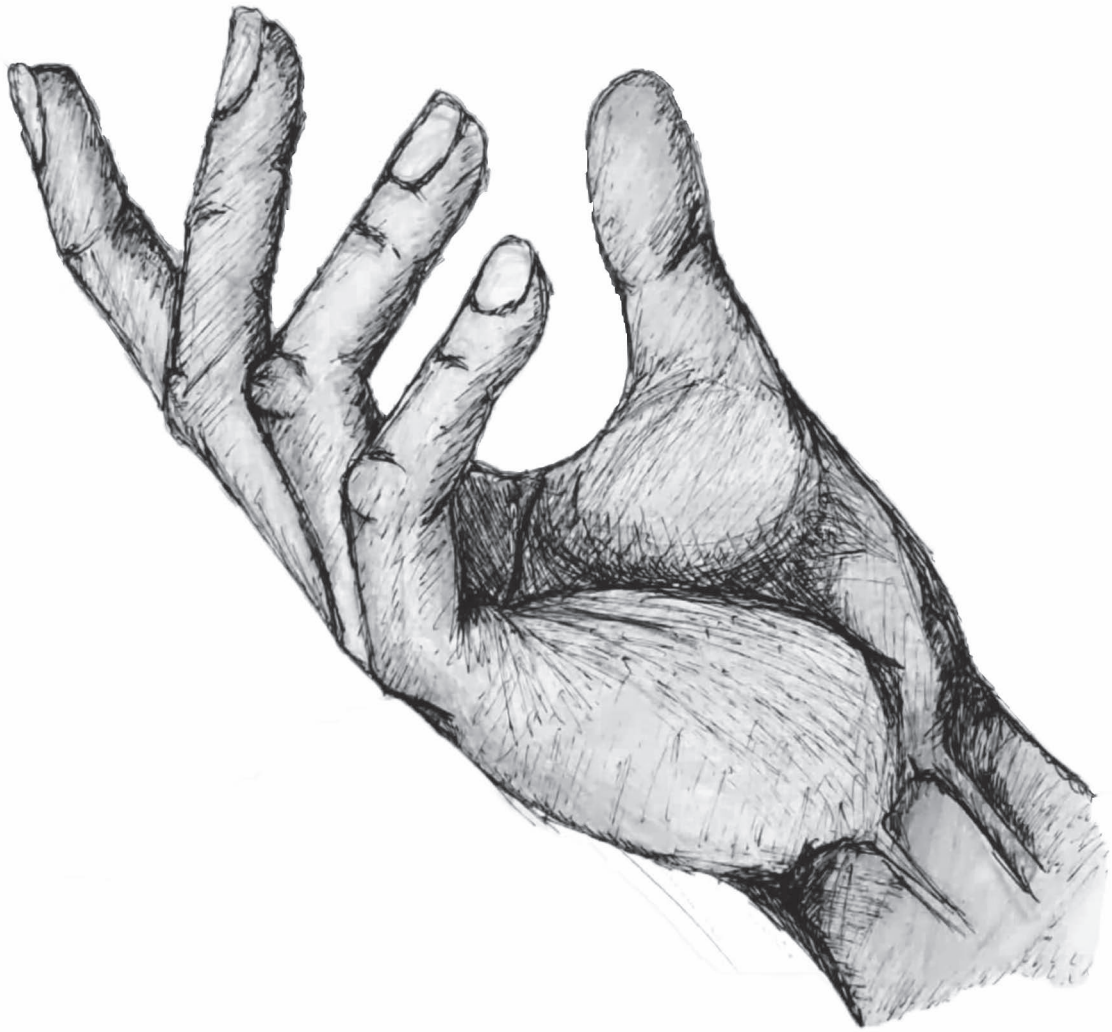
# Active inference for adaptive and fault tolerant control

## An application to robot manipulators

Corrado Pezzato

Master of Science Thesis



TU Delft Delft University of Technology

Delft Center for Systems and Control

# Active inference for adaptive and fault tolerant control

## An application to robot manipulators

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

Corrado Pezzato

July 10, 2019

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology


Delft University of Technology
Department of
Delft Center for Systems and Control (dcsc)

The undersigned hereby certify that they have read and recommend to the Faculty of Mechanical, Maritime and Materials Engineering (3mE) for acceptance a thesis entitled

Active inference for adaptive and fault tolerant control

by

Corrado Pezzato

in partial fulfillment of the requirements for the degree of

Master of Science Systems and Control

Dated: July 10, 2019

Supervisor(s):

Dr. R. Ferrari

Dr. ir. C. Hernández Corbato

Reader(s):

Prof. dr. ir. M. Wisse

Dr. M. Kok

# Abstract

Dealing with inherently unmodeled dynamics and large parameter variations or faults, is a challenging task while controlling robot manipulators. Classical control techniques cannot usually provide satisfactory responses, and often external supervision systems have to be designed to handle the faults. Recent research has shown that active inference, a unifying neuroscientific theory of the brain, bares the potential of intrinsically coping with strong uncertainties in the system, mimicking the adaptability capabilities of humans. However, the current state-of-the-art regarding active inference in robotics is very narrow and limited. This thesis presents a novel active inference controller as a general adaptive fault tolerant solution for control of robot manipulators. The goal of this work is threefold. First, we demonstrate the applicability of active inference in robotics, deriving a control scheme which is computationally efficient and with high performance. Second, we verify the claimed adaptability properties of active inference against a model reference adaptive controller, in a simulated on-line pick and place task with a 7 degrees-of-freedom robot arm. Third, we propose a method to exploit the controller's structure to perform fault detection, isolation and recovery, without the use of external supervision systems. This work showed that not only active inference is applicable to robotics, but it also outperforms the model reference adaptive controller, and it allows to efficiently deal with sensory faults. This thesis represents a leap forward with respect to the current state-of-the-art of active inference for robotics, and it lays the foundations for further research in this direction.

# Table of Contents

# List of Figures

# List of Tables

# Glossary

## List of Acronyms

| | |
|---|---|
| **AIC** | Active inference controller |
| **DOF** | Degrees of freedom |
| **FAR** | False alarm rate |
| **FDI** | Fault detection and isolation |
| **FEP** | Free-energy principle |
| **G-Density** | Generative density |
| **GPR** | Gaussian process regression |
| **LWPR** | Locally weighted projection regression |
| **MRAC** | Model reference adaptive controller |
| **NN** | Neural networks |
| **R-Density** | Recognition density |
| **ROS** | Robot operating system |

## List of Symbols

| | |
|---|---|
| $\boldsymbol{\mu}$ | Current belief about the states $\in \mathbb{R}^n$ where $n$ is the number of states. Mean of a Gaussian distribution, expectation of the states $\boldsymbol{x}$ |
| $\boldsymbol{\varepsilon_\mu}$ | Model prediction errors |
| $\boldsymbol{\varepsilon_y}$ | Sensory prediction errors |
| $\kappa_\mu$ | Learning rate for the state estimation. Tuning parameter of the active inference controller |

| | |
|---|---|
| $\kappa_a$ | Learning rate for the control actions update. Tuning parameter of the active inference controller |
| $\Sigma_\mu$ | Covariance matrix for the internal beliefs |
| $\Sigma_y$ | Covariance matrix for the sensory input |
| $\boldsymbol{u}$ | Control actions, such as applied torques or forces. In the robot arm case $\boldsymbol{u} \in \mathbb{R}^n$ is a vector of torques, where $n$ is the number of joints |
| $\boldsymbol{x}$ | States of the environment which describe well-defined physical entities. The vector $\boldsymbol{x} \in \mathbb{R}^n$ with $n$ number of states |
| $\boldsymbol{y}$ | Vector containing the sensory input $\boldsymbol{y} \in \mathbb{R}^m$ with $m$ number of sensors |
| $\mathcal{F}$ | Free-energy $\in \mathbb{R}^+$. The free-energy is an upper bound on sensory surprisal and it is a scalar value defined as a weighted sum of squared prediction errors |
| $P_\mu$ | Precision matrix for the internal state estimate. Inverse of the covariance $\Sigma_\mu$ |
| $P_y$ | Precision matrix for the sensory input. Inverse of the covariance $\Sigma_y$ |

# Acknowledgements

First, I would like to thank my supervisors, Dr. Carlos Hernández Corbato and Dr. Riccardo Ferrari. During the past year you pushed me to achieve the best out of my work, and you always assisted me with the right advises to succeed. I am grateful to you for making me discover the wonderful world that research is, giving me the freedom to develop my own ideas.

Then, I would like to thank my parents, and all the people who always believed in me and trusted my choices. Mom, Dad, I will never be able to express all my gratitude to you. Even though it was hard sometimes, you always wanted the best for me, and you always let me follow my dreams.

Finally, I want to thank all the amazing friends I made during these two years in Delft. And in particular my flatmates, thank you for making every day just awesome. I really value the memories I have with all of you, it has been a legendary adventure.

Delft, University of Technology — Corrado Pezzato
July 10, 2019

"If you're not having fun, you're not learning. There's a pleasure in finding things out."

— *Richard Feynman*

# Chapter 1

# Introduction

*This introductory chapter focuses on the motivations behind this research, posing the three fundamental questions that this thesis addresses. After briefly reporting the current state-of-the-art in the research area of active inference for robotics, the main contributions of this work are highlighted. The chapter is then concluded with the outline of the document.*

## 1-1 Research motivations

One of the most exciting and at the same time challenging topics in robotics, is the achievement of human-level adaptability in case of unexpected situations and uncertainties. In a dynamic world which is hard to model accurately, the performance of classical control architectures can decay significantly. For robot manipulators, in particular, obtaining a physical model for reliable control is a hard task. For this reason, research focused on the use of machine learning to obtain accurate inverse dynamic models [48, 34]. In general, learning models using Neural Networks (NN) requires experts for defining the best topology for a particular problem [31], and it mainly involves three steps: input/output data collection, heuristic architecture design of the NN, and parameter learning. The resulting structure is usually hard to generalise due to the required balance between overfitting and accuracy [41, 26]. Even though it is possible to exploit the physical knowledge of the system to simplify and improve the learning performance [27], the need of a large amount of training data and the need of several iterations for learning, remains. Other approaches as [21, 20] focused on the learning of task specific inverse dynamics which, even if performing well for the learnt task, cannot be easily generalised to a plurality of motions.

Having said that, in recent years the state-of-the art for robot control in unstructured environments made giant leaps forward, but traditional solutions are still far from what the human capabilities are, in terms of sensorimotor control, inference and adaptability. To try to bridge this gap, bio-inspired control architectures appear to be promising. In the past years, various researchers focused on the understanding of the processes underlining human intelligence and decision making. One of the most influential theories which tries to explain how information

is processed by the human brain, is the so called *active inference*. This theory, based on the *free-energy principle*, was first presented by Karl Friston in the early 2000s [13, 14]. Friston's work represents an influential unifying theory of the brain, with an outstanding explanatory power regarding brain's cognition and motor control functions.

The free-energy principle and active inference relies on Bayesian inference and gradient descent schemes to perform prediction error minimisation in a biologically plausible manner. The possibility of encoding in mathematical terms the biological processes of the brain, and ultimately the intelligence itself, is undoubtedly an interesting research area. Active inference is relevant from a control perspective, not only because of its explanatory power regarding the brain's dynamics, but also because it proposes a structure which is suitable for robot control.

In a robotic context, active inference could provide a general approach which avoids the use of accurate inverse models, removing the need of learning them through an articulated NN. Besides, past work such as [25], highlighted the adaptability properties of the framework proposed by Friston, for state estimation of a robot arm. The free-energy and active inference, intrinsically bare the potential of dealing with unexpected situations. In a sense, active inference qualifies, at least at first glance, as a potential intrinsically fault tolerant controller capable of dealing with sensory faults. This suggests that the framework could be implemented to simplify the control architecture with respect to other conventional techniques, avoiding the use of external supervision systems for fault detection and recovery. The complexity of the brain's processes, however, reflects on the convolution of the current literature about the topic. In this thesis we focus on the most relevant aspects of active inference, and we reformulate them in control engineering terms for their application to robotics.

While active inference has been extensively studied in neuroscience, the application to robotics has been marginally explored. This is mainly due to the neuroscientific formulation of active inference, which is far from the robotics community. The application of this paradigm for robot control is very limited, and at the time of writing, there is no clear evidence of the benefits that active inference can bring in this scenario[1]. Besides, the claimed adaptability properties of the theory in case of noise and unmodeled dynamics, remain to be tested against other similar approaches. To conclude, the research motivations brought to the formulation of the three questions that this work addresses:

- ***Is active inference suitable for robot control?***
  Answering this question on the basis of concrete evidences, was the first motivation of this research. Active inference for robotics is as promising as it is intricate. The fact that, at the time of writing, there was no on-line application for robot control in the literature, indicated a gap that this work proposes to reduce. The novel solution derived in this thesis not only answered the question above, but it also motivated further exploration of the characteristics of the free-energy principle in terms of its adaptive properties. This brought to formulate the second and third research questions.

