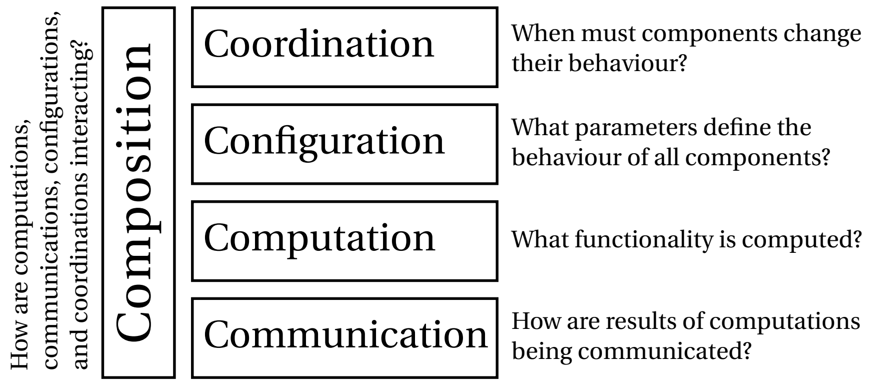

How are computations, communications, configurations, and coordinations interacting?
Composition
Coordination
When must components change their behaviour?
Configuration
What parameters define the behaviour of all components?
Computation
What functionality is computed?
Communication
How are results of computations being communicated?

**Figure 1.2:** Overview of the 5 concerns (Bruyninckx et al., 2013)

## 1.2 Problem Definition

When energy-awareness is not present, i.e. when energy flows are not taken into account, undesired effects such as instability, insufficient fault-handling actions, and degraded performance can occur (Brodskiy, 2014). Therefore, The University of Twente's Robotics and Mechatronics (RaM) research group[1] is working on integrating energy-awareness into the control software.

After analysing the existent work by means of 5Cs, it has been found that energy-aware composition, communication and computation of the control software has been addressed in the RaM group, like the work done by Franken (2011) and Brodskiy (2014), and also EGCS ITP (Energy-Guided Control Stacks) in the RobMoSys[2] project (Hobert, 2020; Cobos Mendez et al., 2020). By doing so, system-level aspects like safety monitoring, fault diagnosis/monitoring and reactive motion task planning and execution can be done based on the energetic state of the system. However, few has been explored for coordination and configuration concerning energy-awareness.

When coordinating and re-configuring the control software during run-time, the behaviour of their computation elements is modified either by changes in their life-cycle, changes in its parameters, or by altering the way its variables are computed. For instance, the instantaneous change of a control law and/or its parameters, if not done carefully, can lead the control software to become energy-inconsistent, compromising system stability and safety. Hence, energy-unaware coordination and configuration of the control software can lead to an undesired behaviour of the system with negative – maybe, catastrophic – results (Cobos Mendez et al., 2020).

[1]RaM website: https://www.ram.eemcs.utwente.nl/
[2]RobMoSys web page: https://robmosys.eu/egcs/

## 1.3 Research Motivation

### 1.3.1 Hypothesis

Since physical interactions can be almost exclusively characterised by energy exchange (Folkertsma and Stramigioli, 2017), the hypothesis is that taking energy information of interactions between the loop control level, the plant, and the environment into account during the configuration and/or coordination action, could enable the controller to assess the behaviour of energy flows produced by configuration and/or coordination. This can lead to preservation of system-level properties such as energy-consistency, stability and safety and their predictability. For example, van Teeffelen (2018) enforces the energy consistence of the system when changing the value of stiffness in impedance control, which is a kind of configuration, then the passivity of system is guaranteed so that the system become safer.

### 1.3.2 Goals and Approach

The goal of this project is to develop a generic design pattern to make the control software be aware of the energetic behaviour when executing configuration and coordination actions. And a basic implementation of the design pattern concept is to be formulated in this thesis. In other words, the main contribution of this project is building the connection between two concerns of the software (configuration and coordination) and the dynamics behaviour of the physical system.

The approach is to address the energy-flow changes derived from controller configuration and coordination. For example, by limiting the amplitude of the sudden change of energy (energy bump) by limiting the generalised forces (efforts, $\boldsymbol{e}$) and/or generalised velocities (flows, $\boldsymbol{f}$), so that an energy-bumpless transfer can be achieved, and the energy consistency can be guaranteed. By using the passivity layer in van Teeffelen (2018) for configuration, and the port-Hamiltonian energy tank in Raiola et al. (2018) for coordination, the energy consistency can be monitored and preserved. As mentioned in Section 1.1, a form of energy-aware configuration has been achieved in van Teeffelen (2018)'s work, it just did not emphasize the configuration behaviour, so in this thesis, the passivity layer in van Teeffelen (2018) is chosen for configuration. The reason why this thesis uses port-Hamiltonian energy tank in Raiola et al. (2018) for coordination is, in Raiola et al. (2018)'s works, the influence of external force disturbance to the energy tank is analysed, which can be used in this thesis, regarding the coordination action as an external disturbance.

### 1.3.3 Research Requirements

The following research requirements and sub-requirements have been devised to structure and focus the work of this thesis.

1. The thesis should be able to acknowledge the impact of run-time coordination and configuration of the loop control software in the behaviour of the robot.
   (a) Analyse the negative effects of sudden energy changes on robot dynamics.
   (b) Analyse which properties are influenced when doing run-time coordination and configuration.
2. A series of requirements should be proposed in the design pattern to preserve the system-level properties, such as energy consistency.
3. The thesis should propose the method on how to implement energy-awareness in coordination and configuration action of the loop controller.
   (a) The design choices should be able to gather the energy information, minimise the negative effects of sudden energy changes on robot dynamics.

4. The assessment of the design pattern should be done in this thesis, to analyse the benefits and drawbacks of energy-aware configuration and coordination.
   (a) Ideally, a physical real-world robot would be implemented. However, due to current restrictions of COVID-19, only simulation is implemented instead of using the physical robots at the university's labs is denied.

## 1.4 Report Outline

The rest of this thesis report is organized as follows:

- **Chapter 2** is introduced as background material, providing the existing knowledge which supports the subsequent research in this thesis.
- **Chapter 3** focuses on the principle behind energy-aware configuration and coordination. The control interface for control "command" and "feedback" signals, and the flow of energy in the system are presented. A use case is designed for evaluating energy-aware coordination.
- **Chapter 4** introduces the design pattern about energy-aware configuration and coordination, which is the main contribution of this project. Corresponding implementation example is proposed.
- **Chapter 5** simulates the implementation example in Chapter 4. Simulation results are presented and discussed here.
- **Chapter 6** summarizes the main contributions of this project. An overview of the future work and recommendations is provided.

# 2 Background

This chapter presents background information on the project's goal of enabling energy-aware coordination and configuration of the control software. First, *Component-Based Software Development* (CBSD) is addressed in Section 1.1, which comprises two parts, i.e. the layered approach in control software, and separation of concerns between the development aspects of Computation, Communication, Coordination, Configuration and Composition (5Cs). Following CBSD, the concepts related to energy exchange during physical interactions are explained. Then, the effect of configuration and coordination in energy level are described. Finally, port-Hamiltonian energy tank has been treated for energy-aware coordination.

## 2.1 CBSD

Applying *Component-Based Software Development* (CBSD) seems to be a clever approach on working through the endless list of potential application development for robotics. Because in CBSD, systems are composed from reusable components, helping engineers concentrate on the prime contribution of their work. Ideally, a CBSD component should be compliant to all possible systems. However, Brugali and Scandurra (2009) mentioned that there is a limit to the extend of compliance of components due to differences in interfaces, e.g. specialistic systems using a proprietary interface or existing systems using an outdated configuration. Therefore, CBSD is about finding the best trade-off between commercial off-the-shelf (COTS), which is too specific (less reusable), and custom development, which is too generic (less valuable)(Brugali and Scandurra, 2009).

In this work, layered approach which separates a control system in layers, as well as 5Cs which separates the system based on specific functional distinctions (concerns) are applied. Separating the system in the development of control software adds to the re-usability of a system and allows work on the system to be focused at one specific system functionality at a time.

### 2.1.1 Layered approach

As shown in Figure 1.1, the complete control system is divided into *(Embedded) Control Software*, *Input/Output Hardware* as well as *physical/mechanical plant (robot)*. Even though this thesis focuses on the loop control layer (part of software), it is still important to keep in mind the full control system including *I/O Hardware* (consists of digital-to-analogue (D/A) and analogue-to-digital (A/D) converters, connecting the discrete-time control stack to the continuous-time plant), as well as the *Plant*. Because this thesis is working on integrating energy awareness into control software, which concerns about the interaction energy between software and hardware.

This section only introduces the concept of two layers in *(Embedded) Control Software* since this thesis is focusing on the development of *loop control layer*, and the *sequence control layer* is needed to send the setpoints to the *loop control layer* directly. For simplicity, the *(Embedded) Control Software* section in Figure 1.1 is taken out separately as Figure 2.1.

A description of *sequence control layer* and *loop control layer* is given by Cobos Mendez et al. (2020) as follows[1]:

- **Sequence Control** is a mid-level controller that typically deals with longer running tasks than loop control layer, e.g. trajectory computation. For example, if the mission is steer-

[1]In fact, Figure 1.1 has been used within the RaM group for years. An early version can be found in the work by Groothuis et al. (2009), after which an upgraded version was presented by Broenink et al. (2010). Based on that, the current version in Figure 1.1 by Bezemer (2013) contains an update on how the real-time guarantees are distributed over the different control layers. And in Cobos Mendez et al. (2020) the definition of each layers are declared

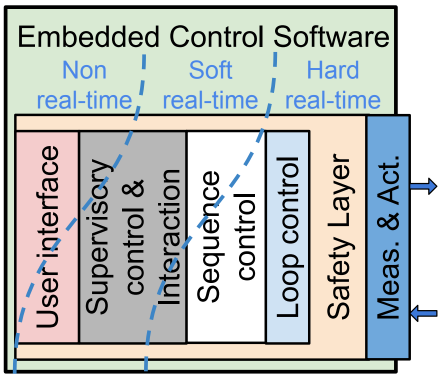


**Figure 2.1:** A general software-architecture representation for layered embedded control systems (Bezemer, 2013)

ing the robotic arm to reach the target state, the sequence control layer needs to generate a serial of setpoints to constitute a feasible trajectory for the robotic arm, just as shown in Figure 2.2, and let loop controller follow the setpoints. Besides setpoints, commands can also be sent to loop control in the form of changing the control parameters, like the PID value etc.. In turn, it receives feedback from loop control layer (or from sensor,e.g. from cameras, lasers, IMUs, GPSes, etc. directly), like whether the robotics arm reach the target state. Sequence control components interacting with loop control components require timing constraint. However if missed deadlines do not jeopardise task execution, a certain degree of tolerance can be implemented, i.e. soft real-time guarantees. For example, when the robotic arm cannot follow the setpoints in time, making the error between current state and setpoint state become bigger and bigger, the sequence control layer can stop generating new setpoint until the setpoint is actually reached.

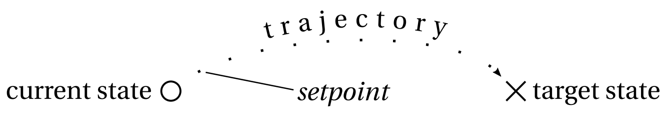


**Figure 2.2:** Setpoints along a trajectory to steer the Loop Controller from its current state towards a target state (Hobert, 2020)

- **Loop Control** is the main operating layer of this thesis because it is responsible for steering the physical system in its environment directly, making the effect of energy awareness easier to observe. The loop-control layer receives setpoint commands from the sequence control layer and sensor measurement feedback from the measuring & actuation layer (the details of this layer can be found in Cobos Mendez et al. (2020)). It computes actuation commands to steer the robot's actuators based on these setpoints and measurements. In turn, it feeds the robot state back to sequence control layer. Usually the robotic control laws are being used in this layer, such as the most classic PID control, impedance control etc.. Loop control layer has to compute with strict timing constraints as it interact directly with the physical world. Otherwise, it will lead to unstable behaviour and cause unacceptable — and sometimes catastrophic — failures.

### 2.1.2 5Cs

Figure 1.2 gives an overview of how the "5Cs" are defined in the context of the BRICS Component Model (BCM). A more detailed vision of the "5Cs" declared by Bruyninckx et al. (2013) is given below:

- **Computation** is the core of a system's functionality, the main role is calculating the *steering* and *measurement* values to control the physical world through a *control law algorithm.* For example, in classic PID control, the calculation of using tracking error $e$ and its three gains $K_P$, $K_I$, $K_D$ to get control action $u$ is part of *computation*:

$$u(t) = K_P e(t) + K_I \int e(t)\mathrm{d}t + K_D \frac{\mathrm{d}e(t)}{\mathrm{d}t}$$

- **Communication** sends data towards the computational components that require it, with the right quality of service, i.e., time, bandwidth, latency, accuracy, priority, etc. For example, the sequence control layer sends the setpoints to the loop control layer mentioned in Section 2.1.1.
- **Coordination** is the main topic in this thesis, which determines how all components in the system should work together, i.e, in each state of the coordinating *finite state machine*, one particular behaviour is configured to be active in each of the components. In other words, Coordination provides the *discrete behaviour* of a component or a system. This thesis focus on one kind of discrete behaviour, i.e. changing the control law in loop control layer. For example, Peng et al. (1996) used the control scheme with the capability of switching between manual and PID control mode, and this example has been detailed in Section 2.3.2.
- **Configuration** is also a main topic in this thesis, which allows users of the *computation* and *communication* functionalities to influence the behaviour and performance, by giving concrete values to the provided configuration parameters, e.g.tuning control or estimation gains, determining *communication* channels and their inter-component interaction policies, providing hardware and software resources and taking care of their appropriate allocation, etc. For example, in classic PID control, in order to speed up the controller's response to errors, $K_P$ needs to be increased, which is a kind of configuration. In this thesis, the energetic effect of tuning the stiffness value in impedance control is being addressed.
- **Composition** models the *coupling* that is always required between components in a particular system.

As mentioned in Section 1.2, among all the five concerns, energy-aware composition, communication and computation of the control software has been addressed in the RaM group. However, energy-aware coordination and configuration is still missing, and has been addressed in this thesis.

## 2.2 Principles of Energy Exchange

This section explains the sentence in Section 1.1 "the physical interactions between the control software and the physical system – e.g., a robot – can be almost exclusively characterised by energy exchange" (Folkertsma and Stramigioli, 2017) and relates to the development of an energy-aware configuration and coordination.

### 2.2.1 Physical quantities

Robotic systems use sensors to measure physical quantity. Depending on the type of sensor, quantities such as those listed in Table 2.1 can be measured. In fact, the physical quantities in

| **physical domain** | **flow** $f$ | **effort** $e$ | **generalised displacement** $q = \int f \, \mathrm{d}t$ | **generalised momentum** $p = \int e \, \mathrm{d}t$ |
|---|---|---|---|---|
| mechanical translation | velocity $v[ms^{-1}]$ | force $F[N]$ | displacement $x[m]$ | momentum $p[N \cdot s]$ |
| mechanical rotation | angular velocity $\omega[rad \cdot s^{-1}]$ | torque $\tau[Nm]$ | angular displacement $\theta[rad]$ | angular momentum $b[Nms]$ |

**Table 2.1:** Physical quantities classified under four physical variables: flow, effort, generalised displacement, and generalised momentum (Breedveld, 1982)

not only the physical domains mentioned in Table 2.1, but also other domains, such as electro-magnetic (includes current $i[A]$, voltage $u[V]$ etc.), hydraulic (includes volume flow $\phi[m^3 s^{-1}]$, pressure $p[Nm^{-2}]$ etc.) and so on can also be measured and classified under four physical variable. However, this thesis is focusing on the mechanical translation and rotation, so only these two domains are listed.

In Table 2.1, the quantities are classified as a generalised velocity or flow ($f$), generalised force or effort ($e$), generalised displacement ($q$), or generalised momentum ($p$). These four physical variables are called the variables of state and can be used to describe the energetic condition of any physical state-determined system (Paynter et al., 1961).

In multi-DOF robotics, these four variables of state are usually in vector form. Consider a 7-DOF robot with a displacement sensor at each joint, the generalised-displacement vector equals $\boldsymbol{q} = [q_1, q_2, q_3, q_4, q_5, q_6, q_7]^{\mathrm{T}}$. In the remainder of this thesis, the variables of state are considered for multi-DOF robotics, thus in vector form ($\boldsymbol{f}, \boldsymbol{e}, \boldsymbol{q}, \boldsymbol{p}$).

### 2.2.2 Basic energy computation

For modelling and control of physical systems, both the *energy* ($E$) and its rate of change are of interest. The *rate of energy change* ($\frac{\mathrm{d}}{\mathrm{d}t}E$) — also known as *power* ($P$) — is defined as the product of two power-conjugated variables: *effort* ($\boldsymbol{e}$) and *flow* ($\boldsymbol{f}$):

$$\frac{\mathrm{d}}{\mathrm{d}t}E = P = \boldsymbol{e}^T \cdot \boldsymbol{f} \tag{2.1}$$

Thus, in continuous time, *energy* ($E$) is equal to the integration of *power* ($P$) over time:

$$E = \int P \mathrm{d}t = \int e^T f \mathrm{d}t \tag{2.2}$$

But the control software interfaces with a physical system through *sampling* and *holding* actions, making the energy computation in discrete time. For an effort-out/flow-in causality, i.e. input the velocities (*flow* $\boldsymbol{f}$), but more commonly input and also used in this thesis is displacement ($\boldsymbol{q}$), while output forces or torques (*effort* $\boldsymbol{e}$), Stramigioli et al. (2005) proposes computing the *energy flow* ($\Delta H$) with sampling *displacement* $\boldsymbol{q}$ and constant *effort* $\boldsymbol{e}$ due to the zero-order hold. Between time $kT$ and $k(T+1)$, where $T$ is the sampling time and $k$ is a positive integer, the *effort* $\boldsymbol{e}$ will be constant:

$$\boldsymbol{e}(t) = \boldsymbol{e}(k+1) \quad t \in [kT, (k+1)T] \tag{2.3}$$

and the computation of *energy flow* $\Delta H$ as shown in Equation 2.4:

$$\begin{aligned}\Delta H(k+1) &= \int_{kT}^{(k+1)T} \boldsymbol{e}^T(t)\boldsymbol{f}(t)\mathrm{d}t \\ &= \int_{kT}^{(k+1)T} \boldsymbol{e}^T(k+1)\boldsymbol{f}(t)\mathrm{d}t \\ &= \boldsymbol{e}^T(k+1)\int_{kT}^{(k+1)T} \boldsymbol{f}(t)\mathrm{d}t \\ &= \boldsymbol{e}^T(k+1)(\boldsymbol{q}((k+1)T) - \boldsymbol{q}(kT))\end{aligned} \tag{2.4}$$

the last step of the Equation 2.4 uses the relation between *flow* ($\boldsymbol{f}$) and *generalised displacement* ($\boldsymbol{q}$) mentioned in Table 2.1. And throughout the sample interval $[kT, (k+1)T]$, the value of the $\boldsymbol{f}$ is continuously changing, here only uses the values at both ends to estimate the total change of $\boldsymbol{f}$. If the sampling time $T$ is small enough, the error will be acceptable.

## 2.3 Energy-tank based controller

Passive systems are a class of stable dynamical systems first defined by Willems (1972), whose total energy is less than, or equal to, the sum of its initial energy and any external energy supplied to it by interaction. However, the configuration and coordination actions can allow internal energy production, resulting in the loss of passivity of the overall system, which has been detailed in Section 2.3.1 and Section 2.3.2, which is the first requirement in Section 1.3.3, i.e. "the thesis should be able to acknowledge the impact of run-time coordination and configuration of the loop control software in the behaviour of the robot".

Thereupon, the energy-tank based controller implementation presented in Raiola et al. (2018) is adopted as a feasible way to circumvent this issue. In this way, only the energy that is put in by the operator or the environment can be used in controlling the robotic devices. If more energy is consumed than what was stored in the energy tank, the control action will be limited such that the energy leakage will not exceed the energy level in energy tank, creating a passive system. In general, energy-tanks can be added to any task oriented controller, let the control laws (PID, impedance control etc.) become black boxes in the eyes of the designer, free the designer from the task of designing the control law.

### 2.3.1 Impact of configuration on energy levels

The configuration of the control law changes the energetic interaction of the robot with the environment. For instance, in impedance control, as shown in Figure 2.3, if the effect of stiffness modulation on the monitoring of energy levels in the system does not be taken into account, when the stiffness increases (the left one in the figure), more energy will be extracted from the tank level than the amount that is added, leading the impedance controller continuously leaking energy because of the overestimation of active behaviour. On the other hand, when the stiffness decreases (the right one in the figure), less energy will be extracted than the amount that is added, i.e. more energy than needed may be supplied to the physical system. van Teeffelen (2018) clarifies that "the reason why these happen is that the energy tanks are not registering the energy flows in and out of the controller in the right way, since the effects of stiffness modulation are not taken into account".

van Teeffelen (2018) also describes how to add energy compensation. For instance, when decreasing the stiffness of the impedance controller as shown in Figure 2.4. The stiffness is changing from $K_1$ to $K_2$, and the relationships between the positional error $X_e$ and the applied control actions (forces) corresponding to the two stiffness levels, $F_{K_1}$ and $F_{K_2}$ are plotted. When the controller uses $K_1$, the energy contained in it is equal to the surface area of the combined white and grey triangles:

$$H_{K_1} = \frac{1}{2}F_{K_1}X_e = \frac{1}{2}K_1X_e^2 \tag{2.5}$$

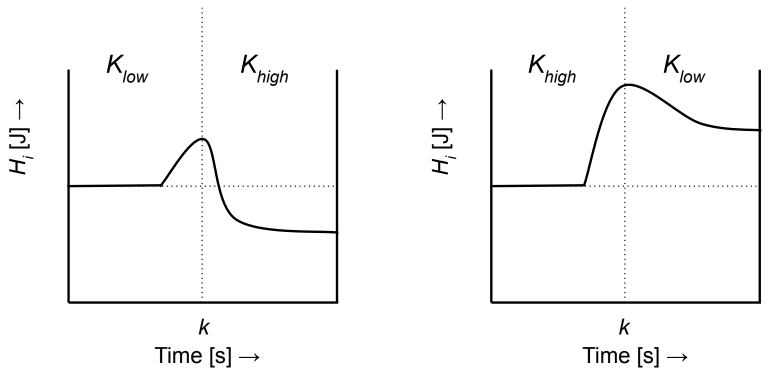


**Figure 2.3:** Energy tank levels of impedance controller in time for switching virtual stiffness levels at time instant $k$ without using energy compensation for impedance modulation. The virtual spring is compressed until $k$, after which it returns to its relaxed state. The left sub-figure increases the stiffness at $k$, while the right sub-figure decreases the stiffness at $k$.(van Teeffelen, 2018)

When decreasing the stiffness to $K_2$, the grey area ($\Delta H_{K_v}$, energy compensation) is added to the energy level of the controller (in this case, $\Delta H_{K_v} < 0$, so the energy becomes lower actually). The surface area of the grey triangle is equal to:

$$\Delta H_{K_v} = \frac{1}{2}(K_2 - K_1)X_e^2 < 0 \tag{2.6}$$

The new energy level of the controller will be equal to the white area:

$$H_{K_2} = H_{K_1} + \Delta H_{K_v} \tag{2.7}$$

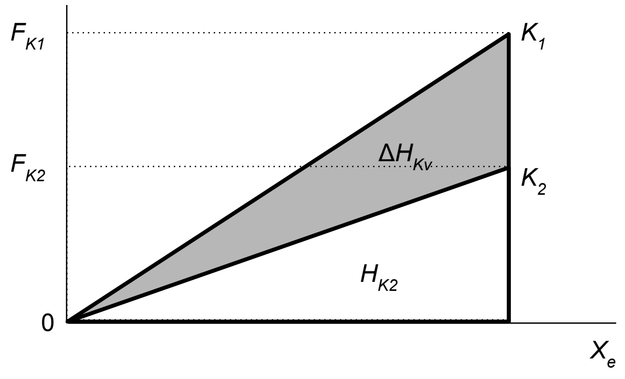


**Figure 2.4:** Energy levels of two impedance controllers with stiffness levels $K_1$ and $K_2$ for a constant positional error $X_e$ resulting in controller efforts $F_{K_1}$ and $F_{K_2}$ respectively (van Teeffelen, 2018).

In this thesis, a generic design pattern has been designed for energy-aware configuration based on van Teeffelen (2018)'s work.

### 2.3.2 Impact of coordination on energy levels

As for coordination action, bumpy transfer, i.e. the mode switching with a jump at the plant input, has been addressed in Peng et al. (1996). They used the control scheme with the capability of switching between manual and PID control mode as shown in Figure 2.5 as an example.

When the switch goes from automatic to manual control, if at that time, the error $e > 0$, then the integral term in automatic mode (PID control) increases in an uncontrolled way to very high

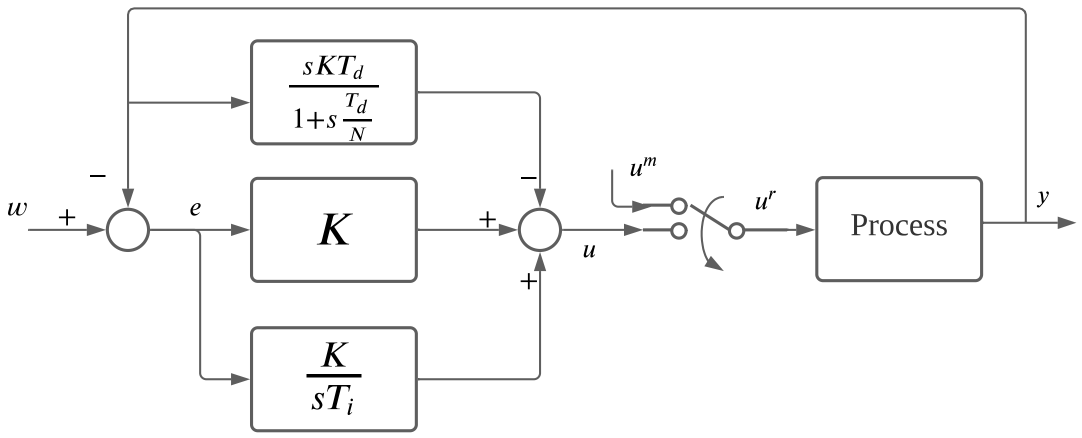


**Figure 2.5:** The control scheme with the capability of switching between manual and PID control mode (Peng et al., 1996)

values and $u$ becomes high and much greater than $u^m$, because the value $e$ will not change when the mode switch to manual control. Now, assume that the switch goes back from manual to automatic control. At that moment, even if $e = 0$, a big jump occurs at $u^r$, due to the previous high values of the integral term. Moreover, $u$ decreases only if $e < 0$ for a sufficiently long time. This leads to a long settling time of the process output. Because of the jump of $u^r$ (e.g. forces, a kind of *effort* $\boldsymbol{e}$), the energy flow made up of $u^r$ also has a big jump and a long settling time.

### 2.3.3 Energy-tank system described through the port-Hamiltonian formulation

Different methodologies are being used to the define the energy tank strategy. In this thesis, the port-Hamiltonian systems theory is chosen to define energy tank as Cardenas (2017) and Raiola et al. (2018) did. "With energy tanks defined through a port-Hamiltonian system, structurally prevents that any action taken would result in spurious energy generation"(Raiola et al., 2018).

The physical representation of the energy-tank based controller for a Multi-DOF system is depicted in Figure 2.6 by Cardenas (2017) and Raiola et al. (2018). The energy tank $H_n(s_n)$ for each joints is modeled as a spring with constant stiffness $k$ (e.g. $k = 1$) connected to the plant through a transmission $MT_n$. This transmission allows power to flow from the controller to the plant, regulated by the ratio $u_n$, whose value is determined by a computational unit $CU$.

The derivation of port-Hamiltonian equation of the energy-tank system for multi-DOF manipulators is not explained in detail here, detailed contents can be found in Cardenas (2017). For multi-DOF system, the port-Hamiltonian equation of the energy-tank system for a joint subsystem $n$ is disclosed as:

$$\begin{pmatrix} \dot{s}_n \\ \tau_{n_{out}} \end{pmatrix} = \begin{pmatrix} 0 & u_n \\ -u_n & 0 \end{pmatrix} \begin{pmatrix} s_n \\ \dot{q}_n \end{pmatrix} \tag{2.8}$$

where $\tau_{n_{out}}$ is the output from the chosen control strategy (e.g. the torque output from impedance controller), $\dot{q}_n$ is the velocity of the joint, and $s_n$ is the state of the spring (energy tank). Then the desired transmission ratio $u_n$ can be set as:

$$u_n = \frac{\tau_{n_{out}}}{s_n} \tag{2.9}$$

If in 1-DOF case, when the tank is empty, the controller's power output instantly drops to zero to guarantee passivity, and if the power "flows back" from the plant to the controller, the energy-tank can be refilled with the injected energy. However, in multi-DOF case, the transmission

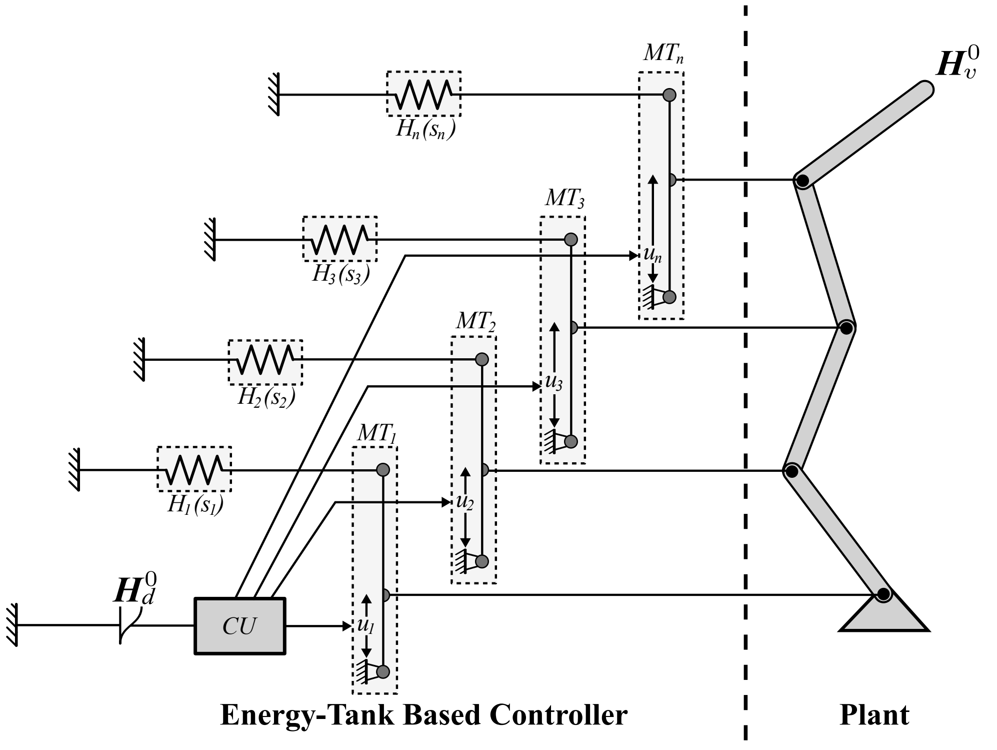


**Figure 2.6:** Multi-DOF energy-tank based controller (Cardenas, 2017)

ratio $u_n$ will not be zero immediately when the energy-tank $H_n(s_n)$ ($H_n(s_n) = \frac{1}{2}ks_n^2$ is the potential energy in the tank) enters to a depleted state, but adopts a smooth decaying behavior, preserving a control action that gradually shuts down joint $n$. Moreover, recharging energy into the tank when there is a power flowing from the plant to the controller is prohibited, due to the presence of noise in the computation of the power flowing between the controller and plant (other links may still be in motion when the tank $H_n(s_n)$ is depleted, and joint $n$ extends its motion due to the inertial motion of the other links, which may unintentionally recharge the tank in case).

Therefore, the transmission ratio $u_n$ is determined as:

$$u_n = \begin{cases} \frac{-\tau_{n_{out}}}{s_n}, & \text{if}\,(H_n\,(s_n) > \epsilon) \\ \frac{-\tau_{n_{out}}}{\gamma^2}s_n, & \text{otherwise} \end{cases} \tag{2.10}$$

where $\epsilon$ is the minimum amount of energy in the tank before the robot and the controller are decoupled, $\gamma = \sqrt{2\cdot\epsilon}$. When $H_n(s_n) \leq \epsilon$, a linear function behavior is chosen, let $\tau_{n_{out}}$ drops proportionally to the state $s_n$. Once the transmission ratio $u_n$ is settled according to the energy levels in $H_n(s_n)$, the torque output sent to joint $n$ is computed as:

$$\tau_{o_n} = -u_n \cdot s_n \tag{2.11}$$

the state of the spring $s_n$ in $H_n(s_n)$ is computed at each time step by integrating $\dot{s}_n$ from the port-Hamiltonian expression in Equation 2.8:

$$\dot{s}_n = u_n\dot{q}_n \tag{2.12}$$

# 3 Analysis

This chapter focuses on the second requirement in Section 1.3.3, i.e. "a series of requirements should be proposed in the design pattern to preserve the system-level properties, such as energy consistency". A list of five projects related to configuration and/or coordination is analysed for the connection points for energy awareness in coordination and configuration. The results thereof are used to analyse interface requirements. Furthermore, a possible use case for evaluating energy awareness of coordination is analysed to identify implementation requirements (the use case for evaluating energy-aware configuration has been analysed by van Teeffelen (2018), so it won't be addressed in this thesis). This chapter is concluded by a list of design requirements and a list of implementation requirements.

## 3.1 Connection points for energy awareness in coordination and configuration

In order to determine requirements for designing and implementing energy-aware coordination and configuration, additional information is needed to verify what are the practical applications of coordination and configuration. Therefore, five projects that consider the configuration and/or coordination are analysed below.

Furthermore, because Section 1.1 mentions that " energy-awareness in robotics implies enabling energy as a system property and using it to enhance system behaviour and make better decisions to accomplish tasks effectively and safely (Brodskiy, 2014)", system-level properties of each project are evaluated to determine how energy awareness could have been beneficial in those projects.

The analyses per project are structured as follows:

- **Project name.** [Purpose of the system].
  - **Measuring.** [The sensors / measurement signals that were used.]
  - **Actuation.** [The actuators / control-command signals that were used.]
  - **Notes.** [Relevant notes specific to the project, e.g. the practical applications of coordination and/or configuration in this project, the method of getting the energy information.]
  - **System-Level properties.** [The system-Level properties improved by implementing energy-aware coordination and/or configuration.]

To make a design pattern that is widely applicable, and allow the task-oriented control system to become black boxes for the engineers, the control systems of the projects have been selected to differ in physical domains, type of actuation, and control laws. No more than the following five projects have been selected, because they are expected to yield sufficient information.

The analyses of the five projects (in chronological order) are listed below:

1. **MACS.** Breemen and Vries (2000) develop an agent-based framework for designing and implementing multi-controller systems, i.e. multi-agent control systems (MACS). In the article, the problem of regulating the water level of the water vessel, is applied as an example.
   (a) **Measuring.** The water level, pump current which steers the pump are measured.
   (b) **Actuation.** Pump.
   (c) **Notes.** When changing from one controller-agent to another, the control algorithm is also changing. For instance, in the example, preventing water overflow situations and normal operation use different controllers, the former one gets active and

turns off the pump when the water level becomes higher than or equal to the vessel height, while the latter one has to operate during normal operation of the water vessel problem. Thus, it is coordination of control software.
The physical system energy can be calculated using the following formula:

$$P = \eta \rho g h Q \tag{3.1}$$

in which, $P$ is the power output; $\eta$ is the efficiency of the turbine; $\rho$ is the density of water; $g$ is the standard acceleration due to gravity; $h$ is the usable fall height; $Q$ is the flow rate, calculated as

$$Q = Av \tag{3.2}$$

in which, $A$ is the cross-sectional area of the channel; $v$ is the flow of water. If using *effort e* and *flow f* to express power output, the equation is:

$$\begin{aligned} P &= \eta A \cdot \rho g h \cdot v \\ e &= \text{pressure} = \rho g h \\ f &= \text{flow} = v \\ P &= \eta A \cdot e \cdot f \end{aligned} \tag{3.3}$$

(d) **System-Level properties.**

- **Energy-consistency and Safety.** The authors mention that one of the problem when using the multiple model approach is bumpless transfer. Switching from one controller module to another can result in a discontinuity of the control signal. By using energy-aware coordination, the energy consistency can be guaranteed. Moreover, the safety risk such as a sudden increase of energy consumption of the pump resulting the increase of water flow can be avoided.

2. **Human-friendly manipulators.** Tadele (2014) identifies suitable metrics to identify the appropriate impedance of robotic manipulators for the two conflicting requirements: motion performance and safety. And the robot “Bobbie-UT" is built as an interconnection of interchangeable mobile platform, torso, robotic manipulator and humanoid head components.

(a) **Measuring.** Each joint of the robotic manipulator is equipped with an absolute joint position sensor, encoder reading for the actuator position and a torque sensor.

(b) **Actuation.** The robot Bobbie-UT has 7 Degrees of Freedom (DOFs) manipulator with a two fingered, compliant and under-actuated gripper.

(c) **Notes.** The author mentions that effective motion based manipulation requires a highly stiff behavior (relatively high impedance values) while important safety requirements are achieved with compliant behaviors (relatively low impedance values). So modifying the impedance value, which is configuration, is important to satisfy both performance and safety requirements.

(d) **System-Level properties.**

- **Safety and Stablity.** The author mentions that energy and power-based metrics are more suitable for human-friendly manipulators, because they are intuitive as well as physically meaningful and can easily fit into the passivity-based impedance controller design. They can handle both clamped as well as unclamped impacts against a human user while the widely known HIC (Head Injury Criteria) is hard to deal with low speed collision injuries. Therefore, using the energy-aware configuration can be beneficial for the safety of human-friendly manipulators.

Besides, the author uses an energy-tank based controller-design approach to implement a variable-impedance controller, which avoids producing the internal energy when varying the stiffness behavior of the virtual controller spring, and maintains the overall passivity of the system, which is beneficial to the stability.

3. **Teleoperated robotic systems.** van Teeffelen (2018) uses an impedance controller to connects a haptic device (Force Dimension omega.7) on the operator's side to a robotic manipulator (KUKA LWR4+) located in the remote environment, implementing a haptic control of a robotic teleoperation system.

   (a) **Measuring.** The haptic device (Force Dimension omega.7) registers the operator's motions and applies force feedback. The Myo Gesture Control Armband receives the Electromyography (EMG) data from human superficial muscles. Angular positions, speed and torque data of the seven joints of the robotic manipulator (KUKA LWR4+) is available.

   (b) **Actuation.** The haptic device (Force Dimension omega.7) has 7 degrees of freedom (DOFs), which can apply forces to the operator. The robotic manipulator (KUKA LWR4+) has 7 motorized joints.

   (c) **Notes.** The author uses variable impedance, allowing the operator to modulate the impedance values, which is configuration. As safety and stability are desired, relatively low impedance values should be applied, which reduces contact forces between the robot and its environment and also reduce active behaviour of the impedance controller. However, when high levels of transparency are desired, for instance during accurate positioning of the robot, relatively high impedance values should be used.