[1] Note that a very recent and still unpublished research, subsequent to this work, formulated a working active inference controller, in a similar fashion to what presented in this thesis. More details about this are given on page 5.

- ***What are the adaptability properties of active inference compared to other adaptive solutions?***
  Active inference tries to capture the adaptability of the brain in terms of state inference and computation of the control actions, based on a set of sensory input. To date, however, there is on actual comparison of this framework with other adaptive control techniques to quantify these properties.

- ***Can the free-energy principle structure be exploited to obtain an intrinsically fault tolerant controller?***
  Past work highlighted how active inference is also able to deal with missing information from redundant sensors, during the state inference process. This suggested to analyse if the free-energy structure can be used to obtain intrinsically fault tolerant schemes, which do not require additional supervision systems.

## 1-2 The state-of-the-art of active inference in robotics

The main idea at the basis of Friston's neuroscientific theory, is that the brain's cognition and motor control functions could be described in terms of energy minimization. It is supposed [11] that we, as humans, have a set of sensory data and a specific internal model to characterize how the sensory data could have possibly been generated. With these models, we approximate the dynamics of the real generative processes of the external world. Then, given these internal models, the causes of sensory data are inferred. Usually, the environment acts on humans to produce sensory impression, and humans can act on the environment to change it. In this view, the motor control of human body can be considered as the fulfillment of a prior expectation about proprioceptive sensations [17]. The fact that this theory tries to capture the adaptive nature of humans' sensorimotor control, suggested the use of the free-energy principle to obtain robust control schemes for robotics. In practice, in a robotic application, the whole sensory input available contributes in understanding the most probable states of the robot through the minimization of the free-energy as cost function. The same minimisation problem provides the control actions to the motors in order to fulfill a prior expectation about a specific desired goal. The use of active inference for robot control allows state estimation and control only using sensory data and internal models for these data.

As mentioned before, while developing this project the state-of-the-art for active inference applied to robotics was narrow and it did not provide enough insights to understand whether or not this brain inspired theory was suitable for real-time robotic applications. Having said that, it is fundamental to carefully analyse the current solutions and to identify the main limitations of the proposed approaches.

### Open-loop active inference

The work presented in [38] by Léo Pio-Lopez et al., is the very first attempt to implement active inference for robot control. In here, the authors simulated the behaviour of a PR2 robot in a reaching task using active inference. The main assumptions taken in this work, are related to the generative processes and models. The generative process, which should corresponds to the true dynamics of the robot, was replaced with a dynamical model defined

by the authors. Besides, the authors assumed that the robot knows its forward model and that it can retrieve the true position of its end effector. Usually, this relation has either to be encoded or estimated, bringing into the control loop noise and unmodeled dynamics. Then, the control law for the robot arm was based on a proportional-integral action using a feedback error computed in Cartesian space. The authors also included a correction term to avoid the occurrence of singularities while controlling the robot. Due to the assumptions taken, this control scheme represented an off-line open-loop implementation of active inference for control in Cartesian space. There is, indeed, no actual feedback from the simulated robot, and all the necessary signals are computed based on the encoded generative process.

This implementation basically provides open-loop position commands to the joints, and it relies on the internal PID controllers of the robot arm to account for the motion of the joints and the compensation of gravity. During the reaching task, the response of the robot was highly affected by the noise in the sensory data, failing to reach the target when all the sensors where noisy. Besides, the computational complexity of the proposed solution precluded the use of active inference for any on-line applications. It is important to notice that the dynamic model of the robot manipulator was based on Newtonian dynamics, using elasticity and viscosity terms to derive an expression for the acceleration of the joints. This may constitute a problem for real applications, mainly because the control actions were computed in open-loop based on the Newtonian model, and fed to the internal position controller of the simulated robot arm. This limits the applicability only to set-ups equipped with position control, and it is not robust. The overall scheme does not provide a general control law for robot manipulators, since the specific design is hard to generalise.

A recent MSc thesis [32] addressed some of the problems of [38], providing a generalisation of the approach detailed in the paper. In contrast with Pio-Lopez, [32] derived a closed-loop implementation of active inference. However, the controller showed lower performance with respect to a standard designed torque controller in simulation. In particular, the active inference scheme was not able to stabilise the system when the gravity compensation was active. One reason behind this behaviour, according to the concluding remarks of [32], could be that the off-line computation of active inference did not include the gravity in its generative model. Furthermore, as in [38], the implementation has been carried out in MATLAB using the Statistical Parametric Mapping (SPM) by Karl Friston. This toolbox is suitable for off-line loops, and it results too computationally expensive for on-line control.

### Closed-loop active inference

More recent research focused on a closed-loop implementation of active inference. In particular, the main contributions were given by Pablo Lanillos and Gordon Cheng, through their publications regarding the free-energy principle [25] and active inference [24].

The authors first focused on the problem of on-line state estimation in a static situation [25]. For the first time, it appears in the literature an application of the free-energy principle on a real UR5 robot arm. The goal of the paper was to demonstrate how a robot equipped with multisensory data (encoders, camera and tactile sensors), can perform body perception using a gradient descent scheme on the free-energy. After formulating the control problem in engineering terms, the authors defined an expression for the free-energy as a sum of squared prediction errors. This was given by the weighted difference between real and expected sensed

values, provided by the generative models of the process and of the sensory data. In contrast with [38], the generative models are estimated using Gaussian Process Regression (GPR), over the collected data from the real robot. The use of GPR with squared exponential kernel, revealed itself a good choice due to the fact that the necessary gradient of the process is available in closed form. The fact that only forward generative models had to be learnt, simplified the complexity of state estimation using fusion of different sensory data. The advantages of using free-energy for state estimation became clear in the experimental results, where the most probable state of the robot was inferred in a static situation. Since the free-energy is based on Bayesian inference, the most probable state of the robot arm is given by the posterior over the whole sensory set. This allowed to compensate for uncalibrated sensors and for missing information from redundant sources. The fact that no actions were included in the framework, however, makes this research incomplete in terms of implementation of the full active inference paradigm. The authors successfully performed static state perception, but left the inclusion of control actions for future work.

Starting from their previous work for static body perception, Lanillos and Cheng attempted to apply active inference for joint space control, in a simulated 2-DOF robot arm. In the solution proposed in [24], the generative models were obtained through continual learning. State-of-the-art regressors, namely locally weighted projection regression (LWPR), were implemented to on-line estimate the non-linear functions for the sensory input and state dynamics. The control actions were defined through a gradient descent of the free-energy with respect to the control input, providing the torques to the two joints of the robot arm. This represents the first closed-loop implementation of active inference for control in joint space. However, during the simulations, the unreliable on-line estimation of the acceleration of the robot from the LWPR, was substituted with the ground truth. Besides, the result showed poor performance of the controlled system with a very slow convergence of the states. Finally, the authors did not specify how the control actions were determined, only providing the general expression which however does not add any information regarding the adopted solution. To conclude, regardless the fact that only forward dynamic models had to be learnt, the problem still remained hard to solve, and the authors pointed out how this approach is not simpler compared with classical inverse dynamics approaches. Table 1-1 summarises the main features of the past work presented so far.

**The most recent closed-loop implementation of active inference** During the development of this work, no suitable solution for on-line robot control using active inference was present in the literature. However, a very recent research [35], unpublished at the time of writing, applied active inference for control of a 3-DOF humanoid robot for a reaching task. Pablo Lanillos and Gordon Cheng, are co-authors of this paper from Guillermo Oliver. The presented solution is based on their previous work [25], extended with the control actions. The approach adopted is very similar to the solution proposed in this thesis, with the difference that Oliver et al. are controlling the robot using velocity commands and not directly torques. Besides, the algorithm was tested in a real 3-DOF set-up, exploiting the inverse Jacobian and a visual camera for a reaching task in the Cartesian space.

Even if similar, the solution we propose in this thesis brings additional contributions and novelties. First of all, we use a 7-DOF robot manipulator instead of a 3-DOF. Then, we provide a more rigorous performance comparison of the active inference controller with a

**Table 1-1:** Main features of past work in active inference for robotics

| Paper | Algorithm | Generative models | Remarks |
|---|---|---|---|
| *Pio-Lopez et al. [38]* | Open-loop Cartesian Space control | Supposed to be known | No gravity compensation, use of built in position controllers, computationally expensive |
| *Lanillos and Cheng [25]* | Closed-loop state estimation | Learnt off-line using GPR | Good state estimation even with noise and missing information from redundant sensors. Static situation with no actions |
| *Lanillos and Cheng [24]* | Closed-loop joint space control | Learnt on-line using LWPR | Used ground truth acceleration, poor performance, unclear explanation of the control actions used |

state-of-the-art model reference adaptive controller. The control strategy presented is in joint space, but similarly to [35] we can use the inverse kinematics to specify the set-point directly from the Cartesian space. Besides, we provide a more detailed tuning procedure for the novel controller, which still remains vague in [35]. Finally, we analysed the fault tolerant properties of the algorithm, deriving a rigorous method to perform fault detection isolation and recovery from sensory faults.

Apart from these differences, the fact that [35] proposes a similar solution with comparable performance to what presented in this thesis, is a confirmation of the relevance of this work. Having acknowledged the presence of another valid active inference controller, the rest of this report is based on what available at the time of the study, so on any cited publication apart from [35].

## 1-3 Main contributions

The goal of this research mainly consisted in developing a novel adaptive fault tolerant active inference control scheme. This scheme is meant to be applied to robot manipulators subject to uncertain dynamics and sensory faults, providing a scalable and robust solution. In detail, the main contributions of this work are listed in the following:

- ***Derivation of an on-line active inference control law for a generic $n$-DOF robot manipulator***
  With this work, we derived a novel control law using active inference for robot control. The presented solution is easily scalable to high degrees-of-freedom, and it maintains high performance even in presence of large unmodeled dynamics. The proposed approach is a step forward with respect to previous work, and, together with the subsequent research in [35], it represents the first application of the free-energy principle for on-line robot control.

- ***Comparison of the adaptability performance with respect to a model reference adaptive controller***
  In order to evaluate the performance of the proposed active inference algorithm, we provided a comparison with another adaptive controller scheme. Due to the similarities that active inference shares with model reference adaptive control (MRAC), the latter has been chosen for a fair performance comparison. The simulations were carried out in a realistic simulator of a real manipulator, for a pick and place task with a 7 degrees-of-freedom robot arm. With the simulation results presented, we provide a strong evidence of the relevance of active inference for robot control, and we confirm the adaptability of the AIC to accommodate large unmodeled dynamics, outperforming the MRAC.

- ***Fault detection, isolation and recovery from sensory faults using active inference***
  A novel solution for fault tolerant control of robot manipulators based on active inference has been devised. The proposed approach simplifies the overall control architecture for fault detection and recovery in case of sensory faults, exploiting the structure of the controller itself. In particular, the free-energy principle is used to facilitate the implementation of the necessary sensory redundancy, and to determine an on-line threshold for the sensory prediction errors.

## 1-4 Thesis outline

This document is organised as follows. Chapter 2 is intended to make the thesis as self-contained as possible. We provide the necessary background knowledge about the three main topics related to this thesis: the free-energy principle, the model reference adaptive control, and the current model based methods for fault detection, isolation and recovery.

In chapter 3 we derive an active inference controller for a general $n$-DOF robot manipulator, and we explain the model assumptions and simplifications taken in order to obtain a suitable scheme for on-line robot control. To verify the performance of the proposed solution, the derived controller is applied to a simulated 7-DOF robot manipulator, and compared to a model reference adaptive scheme in presence of large unmodeled dynamics.

In Chapter 4 we take advantage of the high adaptability properties of active inference, and we propose to exploit the structure of the free-energy principle to obtain an intrinsically fault tolerant controller. We will describe how to detect and isolate sensory faults using a time-varying threshold on sensory prediction errors. Besides, we propose an approach to recover from a detected fault using sensory redundancy.

Finally, Chapter 5 concludes the work presented in this document, and it summarises the answers to the research questions previously posed. Besides, the author included some guidelines for future research in this direction, pointing out the main challenges and the questions that still remains unanswered.

# Chapter 2

# Background knowledge

*In this Chapter, we present some fundamental theoretical concepts at the basis of this work. Section 2-1 and Section 2-3 are extrapolated and adapted from the literature survey prior to this thesis, with the purpose of making this report as self-contained as possible. In particular, the free-energy principle and active inference are presented in control engineering terms to facilitate the understanding of the control algorithm presented later on. Subsequently, the theory behind the model reference adaptive controller for performance comparison is detailed. Finally, the main concepts of classical model based fault detection are reported. The main purpose of the latter is to highlight the intuitions at the basis of the theory presented in Chapter 4, were the the active inference based fault tolerant scheme is devised.*

## 2-1 Free-energy principle and active inference: the biological inspiration

The background knowledge here presented regarding the free-energy and active inference is mainly based on the re-formulation proposed by Buckley et al. [5], since it is closer to the control notation.

The free-energy principle is inspired by the tendency of the living organisms to resist disorder in order to survive, minimising the surprise associated with the occurrence of atypical events. The atypicality of an event can be quantified using the negative logarithm of the probability of its sensory data, which is known as "surprisal":

$$-\ln p(\boldsymbol{y}) \tag{2-1}$$

where $p(\boldsymbol{y})$ is the probability of observing some particular multivariate sensory data $\boldsymbol{y}$ in the environment. The more improbable an event is, the higher is the surprisal. Essentially, according to the free-energy principle (FEP), all living organisms find themselves in a specific sub-set of all the infinite possible *states* in order to survive. From a high level perspective, think about a fish that finds itself outside water. The atypicality of the sensory input will

make the fish perform a sequence of actions that will try to bring the animal back in the water, minimising the sensory surprisal [5]. For clarity, the *states* represent selected physical quantities, such as internal body temperature, the joint configuration of a limb, etc. In this view, in typical conditions, the set of possible states is finite.

**Why is the free-energy necessary?** As we have seen, the sensory surprisal has to be minimised in order to survive. However, the distribution of surprising events is unpredictable and unknown. To solve this paradox, the free-energy is proposed. In a sense, free-energy is a quantity that:

1. Provides an upper bound on the extent of atypicality of sensory data (so an upper bound for surprise);
2. Can be evaluated by an organism, since it depends only on sensory input and on an internal model of how this sensory input was created, so the environmental causes.

In short, the free-energy as a bound on surprise is required since it can be evaluated more easily. Minimising the free-energy, the surprise is also indirectly minimised. A more analytic definition of free-energy is given in the next section.

**The FEP structure** The FEP proposes that organisms implicitly have a best guess of relevant variables around them, a sort of set of probability distributions over all their possible values: this can be seen as a *Bayesian belief*. When an organism receives sensory data, it updates these distributions to better describe the environment. One of these distributions is the *recognition density* (R-Density), which is the internal model of the environmental states. To update the R-Density, some hypothesis on how environmental states are related to the sensory input have to be made. These assumptions are expressed in terms of a probability density function: the *generative density* (G-Density), which encodes the relation between sensory input and environmental states. The G-Density uses a *Bayesian formalism* and it is expressed as the *likelihood* times a certain *prior*. The likelihood is the probability of certain sensory data given specific environmental variables, and the prior is the way to describe the beliefs of the probability distribution over the environmental states. The G-Density describes how sensory data is caused by the environment: generative models are used to specify the brain expectations on environmental states given sensory data in terms of a Gaussian distribution.

The combination of this two densities will result in the definition of an expression for the free-energy. More in detail, the free-energy is a non-negative quantity from the Kullback-Leibler divergence between the two densities $R$ and $G$. So it is not a directly measurable quantity, but it is more a quantity dependent on the interpretation of the brain variables encoded in probability distributions.

### 2-1-1 Mathematical formulation of the free-energy

This section describes the free-energy from a mathematical point of view. Please note that this part will make use of concepts related to *Bayes theory* [28] and in particular *variational*

*Bayes* [3]. To make the mathematical description clear, let us first define the notation for the most important variables as in Table 2-1.

**Table 2-1:** Notation for the definition of the free-energy

| Symbol | Name | Description |
|---|---|---|
| $\boldsymbol{x}$ | Environment states | Well-defined physical entities like internal temperature or joint values. |
| $\boldsymbol{y}$ | Sensory input | The data generated from the available sensory set. |
| $\boldsymbol{u}$ | Control action | The action that an agent can perform on the environment in order to change sensory input. |
| $\boldsymbol{\mu}$ | State estimate | Beliefs about the states of the environment encoded by the controller. Expectation of the states $\boldsymbol{x}$ |

It is assumed [5] that an organism tries to determine the probability of having some environmental states based on the sensory input it has available. This problem can be formulated in Bayesian terms: the agent is trying to find the *posterior* of the state ($\boldsymbol{x}$) of the environment, based on the current sensory data ($\boldsymbol{y}$). To achieve so, let us assume that the prior beliefs of the agents are encoded in the G-Density by the *likelihood* (so the belief about how environmental states cause sensory input), times a *prior* (the belief about the world before the input is received). For the sake of simplicity, we present the free-energy principle for a univariate case such that $\boldsymbol{x} = x$, $\boldsymbol{y} = y$, $\boldsymbol{u} = u$ and $\boldsymbol{\mu} = \mu$. We will later on extend the theory to a multivariate scenario. The G-Density is expressed as:

$$GDensity = p(x, y) = p(y|x)p(x) \tag{2-2}$$

Using Bayes theorem, the posterior that the agent is looking for can be formulated as:

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} = \frac{p(y|x)p(x)}{\int p(y|x)p(x)dx} \tag{2-3}$$

The denominator in the above expression is often intractable, due to high dimensionality and the absence of an analytical solution. Therefore, a *Variational Bayes* is used to approximate the nasty term $p(x|y)$ using optimization. To do so, an approximate probability distribution is introduced, a sort of best guess of the causes of the sensory input, namely the *recognition density* $r_d(x)$. One hypothesis about this quantity is that it normalises as:

$$\int r_d(x)dx = 1 \tag{2-4}$$

At this point a measure of the difference between the recognition density and the true posterior is formulated as a Kullback-Leibler divergence:

$$D_{KL}(r_d(x)||p(x|y)) = \int r_d(x) \ln \frac{r_d(x)}{p(x|y)} dx \tag{2-5}$$

This quantity cannot be evaluated yet since the true posterior $p(x|y)$ is unknown. To solve this problem the G-Density in Eq. (2-2) and Eq. (2-4) can be exploited. It results:

$$D_{KL}(r_d(x)||p(x|y)) = \int r_d(x) \ln \frac{r_d(x)}{p(x, y)} dx + \ln p(y) = \mathcal{F} + \ln p(y) \tag{2-6}$$

where $\mathcal{F}$ is the free-energy. Since the marginal probability density $p(y)$ depends just on sensory inputs, it is sufficient to focus on minimizing only $\mathcal{F}$ with respect to R-Density. Doing so, the R-Density approximates the true posterior $p(x|y)$. In fact when $r_d(x) = p(x|y)$ it results $D_{KL} = 0$. Furthermore, according to the Jensen's Inequality [8], the KL-Divergence is always greater or equal than zero, and this makes the free-energy an upper bound for the surprise:

$$\mathcal{F} \geq -\ln p(y) \tag{2-7}$$

Expanding the G-Density, Eq. (2-6) is rewritten as

$$\mathcal{F} = \int r_d(x) E(x,y) dx + \int r_d(x) \ln r_d(x) dx \tag{2-8}$$

where $E(x,y) = -\ln p(x,y)$. By analogy with the thermodynamic free-energy, the first term in the above equation is called *average energy*.

To conclude, an expression for the free-energy to approximate the surprisal with an upper bound has been found. A method for minimizing the surprisal itself, however, still has to be defined. To do so, an organism can either update its beliefs about the surroundings, or it can perform actions to change the sensory input and match its beliefs. This will result in the complete framework, namely the *active inference*. However, at this point a question should arise: the *R-Density and G-Density are of free choice, but how should they be properly selected*? The next two subsections describe possible approaches.

### Recognition density

To implement the free-energy, the brain must encode the R-Density. To do so, it is supposed [12] that the brain parametrizes the sufficient statistics (e.g. mean and variance) of a probability distribution. More in detail, the neuronal variables encode a family of probability densities over the states $x$. Then, a specific instantaneous state of the brain (represented by its mean value $\mu$) picks up a particular density, namely the recognition density $r_d(x;\mu)$. Note that the semicolon indicates that $\mu$ is more a parameter than a random variable. Finding $r_d(x,\mu)$ is hard and intractable, this is why [12] proposes an approximation assuming that the R-Density is Gaussian. This approximation is called *Laplace approximation*:

$$r_d(x) = \frac{1}{\sqrt{2\pi\sigma}} \exp\{-(x-\mu)^2/(2\sigma)\} \tag{2-9}$$

where $\sigma$ and $\mu$ are the variance and the mean of a single environmental variable $x$. If it is further assumed, according to the Laplace approximation, that the recognition density is sharply peaked at its mean value $\mu$, and that $E(x,y)$ is smooth in $x$, applying the Taylor expansion of $E(x,y)$ around $x = \mu$ allows to re-write Eq. (2-8) as:

$$\mathcal{F} = E(\mu, y) + \frac{1}{2}\left(\left[\frac{d^2E}{dx^2}\right]_\mu \sigma - \ln 2\pi\sigma - 1\right) \tag{2-10}$$

To simplify further, the derivative of the above equation with respect to $\sigma$ is taken. Demanding that this equals zero, the optimal variance (which optimizes the free-energy) is found as:

$$\sigma^* = \left[\frac{d^2E}{dx^2}\right]_\mu^{-1} \tag{2-11}$$

Substituting in Eq. (2-10):

$$\mathcal{F} = E(\mu, y) - \frac{1}{2}\ln(2\pi\sigma^*) \tag{2-12}$$

All these approximations allowed to recast the free-energy in terms of a joint density $p(\mu, y)$. This is a probability over the sensory data $y$ and the sufficient statistic $\mu$ of the recognition density, instead of over some unspecified environmental features $x$. The term $p(\mu, y)$ is, in other words, an approximate G-Density. Finally, neglecting the constant term in Eq. (2-12), an approximate for the free-energy is obtained using the Laplace-encoded energy:

$$\mathcal{F} \approx E(\mu, y) = -\ln p(\mu, y) \tag{2-13}$$

This approximation shows that the brain only represents the most likely environmental causes of some sensory data, neglecting the details of their distributions.

### Generative density

In the previous section, the free-energy has been described in terms of the approximate G-Density, in which the sufficient statistic $\mu$ of the R-Density substituted the environmental state $x$. In order to evaluate $\mathcal{F}$, it still remains unspecified how the brain encodes the generative density. A *generative model* of the environmental causes of sensory data has to be formulated. Once the model is defined, the G-Density can be specified and $\mathcal{F}$ evaluated. There are several techniques to define a generative model, which are introduced in the next subsections.

**Static generative model** In a simple scenario, it is assumed [5] that the sensory data is generated by a non-linear mapping between environmental states $\mu$ in combination with some noise $z \sim (0, \sigma_y)$:

$$y = g(\mu; x) + z \tag{2-14}$$

The belief about environmental states can also be expressed by a fixed parameter plus some noise $w \sim (0, \sigma_\mu)$:

$$\mu = \mu_d + w \tag{2-15}$$

This means that, for a static model, the belief about the states is history-independent, and it fluctuates around a prior mean $\mu_d$. Furthermore, the variance $\sigma_\mu$ represents the agent's confidence about its estimate of future states. Note also that $\mu_d$ and $\sigma_\mu$ are different from the sufficient statistics of the R-Density $\mu$ and $\sigma$, since the latter encode the agent's uncertain belief about its current environment $x$. Now, assuming the two random variables $z = y - g(\mu; x)$ and $w = \mu - \mu_d$ to be Gaussian [12], the generative density can be written as:

$$\begin{aligned} p(\mu, y) &= p(y|\mu)p(\mu) \\ &= \frac{1}{\sqrt{2\pi\sigma_y}}\exp\{-(y - g(\mu; x))^2/(2\sigma_y)\}\frac{1}{\sqrt{2\pi\sigma_\mu}}\exp\{-(\mu - \mu_d)^2/(2\sigma_\mu)\} \end{aligned} \tag{2-16}$$

Substituting now the univariate case from Eq. (2-13), one obtains (up to a constant):

$$\begin{aligned} E(\mu, y) &= -\ln p(y|\mu) - \ln p(\mu) \\ &= \frac{1}{2\sigma_y}\varepsilon_y^2 + \frac{1}{2\sigma_\mu}\varepsilon_\mu^2 + \frac{1}{2}\ln(\sigma_y\sigma_\mu) \end{aligned} \tag{2-17}$$

Two new important quantities emerged from Eq. (2-17). They are the so called *prediction errors*. In particular:

- $\varepsilon_y \equiv y - g(\mu; x)$ is the sensory prediction error. This is the discrepancy between the sensory data $y$ and the outcome of its predictions from $g(\mu; x)$;
- $\varepsilon_\mu \equiv \mu - \mu_d$ is the model prediction error. This is the extent to which $\mu$ differs from $\mu_d$.