   The author uses force and displacement data to get the energy information. The effects of control stiffness modulation as well as saturating control actions on energy levels are also analysed, and the solutions to these problems are proposed.

   (d) **System-Level properties.**

   - **Safety.** The author mentions that in haptic control, the delays in the communication channel or the digital sampling in the control architecture could lead to active behavior (i.e. production of energy), making the system loss the passivity. Therefore, the passivity layer is adopted by the author to guarantee the stable behaviour of the impedance controller, making the system safer.
   - **Energy consistency and passivity.** The author mentions that high levels of transparency are often desired because of better motions tracking and force application result, and the haptic 'image' of the remote environment that is presented to the operator will be clearer. By using energy-aware configuration, the passivity of impedance controller is guaranteed, so the impedance value could be as high as possible to satisfy the transparency.

4. **Reconfigurable navigation system for service robots.** Brugali et al. (2018) proposes a methodology for modeling the variability and the associated Quality-of-Service (QoS) characteristics of reconfigurable software systems.

   (a) **Measuring.** In the example, when the task is urgent delivery, the robot chooses marker-based navigation, a monocular camera is used to detect visual landmarks and obstacles, and if the illumination becomes suddenly inadequate, the laser scan will be activated. When the task is regular delivery, the robot chooses map-based navigation, uses a stereo camera for map-localization.

(b) **Actuation.** The motor for driving the wheels.

(c) **Notes.** Brugali (2020) mentions that switching between two sensors requires to replace the algorithm that recognizes obstacles from sensory data. This is coordination. Moreover, the different performance of the two sensors (e.g. the scan rate) and of the corresponding obstacle detection algorithms (e.g. the response time) requires to adapt some parameters of the navigation control system (e.g. the maximum robot speed). This is configuration.

The combination of the velocity and force data could be used to get the power data, which can be integrated to estimate energy exchange.

(d) **System-Level properties.**

- **Robustness (stability) and Safety.** Brugali (2020) mentions that for autonomous robots operating in everyday environments, such as hospitals, private houses, and public roads, one of the most important challenges is to guarantee robustness to changing operational conditions. Using energy-awareness, the total amount of energy will be limited, guaranteeing the passivity. And the bumpless transfer can be achieved, so that the dangerous behavior such as the sudden change in speed will not happen.

5. **Fully-Actuated Hexarotor.** Rashad et al. (2019) present an observer-based wrench/impedance controller for a fully-actuated hexarotor.

(a) **Measuring.** By using wrench observers, the authors achieve the interaction without external force/torque sensing.

(b) **Actuation.** The fully-actuate hexarotor has six canted rotors, six propellers are activated by electric motors on the revolute joints.

(c) **Notes.** The Port-Hamiltonian control get the kinetic energy as well as the potential energy information. And the value of total wrench applied to the rigid body of Unmanned Aerial Vehicle (UAV) is modified by impedance control. The behavior of changing the value of parameter, is configuration.

(d) **System-Level properties.**

- **Stability and Safety.** The authors use the energy tanks to guarantee the system's overall the contact stability of the aerial robot interacting with any conceivable passive environment. On the other hand, energy tanks allow the energy flow within the system to be observed, a high-level strategy can be designed to analyze whether it is safe or not to allocate more energy in the energy tank (modify the energy budget) to allow further regulation of the interaction wrench.

In line with Section 2.2, the systems in the examples above typically output a generalised force—i.e. *effort*—command based on a generalised displacement (or sometimes generalised velocity—i.e *flow*) input feedback. Except the first example, which inputs the water level (constitutes pressure-i.e. *effort*) while outputs the pump current—i.e *flow*. No matter what kind of causality (effort-out/flow-in or flow-out/effort-in) is being used in the system, both *effort* and *flow* signals are indispensable to get the energy information.

## 3.2 Control interface

Following *Component-Based Software Development* (CBSD, Section 2.1), data communication within the energy-aware configuration and coordination should be generic and standardised to allow for component-based design. And the work is focusing on the loop control layer as suggested in Section 1.1. Therefore, the generic interfaces connecting the loop control layer, energy tank and the physical plant should be explored.

Typically, at least 1 "command" (e.g. actuation torques) and at least 1 "feedback" (e.g. measured velocities) signal are communicated between the control software and physical plant can be used to integrate energy awareness. Notice that "command" and "feedback" are generic terms that include any sort of transmitted control signal. And both effort-out/flow-in and flow-out/effort-in causalities can be used. To be precise, not only *effort* ($\boldsymbol{e}$) and *flow* ($\boldsymbol{f}$), but also *generalised momenta* ($\boldsymbol{p}$) and *generalised displacement* ($\boldsymbol{q}$) can also be used in the causalities, since energy exchange can be calculated with either $\boldsymbol{e}$ or $\boldsymbol{p}$ together with either $\boldsymbol{f}$ or $\boldsymbol{q}$, and are belonging to the same coordinate frame (can refer to Table 2.1) and same sample time.

This thesis reuses the control interface proposed by Hobert (2020) and makes some changes on that, is illustrated in Figure 3.1. In which, any two entities interfacing with each other share an input/output connector over which a "command" or "feedback" signal can be sent. And the 8 possible combinations of physical quantities ($\boldsymbol{f}, \boldsymbol{e}, \boldsymbol{q}, \boldsymbol{p}$) corresponding to "command" and "feedback", with which energy exchange can be calculated, are listed in Table 3.1.

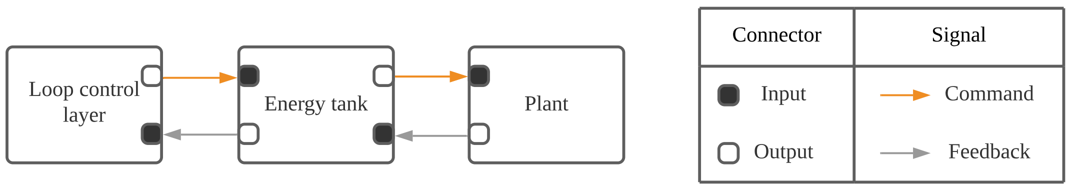


**Figure 3.1:** Control interface consists of connector, signal and 3 entities (loop control layer, energy tank and plant). There should be a sequence control layer before the loop control layer, and Measuring & Actuation layer before plant, but due to the main contribution of this thesis is about loop control layer, other layers in the Embedded Control Software are ignored.

| Command | Feedback |
|---|---|
| $\boldsymbol{e}$ | $\boldsymbol{f}$ |
| $\boldsymbol{e}$ | $\boldsymbol{q}$ |
| $\boldsymbol{p}$ | $\boldsymbol{f}$ |
| $\boldsymbol{p}$ | $\boldsymbol{q}$ |
| $\boldsymbol{f}$ | $\boldsymbol{e}$ |
| $\boldsymbol{f}$ | $\boldsymbol{p}$ |
| $\boldsymbol{q}$ | $\boldsymbol{e}$ |
| $\boldsymbol{q}$ | $\boldsymbol{p}$ |

**Table 3.1:** Possible combinations of physical quantities ($\boldsymbol{f}, \boldsymbol{e}, \boldsymbol{q}, \boldsymbol{p}$) corresponding to "command" and "feedback"

### 3.2.1 Top view of the energy flow

In order to integrate energy awareness into the control software, description of the composition of the energy in the system and the energy-communication way are necessary. To provide an energy-communication standard for the system, the top view of the energy flow is needed.

Using the energy-aware system with a port-Hamiltonian energy tank by Raiola et al. (2018) as an example, the top view of the energy flow is illustrated in Figure 3.2. And the precise names and brief descriptions of each energy signals proposed by Hobert (2020) [1] in Figure 3.2 are listed in Table 3.2.

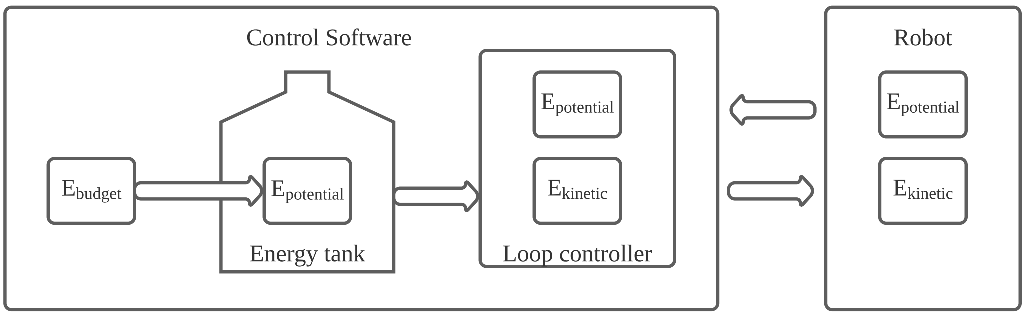


**Figure 3.2:** The top view of the energy flow consists of energy signals ($E_{budget}$ etc.) and 3 entities (loop control layer, energy tank and plant). The arrows represent the flow of energy quanta. There should be a sequence control layer before the loop control layer, and Measuring & Actuation layer before plant, but due to the main contribution of this thesis is about loop control layer, other layers in the Embedded Control Software are ignored.

| Proposed Energy Signal | Description |
|---|---|
| $E_{budget}$ | energy supplied to energy tank |
| $E_{tank,\ potential}$ | the potential energy of the virtual spring representing the energy tank |
| $E_{loop,\ potential}$ | the potential energy in loop controller |
| $E_{loop,\ kinetic}$ | the kinetic energy in loop controller |
| $E_{robot,\ potential}$ | the potential energy in robot |
| $E_{robot,\ kinetic}$ | the kinetic energy in robot |

**Table 3.2:** The brief description of the energy signals in the system, ignoring the interfaces in other layers of the Embedded Control Software except loop control layer.

In Figure 3.2, according to Brodskiy (2014), "the level of the energy tank should be interpreted as a tight energy budget which is used to pay for all actions". Therefore, the maximum value in the energy tank depends on the energy budget that supplies to it. However, calculating the accurate energy budget is difficult. According to Brodskiy (2014), determining model-based energy budget for the energy tank need to know the real-time models of the loop control layer, the robot, and the environment. Besides, the author states that incorrect estimation of the energy budget results in a negative impact on system performance by the energy tank. As the main goal of this thesis serves as a basic proof of concept for the energy-aware configuration and coordination, it is not in the scope of this thesis to develop highly-accurate component models for estimating the energy budget. Moreover, for energy-aware coordination, it is ideal to make the design pattern that is widely applicable, and allow the task oriented control system to become black boxes for the engineers, the model of the loop control layer will not be available for estimating the energy budget. As an alternative, the energy budget ($E_{budget}$) for coordination in this thesis is responsible for filling the energy tank under the maximum limit at the beginning of energy tank working. The specific value of maximum energy level can be

[1] In Hobert (2020)'s work, they called energy interfaces, but in this thesis, they are just used to describe how the energy flows between the components, so some changes are made.

determined by the specific use cases. For example, the HIC (Head Injury Criteria), or the energy and power-based safety metrics applied in the second project of Section 3.1 (Human-friendly manipulators) etc..

Groothuis et al. (2018) mentioned "A certain motion task that is to be executed by the robot requires an amount of energy to be converted into kinetic energy. An accelerated motion will always correspond to a change of kinetic energy ($E_{robot,\ kinetic}$), and if a system moves along a gravitational field, for instance increasing and decreasing its height, the potential energy ($E_{robot,\ potential}$) will change as well." In a Hamiltonian energy tank, the potential energy of a virtual spring is used to represent the virtual energy in the tank ($E_{tank,\ potential}$). And the $E_{tank,\ potential}$ is partially converted into kinetic energy ($E_{loop,\ kinetic}$), which is corresponding to $E_{robot,\ kinetic}$ as well as $E_{robot,\ potential}$, needed by the robot to executed a certain motion task. Another part of $E_{tank,\ potential}$ is stored in loop controller as potential energy ($E_{loop,\ potential}$), such as a the virtual spring in Cartesian impedance controller. This can be derived from Equation 2.12, in which the state of the spring $s_n$ is influenced by the velocity of the joint $\dot{q}_n$. The velocity of the joint $\dot{q}_n$ will not only constitute the $E_{loop,\ kinetic}$ ($E_{loop,\ kinetic} = \frac{1}{2}m\dot{q}_n^2$, $m$ is the mass of the joint), but also the $E_{loop,\ potential}$ ($E_{loop,\ potential} = \frac{1}{2}kq_n^2$, $k$ is the stiffness of the virtual spring). However, actually not all the $E_{robot,\ potential}$ can be controlled by the engineer, because robot controllers often apply active gravity compensation, which is an untouchable area for the engineer.

When the scalar energy transported in the direction of the arrow, it is along with a sign-added or subtracted. For example, in van Teeffelen (2018)'s work, when decreasing the stiffness in impedance control, more energy than needed may be supplied to the physical system. Then the energy budget should be negative to remove the extra energy in the energy tank.

## 3.3 Use case for evaluating energy awareness

To allow energy-aware coordination and configuration to be evaluated, suitable evaluation scenarios are needed. Section 1.1 states the example of autonomous robot (e.g. self driving car) that changes the control law, i.e. coordination. From Section 3.1, it can be observed that there exist many scenarios in which energy-aware coordination and configuration could be useful.

An energy-aware coordination and configuration should be able to detect or even predict the energy inconsistency, active (non-passive) action, and some unsafe action during configuration and coordination, by assessing the energetic state of the system. After detection or prediction, the corresponding solution should be planned and executed, like adjusting the value of parameters (e.g. velocity, force etc.), increasing the transition period of configuration and coordination, or even stop the configuration/coordination. Then the detection should be activated again to check whether the undesired situation is solved or not.

As mentioned in Section 1.1, a form of energy-aware configuration has been addressed in van Teeffelen (2018)'s work. Therefore, only focusing on the energy-aware coordination. The proposed use case is illustrated in Figure 3.3, and has three sequences:

1. Interval $[t_0 - t_1]$: The loop control layer uses *controller A* to execute the task commanded by sequence control layer.

2. Interval $[t_1 - t_2]$: Doing coordination action, i.e. switching the control law of loop control layer from *controller A* to *controller B*.

3. Interval $[t_2 - t_3]$: The loop control layer uses *controller B* to execute the task commanded by sequence control layer.

In the proposed use case, the loop control layer applies two control law successively. Since the proposed design pattern makes the control law become a black box for engineer, the two

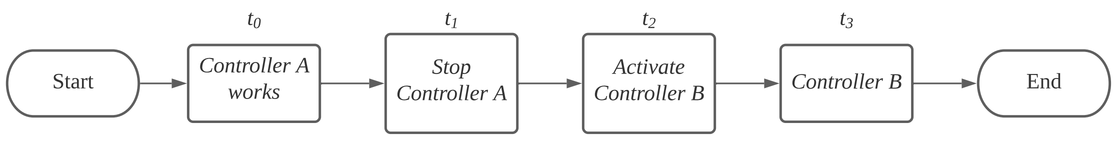


**Figure 3.3:** Use case

control laws could be arbitrarily selected, as long as they can provide the control interfaces mentioned in Section 3.2. In this thesis, the two control laws named *controller A* and *controller B* respectively. In order to make the effect of energy awareness obvious in coordination, the sequence control layer should only publish one task from start to finish, such as commanding the robotic manipulator to draw a circle.

## 3.4 Requirements

In this section, a list of requirements for the energy-aware configuration and coordination of control software for robot has been created. They are prioritized following the MoSCoW (Must, Should, Could, Won't) criterion, which is explained in Table 3.3.

| | |
|---|---|
| **Must** | Defines the project. Must haves are critical and failing to achieve one equals failing the project. |
| **Should** | Structures the project. Should haves are important to the project. Failure of achieving these would be notable, but would not result in failure of the project. |
| **Could** | Improves the project. Could haves are relevant to the project, but not necessary. Only if time permits should they be pursued. Failure of achieving a Could have has no notable consequences on the project outcome. |
| **Won't** | Extends the project. Won't haves are what would be interesting extensions to the project. They are currently not needed or not feasible and may only be pursued if all other tasks are done. Won't haves can be part of the recommendations for future work. |

**Table 3.3:** MoSCoW (Must, Should, Could,Won't) prioritisation of tasks explained (Clegg and Barker, 1994)

The requirements are as follows:

- **Must**
  - The design pattern for energy-aware configuration of the loop controller *must* be developed based on van Teeffelen (2018)'s work.
  - The design pattern for energy-aware coordination at loop control level *must* be developed. And the energy awareness for the proposed use case in Section 3.3 *must* be achieved by using the design pattern.
  - A demo of the final product (design pattern) *must* showcase the effects of energy-aware coordination of control software.
  - The design *must* based on CBSD mentioned in Section 2.1.
    - The design pattern *must* incorporate the control interfaces consisting of "command" and "feedback" inputs and outputs (illustrated in Figure 3.1) that can communicate the four physical quantities ($\boldsymbol{f}, \boldsymbol{e}, \boldsymbol{q}, \boldsymbol{p}$) as indicated in Section 3.2.

        * The flow way of energy in design pattern *must* be consistent with the standardised communication of energy data as indicated by Section 3.2.1.

- **Should**
    - The design *should* consider the connection point for task commands coming from the sequence control layer.
    - The generic requirements at loop control level for energy-aware coordination and configuration *should* be set. For instance, the loop control layer *should* provides either $\boldsymbol{e}$ or $\boldsymbol{p}$ together with either $\boldsymbol{f}$ or $\boldsymbol{q}$, and are belonging to the same coordinate frame (refer to Table 2.1).
    - The specific requirements for specific application at loop-control level for energy-aware coordination and configuration *should* be set. For instance, in the demo's use case, if the energy peak exceed the maximum allowable energy level, the torques which are inputted to the robot will be attenuated.
- **Could**
    - An energy budget *could* be set for the loop control level.
        * Only the value of energy budget *could* be set, this *could* be achieved by implementing a real-time system model and/or setting pre-configured reference values.
    - More than one use case *could* be used to show case the effects of energy-aware coordination of control software.
- **Won't**
    - The optimization of configuration and coordination for performance *won't* be done in this work.
    - Energy awareness for other control-software configuration situations *won't* be implemented in this work.
    - Energy awareness for other control-software coordination situations *won't* be implemented in this work.
    - The demo *won't* be given to showcase the effects of energy-aware configuration of control software, as this has been done in van Teeffelen (2018)'s work.
    - The function of sequence controller which used to predict the energy budget *won't* be developed in this work.

# 4 Design and Implementation

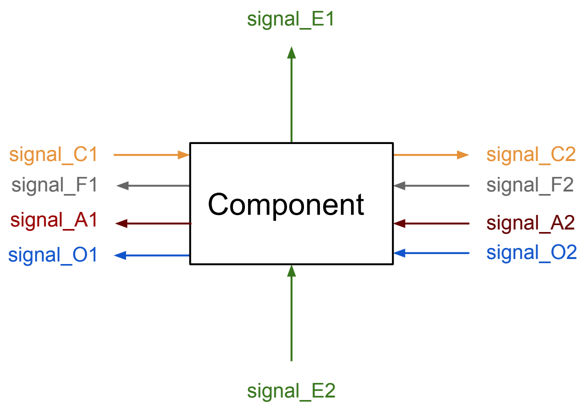
In Section 1.3.3, a series of research requirements are presented. This chapter focus on the third requirement, i.e. "the thesis should propose the method on how to implement energy-awareness in coordination and configuration action of the loop controller".

## 4.1 Design of energy-aware coordination and configuration architecture

This section describes the design of the energy-aware coordination and configuration architecture by using CBSD approach mentioned in Section 2.1, dividing and classifying all system functions into several components, as well as setting up the connections among the components. To explain how to take into account the system's energetic state in the control strategy, the design is presented in steps of increasing detail for clarity and intelligibility.