Note that each error is multiplied by the inverse of its variance. This is an indication of the relative confidence about these errors. The extension to a multivariate case follows simply: for each sensory input available, a sensory prediction error is collected in a vector $\boldsymbol{\varepsilon}_y$. The variance $\sigma_y$ is also substituted by a covariance matrix $\Sigma_y$. Analogously for $\boldsymbol{\varepsilon}_\mu$ and $\Sigma_\mu$.

To conclude, the static model suggests that $\mathcal{F}$ is nothing more than a quadratic sum of prediction errors, regulated by the relative precision (inverse variances). With this definition, the free-energy can be computed once a prior $\mu_d$ and the generative model $g(\mu; x)$ are given.

**Dynamic generative model** A dynamic generative model is a step forward with respect to the previous approach. Now, the possibility of the environment to change dynamically is taken into account. A dynamic generative model is necessary if one wants to perform state inference of time varying states. Let us assume again a single brain state $\mu$ and a single sensory input $y$. The Langevin-type equation [52] is considered instead of using the static Eq. (2-15):

$$\frac{d\mu}{dt} = f(\mu) + w \tag{2-18}$$

Now, combining Eq. (2-18) with Eq. (2-14), a dynamical generative model can be obtained. To do so, however, a new concept has to be introduced first: the *generalised motions*.

**Generalised motions** The generalised motions are useful to describe the evolution of the states of a dynamical system in terms of increasingly higher order derivatives of its state variables, allowing a more precise description of the system itself. In mathematical terms, neglecting non-linear derivative terms under local linearity assumption [12], the generalised sensory data is expressed as:

$$\begin{aligned} y &= g(\mu) + z \\ y' &= \frac{\partial g}{\partial \mu}\mu' + z' \\ y'' &= \frac{\partial g}{\partial \mu}\mu'' + z'' \\ &\vdots \end{aligned} \tag{2-19}$$

where $z, z', ...$ are independent noise sources for every dynamic order, and the symbol "′" means the time derivative $d/dt$. Similarly, for the Langevin equation it holds:

$$\begin{aligned} \mu &= f(\mu) + w \\ \mu' &= \frac{\partial f}{\partial \mu}\mu' + w' \\ \mu'' &= \frac{\partial f}{\partial \mu}\mu'' + w'' \\ &\vdots \end{aligned} \tag{2-20}$$

Finally, for a compact notation, the generalised sensory data $\tilde{y}$, the states $\tilde{\mu}$, the noise, and the functions $g$, $f$, are defined as:

$$\begin{aligned} \tilde{y} &= (y, y', ...) = (y^{(0)}, y^{(1)}, ...) \qquad \tilde{\mu} = (\mu, \mu', ...) = (\mu^{(0)}, \mu^{(1)}, ...) \\ \tilde{z} &= (z, z', ...) = (z^{(0)}, z^{(1)}, ...) \qquad \tilde{w} = (w, w', ...) = (w^{(0)}, w^{(1)}, ...) \\ \tilde{g} &= (g, g', ...) = (g^{(0)}, g^{(1)}, ...) \qquad \tilde{f} = (f, f', ...) = (f^{(0)}, f^{(1)}, ...) \\ \tilde{\mu}' &\equiv D\tilde{\mu} \equiv \frac{d}{dt}(\mu, \mu', ...) = \begin{pmatrix} 0 & 1 & 0 & \dots \\ 0 & 0 & 1 & \vdots \\ \vdots & \vdots & \ddots & \ddots \\ 0 & 0 & \dots & 0 \end{pmatrix} (\mu, \mu', ...) = (\mu', \mu'', ...) \end{aligned} \tag{2-21}$$

Having said that, the sensory data and the evolution of the states can be written as

$$\tilde{y} = \tilde{g}(\tilde{\mu}) + \tilde{z} \tag{2-22}$$

$$D\tilde{\mu} = \tilde{f}(\tilde{\mu}) + \tilde{w} \tag{2-23}$$

The first equation describes how sensory data $\tilde{y}$ are inferred using their causes $\tilde{\mu}$. At each dynamical order $i$, $y^{(i)}$ only depends on $\mu^{(i)}$. The second equation is the generalised equation of motion and it specifies how the coupling between adjacent dynamical orders is. Now, assuming [12] that all the fluctuations at each dynamical order $z^{(i)}$ and $w^{(i)}$ are Gaussian, the expression for the Laplace-encoded energy of Eq. (2-13), and thus an approximate for $\mathcal{F}$, reduces up to a constant to:

$$\mathcal{F} \approx E(\tilde{\mu}, \tilde{y}) = \sum_{i=0}^{\infty} \left( \frac{1}{2\sigma_{y^{(i)}}} {\varepsilon_y^{(i)}}^2 + \frac{1}{2} \ln \sigma_{y^{(i)}} \right) + \sum_{i=0}^{\infty} \left( \frac{1}{2\sigma_{\mu^{(i)}}} {\varepsilon_\mu^{(i)}}^2 + \frac{1}{2} \ln \sigma_{\mu^{(i)}} \right) \tag{2-24}$$

where:

- $\varepsilon_y^{(i)} \equiv y^{(i)} - g^{(i)}$ is the prediction error between sensory data $y^{(i)}$ and its prediction $g^{(i)}$;
- $\varepsilon_\mu^{(i)} \equiv \mu^{(i+1)} - f^{(i)}$ is the discrepancy between expected higher order output $\mu^{(i+1)}$ and its prediction $f^{(i)}$;

Usually, only finite dimension systems are considered. As [15] suggests, order six should be sufficient to describe even complex processes. Moreover, the variance associated to high derivatives is high, and the contribution of the errors squared in Eq. (2-24) is limited. To sum up, the Laplace-encoded energy (an approximation for $\mathcal{F}$) is expressed as a quadratic sum of sensory prediction errors $\varepsilon_y^{(i)}$ and model prediction errors $\varepsilon_\mu^{(i)}$, between various dynamical orders, weighted by their variances.

#### Free-energy minimization using perception

In the previous sections it has been discussed how to obtain the free-energy. We focus now on how to minimise free-energy to make the R-Density a good posterior approximation. There are reasons to believe [16, 11] that the brain implements a gradient descent to minimize $\mathcal{F}$. A proposal is that a generic brain state updates according to the gradient of the Laplace-encoded energy. Before proceeding further, an important remark has to be done. In fact, there is a difference between the expectation of the state motion $\dot{\mu}$ under the generative model, and the agent's current belief of that motion $D\mu$ [18] (i.e. for a generic $i$-th component of the generalised state $\mu^{(i)'} \neq \dot{\mu}^{(i)}$). In other words, for instance, the velocity of a point in the generalised state space, is different from the trajectory model that the brain encodes regarding that motion. Remember that this trajectory is denoted with $D\tilde{\mu}$. This is a crucial point to understand so, in detail, it holds:

$$D\tilde{\mu} \equiv \begin{pmatrix} 0 & 1 & 0 & \dots \\ 0 & 0 & 1 & \vdots \\ \vdots & \vdots & \ddots & \ddots \\ 0 & 0 & \dots & 0 \end{pmatrix} \tilde{\mu} = (\mu^{(0)'}, \mu^{(1)'}, ...) = (\mu', \mu'', ...) \tag{2-25}$$

Having said that, considering that $D\mu^{(i)} = \mu^{(i)'}$, the gradient descent for the free-energy can be written as:

$$\dot{\mu}^{(i)} - D\mu^{(i)} = -\kappa_\mu \partial_{\tilde{\mu}} E(\tilde{\mu}, \tilde{y}) \tag{2-26}$$

where $\kappa_\mu$ is the learning rate for the state update. The term $E$ vanishes when $\dot{\tilde{\mu}} = D\tilde{\mu}$ so the expectation of the motion equals the motion of the expectation. Eq. (2-26) in a multivariate case will represent a set of first order differential equations. If this set was continuously integrated in presence of sensory input, it would make the brain perform approximate inference about the environmental states, minimizing $\mathcal{F}$. The next section presents the last piece of the framework to perform active inference: the definition of the actions.

### 2-1-2 Active inference

An organism in the environment has usually the possibility to take actions in order to modify the world around itself. In this view, an organism can change its predictions about environmental states through perception, but it can also change its sensory input through action to minimize prediction errors. We present now the last piece of theory to define the whole active inference construct.

#### Free-energy minimization using actions

While perception minimises the free-energy changing the beliefs about the states to better predict sensory data, actions act on the environment to indirectly change sensory input to match sensory predictions.

To perform active inference, an agent must have a model of how sensory data changes with actions. This can be formulated in mathematical terms. First of all, a one dimensional case is considered (one sensory input and one brain state). The sensory data $y$ is written as a

function of the action $u$ such that $y = y(u)$. Assuming a gradient descent scheme as before [5], the gradient of the Laplace-encoded energy with respect to the control action becomes:

$$\frac{\partial E(\mu, y)}{\partial u} \equiv \frac{\partial y}{\partial u} \frac{\partial E(\mu, y)}{\partial y} \tag{2-27}$$

The action that minimises the Laplace-encoded energy is thus:

$$\dot{u} = -\kappa_a \frac{\partial y}{\partial u} \frac{\partial E(\mu, y)}{\partial y} \tag{2-28}$$

where $\kappa_a$ is the learning rate for the actions update. Even though the definition of control actions appears simple, the partial derivatives of the sensory input with respect to the control action is hard to determine. This can be seen as a forward dynamics problem, and it represents the main challenge during the definition of an active inference controller. In Chapter 3, a possible approach to tackle this problem for the definition of the actions update is presented.

## 2-2 Model Reference Adaptive Control

Even though there is already evidence of the adaptability performance of active inference, as highlighted in [25], an actual comparison with another adaptive controller is still missing in the literature. To validate the performance of the active inference scheme, then, a comparable control architecture has to be chosen. Considering the fact that one of the goals of this research is to verify the adaptive properties of the algorithm, our choice falls on an adaptive controller. Besides active inference, in fact, there exist other well established control solutions to deal with robotic manipulators subject to parameters variation and abrupt changes in the dynamics. The adaptive control branch of control theory is the main example [1]. Within the adaptive controllers, two main categories can be identified: the model reference adaptive systems, and the self-tuning regulators [44]. The first technique being studied for robot manipulators was the model reference adaptive control (MRAC) [51]. The idea behind this technique, is to derive an adaptive control signal to be applied to the robot actuators, which will force the system to behave as specified by a chosen reference model. Furthermore, the adaptation law is designed to guarantee stability using either Lyapunov theory or hyperstability theory [45]. The other most common approach for robot control is the self-tuning adaptive control [50] [22]. The main difference between this technique and the MRAC, is that the self-tuning approach represents the robot as a linear discrete-time model, and it estimates on-line the unknown parameters substituting them in the control law. The literature for adaptive control of robot manipulators shows the ability of these techniques to perform well in presence of uncertain dynamics and varying payloads. Having said that, the complexity of the controllers usually increases with increasing number of DOF.

Among all the possible adaptive controllers, in this thesis we choose the MRAC with hyperstability theory for comparison [45]. As we sill see later on, this choice is motivated by the fact that this approach provides adaptability to abrupt changes in the robot dynamics, and it does not require the kinematic or dynamic description of the manipulator. These characteristics make the MRAC suitable for a fair comparison with the derived active inference controller.