### 4.1.1 Interface Designs

Before presenting the components that make up the energy-aware coordination and configuration architecture, the interface designs are explained for the connections among the components. The classification of the interface inspired by the the method used by Hobert (2020).



| signal notation | | description |
|---|---|---|
| command signal | → | can be one of [***f, e, q, p***] |
| feedback signal | → | can be one of [***f, e, q, p***] |
| energy signal | → | can be one of [$E_{budget}$, $E_{tank, potential}$, $E_{loop, potential}$, $E_{loop,kinetic}$, $E_{robot, potential}$, $E_{robot, kinetic}$] |
| activate signal | → | activate processes corresponding to configuration/coordination |
| other signal | → | a task / control state / camera data etc. |

**Figure 4.1:** Descriptions of proposed interface signals (inspired by Hobert (2020)). The command and feedback signal arrows represent vector signals while energy signal arrows represent scalar signals, they all contain multiple quantities, the details are in Section 3.2 and Section 3.2.1 respectively. The activate signal activates processes of component corresponding to configuration or coordination action. The 'other' signal can contains all signals that cannot be classified as command, feedback energy or activate signal. The names beside the arrows (e.g. signal_C1, signal_F1, etc.) are merely a label to distinguish the signal, not the arrows' meaning. The layout of signals in this diagram just serves as an example, not all the types of signal notation must appear in one component.

Figure 4.1 shows signal notations used in all design architectures in this chapter. The arrows are colour coded to distinguish the different types of signals. The orange and grey arrows represent command and feedback signals respectively, being consistent with the control interface in Section 3.2. The green arrows represent energy signals and consistent to the energy flowing way in Section 3.2.1. The red arrows represent the signals responsible to activate the processes of the components corresponding to configuration or coordination action. The blue arrows represent other signals which are not in this project's scope to define a specific interface, such as the camera photo signal.

The command (orange) and feedback (grey) signals transmit arrays consisting of four physical variables ($\boldsymbol{f}, \boldsymbol{e}, \boldsymbol{q}, \boldsymbol{p}$) that can contain any of the quantities listed in Table 2.1. However, the control interface does not restrict the number of concurrently transmitting variables. For example, at one sample time, the feedback signal transmits the measured displacement signal from robot to controller, but at next sample time, both measured displacement and measured torque signals can be transmitted to controller through feedback signal. The reason for this design choice is to establish a control interface that is flexible for all kind of control strategy. For example, for coordination action, the interface should be compatible with various control laws. Like the control system changing the control law from torque control to position control, then the control interface should allow both torque ($\boldsymbol{e}$) and position ($\boldsymbol{q}$) signal to be commanded.

The energy signals (green) correspond to the energy flow in Section 3.2.1, and transmits an array of the energy variables listed in Table 3.2. Same as the control interface, the energy flow does not restrict the number of concurrently transmitting variables, i.e. the energy flow enables any number of the energy signals listed in Table 3.2 to be filled in. For example, the robot can feed both $E_{robot,\ potential}$ as well as $E_{robot,\ kinetic}$ through one energy flow back to controller at the same sample time.

The activate signals (red) are separated from other signals (blue), because this thesis focus on configuration and coordination actions, which need a specific signal to tell the system the start time and end time of the actions. Thus, the activate signals are responsible for starting and stopping the processes of the components handling configuration/coordination action.

### 4.1.2 Top-view based on the layered approach

This section uses the interface designs in Section 4.1.1 to establish the interconnections among the main control layers mentioned in Section 2.1.1, providing a top view of the full control-system design for configuration and coordination separately.

The design omits other layers in the *(Embedded) Control Software* except *sequence control layer* and *loop control layer*. Because as mentioned in Section 2.1.1, this thesis focuses on the development of *loop control layer*, and the *sequence control layer* is needed to send the setpoints to the *loop control layer* directly. The design also integrates the *I/O Hardware* as well as *Plant* in Figure 1.1 into a *Robot* component. Besides the signals marked with arrows in the figure, there are many other signals in the system, such as the robot can send the data from camera, laser etc. back to sequence control as mentioned in Section 2.1.1. But these signals will not be used in this thesis, so they do not appear in the figure.

In order to integrate energy awareness into the control software, a specific energy-aware component is needed in control software. Section 1.3.2 proposes using Passivity layer in van Teeffelen (2018) for configuration while using port-Hamiltonian energy tank in Raiola et al. (2018) for coordination.

For the configuration action, the top view of the design is illustrated in Figure 4.2. As mentioned in Section 1.2, the energy-aware composition, communication and computation of the control software has been addressed in the RaM group. They used Passivity layer to implement energy guards, enforcing system passivity. Reusing proven concepts is consistent with the CBSD and can therefore also be applied in this project. The main component in the Passivity layer is the energy tank which needs accurate energy budget according to Brodskiy (2014).

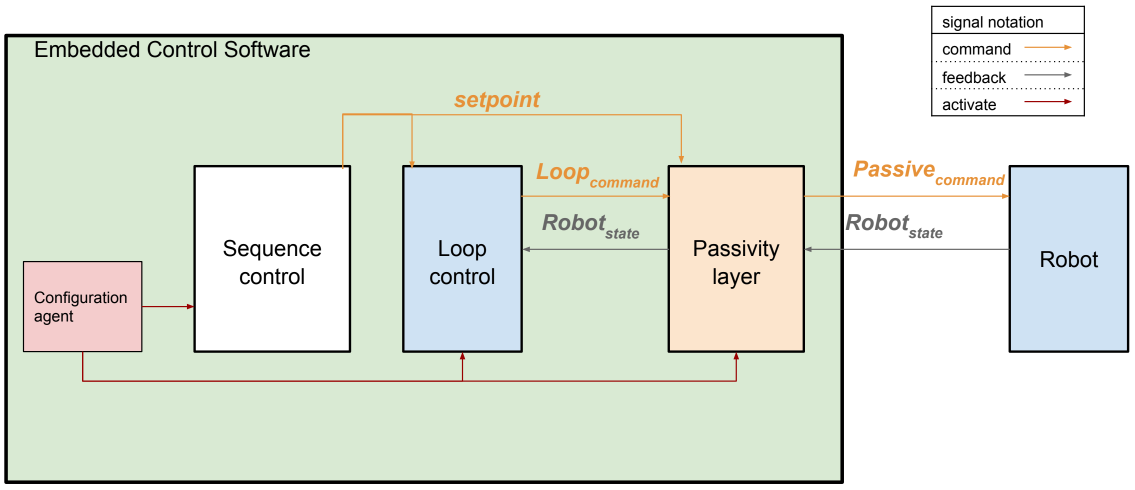


**Figure 4.2:** Design top view of energy-aware configuration control system based on layered approach.

For the configuration action, the top view of the design is illustrated in Figure 4.3. As mentioned in Section 3.2.1, for energy-aware coordination, it is ideal to make the design pattern that is widely applicable, and allow the task oriented control system to become black boxes for the engineers. By using this design, it is possible. If the control software is energy-unaware, the Loop control layer uses the control interface to communicate a command signal and a feedback signal with Robot directly. By adding port-Hamiltonian Energy Tank, the command and feedback signal is processed by port-Hamiltonian Energy Tank first and then transmitted to the original destination.

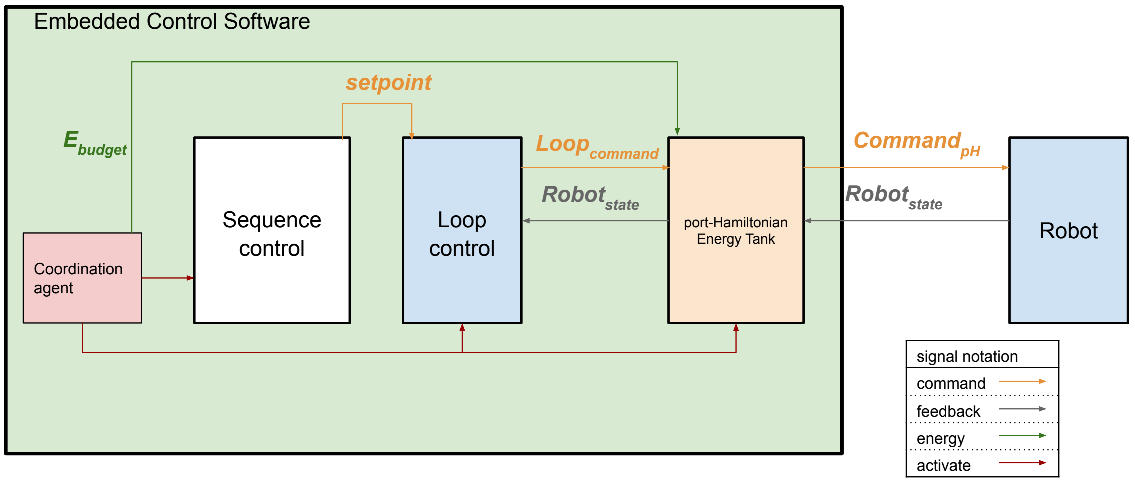


**Figure 4.3:** Design top view of energy-aware coordination control system based on layered approach.

Both in Figure 4.2 and Figure 4.3, an extra component is separated from the higher layer of *Sequence control* (perhaps in the *User interface* if the command is sent by the user manually, or in the *Supervisory layer* if the command is sent automatically). This component is the Configuration/Coordination agent who sends the activate signal (red arrow) to tell the receivers the configuration/coordination action is activated, appropriate processes need to be implemented. For example, when the coordination action is activated, the robotic arm cannot follow the setpoints in time as mentioned in Section 2.1.1, making the error between current state and setpoint state become bigger and bigger. As a coordination action, the sequence control layer can stop generating new setpoint until the setpoint is actually reached. Moreover, for Coordination agent, it also responsible to send the $E_{budget}$ to the energy tank, and the value of $E_{budget}$ is depending on the use cases.

### 4.1.3 Detail design of Passivity layer for configuration

The Passvity layer of configuration is inspired from the block with the same name by Hobert (2020) as shown in Figure 4.4. The Passivity Layer consists of an Energy Tank block, a Passive Zero-Order Hold (ZOH) block, an Energy Sampling block, and the most important block for configuration, i.e. Energy compensation block, as shown in Figure 4.4.

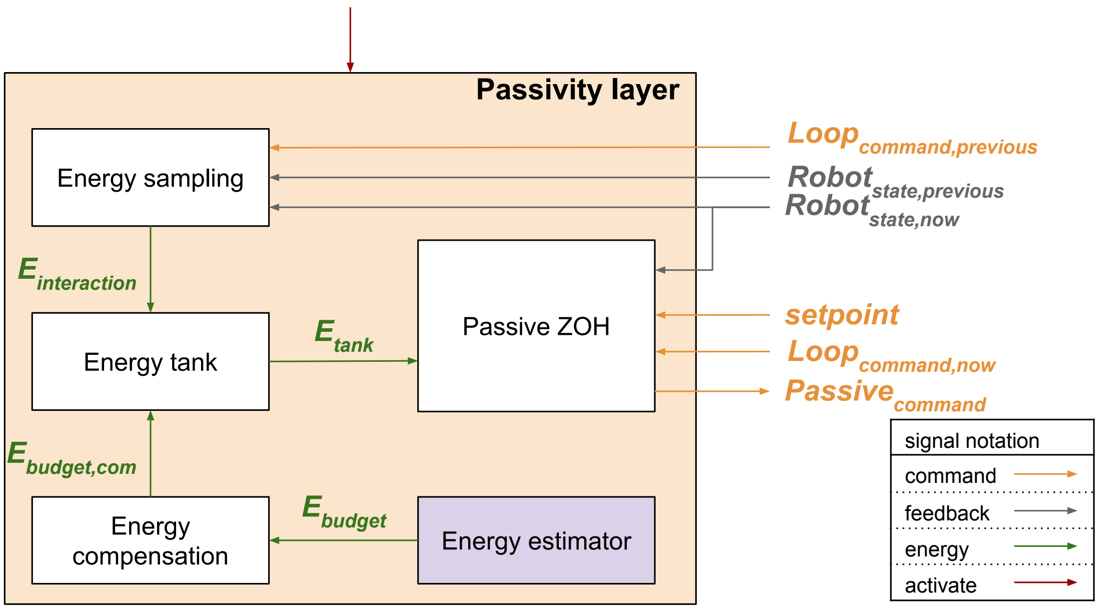


**Figure 4.4:** Passivity layer for configuration inspired by Hobert (2020), consists of an Energy Tank block, a Passive Zero-Order Hold (ZOH) block, an Energy Sampling block, and an Energy compensation block.

To increasing the comprehensibility, the explantion of the Figure 4.4 is combined with an displacement-in/effort-out system as an example.

The Energy tank block computes the current energy-tank level ($E_{tank}(k)$) as a function of the old energy-tank level ($E_{tank}(k-1)$), energy budget with compensation ($E_{budget,com}(k-1)$) and the amount of energy consumed by the Loop-Control layer during the last sample interval($E_{interaction}$):

$$E_{tank}(k) = E_{tank}(k-1) + E_{budget,com}(k-1) - E_{interaction}$$

The Energy compensation block is responsible for calculating the energy compensation as mentioned in Section 2.3.1. Which is important for the energy tank to register the energy flows in and out of the controller in the right way. The output of this block is:

$$E_{budget,com} = E_{budget} + \Delta H_{K_v}$$

in which $\Delta H_{K_v}$ is the symbol representing energy compensation in Section 2.3.1.

Assuming the system uses *effort* ($\boldsymbol{e}$) as "command" signal and *position* ($\boldsymbol{q}$) as "feedback" signal. The Energy sampling block calculates the $E_{interaction}$ using Equation 2.4 too, but for the previous sample interval $[(k-1)T, kT]$:

$$\begin{aligned}
&E_{interaction}(k) = \boldsymbol{e}^T(k)(\boldsymbol{q}(kT) - \boldsymbol{q}((k-1)T))\\
&\boldsymbol{e}(k) = Loop_{command,previous}\\
&\boldsymbol{q}(kT) = Robot_{state,now}\\
&\boldsymbol{q}((k-1)T)) = Robot_{state,previous}
\end{aligned} \tag{4.1}$$

For Eenrgy estimator block, according to Brodskiy (2014), determining model-based energy budget for the energy tank needs to know the real-time models of the loop control layer, the robot, and the environment. As for configuration action, the model of loop control layer should be available. Because as mentioned in Section 2.1.2, the configuration action means to tune the parameters such as the stiffness value in the loop controller, which means the engineer is possible to know and understand the control law used in the loop control layer. For example, in *Teleoperated robotic systems* (van Teeffelen, 2018) mentioned in Section 3.1, if the engineer desired safety and stability, he needs to know the loop control layer uses Cartesian impedance control, and what he needs to do is decreasing the impedance values. Then when calculating the energy budget, the potential energy stored in virtual spring should be considered. However, as mentioned in Section 3.2.1, it is not in the scope of this thesis to develop highly-accurate component models for estimating the energy budget. So the design of Energy estimator block is arranged in Appendix A.

The Passivity layer enforces passivity of the configuration action by making sure the energy tank level remains positive. This is achieved by the Passive ZOH block, which modulates the command signal from Loop control layer if need be. It estimates the amount of energy required for the upcoming sample interval $[kT, (k+1)T]$ ($E_{req}(k+1)$) using Equation 2.4, in which

$$\begin{aligned}
&E_{req}(k+1) = \boldsymbol{e}^T(k+1)(\boldsymbol{q}((k+1)T) - \boldsymbol{q}(kT))\\
&\boldsymbol{e}(k+1) = Loop_{command,now}\\
&\boldsymbol{q}((k+1)T) = setpoint\\
&\boldsymbol{q}(kT)) = Robot_{state,now}
\end{aligned}$$

and, if this energy estimate ($E_{req}(k+1)$) exceeds the current energy-tank level ($E_{tank}(k)$), the Passivity layer attenuates the commanded signal accordingly. When the energy in the tank is depleted, no movement is allowed:

$$e_{passive}(k+1) = \begin{cases} e(k+1)\cdot\frac{E_{tank}(k)}{E_{req}(k+1)}, & \text{if}\big(E_{req}(k+1) \geq E_{tank}(k) > 0\big)\\ e(k+1), & \text{if}\big(E_{req}(k+1) < E_{tank}(k)\big)\,\&\,(E_{tank}(k) > 0)\\ 0, & \text{if}\,(E_{tank}(k) \leq 0)\end{cases} \tag{4.2}$$

### 4.1.4 Detail design of port-Hamiltonian energy tank for coordination

Due to the possibility of missing the model of loop control layer when dealing with the coordination action, the Energy Estimator will not be used. And the energy budget for coordination in this thesis is only responsible for filling the energy tank under the maximum limit at the beginning of energy tank working as mentioned in Section 3.2.1.

The port-Hamiltonian Energy Tank for coordination is inspired by Raiola et al. (2018). Which treats the port-Hamiltonian Energy Tank as a whole, making the physical description of the port-Hamiltonian Energy Tank intelligible. And Raiola et al. (2018) regards each joint of the manipulator as a subsystem. Hence, the port-Hamiltonian Energy Tank in Figure 4.5 is for one joint $n$.

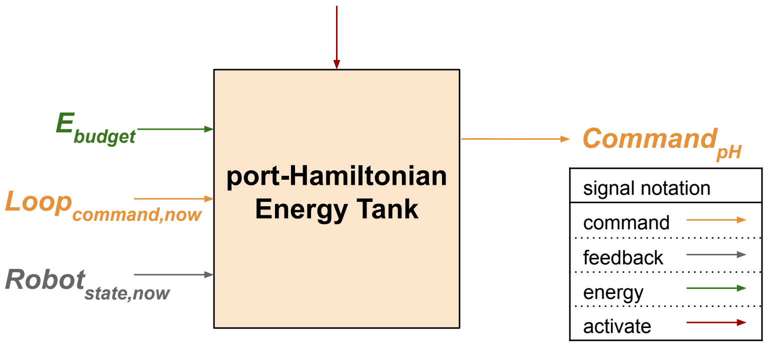


**Figure 4.5:** Port-Hamiltonian Energy Tank for coordination inspired by Raiola et al. (2018).

The Energy tank block inputs the velocity of the joint $\dot{q}_n$ ($Robot_{state,now}$), using Equation 2.12 to get the state of the spring ($s_n$), and the energy level in the tank $E_{tank} = H_n(s_n) = \frac{1}{2}ks_n^2 = \frac{1}{2}s_n^2$, in which the stiffness of the virtual spring choose to be $k = 1$.