### 2-2-1 MRAC for robot manipulators

In the following, we will describe a model reference adaptive controller for a generic $n$-DOF robot manipulator. The control scheme here reported, is taken from [45] and it makes use of hyperstability theory. The control signal is a set of reference torques for the motors. These torques are computed through an adaptive law, which will make each joint of the robot manipulator behave as a desired second order linear system. To present the control structure, let us start with the dynamic description of the robot. In general it holds:

$$\boldsymbol{u}(t) = M(\boldsymbol{q}(t))\ddot{\boldsymbol{q}}(t) + N(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t))\dot{\boldsymbol{q}}(t) + G(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t))\boldsymbol{q}(t) \tag{2-29}$$

where $\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t),\ \ddot{\boldsymbol{q}}(t)$ are the joint positions, velocities and accelerations $\in \mathbb{R}^n$. $M \in \mathbb{R}^{n\times n}$ is the inertia matrix, $N \in \mathbb{R}^{n\times n}$ is the Coriolis and friction matrix, and $G \in \mathbb{R}^{n\times n}$ is the gravity terms matrix. Let us suppose that the robot has to follow a specified reference trajectory, and that the reference position $\boldsymbol{q}_r$ and velocity $\dot{\boldsymbol{q}}_r$ are finite and continuous functions. A general control law, then, can be expressed as:

$$\boldsymbol{u}(t) = K_0(t)\boldsymbol{q}(t) + K_1(t)\dot{\boldsymbol{q}}(t) + Q_0(t)\boldsymbol{q}_r(t) + Q_1(t)\dot{\boldsymbol{q}}_r(t) + \boldsymbol{f}(t) \tag{2-30}$$

where $K_0(t),\ K_1(t) \in \mathbb{R}^{n\times n}$ are matrices of position and velocity feedback, $Q_0(t),\ Q_1(t) \in \mathbb{R}^{n\times n}$ are matrices acting on the reference signal and $\boldsymbol{f}(t) \in \mathbb{R}^n$ is an auxiliary feedforward vector. By substituting Eq. (2-30) in Eq. (2-29) it results:

$$\begin{aligned} M(\boldsymbol{q}(t))\ddot{\boldsymbol{q}}(t) \quad &+ \quad (N(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t)) - K_1(t))\dot{\boldsymbol{q}}(t) + (G(\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t)) - K_0(t))\boldsymbol{q}(t) \\ &= \quad Q_0(t)\boldsymbol{q}_r(t) + Q_1(t)\dot{\boldsymbol{q}}_r(t) + \boldsymbol{f}(t) \end{aligned} \tag{2-31}$$

Now, let us assume [45] that the desired dynamics that we want to impose for each joint of the manipulator, are described by the following model:

$$\ddot{\boldsymbol{q}}_m(t) + A_1\dot{\boldsymbol{q}}_m(t) + A_0\boldsymbol{q}_m(t) = A_0\boldsymbol{q}_r(t) \tag{2-32}$$

where $\boldsymbol{q}_m(t)$ is the $n$-dimensional joint angle vector of the reference model. The elements $A_0,\ A_1 \in \mathbb{R}^{n\times n}$ are constant matrices to be chosen such that the error dynamics is asymptotically stable, and the joint angles are decoupled. In this manner, $\boldsymbol{q}_m(t)$ approaches $\boldsymbol{q}_r(t)$. To do so, the following matrices are used:

$$A_0(t) = diag\{\omega_i^2\}, \qquad A_1(t) = diag\{2\zeta_i\omega_i\} \tag{2-33}$$

where $\omega_i,\ \zeta_i,\ i = 1...n$ are the natural frequency and damping of the model of the $i$-th joint response, which is given by:

$$q_{mi}(s) = \frac{\omega_i^2}{s^2 + 2\zeta\omega_i s + \omega_i^2} q_{ri}(s) \tag{2-34}$$

The MRAC will steer the joints to behave as a second order linear system: the joint angle $q_{mi}$ can be made as close to its reference value $q_{ri}$ as desired, choosing proper $\zeta_i$ and $\omega_i$. Usually, it is supposed critical damping $\zeta_i = 1, \forall i$ [45].

The problem is now to find an adaptation law such that the response of the joints' angles from Eq. (2-31) approaches the one of the reference model described in Eq. (2-32). Let us define the adaptation error as:

$$\boldsymbol{e}(t) = \boldsymbol{q}_m(t) - \boldsymbol{q}(t) \tag{2-35}$$

Substituting equations Eq. (2-31) and Eq. (2-32) in Eq. (2-35), it results:

$$\ddot{\boldsymbol{e}}(t) + A_1\dot{\boldsymbol{e}}(t) + A_0\boldsymbol{e}(t) = -\boldsymbol{u}(t) \tag{2-36}$$

where $\ddot{\boldsymbol{e}}(t) = \ddot{\boldsymbol{q}}_m(t) - \ddot{\boldsymbol{q}}(t)$, $\dot{\boldsymbol{e}}(t) = \dot{\boldsymbol{q}}_m(t) - \dot{\boldsymbol{q}}(t)$ and:

$$\begin{aligned} \boldsymbol{u}(t) &= [A_0 + M^{-1}(K_0(t) - G)]\boldsymbol{q}(t) + [A_1 + M^{-1}(K_1(t) - N)]\dot{\boldsymbol{q}}(t) \\ &+ [2M^{-1}Q_0(t) - A_0]\boldsymbol{q}_r(t) + M^{-1}Q_1(t)\dot{\boldsymbol{q}}_r(t) + M^{-1}\boldsymbol{f}(t) \end{aligned} \tag{2-37}$$

The only remaining part is to find the feedback and cascade matrices $K_0(t),\ K1(t),\ Q_0(t),\ Q_1(t)$ and the feedforward term $\boldsymbol{f}(t)$ such that the adaptation error of Eq. (2-35) goes to zero. Only the final results will be here reported, for the complete mathematical derivation the interested reader is referred to [45]. For the implementation of the control algorithm, the terms which has to be specified are $K_0(t),\ K_1(t),\ Q_0(t),\ Q_1(t),\ f(t)$. In particular:

$$\begin{aligned} K_0(t) &= \hat{K}_0 + E_{01}\bar{\boldsymbol{q}}_e(t)\boldsymbol{q}(t)^T + E_{02}\int_0^t \bar{\boldsymbol{q}}_e(\tau)\boldsymbol{q}(\tau)d\tau + E_{03}\frac{d}{dt}[\bar{\boldsymbol{q}}_e(t)\boldsymbol{q}(t)^T] \\ K_1(t) &= \hat{K}_1 + E_{11}\bar{\boldsymbol{q}}_e(t)\dot{\boldsymbol{q}}(t)^T + E_{12}\int_0^t \bar{\boldsymbol{q}}_e(\tau)\dot{\boldsymbol{q}}(\tau)d\tau + E_{13}\frac{d}{dt}[\bar{\boldsymbol{q}}_e(t)\dot{\boldsymbol{q}}(t)^T] \\ Q_0(t) &= \hat{Q}_0 + F_{01}\bar{\boldsymbol{q}}_e(t)\boldsymbol{q}_r(t)^T + F_{02}\int_0^t \bar{\boldsymbol{q}}_e(\tau)\boldsymbol{q}_r(\tau)d\tau + F_{03}\frac{d}{dt}[\bar{\boldsymbol{q}}_e(t)\boldsymbol{q}_r(t)^T] \\ Q_1(t) &= \hat{Q}_1 + F_{11}\bar{\boldsymbol{q}}_e(t)\dot{\boldsymbol{q}}_r(t)^T + F_{12}\int_0^t \bar{\boldsymbol{q}}_e(\tau)\dot{\boldsymbol{q}}_r(\tau)d\tau + F_{13}\frac{d}{dt}[\bar{\boldsymbol{q}}_e(t)\dot{\boldsymbol{q}}_r(t)^T] \\ \boldsymbol{f}(t) &= \alpha_1\bar{\boldsymbol{q}}_e(t) + \alpha_2\int_0^t \bar{\boldsymbol{q}}_e(\tau)d\tau + \alpha_3\frac{d\bar{\boldsymbol{q}}_e}{dt} \end{aligned} \tag{2-38}$$

The quantity $\bar{\boldsymbol{q}}_e$ is the modified joint angle error vector defined as:

$$\bar{\boldsymbol{q}}_e = P_2[\boldsymbol{q}_r(t) - \boldsymbol{q}(t)] + P3[\dot{\boldsymbol{q}}_r(t) - \dot{\boldsymbol{q}}(t)] \tag{2-39}$$

and the matrices $P2,\ P3$ are $P2 = diag\{p_{2i}\},\ P3 = diag\{p_{3i}\},\ i = 1...n$ with

$$p_{2i} = \frac{l_{1i}}{2\omega_i^2}, \qquad p_{3i} = \frac{l_{2i}}{4\zeta_i\omega_i} + \frac{l_{1i}}{4\zeta_i\omega_i^3}, \qquad i = 1...n \tag{2-40}$$

The coefficients $l_{1i},\ l_{2i},\ i = 1...n$ are such that the matrix $L$ is positive semi-definite, to guarantee stability. The matrix $L$ is defined as:

$$L = \begin{pmatrix} diag\{l_{1i}\} & 0 \\ 0 & diag\{l_{2i}\} \end{pmatrix}, \qquad i = 1...n \tag{2-41}$$

Regarding the terms described in Eq. (2-38), some considerations can be made:

- When the adaptation process is fast, the matrices $\hat{K}_0(t),\ \hat{K}_1(t),\ \hat{Q}_0(t),\ \hat{Q}_1(t)$ can be treated as arbitrary constants, often set to zero;
- The gain matrices $E_{02},\ E_{12}, F_{02},\ F_{12}$ are positive definite matrices, while $E_{01},\ E_{11}, E_{03},\ E_{13}$, $F_{01},\ F_{11}, F_{03},\ F_{13}$ are positive semi-definite;
- If noise is present in the position measurement the matrices $E_{03},\ E_{13},\ F_{03},\ F_{13}$ can be set to small values, often zero.

### Controller characteristics and analysis

The adaptive control law offers large flexibility regarding the structure and the parameters of the controller. According to Eq. (2-38), one can realise that the adaptation law is of a proportional-integral-derivative type. Also, the control structure allows a feedforward signal $\boldsymbol{f}$ to improve tracking performance. This signal can include position, velocity, acceleration and so on, of the reference signal. Notice that, the adaptation algorithm allows to drastically improve the performance in case of system's parameters variation, uncertainties and disturbances. This, in combination with the scalability of the MRAC, and the fact that no dynamical model of the robot is needed for control, are the reasons why this controller has been chosen for comparison. However, while all this flexibility can be beneficial, it can also represent a challenge during the tuning of the large number of parameters.

## 2-2-2 MRAC tuning

Based on the controller description previously given, the following block schemes can be derived:

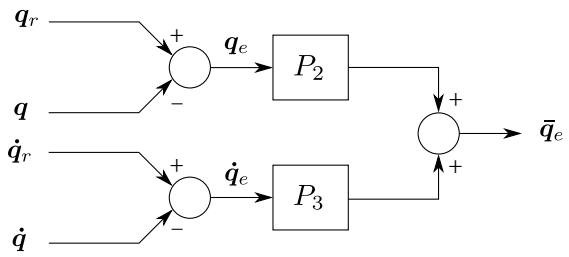



**Figure 2-1:** Modified joint angle error vector

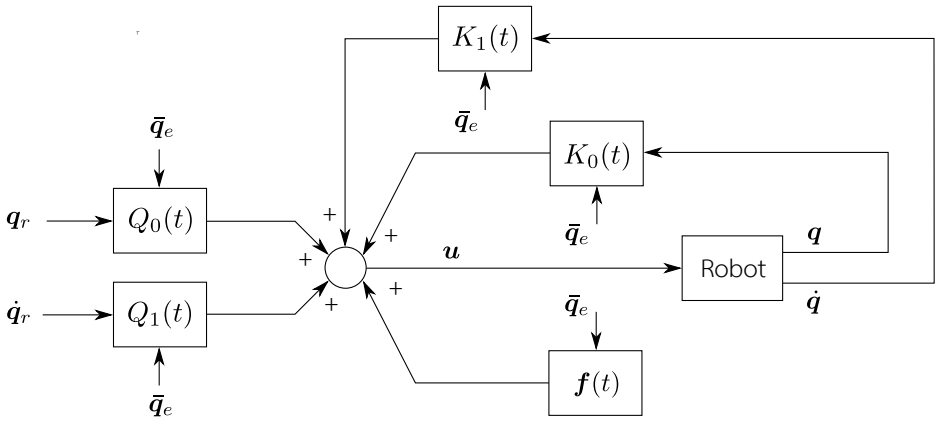



**Figure 2-2:** Block scheme for the MRAC with hyperstability approach

There are several constant gain matrices for the feedback, cascade and feedforward signal to tune. Furthermore, the choice of the positive definite matrix $L$ in equation Eq. (2-41) influences the controller performance and feasibility. Some considerations from [45] have

been taken into account for the tuning or the MRAC. This is done to appreciate later on the simpler tuning procedure of the active inference controller. First of all, the derivative terms have been set to zero, supposing that noise is present in the position measurements. Thus $E_{03},\ E_{13},\ F_{03},\ F_{13}$ can be set to zero. The gain matrices have been considered as a scalar constant (so the equivalent of a diagonal matrix of a single value) thus every joint will have the same adaptive gain. Due to fast convergence of the algorithm, the values $\hat{K}_0(t),\ \hat{K}1(t),\ \hat{Q}_0(t),\ \hat{Q}_1(t)$ have been set to zero. Moreover, the matrix $L$ has been initially set to the identity and subsequently the values $l_{1i},\ l_{1i}$ have been increased, to have a faster step response. In fact, $L$ directly influences the modified joint angle vector $\bar{\boldsymbol{q}}_e$, through $P_2$ and $P_3$. The remaining parameters to tune are the proportional and integral matrices, considered as well just as positive constants to use the same gains for all the joints. The values $E_{01},\ E_{02},\ F_{01},\ F_{02}$ have been increased until the response was satisfactory. The same procedure has been followed for the proportional and integral terms $\alpha_0$ and $\alpha_1$.

## 2-3 Fault tolerant control

*Beside assessing the adaptability performance of the neuroscientific theory proposed by Friston, this thesis also investigated the fault tolerant properties of active inference. To understand the connection between the latter and the approaches nowadays implemented for fault tolerance, the main components of a fault tolerant scheme are presented. The concepts reported in this section and the related subsections are mainly based on the theory explained in [2, 7].*

In a non-ideal world, technology is vulnerable to faults and failures. Sensor faults, actuator faults, or plant faults, can make a system behave in an non-optimal manner, leading in some cases to a complete break-down. It is clear, then, the importance of designing a control system capable of managing critical situations. But what do faults and failures exactly mean? A failure represents the inability of a system (or a component) to fulfill its function, and it is irrecoverable. A fault is a modification of the characteristics of a component, such that its performance (or mode of operation) is changed in an undesired way. In this view, a fault tolerant controller has to prevent a component fault to cause a failure. The design of such a controller consists of two main steps [2]:

- **Fault diagnosis**: the fault detection and isolation (FDI) of the faults;
- **Control re-design**: the adaptation of the controller parameters or structure to the faulty situation.

Usually, fault diagnosis and control re-design are not carried out by a classical feedback controller but from another *supervision system*, as depicted in Figure 2-3.

It has to be noticed that faults $\boldsymbol{f}$ and disturbances $\boldsymbol{d}$ are different. In fact, only the latter are usually taken into account by the default controller, which compensates for them. Moreover, from Figure 2-3, it can be seen that extra elements have to be added to the overall scheme to perform fault detection and classification, and to adapt the controller to new situations. Furthermore, in the diagnosis block, a dynamic model of the system subject to fault has to be implemented for FDI. There are, to some extent, some ways to avoid these external components, using for instance adaptive schemes. However, these are usually suitable for a restricted family of faults, or for slowly changing process dynamics.

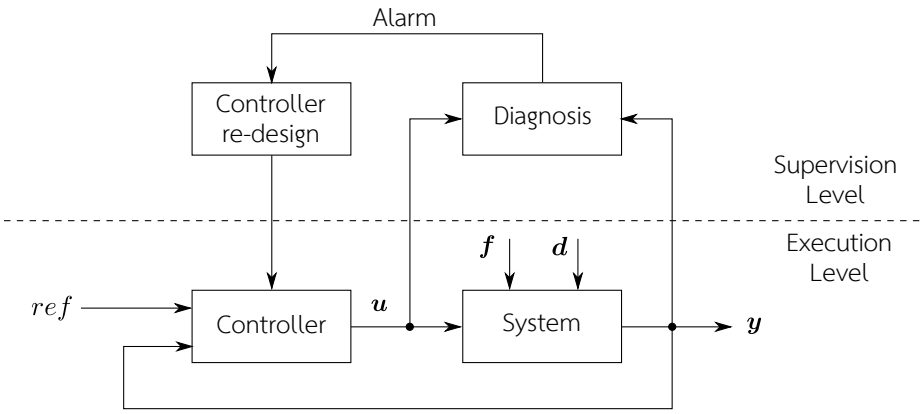



**Figure 2-3:** Architecture of a fault tolerant control system from [2]

### 2-3-1 Model-based fault diagnosis

The literature about fault diagnosis, detection and isolation is vast. In this work attention is paid to the most advanced technique which is nowadays implemented, namely a model-based approach [7]. This choice has been made since this class of methods makes use of process models and analytical redundancy. As it will be explained in Chapter 4, this peculiarity allows to extend the concepts from model-based approaches to the free-energy principle, in order to perform FDI.

The model-based methods monitor the trend of certain signals, called *residuals*, for fault diagnosis. These signals are then compared to a threshold, and faults are recognised if the threshold is exceeded.

More in detail [7], the first step in a model-based approach is to build a mathematical model of the system in play, to generate the residuals. The latter are defined as quantities that reflect the inconsistency between the actual system variables and their mathematical description. Model-based approaches exploit the concept of analytical redundancy, intended as a mathematical model of a measurable quantity. Analytical redundancy is used to cross-check the value of a variable, and it is the main concept on which the residual generation rests. In order to use the residuals for fault diagnosis, as mentioned before, both *residual generation* and *evaluation* have to be performed.

#### Residual generation

A general framework for residual generation, as described in [7], is now presented. To introduce the idea of residual, the dynamics of a system in which possible faults could occur has to be defined. To explain this, a MIMO linear system is considered as an example. Such a system, in absence of any fault, is described by the following dynamics:

$$\begin{cases} \dot{\boldsymbol{x}}(t) = A\boldsymbol{x}(t) + B\boldsymbol{u}_R(t) \\ \boldsymbol{y}_R(t) = C\boldsymbol{x}(t) + D\boldsymbol{u}_R(t) \end{cases} \tag{2-42}$$

where $\boldsymbol{x}(t)$ is the state vector, $\boldsymbol{y}_R(t)$ is the real output vector of the system, $\boldsymbol{u}_R(t)$ is the real input vector to the the system. The matrices $A$, $B$, $C$, and $D$ are the conventional state-space

matrices. In the presence of a fault, the above dynamics have to be adjusted slightly. First, a general scheme of a system subject to fault is reported:

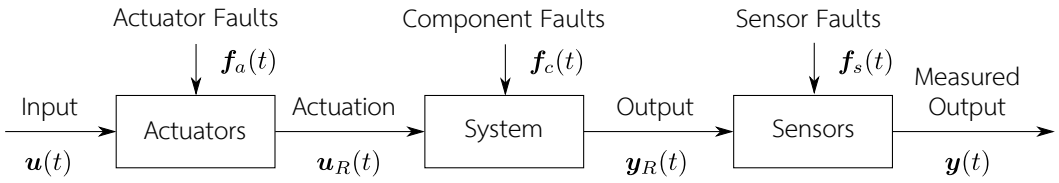



**Figure 2-4:** Faulty open-loop system from [7]

From Figure 2-4, one can observe the presence of three possible kinds fault, described by their relative functions. In particular there can be: actuator faults $\boldsymbol{f}_a(t)$, plant components faults $\boldsymbol{f}_c(t)$, and sensor faults $\boldsymbol{f}_s(t)$. Remember that $\boldsymbol{u}_R$ is the actuator response to an actuator command $\boldsymbol{u}(t)$. Neglecting the actuator dynamics it holds that:

$$\boldsymbol{u}_R(t) = \boldsymbol{u}(t) + \boldsymbol{f}_a(t) \tag{2-43}$$

Considering also the other two fault functions and neglecting the sensors dynamics, one can define a new system of equations to describe the plant dynamics in a more general way:

$$\begin{cases} \dot{\boldsymbol{x}}(t) = A\boldsymbol{x}(t) + B(\boldsymbol{u}(t) + \boldsymbol{f}_a(t)) + \boldsymbol{f}_c(t) \\ \boldsymbol{y}_R(t) = C\boldsymbol{x}(t) + D(\boldsymbol{u}(t) + \boldsymbol{f}_a(t)) + \boldsymbol{f}_s(t) \end{cases} \tag{2-44}$$

which can be rewritten as:

$$\begin{cases} \dot{\boldsymbol{x}}(t) = A\boldsymbol{x}(t) + B\boldsymbol{u}(t) + R_1\boldsymbol{f}(t) \\ \boldsymbol{y}(t) = C\boldsymbol{x}(t) + D\boldsymbol{u}(t) + R_2\boldsymbol{f}(t) \end{cases} \tag{2-45}$$

where $\boldsymbol{f}(t)$ is a fault vector and each entry corresponds to a specific fault. $R_1$ and $R_2$ are the fault matrices representing the effects of the faults on the system. Note that here $\boldsymbol{u}(t)$ is the input to the actuators (from the controller) and $\boldsymbol{y}(t)$ is the measured output (from the sensors). These two quantities are known.

The **residual generation** considering the available signals $\boldsymbol{u}(t)$ and $\boldsymbol{y}(t)$ is now analysed. For this purpose, a schematic is reported in Figure 2-5, where the system in Figure 2-4 is grouped in a single block and the general mechanism of the residual generation is illustrated.

Figure 2-5 is a general scheme which highlights two different steps, namely the generation of the redundant signal $\boldsymbol{z}(t)$ and the generation of the residual signal $\boldsymbol{r}(t)$. In particular, in this scheme, a function of the input and output $F_1(\boldsymbol{u}, \boldsymbol{y})$ generates a redundant signal which is used by the function $F_2(\boldsymbol{z}, \boldsymbol{y})$ to compute $\boldsymbol{r}(t)$:

$$\boldsymbol{r}(t) = F_2(\boldsymbol{z}(t), \boldsymbol{y}(t)) = F_2(F_1(\boldsymbol{u}(t), \boldsymbol{y}(t)), \boldsymbol{y}(t)) \tag{2-46}$$

Different approaches can be used to define the functions $F_1$ and $F_2$. The simplest solution is to use a copy of the system dynamics as $F_1$. In this way, only the input $\boldsymbol{u}(t)$ is required by

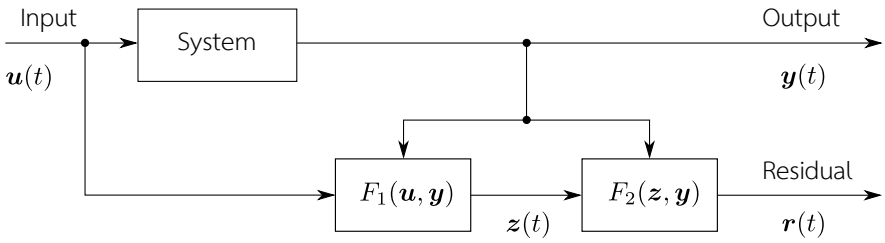



**Figure 2-5:** Redundant structure for residual generation from [7]

$F_1$, which acts as a *system simulator*, replicating the system output $\boldsymbol{y}(t)$ as $\boldsymbol{z}(t)$. Then $F_2$ can be, for instance, a simple subtraction. However, this would result in an open-loop approach which is prone to output drifting due to unavoidable unmodeled dynamics. An extension to this simulator-based residual generation, is to use an *output estimator* instead of a copy of the system. Doing so, both signals $\boldsymbol{u}(t)$ and $\boldsymbol{y}(t)$ are used. In any case, no matter what type of approach is implemented to define $F_1$ and $F_2$, a residual generator is usually just a linear processor with input the signals $\boldsymbol{u}(t)$ and $\boldsymbol{y}(t)$ of the system being monitored. In a mathematical formulation [7]:

$$\boldsymbol{r}(t) = F_2(\boldsymbol{z}(t), \boldsymbol{y}(t)) = Q(\boldsymbol{z}(t) - \boldsymbol{y}(t)) \tag{2-47}$$

where $Q$ is a static (or dynamic) weighting matrix.

The framework presented so far is general, and the selection of the two functions $F_1$ and $F_2$ defines different parametrizations of the residuals. When the residual generation is properly defined, a *residual evaluation function* $J(\boldsymbol{r}(t))$ and a certain *threshold* $\boldsymbol{\tau}(t)$ are necessary to detect the presence of a fault.

#### Residual evaluation

Residual evaluation is a fundamental part of fault detection. A number of evaluation schemes are available and described in the literature. In particular, statistical methods and norm based evaluation methods are the most popular [9] and often used for residual evaluation in model-based approaches. Both schemes create a threshold according to model uncertainties, noise and residual features. In general, residual evaluation can be represented as in Figure 2-6, where exceeding the threshold indicates a fault in the system. In particular, it holds that:

$$\begin{cases} J(r(t)) \leq \boldsymbol{\tau}(t), & \text{for } \boldsymbol{f}(t) = 0 \\ J(r(t)) > \boldsymbol{\tau}(t), & \text{for } \boldsymbol{f}(t) \neq 0 \end{cases} \tag{2-48}$$

We will see later on how the free-energy could be used to facilitate the definition of $\boldsymbol{r}(t)$, $J(\boldsymbol{r}(t))$ and $\boldsymbol{\tau}(t)$.