The port-Hamiltonian Energy Tank tries to preserve energy-consistency of the coordination action by modulating the transmission ratio $u_n$. In Raiola et al. (2018)'s experiment, under the presence of an external force, the energy tanks level increases. But the authors didn't limit the increasing of the energy tanks level. The disturbance due to coordination action can be regard as an external force applying to the system. In order to preserve energy-consistency, limit the increasing of the energy tanks level, proportionally attenuate the torques (from torques output by Loop controller $\tau_{c_n}$, i.e. $Loop_{command,now}$ in Figure 4.5, to torques output by port-Hamiltonian Energy Tank $\tau_{n_{out}}$, i.e. $Command_{pH}$ in Figure 4.5) with the difference in maximum tank level (comes from the Coordination agent, which is equal to $E_{budget}$, depending on the specific use cases, such as using Head Injury Criteria) and actual tank level, i.e.

$$\frac{\tau_{n_{out}}}{\tau_{c_n}} = \frac{E_{max}}{H_n(s_n)},$$

following derivation is derived as a supplement to Equation 2.10:

$$\begin{aligned}
&\because \tau_{n_{out}} = \tau_{c_n}\frac{E_{max}}{H_n(s_n)}\\
&\because H_n(s_n) = \frac{1}{2}s_n^2\\
&\therefore \tau_{n_{out}} = \tau_{c_n}\cdot\frac{2E_{max}}{s_n^2} = \tau_{c_n}\frac{2E_{max}}{s_n^3}\cdot s_n\\
&\because \tau_{n_{out}} = -u_n\cdot s_n \; (Equation\, 2.11)\\
&\therefore u_n = \frac{-2E_{max}}{s^3}\tau_{c_n}
\end{aligned} \tag{4.3}$$

then the transmission ratio $u_n$ is determined using $\tau_{c_n}$ ($Loop_{command,now}$) and $s_n$ (related to $\dot{q}_n$, i.e. $Robot_{state,now}$) as:

$$u_n = \begin{cases} \frac{-2E_{max}}{s^3}\tau_{c_n}, & \text{if}\,(H_n(s_n) \geq E_{max})\\ \frac{-\tau_{cn}}{s_n}, & \text{if}\,(H_n(s_n) > \epsilon)\,\&\,(H_n(s_n) < E_{max})\\ \frac{-\tau_{cn}}{\gamma^2}s_n, & \text{otherwise} \end{cases} \tag{4.4}$$

Once the transmission ratio $u_n$ is settled according to the energy levels in $H_n(s_n)$, the torque output sent to joint $n$ (the $Command_{pH}$ signal in Figure 4.5) is computed as:

$$\tau_{o_n} = -u_n\cdot s_n \tag{4.5}$$

And the specific value of maximum energy level ($E_{max}$) can be determined by the specific use cases as mentioned in Section 3.2.1. For example, the HIC (Head Injury Criteria), or the energy and power based safety metrics applied in the second project of Section 3.1 (Human-friendly manipulators) etc.. And it will be sent by Coordination agent in Figure 4.3.

## 4.2 Conceptual implementation of the energy-aware coordination

This section presents a method of implementation the goals posed in Section 1.3.2. As mentioned in Section 1.1, a form of energy-aware configuration has been achieved in van Teeffelen (2018)'s work. Hence, the implementation of energy-aware configuration will not be processed in this thesis. For energy-aware coordination, the implementation serves a proof of concept. It is intended to form a basis on implementing rather than being a fully finished implementation with extensive functionalities. And it choose the use case in Section 3.3 as the implementation scenario.

### 4.2.1 Robot platform

As posed in Section 2.1.1, even though this thesis focuses on the loop control layer (part of software), it is still important to keep in mind the full control system including *I/O Hardware* as well as the *Plant*. Therefore, the implementation needs to be connected to a plant (robot plus environment). Since the robot's development is out of this project's scope, it should be reused from existing platform.

The KUKA Lightweight Robot (LWR) manipulator, developed by the German Aerospace Center (DLR)[1], is one of the earliest generations of manipulators designed for human interaction. (Cardenas, 2017) introduces the robot features moderate joint compliance, suitable sensing, and control capability. And due to its lightweight essence, dynamic performance is increased by reducing power consumption . The manipulator has 7 DOF (degree of freedom), where each joint is equipped with position and torque sensors, and the components characterization and relevant dimensions of the manipulator are displayed in Figure 4.6. Thus, the robot can be operated with position, velocity and torque control. And the 7 DOF gives the LWR 4+ greater flexibility and precision than the more common 6-axis robots.

The Research Center "E.Piaggio" provided a series of software related to the KUKA LWR 4+, either for real and for simulation[2], where the controller can be applied in loop control layer. However, not all the controllers can be used in simulation. For example, the position controllers such as "one_task_inverse_kinematics" controller cannot be used, which command position (displacement) but cannot get the force (effort) feedback. The reason why this happens is answered by the "E.Piaggio": "Since there is no actual PID, but the position is set directly. This is because we didn't want to tune a PID for simulation, and in the real case, KUKA already did a great job with the joint position control"[3]. Only if the controller can provide the control interface mentioned in Section 3.2, it can be applied to integrate energy awareness. In this thesis, the "minimum_effort_inverse_dynamics" controller and the "backstepping_controller" controller are being used in the loop control layer, since the loop control layer can be a black box, the detail of the control laws is not important. They both use "flow-in/effort-out" causality, to be precise, is "displacement-in/effort-out", i.e. input position data, output torque data to each joints. However, the ways of calculating the output torque are different, leading to the bump of energy appears when changing the control laws.

[1] DLR Portal-German Aerospace Center: http://www.dlr.de
[2] https://github.com/CentroEPiaggio/kuka-lwr
[3] https://github.com/CentroEPiaggio/kuka-lwr/issues/69

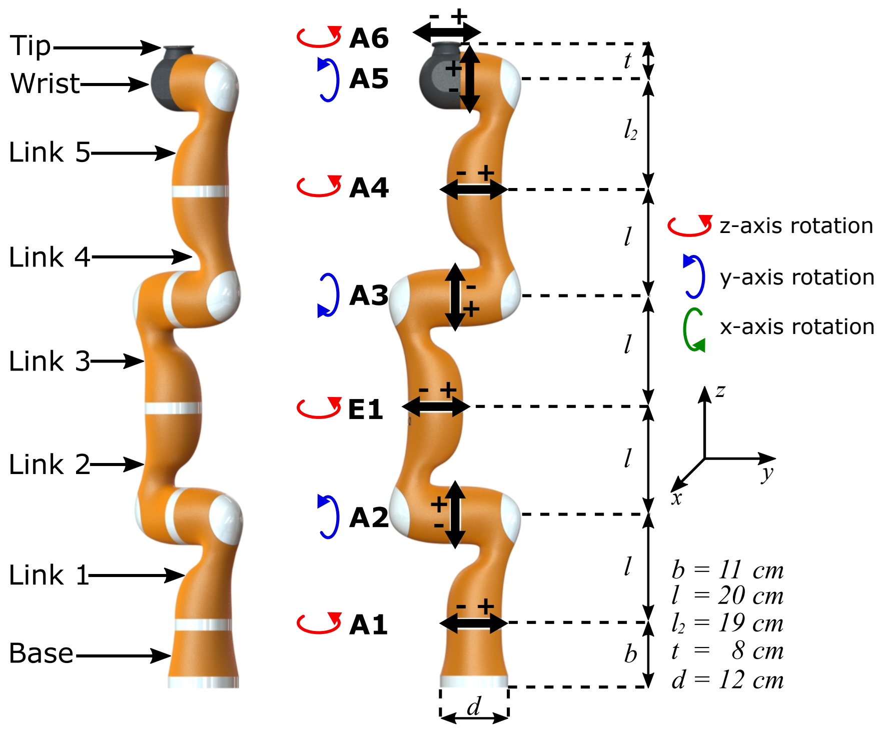


**Figure 4.6:** Schematic of the KUKA LWR 4+ manipulator with components characterization.

### 4.2.2 Overview of the implementation

This section clarified how the whole energy-aware control software works to handle the coordination action of the use case mentioned in Section 3.3, the flow chart is shown in Figure 4.7.

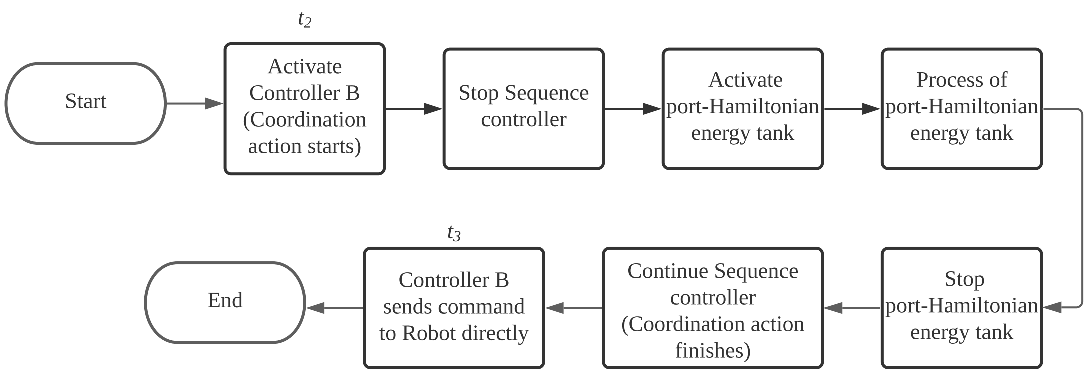

$t_2$
Start
Activate Controller B (Coordination action starts)
Stop Sequence controller
Activate port-Hamiltonian energy tank
Process of port-Hamiltonian energy tank
Stop port-Hamiltonian energy tank
Continue Sequence controller (Coordination action finishes)
$t_3$
Controller B sends command to Robot directly
End

**Figure 4.7:** Flow chart for the steps of the whole energy-aware control software to handle the coordination action of the use case mentioned in Section 3.3

The handling of the coordination action happens in the interval $[t_2 - t_3]$ of the use case. Hence, the process before $t_2$ is omitted. The reason why the Sequence controller needs to stop during handling the coordination action, and why the working time of port-Hamiltonian Energy Tank is limited will be analysed in the coming sections respectively. And the detail of the "Process of port-Hamiltonian energy tank" will be illustrated as a flow chart in Section 4.2.4. This flow chart just provides an overview of the entire process.

### 4.2.3 Implementation of the sequence control layer

As can be seen from Figure 4.3, the implementation also need to include a Sequence control layer, sending setpoints to Loop control layer and Passivity layer. The setpoints are implemented in a form of H-matrix, which describes a target pose of the robot's end effector with respect to the robot's inertial reference frame ($\boldsymbol{H^0_{target}}$).

For the experiments, a Cartesian reference trajectory was defined as a periodic motion. Because development of a sequence controller has no priority in this project, thus the setpoints just simply let the end effector move back and forth in a straight line as shown in Figure 4.8, and the orientation is kept unchanged:

$$H^0_{target}(t) = \begin{cases} p^0_{target}(t) = \begin{bmatrix} x(t) \\ y(t) \\ z \end{bmatrix} = \begin{bmatrix} 0.25 + 0.05 sin(ft) \\ 0.05 sin(ft) \\ 1.75 \end{bmatrix} \\ R^0_{target} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{cases} \tag{4.6}$$

in which, frequency $f = \frac{2\pi}{T} = 3$.

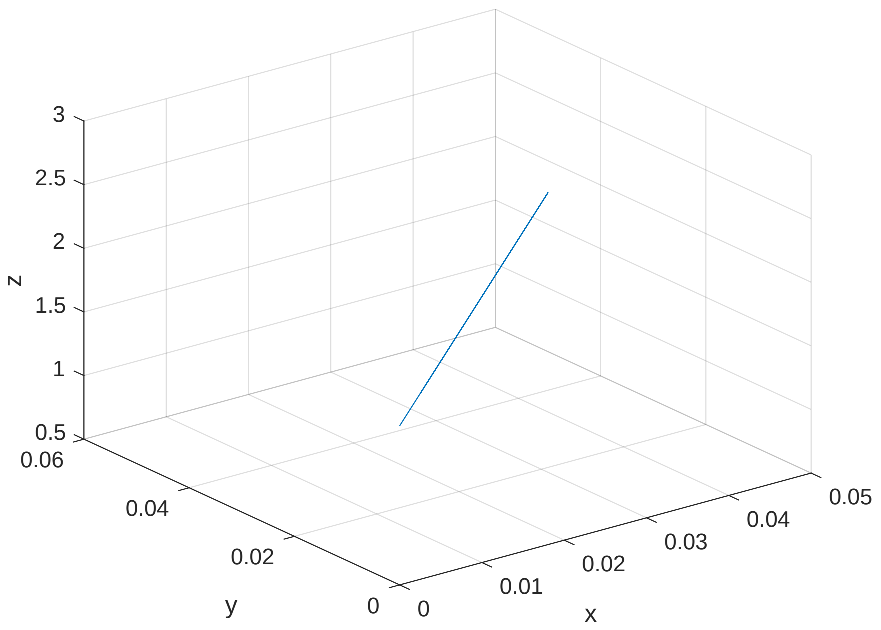


**Figure 4.8:** Desired trajectory of robot's end effector.

In order to avoid the situation that the robotic arm cannot follow the setpoints in time during coordination action, making the error between current state and setpoint state become bigger and bigger, the sequence control layer will stop generating new setpoint until the coordination action finishes.

### 4.2.4 Implementation of the port-Hamiltonian Energy Tank for coordination action

The port-Hamiltonian Energy Tank for coordination action has been implemented in line with its design shown in Figure 4.3 and Figure 4.5. Its main functionalities are illustrated by the flowchart in Figure 4.9.

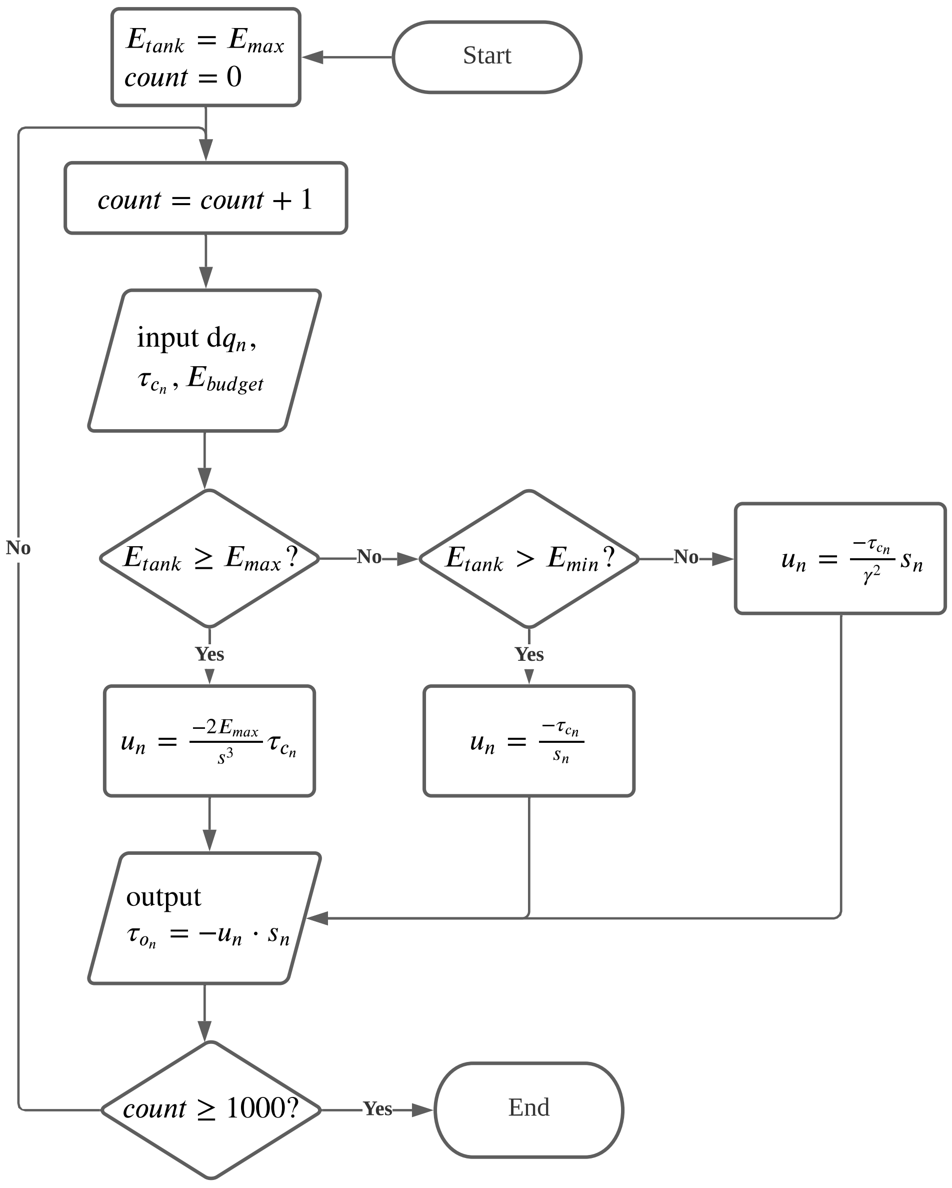

Start
$E_{tank} = E_{max}$
$count = 0$
$count = count + 1$
input $dq_n$, $\tau_{c_n}$, $E_{budget}$
$E_{tank} \geq E_{max}$?
No
$E_{tank} > E_{min}$?
No
$u_n = \frac{-\tau_{c_n}}{\gamma^2} s_n$
Yes
$u_n = \frac{-2E_{max}}{s^3} \tau_{c_n}$
Yes
$u_n = \frac{-\tau_{c_n}}{s_n}$
output $\tau_{o_n} = -u_n \cdot s_n$
$count \geq 1000$?
Yes
End
No

**Figure 4.9:** Flow chart of process for port-Hamiltonian Energy Tank

First fill the energy tank using the $E_{budget}$ ($E_{tank} = E_{max} = E_{budget}$) and activate a counter to limit the working time ($WorkingTime = count \times SampleTime$) of the energy tank. Because if the energy tank works too long, the output torques will be continuously limited until $E_{tank}$ drops below $E_{max}$, which consumes a lot of time, and the error between current state and set-point state may be worse. If the loop controller using the Cartesian forces of the end-effector as the intermediate variable, and finally use transpose of the Jacobian, mapping to the torques in joint coordinates, like what Cartesian impedance controller does, the error may keep decreasing, because only the amplitude of Cartesian forces will be limited, not the orientation.

However, the design pattern of energy-aware coordination needs to be generic, and usually the controller only has joint torques, for which, a balancing end-effector force does not always exist. There might be a better way to deal with it rather than limit the working time, but it is not the scope of this thesis.

In line with the design presented in Section 4.1.4, the energy tank uses the input signal ($\dot{q}_n$) to calculate the energy level in the tank ($E_{tank}$). Then the port-Hamiltonian Energy Tank will keep or attenuate the output torques depending on the $E_{tank}$. Once reaching the limit time, such as $count \geq 1000$, the port-Hamiltonian Energy Tank will stop working, and the computed torques of Loop controller will be sent to robot directly, the sequence controller will continue generating new setpoints.

## 4.3 Conclusion

This chapter describes the design pattern of the energy-aware configuration and coordination respectively, and the conceptual implementation of energy-aware coordination, on the basis of requirements posed in Section 3.4. The design pattern serves as an architectural guide for the implementation, while the implementation described in this chapter is conceptual and serves a take on how the design could be implemented. The energy-aware configuration and coordination design both consist of control interfaces responsible for the communication of the control signals, and the energy flowing way are consistent with the one posed in Section 3.2.1. But the main components are different, for energy-aware configuration, it is a passivity layer, for energy-aware coordination, it is port-Hamiltonian energy tank. Although the architecture of main components for configuration and coordination are different, the cores are the same, i.e. limiting the active behavior by limiting the available energy for robot supplied by the control system. The energy-aware configuration has been implemented by van Teeffelen (2018), while the energy-aware coordination is implemented in simulation and uses the use case in Section 3.3 as an implementation example.

# 5 Evaluation

This chapter treats the project's requirement of "a demo of the final product (design pattern) must showcase the effects of energy-aware coordination", stated in Section 3.4, and also the forth research requirement in Section 1.3.3, i.e. "the assessment of the design pattern should be done in this thesis, to analyse the benefits and drawbacks of energy-aware configuration and coordination". Four different experiments on energy-aware coordination are treated, they serve as an assessment of concept presented in Chapter 4. With the results of these experiments, the design and implementation of the energy-aware coordination architecture presented in Chapter 4 are evaluated on the basis of the system-level properties, such as energy consistency.