### 2-3-2 Controller re-design

The last part of a fault tolerant control scheme is the controller-redesign. The ability to change the default controller parameters (or structure) in order to fulfill the specifications in

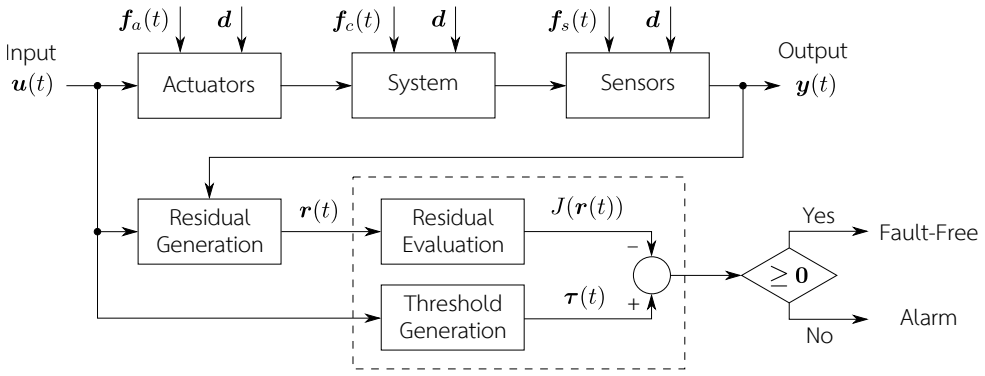



**Figure 2-6:** Residual evaluation and threshold generation from [9]

case of fault, is indeed crucial. This task can be really complicated and sometimes unfeasible. This can happen when unstable modes of the system become uncontrollable or unobservable due to the fault. To try to solve the problem of controller re-design, two general approaches can be found in the literature:

- **Fault accommodation**: adapting the controller parameters in relation to the dynamical properties of the plant subject to a fault;
- **Control reconfiguration**: selecting a new control configuration and re-designing the controller online. This is necessary when severe faults occur, like sensor faults, actuators or plant faults. This kind of faults usually leads to unobservability or uncontrollability.

Some considerations for-real time applications deserve to be done here. Fault accommodation and control reconfiguration have to be completely automatic, and a solution, even if not optimal, has to be found as quickly as possible. The switching and/or adaptation of controllers can require a certain amount of time, and this interval can be seen as a time delay that can possibly lead to instability. The controller re-design block is a hard component to design, which depends on the controlled system and the supposed faults. Common approaches consider modality switching to recover from large faults [2, 19, 33]. This means that for each supposed faulty situation, a working controller is designed. Then, once a fault is detected and isolated, the supervision system switches to the specifically designed controller for recovery. This clearly increases the complexity of the overall architecture.

### Relation between free-energy and model-based FDI and recovery

So far we have seen the main steps to be performed to achieve fault tolerant control. The choice of the external supervision system for the residual generation and evaluation, is influenced by the process to be controlled and by the controller itself. The same happens for the controller re-design part. Designing an external supervision system can then become a quite challenging task.

One question might then arise: is there a way to avoid the external residual generation and evaluation using an intrinsically fault tolerant control scheme? And also, can the modality switching be avoided for recovery? Active inference is a possible candidate to achieve so.

The FEP provides a structure which could be used to obtain an intrinsically fault tolerant controller. First of all, its adaptability properties could be beneficial in case of smaller faults such as limited decrease of actuator efficiency, or increase of friction coefficients etc. Besides, the framework considers as sensory input $\boldsymbol{y}$ for the state inference, a generic $m$-dimensional vector. Since the whole sensory set contributes to the state estimation, this could be exploited to facilitate the inclusion of redundant sensors, to recovery in case of sensory faults. Analysing both the sensory prediction errors in the FEP, and the residuals in a fault tolerant sense, one can notice some strong similarities. Due to the free-energy minimisation, in fact, the sensory prediction errors are quantities which measure the mismatch between what it is sensed and what it is predicted to be sensed. In normal working conditions these terms should approach zero. However, in case of sensory faults, the sensory prediction error relative to the faulty sensor should manifest an unusual increase. Besides, since the free-energy is a positive quantity, one could calculate an upper bound to be used as threshold for fault detection. Finally, the recovery actions in case of sensory faults, could be directly embedded in the active inference controller. Remember that the controller encodes its confidence with respect to the sensory input through the covariance matrices $\Sigma_{y^{(i)}}$. Increasing the variance associated to a faulty sensor would exclude the malfunctioning sensor from both the inference process and the control action computation. Thus, also the recovery could be implmented in the controller in a natural way. To conclude, Table 2-2 summarises the intuitions behind free-energy for fault tolerance.

**Table 2-2:** Intuitions regarding free-energy for fault tolerance

| **Method** | **Residual generation** | **Residual evaluation** | **Recovery** |
|---|---|---|---|
| *Classical model based* | External, using a model of the process | Norm or statistic based threshold | External, usually with modality switching |
| *Free-energy* | Internal, using the available sensory prediction errors | On-line threshold from sensory prediction errors and $\mathcal{F}$ | Internal, changing the confidence on the sensory input |

## 2-4 Concluding remarks

This chapter introduced the three main topics on which this study is based. The definition of the free-energy principle and active inference provides that both state estimation and control can be performed through the minimisation of one single cost function. The framework heavily relies on the defined generative models for the sensory data and the state dynamics. But how should these models be defined? And also, what kind of benefits can active inference bring for robot control? If we find a way of properly defining these generative models, we could obtain a lightweight formulation of active inference for on-line applications. Then, we could take advantage of the adaptability performance of the framework, and obtain a novel control scheme.

In order to understand the relevance of such a novel controller, a measure of performance had to be defined. The choice of using a model reference adaptive controller for performance comparison, is motivated by the similar characteristics that the controllers share. The two

schemes are indeed both scalable to high DOF, and they do not require the dynamic equations of the robot to be controlled. Besides, the MRAC is a good candidate as a benchmark for evaluating the claimed adaptability performance of the active inference controller (AIC).

Finally, the initial study of active inference suggested its possible applicability as a fault tolerant controller. To verify this, we reported the general theory regarding model-based fault tolerant control, and we pointed out the similarities in terms of residual generation that this approach shares with the free-energy. We also saw how traditional fault tolerant solutions require external supervision systems, and a dynamical model of the process. The free-energy appears to be useful to avoid all these components. In particular, the intuition which lies behind this work, is that the sensory prediction errors could be used as residuals for fault detection and isolation. Besides, the free-energy would facilitate the implementation of sensory redundancy, and it would provide a way to weight the sensory input to perform fault recovery. This idea will result in an elegant way of performing fault tolerance only exploiting the internal signal of the controller.

# Chapter 3

# Robot arm control with active inference

*This chapter presents a novel active inference control scheme for a generic n-DOF manipulator. The algorithm allows to control the robot in joint space, directly providing the torques to be applied to the motors. To illustrate the properties of active inference, two examples with a 2-DOF robot arm are reported. Then, the adaptability properties of the control algorithm are tested in a simulated pick and place cycle with a 7-DOF robot arm. The resulting performance is compared to an MRAC in case of large model uncertainties, showing that the AIC can outperform the other adaptive controller. The chapter is then concluded with a summary regarding the benefits and limitations of the proposed solution.*

## 3-1 Free-energy and active inference from a control perspective

In Chapter 2, the free-energy principle and active inference have been presented, focusing on their biological inspiration. The concepts of sensory set, control actions and environmental states, can however be seen from a more control engineering perspective. In a classical control problem, the system can be described with a set of states representing the physical quantities to be controlled. Then, the control input is computed making use of a model of the system, a set of sensors, and a control law. The analogy with the quantities described in the active inference framework, is then self explanatory. Since the goal of this thesis is to perform on-line control of a robot manipulator in joint space, a clear connection with active inference can be drawn, as reported in Table 3-1.

It is important to notice that, in a classical control scheme, there will be a dedicated filtering block for the state estimation, and a control block to perform reference tracking. From this point of view, given a system to be steered to a desired goal, active inference provides both state estimation and control input through the minimization of one single cost function, the free-energy.

**Table 3-1:** Biological inspiration in engineering terms for robot control

| Inspiration | Sensory data | Beliefs update | Control actions |
|---|---|---|---|
| *Biological* | Proprioception | Sensory explanation | Stimuli to the muscles |
| *Control engineering* | Encoders and tachometers | Filtering, state estimation | Torques to the motors |

In the next sections, we adapt the theory presented in the previous chapter to the robot control case. First, the simplest case of static state estimation through free-energy minimisation in a multivariate case is explained. Then, the control actions are included in the framework to obtain a closed-loop implementation of active inference. The results are first presented through a 2-DOF example, to appreciate the theoretical aspects described in Chapter 2. Then, the derived general control law is applied to a more complex 7-DOF manipulator, for performance comparison with the MRAC.

## 3-2 Free-energy minimisation for static state estimation

The simplest implementation of the free-energy principle, is to perform state estimation in a static scenario without any control action allowed. Such a case is analysed in the following, where the general state update law for an $n$-DOF robot manipulator is derived. Since we consider only perception in a static case, there is no need to use the generalised motions to describe the dynamic evolution of the states.

For this case, we suppose that the only sensory information available is the one from noisy encoders. In a more complex scenario, multisensory data can be considered to have a better a posteriori approximation of the states. This is particularly useful in presence of a relevant level of noise, but also in case of uncalibrated or even faulty sensors. The multisensory case is presented in Chapter 4, when sensory faults are injected in the system.

As a starting point to derive the static state update expression, we recall the general formulation of the free-energy given in Chapter 2. In a multivariate case, the free-energy is represented by:

$$\mathcal{F} = -\ln p(\boldsymbol{\mu}, \boldsymbol{y}_q) \tag{3-1}$$

where $\boldsymbol{\mu}$ is the belief about the states of the robot, and $\boldsymbol{y}_q$ is the available noisy measurement of the joints position $\boldsymbol{q}$. The static state perception will make the beliefs converge to the true states. But *what should be chosen as state*? A reasonable choice is to consider the joint positions as states of the robot to be estimated. This will be helpful also later on when the control of the robot in joint space will be presented. This choice, indeed, considerably simplifies the overall control structure. To summarise, the state estimate $\boldsymbol{\mu} \in \mathbb{R}^n$ represents the posterior approximation of the joint values, given the noisy encoders' output $\boldsymbol{y}_q$. The inference process will make $\boldsymbol{\mu}$ converge to the true joint values $\boldsymbol{q}$. According to the fundamental rule of probability it holds that:

$$p(\boldsymbol{\mu}, \boldsymbol{y}_q) = p(\boldsymbol{y}_q|\boldsymbol{\mu})p(\boldsymbol{\mu}) \tag{3-2}$$

Eq. (3-2) reads as: the joint probability of being in a state $\boldsymbol{\mu}$ and sensing $\boldsymbol{y}_q$, is given by the probability of sensing $\boldsymbol{y}_q$ once given the state $\boldsymbol{\mu}$, times the probability of being in $\boldsymbol{\mu}$. Substituting Eq. (3-2) in Eq. (3-1) leads to:

$$\mathcal{F} = -\ln p(\boldsymbol{y}_q|\boldsymbol{\mu}) - \ln p(\boldsymbol{\mu}) \tag{3-3}$$

To express these two probabilities, the generative models of the sensory data and state dynamics need to be specified.

**Generative model of the sensory data** The sensory data is modeled through a noisy non-linear mapping between the state estimate $\boldsymbol{\mu}$ and the encoders' output. Following [11], we can write:

$$\boldsymbol{y}_q = \boldsymbol{g}_q(\boldsymbol{\mu}) + \boldsymbol{z} \tag{3-4}$$

where $\boldsymbol{g}_q(\boldsymbol{\mu})$ is a the generative function, and $\boldsymbol{z}$ is some Gaussian noise $\boldsymbol{z} \sim (\boldsymbol{0}, \Sigma_{y_q})$. Since we chose the states to be the joint positions, and the sensory data provides directly $\boldsymbol{q}$, it holds:

$$\boldsymbol{g}_q(\boldsymbol{\mu}) = \boldsymbol{\mu} \tag{3-5}$$

Rewriting Eq. (3-4) as $\boldsymbol{z} = \boldsymbol{y}_q - \boldsymbol{g}_q(\boldsymbol{\mu})$ and making use of the Laplace assumption, allows to express $p(\boldsymbol{y}_q|\boldsymbol{\mu})$ as:

$$p(\boldsymbol{y}_q|\boldsymbol{\mu}) = \frac{1}{|\Sigma_{y_q}| \sqrt[n]{2\pi}} exp\left\{-\frac{1}{2}(\boldsymbol{y}_q - \boldsymbol{g}_q(\boldsymbol{\mu}))^\top \Sigma_{y_q}^{-1}(\boldsymbol{y}_q - \boldsymbol{g}_q(\boldsymbol{\mu}))\right\} \tag{3-6}$$

where $|\Sigma_{y_q}|$ is the determinant of the covariance matrix.