## 5.1 Experiments

The experiments conducted in this project serve a common purpose of demonstrating system-level properties, such as the control software energy awareness when dealing with coordination action. The steps of the experiments are consistent with the steps in Figure 4.7, as shown in

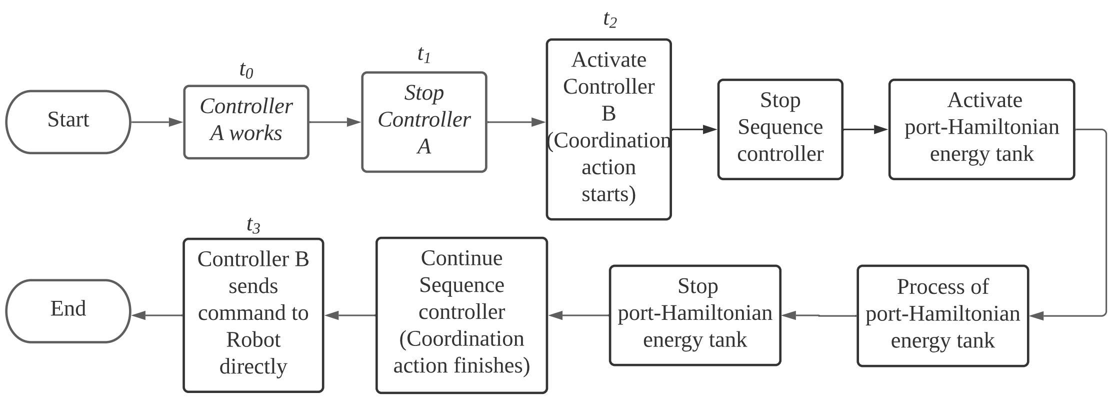

Start
$t_0$
Controller A works
$t_1$
Stop Controller A
$t_2$
Activate Controller B (Coordination action starts)
Stop Sequence controller
Activate port-Hamiltonian energy tank
Process of port-Hamiltonian energy tank
Stop port-Hamiltonian energy tank
Continue Sequence controller (Coordination action finishes)
$t_3$
Controller B sends command to Robot directly
End

**Figure 5.1:** Flow chart for the steps of experiments, is consistent with Figure 4.7, but adding the steps before $t_2$ of the use case in Section 3.3.

Four experiments have been conducted, each with the same duration and trajectory (explained in Section 4.2.3). The difference between these four experiments is in the form of simulation types (with or without port-Hamiltonian energy tank working), maximum value of the energy level in the energy tank (mentioned in Section 4.1.3), as explained below:

1. Without port-Hamiltonian energy tank
   The first experiment performs a simulation without port-Hamiltonian Energy Tank working, i.e. the torques output by the Loop controller will send to the robot directly. This experiment serves as a reference scenario to compare with the other experiments in terms of interaction energy, evaluating experiment results from the perspective of energy.

2. With port-Hamiltonian energy tank, $\boldsymbol{E_{max} = 5J}$
   The second experiment performs a simulation with port-Hamiltonian Energy Tank working, i.e. the output torques may be attenuated by the port-Hamiltonian Energy Tank. The purpose of this experiment is to verifying whether the design of energy awareness is effective or not for coordination action, and evaluating the system-level properties when adding port-Hamiltonian Energy Tank to the control software. $5J$ has empirically been

chosen as a starting $E_{max}$, because it leads to insightful data where the torques will be attenuated (the value is not too high, and the phenomenon of too high value is shown in **Experiment4**) and the robot manipulator will not out of control (the value is not too low, and the phenomenon of too low value is shown in **Experiment 3**).

3. With port-Hamiltonian energy tank, $\boldsymbol{E_{max}} = \mathbf{4}\boldsymbol{J}$
   The third simulation comprises a simulation that allows port-Hamiltonian Energy Tank to work too, but making $E_{max} = 4J$, which is smaller than the $E_{max}$ applied in the second simulation. The purpose of this experiment is to verify the negative effect of port-Hamiltonian Energy Tank as mentioned in Section 4.2.4, i.e. too much restriction on the output torques will degrade the performance, increasing the error between current state and setpoint state.

4. With port-Hamiltonian energy tank, $\boldsymbol{E_{max}} = \mathbf{6}\boldsymbol{J}$
   The forth simulation comprises a simulation that allows port-Hamiltonian Energy Tank to work too, but making $E_{max} = 6J$, which is bigger than the $E_{max}$ applied in the second simulation. The purpose of this experiment is to verify the negative effect of port-Hamiltonian Energy Tank if allow too much energy in the energy tank, will also degrade the performance, weaken the effect of limiting the bump of the torque output, which is bad for the energy consistency.

In all experiments, the setpoints sended by the seqenunce control layer follow the same trajectory (explained in Section 4.2.3). And the $t_1$, $t_2$ and $t_3$ in Figure 5.1 are as unified as possible in all experiments with the help of ROS script, making the coordination action happens almost at the same pose and having the same coming setpoints (due to the simulation environment, complete consistency cannot be guaranteed), allowing easier comparison between the four experiments. The ROS computation graph is in Appendix B. A list of hardware and software materials that have been used in the experiments is given in Appendix C. The results of the experiments are presented in Section 5.2.

## 5.2 Results and Discussion

This section presents the data resulting from the four experiments described in Section 5.1, and then discusses the result.

For the first experiment, only one figure is shown, that is, the interaction energy ($E_{interaction}$) calculated by Equation 4.1 versus time. It consists of two sub-figures, the left sub-figure is the interaction energy using Controller A, and the right one is the interaction energy after changing to Controller B.

For the last three experiments, three figures are shown. In the first figure, the energy-tank level of each joint versus time is shown. The second one consists of three sub-figures. Counting from left to right, the leftmost one presents the output torque of each joint versus time using the first controller (Controller A in use case mentioned in Section 3.3). The middle one presents the time series of the each joint torque output by the loop controller directly after changing to the second controller (Controller B in use case mentioned in Section 3.3). And the rightmost figure, presents the the time series of the each joint torque output by the port-Hamiltonian Energy Tank (i.e. sent to the robot) after changing to the Controller B The third figure shows the interaction energy, also consists of two sub-figures. The left sub-figure is the interaction energy using Controller A, and the right one is the interaction energy after changing to Controller B.

### 5.2.1 Experiment 1: Without port-Hamiltonian energy tank

**Result**

Following are the results for Experiment 1 mentioned in Section 5.1.

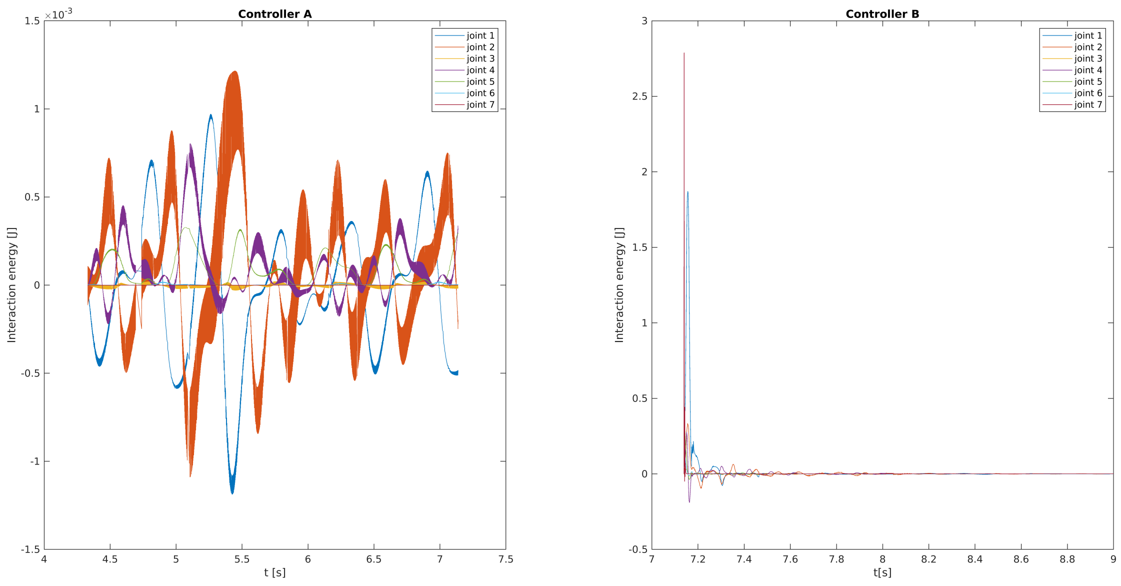


**Figure 5.2:** The interaction energy of each joint versus time without port-Hamiltonian energy tank working.

### Discussion

A zoom in version of Figure 5.2 will be shown in Section 5.2.2 to make a clear comparison, showing the effect of port-Hamiltonian energy tank from the perspective of energy.

### 5.2.2 Experiment 2: With port-Hamiltonian energy tank, $E_{max} = 5J$

### Result

Following are the results for Experiment 2 mentioned in Section 5.1.

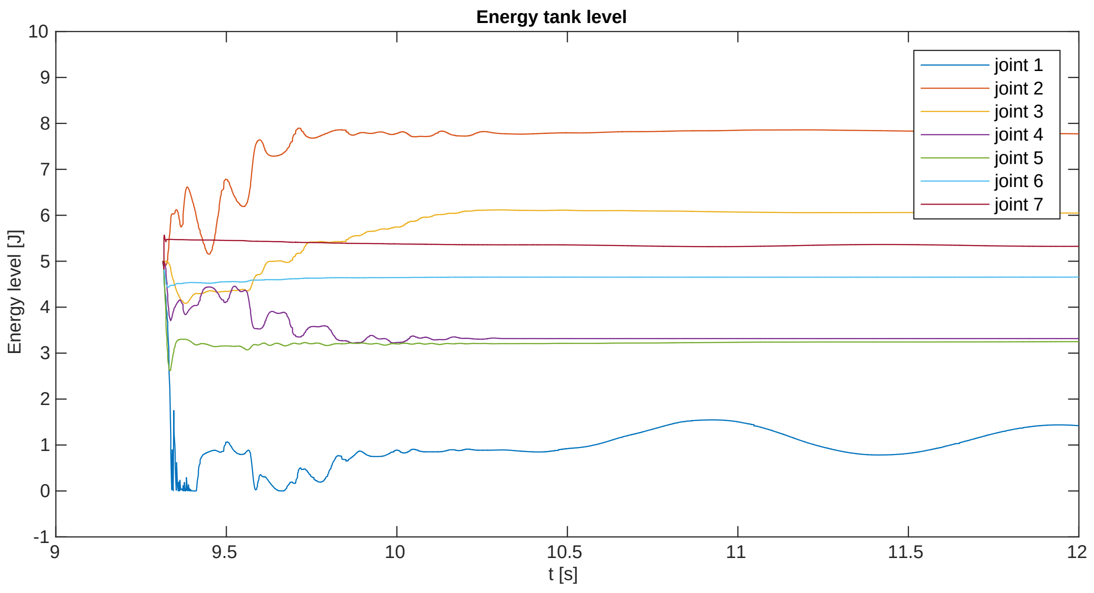


**Figure 5.3:** The energy-tank level of each joint versus time with $E_{max} = 5J$

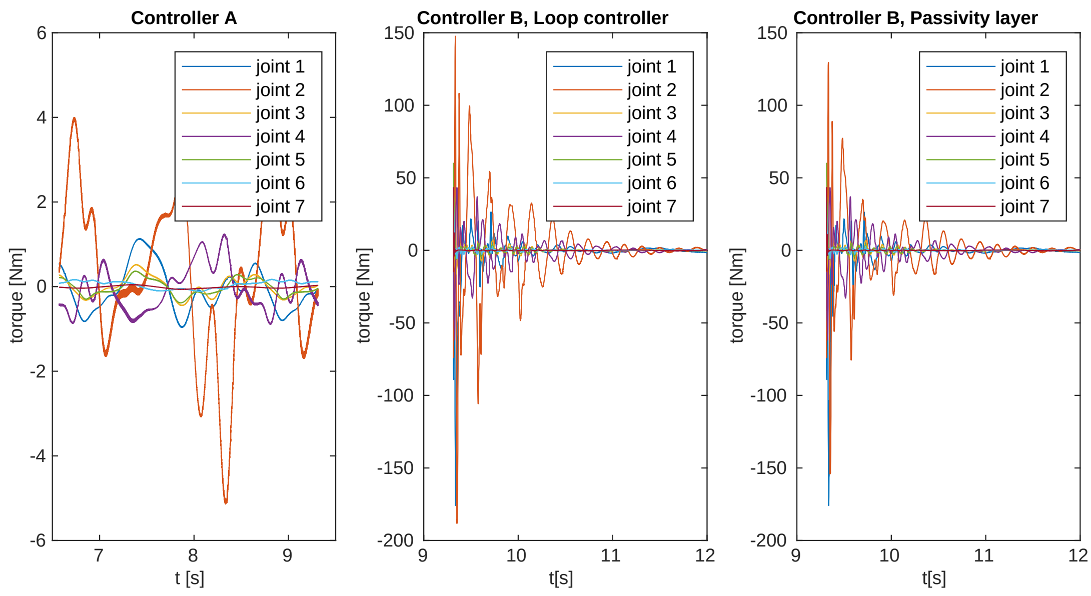


**Figure 5.4:** Time series of the robot's joint torque of the first experiment ($E_{max} = 5J$). Counting from left to right, the leftmost one is the joint torque versus time using Controller A. The middle one is the time series of the each joint torque outputting by the loop controller directly after changing to Controller B. The rightmost sub-figure, presents the the time series of the each joint torque output by the port-Hamiltonian Energy Tank (i.e. sent to the robot) using the Controller B.

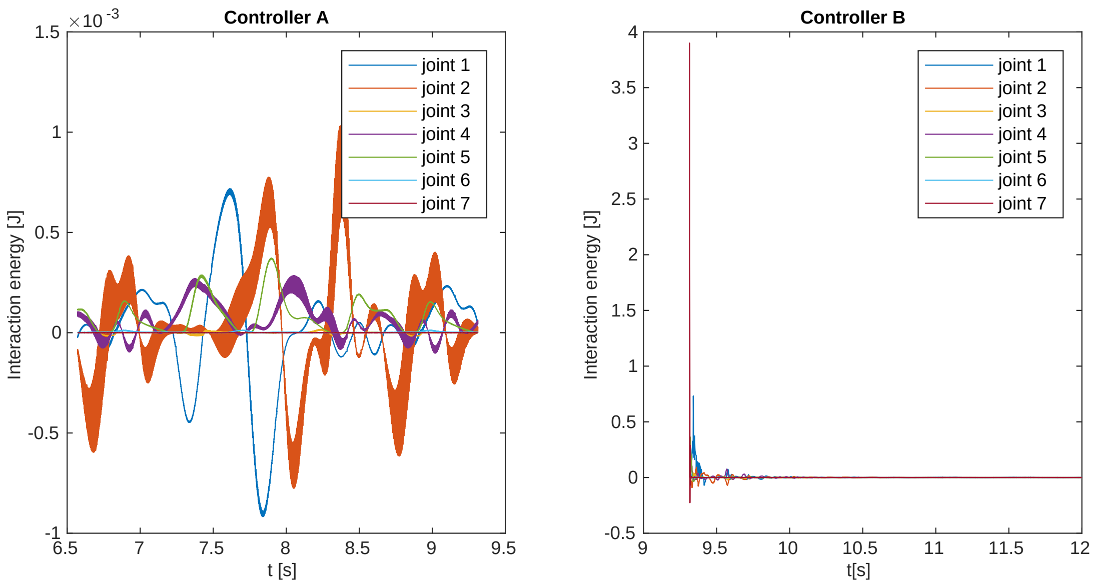


**Figure 5.5:** The interaction energy of each joint versus time with $E_{max} = 5J$. The left one is the interaction energy using Controller A, and the right one is the interaction energy after changing to Controller B.

## Discussion

For Experiment 2, as shown in Figure 5.4, comparing the first and second sub-figures, a big bump of the torques output by the Loop control layer appears, especially the torque of joint

2. And in Figure 5.3, a sudden change of energy tank level also appears at the beginning of coordination action (the lines appear in figure when the coordination action starts). This phenomenon proofs the concept argued in Section 2.3.2, that is, using the energy information, the impact of the coordination action on the system-level properties can be observed.

With the help of energy awareness, it's possible for the engineer to enhance system behaviour and make better decisions to accomplish tasks effectively and safely (Brodskiy, 2014) as mentioned in Section 1.1. And this can be proved by comparing the second and third sub-figures. Observing Figure 5.3, it can be found that the energy-tank level of joint 2 is the biggest one among all the energy-tank levels, which means the energy level in the tank of joint 2 far exceeds the maximum limit ($E_{max} = 5J$). By analysing Equation 4.4 and Equation 2.11, the conclusion can be drawn that, the more energy in energy tank ($E_{tank}$) exceeds the maximum value ($E_{max}$), the stronger the attenuating of the Loop controller outputting torques. Hence, the torque of joint 2 outputted by the port-Hamiltonian Energy Tank is much smaller than the one outputted by the Loop controller. In other words, the port-Hamiltonian Energy Tank puts a restriction on the Loop controller outputting torque, just like a filter.

In Figure 5.5, the interaction energy of each joints is shown, to evaluate the experiment result on the view of energy. However, due to the size of the figure, it cannot be seen clearly. Hence, a zoom in version of the interaction energy after changing to Controller B is applied and compares to the zoom in version of the corresponding one in Figure 5.2. Then also separate each joints and indicate them separately as shown in Figure 5.6. It can be seen that bump of interaction energy of joint 2 is attenuated obviously at the beginning of coordination action.

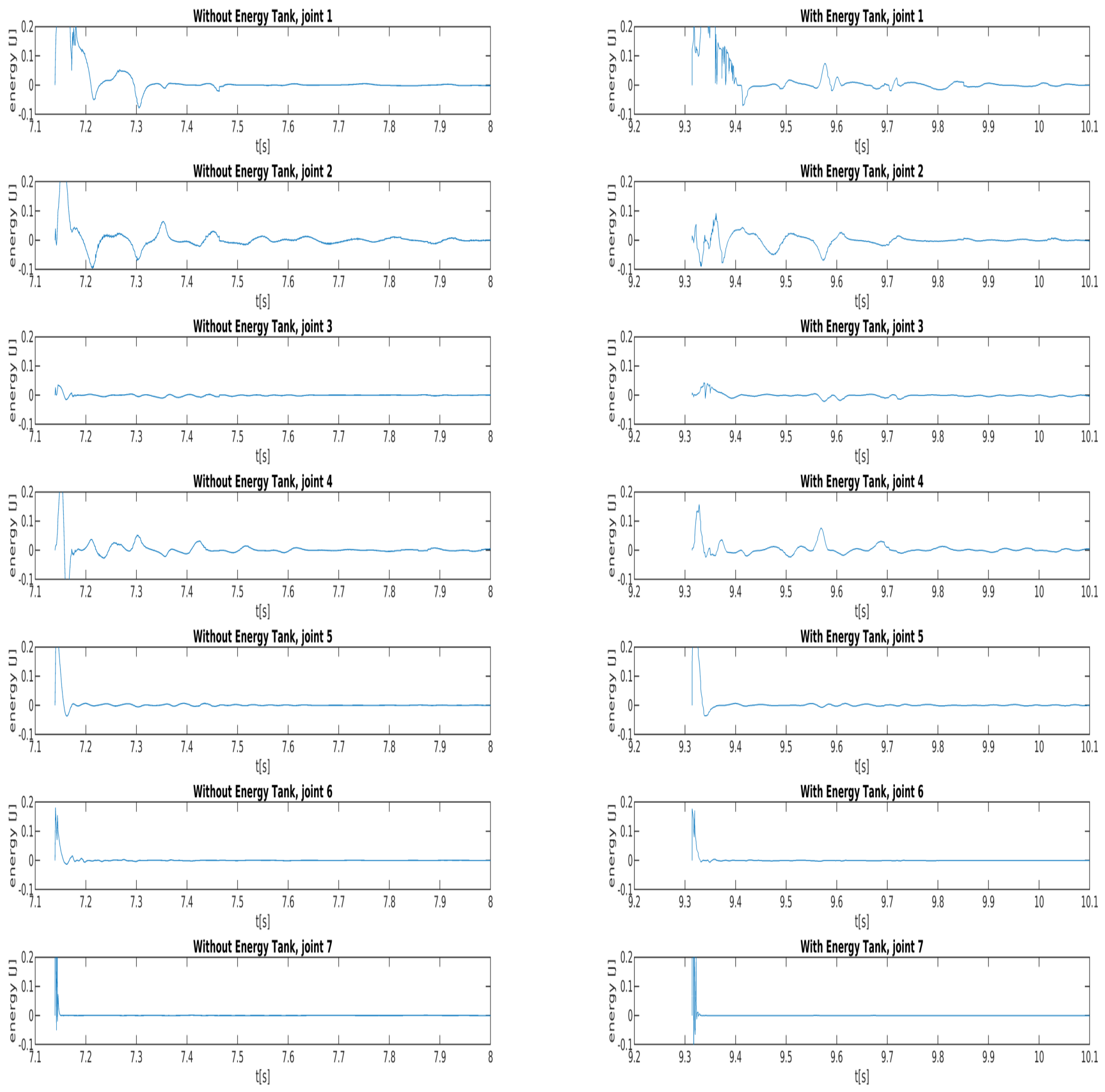


**Figure 5.6:** Compare the interaction energy of each joints after changing to Controller B, between Experiment 1 and Experiment 2. The left sub-figures are for Experiment 1, i.e. without port-Hamiltonian Energy Tank. The right sub-figures are for Experiment 2, i.e. with port-Hamiltonian Energy Tank, $E_{max} = 5J$.

### 5.2.3 Experiment 3: With port-Hamiltonian energy tank, $E_{max} = 4J$

#### Result

Following are the results for Experiment 3 mentioned in Section 5.1.

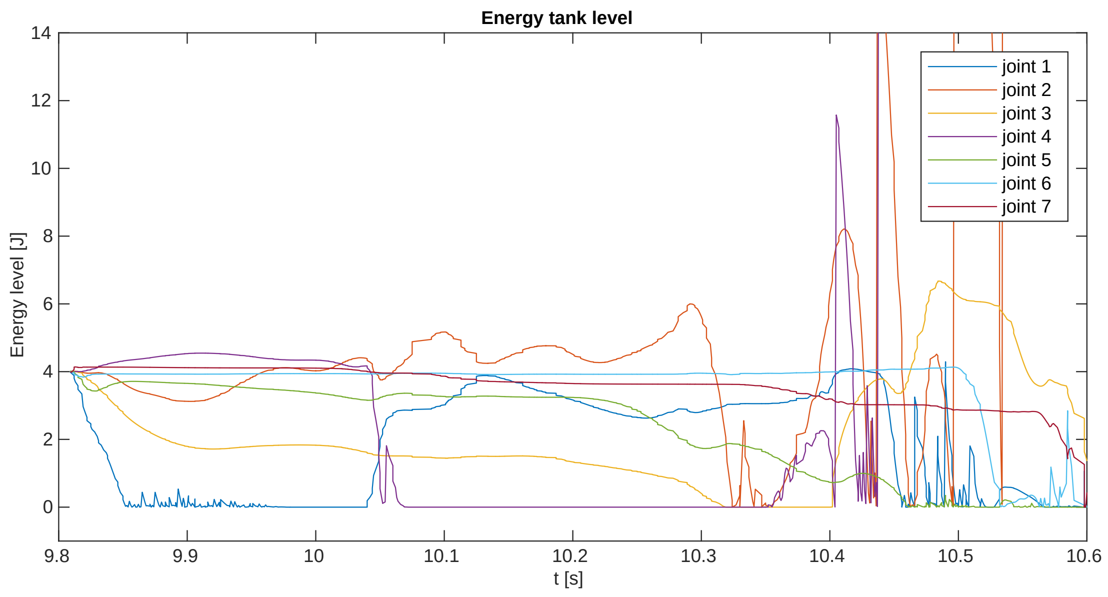


**Figure 5.7:** The energy-tank level of each joint versus time with $E_{max} = 4J$

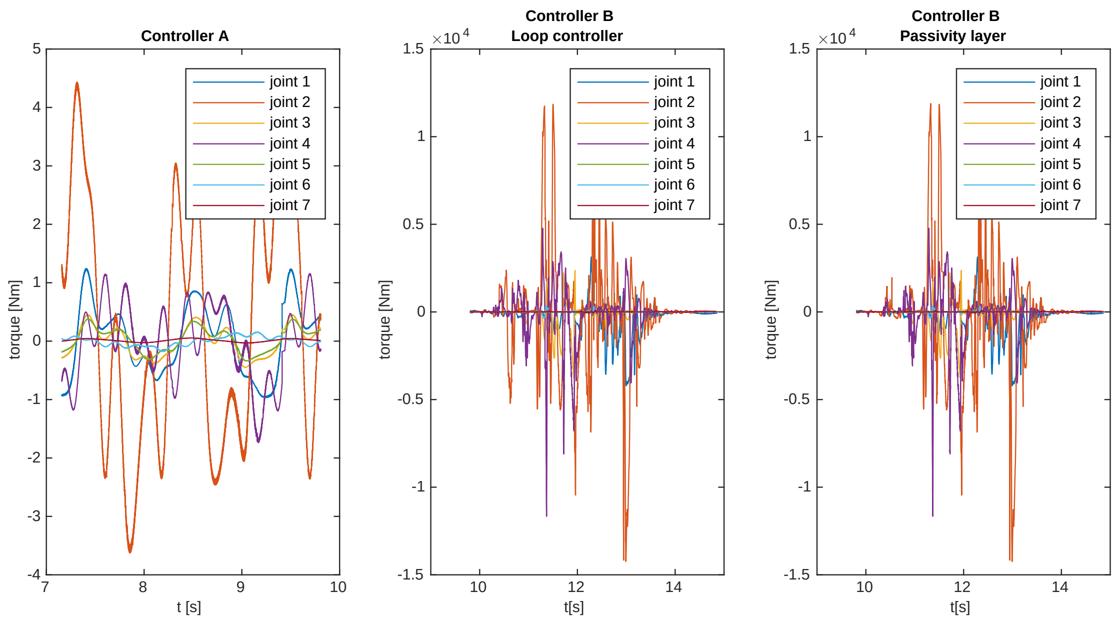


**Figure 5.8:** Time series of the robot's joint torque of the second experiment ($E_{max} = 4J$). Counting from left to right, the leftmost one is the joint torque versus time using Controller A. The middle one is the time series of the each joint torque output by the loop controller directly after changing to Controller B. The rightmost sub-figure, presents the the time series of the each joint torque output by the port-Hamiltonian Energy Tank (i.e. sent to the robot) using the Controller B.

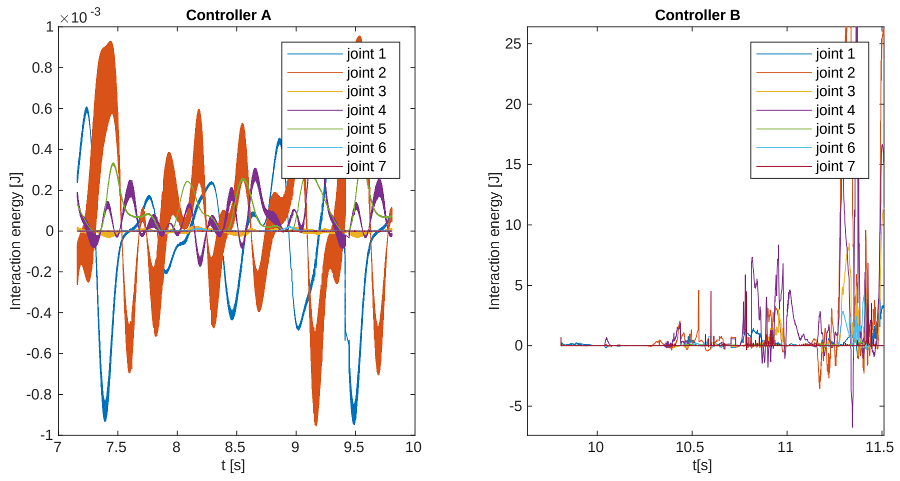


**Figure 5.9:** The interaction energy of each joint versus time with $E_{max} = 4J$. The left one is the interaction energy using Controller A, and the right one is the interaction energy after changing to Controller B.

## Discussion

For Experiment 3, as shown in Figure 5.7, the coordination action produces a great amount of internal energy to the energy tank after 10.4s, making the robot manipulator out of control. This is because the $E_{max}$ is too small, letting the port-Hamiltonian Energy Tank put too much restriction on the Loop controller outputting torque in the beginning of the coordination action. By zooming in the last two sub-figures in Figure 5.8 as shown in Figure 5.10, it can be found that although when the port-Hamiltonian Energy Tank working (in the beginning of the two figures), the torques can be limited to a smaller range, once the port-Hamiltonian Energy Tank stop its functionality, the torques will be out of control due to the big amount of error between setpoint position and current robot joint position. And also observing the right sub-figure in Figure 5.9, the interaction energy is small when the port-Hamiltonian Energy Tank working, but once the port-Hamiltonian Energy Tank stop its functionality, the bump of the interaction energy appears.

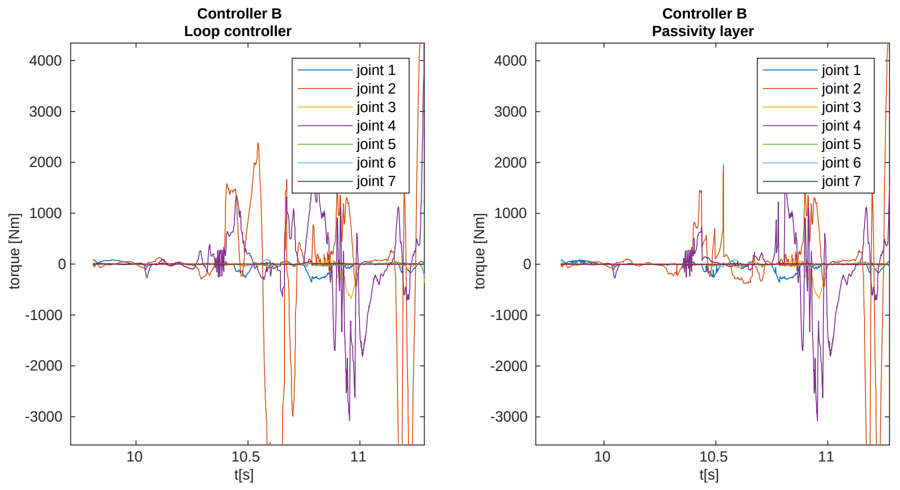


**Figure 5.10:** Zoom in the last two sub-figures in Figure 5.8.

### 5.2.4 Experiment 4: With port-Hamiltonian energy tank, $E_{max} = 6J$

#### Result

Following are the results for Experiment 4 mentioned in Section 5.1.

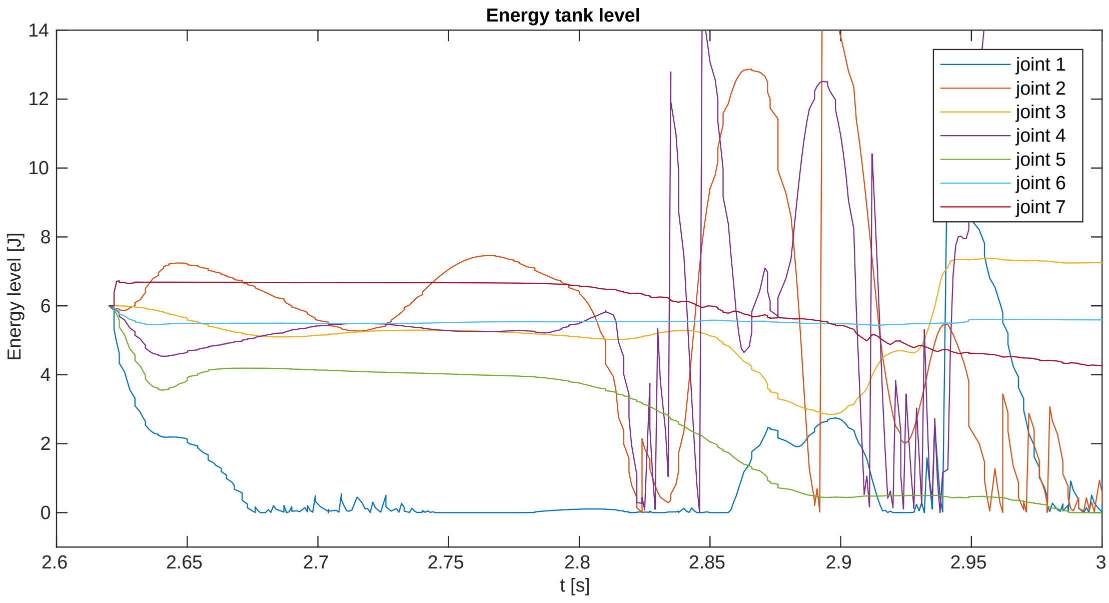


**Figure 5.11:** The energy-tank level of each joint versus time with $E_{max} = 6J$

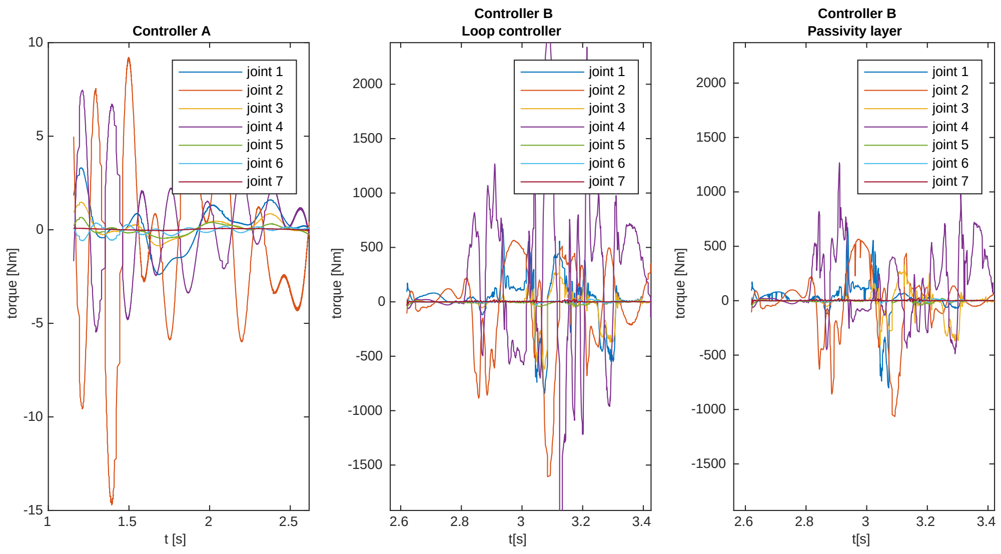


**Figure 5.12:** Time series of the robot's joint torque of the second experiment ($E_{max} = 6J$). Counting from left to right, the leftmost one is the joint torque versus time using Controller A. The middle one is the time series of the each joint torque output by the loop controller directly after changing to Controller B. The rightmost sub-figure, presents the the time series of the each joint torque output by the port-Hamiltonian Energy Tank (i.e. sent to the robot) using the Controller B.

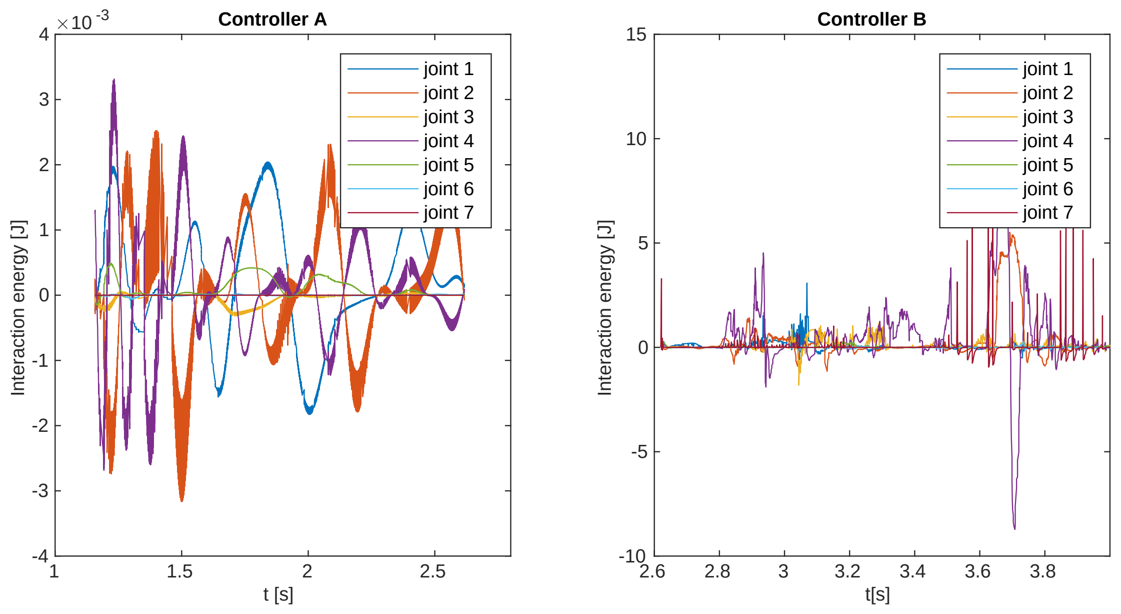


**Figure 5.13:** The interaction energy of each joint versus time with $E_{max} = 6J$. The left one is the interaction energy using Controller A, and the right one is the interaction energy after changing to Controller B.

### Discussion

For Experiment 4, as shown in Figure 5.11, the coordination action produces a great amount of internal energy to the energy tank after 2.83s, making the robot manipulator out of control. This is because the $E_{max}$ is too big, allowing too much energy in the energy tank, weaken the effect of limiting the bump of the torque output. As shown in Figure 5.13, in the beginning of the right sub-figure, even when the port-Hamiltonian Energy Tank working, there is still a peak of the interaction energy, which means the limitation of energy is not enough.

# 6 Conclusions and Recommendations

## 6.1 Conclusions

The project presented in this thesis has focused on integrating energy awareness into the control software when dealing with the configuration and coordination actions. In Section 1.3.3, a series of research requirements are presented. In this section, the main contributions of this thesis are illustrated in the order of the requirements listed in Section 1.3.3.

1. The thesis should be able to acknowledge the impact of run-time coordination and configuration of the loop control software in the behaviour of the robot.

   First, the robot dynamics will be influenced. For example, the sudden energy changes signify the sudden velocity and/or forces changes (such as the sudden torques changes after changing to Controller B in the experiments of Chapter 5), which leads to dangerous motion of the robot. And the system-level properties are degraded, such as energy-consistency, system stability and safety as analysed in Section 3.1.

2. A series of requirements should be proposed in the design pattern to preserve the system-level properties, such as energy consistency.

   In order to let the design pattern preserve the system-level properties, it is important to integrate energy awareness into control software. The control software and the hardware (e.g. robot) need to supply suitable control interfaces as mentioned in Section 3.2 to let the energy-aware component (the Passivity layer for configuration and the port-Hamiltonian energy tank for coordination) plug into the system. And after plugging in the energy-aware component, the energy communication of the system should follow the energy-communication standard in Section 3.2.1.

3. The thesis should propose the method on how to implement energy-awareness in coordination and configuration action of the loop controller.

   The design choice can refer to the design pattern in Chapter 4. The design pattern for the energy-aware configuration and coordination both applies the control interface and energy-communication standard mentioned above, and consists of 4 components, i.e. agent, sequence controller, loop controller and energy-aware component. The agent handles the configuration and coordination actions by altering the behaviour of the other 3 components. The sequence controller sends the setpoints to the loop controller directly. The loop controller runs the control law that steers the physical system to follow the setpoint value. Last, the energy-aware component integrates energy awareness into the control software, and tries to reduce the undesired effect of configuration and coordination actions by limiting the energy supplied by the control software to robot, then the sudden interaction energy change can be attenuated. As the experiment 2 ($E_{max} = 5J$) result shown in Chapter 5, the output torques are attenuated when the energy level in the tank ($E_{tank}$) exceeds the upper limit, then the system-level properties like energy consistency can be preserved.

4. The assessment of the design pattern should be done in this thesis, to analyse the benefits and drawbacks of energy-aware configuration and coordination.

   Advantages: It's possible to acknowledge the consequences in robot behavior from an energy point of view when doing configuration and coordination actions, allowing the system to have much more information available to make a better decision; Improving the system-level properties if proper maximum energy level of energy tank is applied.

Disadvantages: The control algorithm may become more complex; Suitable physical pair (such as $\boldsymbol{e}$ and $\boldsymbol{f}$ in the same coordinate frame) should be available to the control interface, and the causality, timeliness are also the constraints; System performance depends on accurate energy budget for the energy-aware component (the Passivity layer for configuration and the port-Hamiltonian energy tank for coordination). As shown in Chapter 5, if the $E_{max}$ is appropriate, the system-level properties such as bumpless transfer for the output torques is possible to realize. However, if the $E_{max}$ is too small, when the passivity layer stop working, the robot manipulator may be out of control. And if the $E_{max}$ too high, allowing too much energy in the energy tank, the effect of limiting the bump of the energy is weakened.

## 6.2 Recommendations

Since in this project, the energy-aware configuration and coordination designs have been implemented to serve only as a proof of concept, most of the recommendations for future work are focused on the implementation and improvement. The recommendations refer to the requirements list in Section 3.4.

- In the future work, more use cases can be applied to increase the functions of the design pattern. For energy-aware configuration, only refer to van Teeffelen (2018)'s work. And for energy-aware coordination, only design the conceptual implementation and do the experiment under the use case in Section 3.3.
- The bumpy transfer has not been completely resolved in the experiments, and the value of $E_{max}$ is chose empirically. Hence the design pattern of the port-Hamiltonian energy tank may need an improvement in the future.
- In this thesis, only simulation is used for evaluation, which is not enough. In the future work, implementation on a real robot platform is necessary.

# A Detailed design of Energy estimator for configuration

Like the Passivity layer presented in Section 4.1.3, the Energy estimator is also inspired by the work of Hobert (2020). Accordingly, it consists of real-time models of the Loop control layer, the Robot, and the Environment to determine model-based energy budgets for the Passivity layer's energy tank. These energy budgets are an estimation of the amount of energy that the Loop control layer will require for the next sample time.

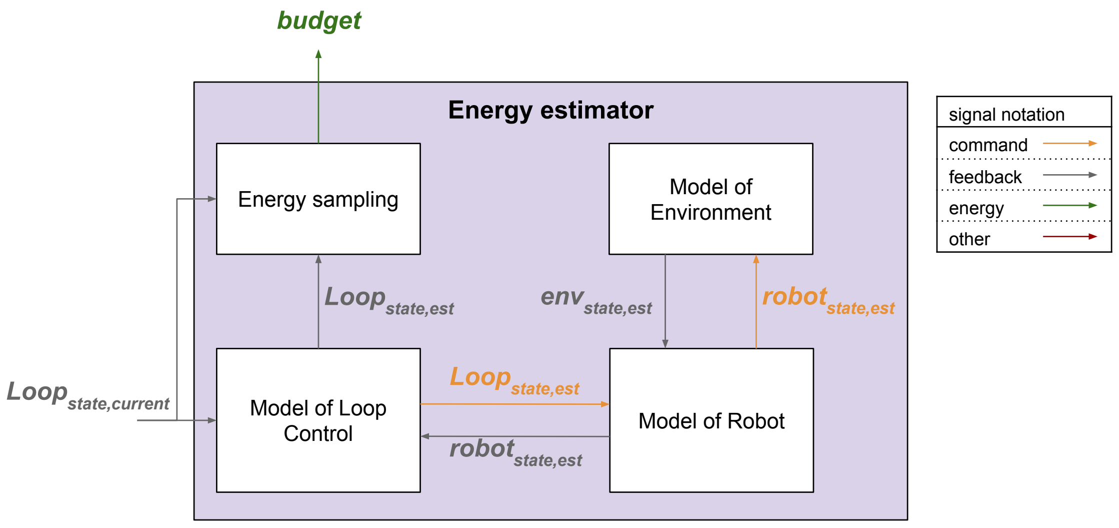


**Figure A.1:** Energy estimator design inspired by Hobert (2020).

To be easier to understand, the Figure A.1 will be combine with Equation 2.4 as an example to illustrate. Following computation will be processed in Energy sampling block:

$$
\begin{aligned}
&\Delta H(k+1) = \boldsymbol{e}^T(k+1)(\boldsymbol{q}((k+1)T) - \boldsymbol{q}(kT)) \\
&\Delta H(k+1) = E_{budget} \\
&\boldsymbol{e}(k+1) = Loop_{state,current\_}1 \\
&\boldsymbol{q}((k+1)T) = Loop_{state,est} \\
&\boldsymbol{q}(kT) = Loop_{state,current\_}2 \\
&\therefore\ E_{budget} = Loop_{state,current\_}1(Loop_{state,est} - Loop_{state,current\_}2)
\end{aligned}
\tag{A.1}
$$

First, the loop control layer (e.g. Cartesian impedance control) computes the commanded forces ($Loop_{state,current\_}1$) by measured positions and setpoints. Then, using the *Model of Loop Control*, *Model of Robot* and *Model of Environment* to estimate the next positions ($Loop_{state,est}$). Finally, the rough energy budget ($E_{budget}$) is equal to the product of commanded forces and, the difference between two positions (next positions and current positions).

However, using this way, only obtain a rough energy budget. As mentioned in Section 2.2.2, throughout the sample interval $[kT, (k+1)T]$, the value of the $\boldsymbol{f}$ is continuously changing, here only uses the position values at both ends to estimate the total change of $\boldsymbol{f}$. Brodskiy (2014) identified that inaccurate energy-budget estimations have a negative impact on system performance. Only if the sampling time $T$ is small enough, and the accuracy requirements are not so high, the error could be acceptable.

As mentioned in Section 4.1.3, it's possible to get the *Model of Loop Control*. And as clarified in Hobert (2020), the behaviour of the *Model of Loop Control* depends on the behaviour of the *Model of Robot*, the interaction between these two blocks is also necessary. Hence, the signals $Loop_{state,est}$ and $robot_{state,est}$ are necessary. However the *Model of Environment* can be omitted due to the development of *Model of Robot* without considering its environment is possible. With the help of above Models, calculating an accurate energy budget is possible (Groothuis et al., 2018). Or the energy budgets per time step can be determined by running a simultaneous simulation of the controlled system dynamics ((Brodskiy, 2014; Hobert, 2020). However, as mentioned in Section 3.2.1, it is not in the scope of this thesis to develop highly-accurate component models for estimating the energy budget.

# B ROS Computation Graph

The following graphs are the ROS computation graphs using the command:

```
$ rqt_graph
```

When the loop controller uses "minimum_effort_inverse_dynamics" controller, the ROS graph is:

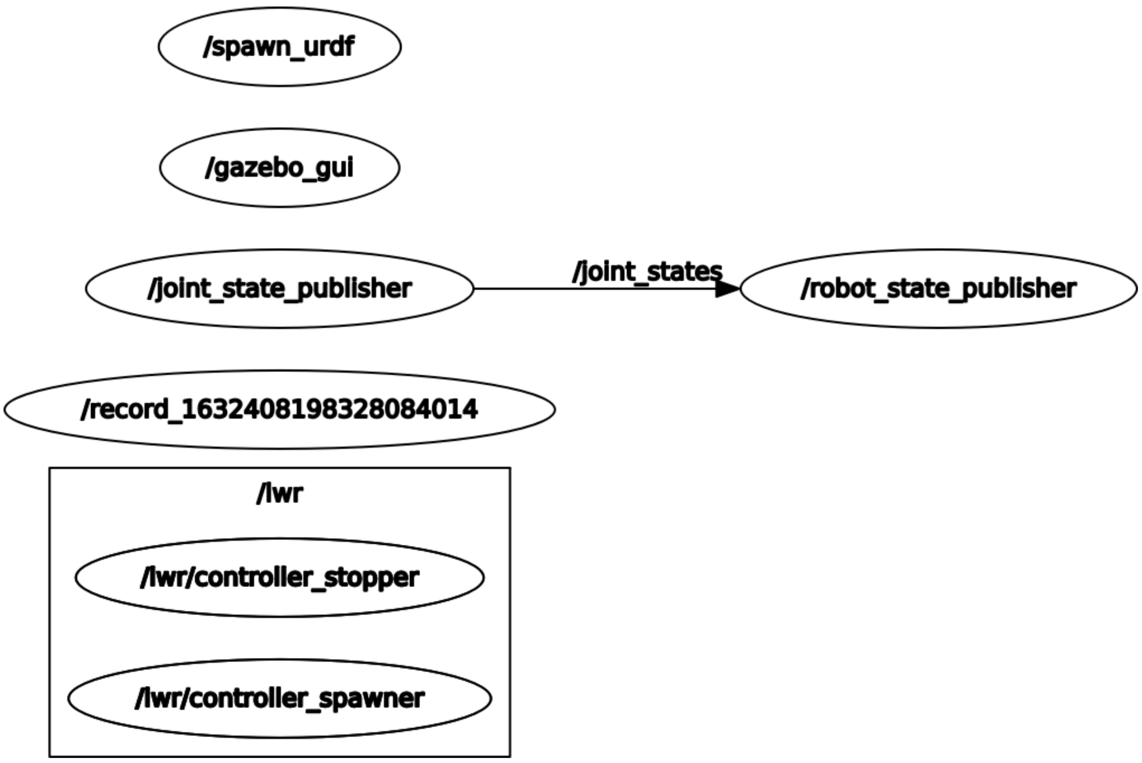


**Figure B.1:** The ROS graph when using "minimum_effort_inverse_dynamics" controller.

Then use the command:

```
$ rosservice call /lwr/controller_manager/switch_controller
"{start_controllers:␣['ecca_back_stepping_controller'],
stop_controllers:['ecca_minimum_effort_inverse_dynamics'],
strictness:␣2}"
```

The control law in loop controller switch to "backstepping_controller" controller, and the ROS graph is following:

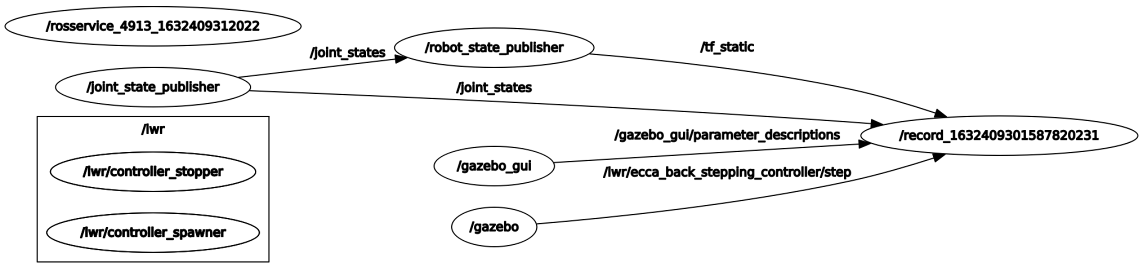


**Figure B.2:** The ROS graph when using "backstepping_controller" controller.

# C Setup

*The following corresponds to Section 5.1.*

The setup used for the experiments described in Section 5.1 is:

- Hardware: Lenovo XiaoXin-14/WL 2019, Intel(R) Core(TM) i7-8565U CPU @ 1.80GHz, RAM 20.0 GB, Window-10.
- Virtual machine software: Oracle VM VirtualBox[1] v6.1.18.
  Settings:
  - Ubuntu[2] 16.04
    *Note: the simulation is not yet functional on newer Ubuntu versions.*
  - 3 CPUs
  - 12 GB memory
  - 60 GB storage.
- Middleware: ROS[3] Kinetic
- Simulator: Gazebo[4] 7.0.0

[1]Oracle VM VirtualBox website: https://www.virtualbox.org/
[2]Ubuntu website: https://ubuntu.com/
[3]ROS website: https://www.ros.org/
[4]Gazebo website: http://gazebosim.org/

# Bibliography

Bezemer, M. M. (2013), Cyber-physical systems software development: way of working and tool suite, doi:10.3990/1.9789036518796.
https://research.utwente.nl/en/publications/cyber-physical-systems-software-development-way-of-working-and-to

Breedveld, P. (1982), Thermodynamic bond graphs and the problem of thermal inertance, *J. Franklin Inst*, pp. 15–40.

Breemen, A. v. and T. J. A. d. Vries (2000), An agent-based framework for designing multi-controller systems, in *PAAM 2000: proceedings of the Fifth International Conference on the Practical Application of Intelligent Agent and Multi Agent Technology 10th-12th April, 2000, Manchester, UK*, Practical Application Company, pp. 219–235.
https://research.utwente.nl/en/publications/an-agent-based-framework-for-designing-multi-controller-systems

Brodskiy, Y. (2014), Robust autonomy for interactive robots, doi:10.3990/1.9789036536202.
https://research.utwente.nl/en/publications/robust-autonomy-for-interactive-robots

Broenink, J. F., Y. Ni and M. A. Groothuis (2010), On model-driven design of robot software using co-simulation, in *Proceedings of SIMPAR 2010 Workshops International Conference on Simulation, Modeling, and Programming for Autonomous Robots*, TU Darmstadt, pp. 659–668.
https://research.utwente.nl/en/publications/on-model-driven-design-of-robot-software-using-co-simulation

Brugali, D. (2020), Runtime reconfiguration of robot control systems: a ROS-based case study, in *2020 Fourth IEEE International Conference on Robotic Computing (IRC)*, pp. 256–262, doi:10.1109/IRC.2020.00047.

Brugali, D., R. Capilla, R. Mirandola and C. Trubiani (2018), Model-Based Development of QoS-Aware Reconfigurable Autonomous Robotic Systems, in *2018 Second IEEE International Conference on Robotic Computing (IRC)*, pp. 129–136, doi:10.1109/IRC.2018.00027.

Brugali, D. and P. Scandurra (2009), Component-based robotic engineering (Part I) [Tutorial], **vol. 16**, no.4, pp. 84–96, ISSN 1558-223X, doi:10.1109/MRA.2009.934837, conference Name: IEEE Robotics Automation Magazine.

Bruyninckx, H., M. Klotzbücher, N. Hochgeschwender, G. Kraetzschmar, L. Gherardi and D. Brugali (2013), The BRICS component model: a model-based development paradigm for complex robotics software systems, in *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, Association for Computing Machinery, New York, NY, USA, SAC '13, pp. 1758–1764, ISBN 978-1-4503-1656-9, doi:10.1145/2480362.2480693.
https://doi.org/10.1145/2480362.2480693

Cardenas, C. A. (2017), Development of a safety-aware intrinsically passive controller for a multi-DOF manipulator, publisher: University of Twente.
https://essay.utwente.nl/73878/

Clegg, D. and R. Barker (1994), *Fast-track: a RAD approach*, Addison-Wesley Pub. Co. ; Oracle, Wokingham, England; Reading, Mass.; Berkshire, UK; Redwood Shores, CA, USA, ISBN 978-0-201-62432-8, oCLC: 31134262.

Cobos Mendez, R., K. van Teeffelen, J. de Oliveira Filho, T. de Vries, D. Dresscher, B. Driessen and J. Broenink (2020), Energy-Guided Control Stacks and Robot-Software Architectures using Model-Driven Design, Internal D2.2, University of Twente, TNO, VIRO, Enschede, The Netherlands, issue: D2.2.

https://cloud.ram.eemcs.utwente.nl/index.php/s/2BARxQtJdA5t5am

Folkertsma, G. A. and S. Stramigioli (2017), Energy in Robotics, **vol. 6**, no.3, pp. 140–210, ISSN 1935-8253, 1935-8261, doi:10.1561/2300000038.
https://www.nowpublishers.com/article/Details/ROB-038

Franken, M. C. J. (2011), Control of haptic interaction : an energy-based approach, doi:10.3990/1.9789036531894.
https://research.utwente.nl/en/publications/control-of-haptic-interaction-an-energy-based-approach

Groothuis, M. A., R. Frijns, J. Voeten and J. F. Broenink (2009), Concurrent Design of Embedded Control Software, in *Proceedings of the 3rd International Workshop on Multi-Paradigm Modeling (MPM2009)*, European Association for the Study of Science and Technology.
https://research.utwente.nl/en/publications/concurrent-design-of-embedded-control-software

Groothuis, S. S., G. A. Folkertsma and S. Stramigioli (2018), A General Approach to Achieving Stability and Safe Behavior in Distributed Robotic Architectures, *Frontiers in robotics and AI*, **vol. 5**, p. 108, ISSN 2296-9144, doi:10.3389/frobt.2018.00108, publisher: Frontiers Media S.A.
https://research.utwente.nl/en/publications/a-general-approach-to-achieving-stability-and-safe-behavior-in-di

Guiochet, J., M. Machin and H. Waeselynck (2017), Safety-critical advanced robots: A survey, *Robotics and Autonomous Systems*, **vol. 94**, pp. 43–52, ISSN 0921-8890, doi:10.1016/j.robot.2017.04.004.
https://www.sciencedirect.com/science/article/pii/S0921889016300768

Hobert, E. C. (2020), Integrating Energy Awareness into Sequence Control, publisher: University of Twente.
http://essay.utwente.nl/82739/

Paynter, H. M., P. Briggs and Massachusetts Institute of Technology (1961), *Analysis and design of engineering systems: class notes for M.I.T. course 2.751*, M.I.T. Press, Cambridge, Mass.
https://catalog.hathitrust.org/Record/005135763

Peng, Y., D. Vrancic and R. Hanus (1996), Anti-windup, bumpless, and conditioned transfer techniques for PID controllers, **vol. 16**, no.4, pp. 48–57, ISSN 1941-000X, doi:10.1109/37.526915, conference Name: IEEE Control Systems Magazine.

Raiola, G., C. A. Cardenas, T. S. Tadele, T. de Vries and S. Stramigioli (2018), Development of a Safety- and Energy-Aware Impedance Controller for Collaborative Robots, **vol. 3**, no.2, pp. 1237–1244, ISSN 2377-3766, doi:10.1109/LRA.2018.2795639, conference Name: IEEE Robotics and Automation Letters.

Rashad, R., J. B. C. Engelen and S. Stramigioli (2019), Energy Tank-Based Wrench/Impedance Control of a Fully-Actuated Hexarotor: A Geometric Port-Hamiltonian Approach, in *2019 International Conference on Robotics and Automation (ICRA)*, IEEE, pp. 6418–6424, doi:10.1109/ICRA.2019.8793939.
https://research.utwente.nl/en/publications/energy-tank-based-wrenchimpedance-control-of-a-fully-actuated-hex

Stramigioli, S., C. Secchi, A. van der Schaft and C. Fantuzzi (2005), Sampled data systems passivity and discrete port-Hamiltonian systems, **vol. 21**, no.4, pp. 574–587, ISSN 1941-0468, doi:10.1109/TRO.2004.842330, conference Name: IEEE Transactions on Robotics.

Tadele, T. S. (2014), Human-friendly robotic manipulators: safety and performance issues in controller design, doi:10.3990/1.9789036537841.
https://research.utwente.nl/en/publications/human-friendly-robotic-manipulators-safety-and-performance-issues

van Teeffelen, K. (2018), Intuitive Impedance Modulation in Haptic Control using Electromyography.
https://essay.utwente.nl/74440/

Willems, J. C. (1972), Dissipative dynamical systems part I: General theory, **vol. 45**, no.5, pp. 321–351, ISSN 1432-0673, doi:10.1007/BF00276493.
https://doi.org/10.1007/BF00276493