**Generative model of the state dynamics** Since the environment is static, we encode in the controller a static generative model. In this way, the controller assumes that the states are fluctuating around a mean $\boldsymbol{\mu}_d$ [5]:

$$\boldsymbol{\mu} = \boldsymbol{\mu}_d + \boldsymbol{w} \tag{3-7}$$

where $\boldsymbol{w}$ is Gaussian noise $\boldsymbol{w} \sim (\boldsymbol{0}, \Sigma_\mu)$. Similarly, then:

$$p(\boldsymbol{\mu}) = \frac{1}{|\Sigma_\mu| \sqrt[n]{2\pi}} exp\left\{-\frac{1}{2}(\boldsymbol{\mu} - \boldsymbol{\mu}_d)^\top \Sigma_\mu^{-1}(\boldsymbol{\mu} - \boldsymbol{\mu}_d)\right\} \tag{3-8}$$

### 3-2-1 Free-energy and beliefs update in a static environment

*The static state perception here presented, is an extended version to a robotic example of the 1-DOF thermostat example presented in [5], making use of [4] and [25].*

#### Free-energy expression

The Laplace assumption regarding Gaussian distributed probability densities, allows to simplify the expression of $\mathcal{F}$. Substituting Eq. (3-6) and Eq. (3-8) in Eq. (3-3), leads to:

$$\mathcal{F} \approx \frac{1}{2}(\boldsymbol{y}_q - \boldsymbol{g}_q(\boldsymbol{\mu}))^\top P_{y_q}(\boldsymbol{y}_q - \boldsymbol{g}_q(\boldsymbol{\mu})) + \frac{1}{2}(\boldsymbol{\mu} - \boldsymbol{\mu}_d)^\top P_\mu(\boldsymbol{\mu} - \boldsymbol{\mu}_d) \in \mathbb{R} \tag{3-9}$$

In the expression above, the positive constant terms resulting from the logarithm of the Gaussian distribution have been neglected, since they will not play any role in the minimisation of $\mathcal{F}$ for state estimation. Furthermore, the inverse covariance matrices $\Sigma_{y_q}^{-1}$ and $\Sigma_{\mu}^{-1}$ have been renamed as precision matrices $P_{y_q}$ and $P_{\mu}$ respectively. In general, $P_y$ indicates a precision matrix for a generic sensory input. In this view, a lower precision matrix would result in a lower confidence of the controller regarding either the sensory input or the internal estimate about the states. This term will be crucial while performing recovery from sensory faults, as explained in Chapter 4.

### Beliefs update equation for static state estimation

To perform state estimation in the static case, a gradient descent on the free-energy is used. In particular, the state inference is provided by:

$$\dot{\boldsymbol{\mu}} = -\kappa_{\mu}\partial_{\mu}\mathcal{F} \in \mathbb{R}^n \tag{3-10}$$

were $\kappa_{\mu} > 0$ is the learning rate to adjust the rate of convergence. Recalling that, for a generic vector $\boldsymbol{v}$ it holds $\partial_v(\boldsymbol{v}^\top A\boldsymbol{v}) = \boldsymbol{v}^\top(A + A^\top)$, and that the precision matrices are symmetric, we can write the following state update equation:

$$\dot{\boldsymbol{\mu}} = -\kappa_{\mu}\left[-(\boldsymbol{y}_q - \boldsymbol{g}_q(\boldsymbol{\mu}))P_{y_q}\partial_{\mu}\boldsymbol{g}_q(\boldsymbol{\mu}) + (\boldsymbol{\mu} - \boldsymbol{\mu}_d)P_{\mu}\right] \tag{3-11}$$

Finally, since the states have been chosen as the joint values of the robot manipulator, and $\boldsymbol{g}_q(\boldsymbol{\mu}) = \boldsymbol{\mu}$, it holds $\partial_{\mu}\boldsymbol{g}_q(\boldsymbol{\mu}) = 1$.

### Remarks

A detail which deserves attention, is the role of the mean $\boldsymbol{\mu}_d$. The FEP defines this quantity as a prior expectation regarding the states of the environment. In control terms, $\boldsymbol{\mu}_d$ can be seen as the desired goal to be reached. In a sense, this term encodes the desire of the controller to steer the states to a specific value. If, as in this case, the control actions are suppressed, the only way to completely minimise $\mathcal{F}$, is when the desired goal coincides with the actual robot position. The free-energy minimisation will then fulfill the prior expectation encoded in the controller, through state estimation only.

In case $\boldsymbol{\mu}_d$ differs from the actual joint values of the robot, the free-energy minimisation will settle to a local minimum. The controller cannot satisfy its desire to steer the states to a specific set-point since no actions are allowed. In this situation, the most probable state $\boldsymbol{\mu}$, will be a trade-off between the sensed and the desired states. The posterior estimate qualifies as minimiser for $\mathcal{F}$, which in this case will be stuck in a local minimum. The only way to solve this discrepancy is through the control actions. Once again, it becomes clear the coupling between state estimation and control actions which usually remains separate in classical control solutions.

However, this phenomenon should not be seen as a limitation of the algorithm. In fact, the interconnection between actions and perception is the main reason behind the adaptability properties of active inference, as we will see in the next sections. In other words, the actions computed by the controller aims at changing the sensed values in order to match the prior

expectation. To conclude, the best situation is found when the controller is sensing what it expects to sense, and when its internal beliefs about the states coincide with their desired values. With the next example we illustrate these concepts.

### 3-2-2 2-DOF example

To illustrate the free-energy minimisation for state estimation in a static environment, a simple 2-DOF example is here presented. The complete MATLAB implementation is reported in Appendix A-1. The initial free-energy minimisation was based on the tutorial from Bogacz [4], and subsequently adapted and extended with the control actions. First of all, the dynamic model of a 2-DOF robotic arm as in Figure 3-1 is specified.

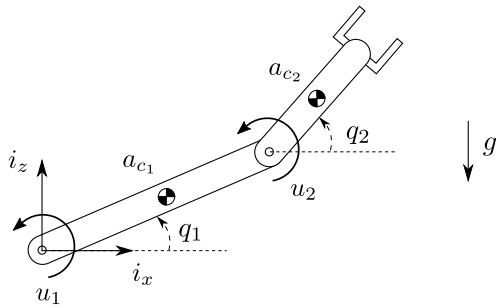



**Figure 3-1:** 2-DOF robot manipulator

Let us consider the dynamic description of the manipulator as follows [43]:

$$\boldsymbol{u}(t) = M(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + D\dot{\boldsymbol{q}} + g_t(\boldsymbol{q}) \tag{3-12}$$

where $\boldsymbol{q} = [q_1,\ q_2]^\top$, $\boldsymbol{u}(t) = [u_1,\ u_2]^\top$. Furthermore, the inertia matrix is given by:

$$M(\boldsymbol{q}) = \begin{pmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{pmatrix} \tag{3-13}$$

Each term of the symmetric inertia matrix are defined as:

$$\begin{aligned} M_{11} &= m_1 a_{C1}^2 + m_2[a_1^2 + a_{C_2}^2 + 2a_1 a_{C_2} cos(q_2)] + J_1 + J_2 \\ M_{12} &= M_{21} = m_2[a_{C_2}^2 + a_1 a_{C_2} cos(q_2)) + J_2 \\ M_{22} &= m_2 a_{C_2}^2 + J_2 \end{aligned} \tag{3-14}$$

The terms $J_i$, $m_i$ and $a_i$ for $i = 1, 2$, are respectively the inertia, the masses and the lengths of the two links. The Coriolis matrix is instead:

$$C(q, \dot{q}) = \begin{pmatrix} -m_2 a_1 a_{C_2} sin(q_2)\dot{q}_2 & -m_2 a_1 a_{C_2} sin(q_2)(\dot{q}_1 + \dot{q}_2) \\ m_2 a_1 a_{C_2} sin(q_2)\dot{q}_1 & 0 \end{pmatrix} \tag{3-15}$$

The gravity term $g_t = [g_1\ g_2]^\top$ is defined as:

$$\begin{aligned} g_1 &= (m_1 a_{C_1} + m_2 a_1) cos(q_1) g + m_2 a_{C_2} cos(q_1 - q_2) g \\ g_2 &= m_2 a_{C_2} cos(q_1 - q_2) g \end{aligned} \tag{3-16}$$

and the matrix $D$ of friction coefficients is simply $D = diag\{d_1, d_2\}$. The full model will be used when the control actions will be allowed, to simulate the response of the controlled robot arm. For now, the robot is assumed to be in a the static configuration, and the noisy joint values from the encoders are used to perform state estimation.

### Simulation results

For this simulation, the robot arm is fixed in a configuration in joint space. Specifically, $\boldsymbol{q} = [0.3,\ 0.8]^\top\ [rad]\ \forall t > 0$. The sensory input is affected, without loss of generality, by normally distributed Gaussian noise with standard deviation 0.001 $[rad]$. The noise in the two encoders is supposed uncorrelated [5] such that $\boldsymbol{z} \sim (\boldsymbol{0}, 0.001 I_2)$, with $I_2$ two dimensional identity matrix. The actual simulated measurement of the joint positions is then coded as $\boldsymbol{y}_q = \boldsymbol{q} + \boldsymbol{z}$.

**State estimation which meets the desired prior expectation** The prior beliefs about the states $\boldsymbol{\mu_d}$ is initially set equal to the actual robot position, so the desire of the controller is actually met when the states are correctly estimated. In other words when $\boldsymbol{\mu}_d = \boldsymbol{q}$. This allows to evaluate just the effect of perception using to sensory data.

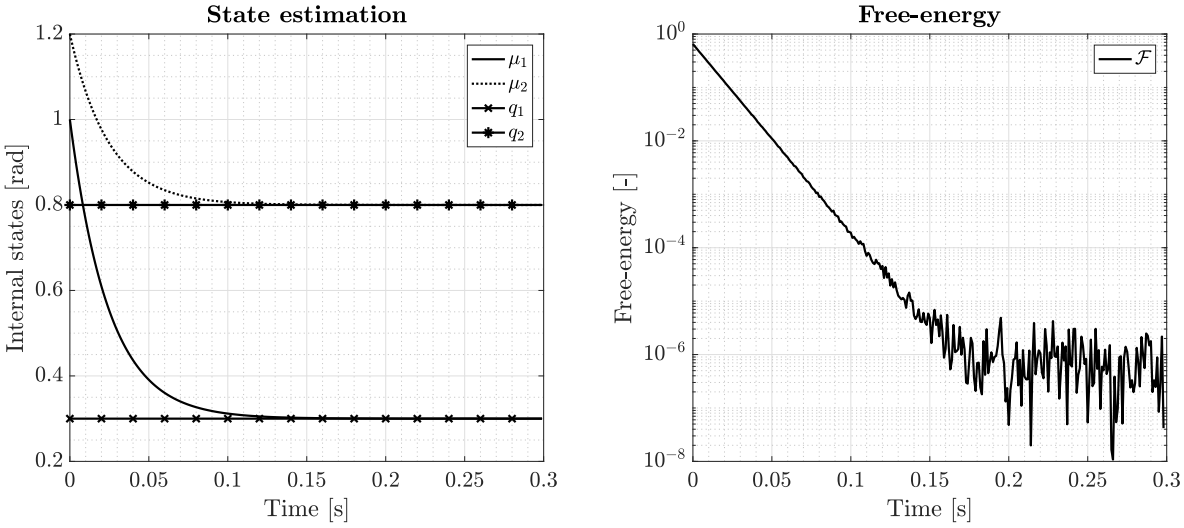



**Figure 3-2:** Free-energy minimisation for state estimation in a static case. $\boldsymbol{\mu}_d = \boldsymbol{q}$

The two equations to be implemented for static state estimation, are given by Eq. (3-9) and Eq. (3-11). The precision matrices $P_{y_q}$ and $P_\mu$ are assumed to be the identity, to evaluate the state estimation with equally weighted sensory input and internal beliefs. The learning rate $\kappa_\mu$ for the update is set to 20 for a fast convergence, and the integration step is set to 1 $[ms]$.

**State estimation with different prior expectation** We now assume that the prior expectation $\boldsymbol{\mu_d}$ differs from the actual robot configuration, imposing for instance, $\boldsymbol{\mu_d} = [0.4,\ 0.6]$.

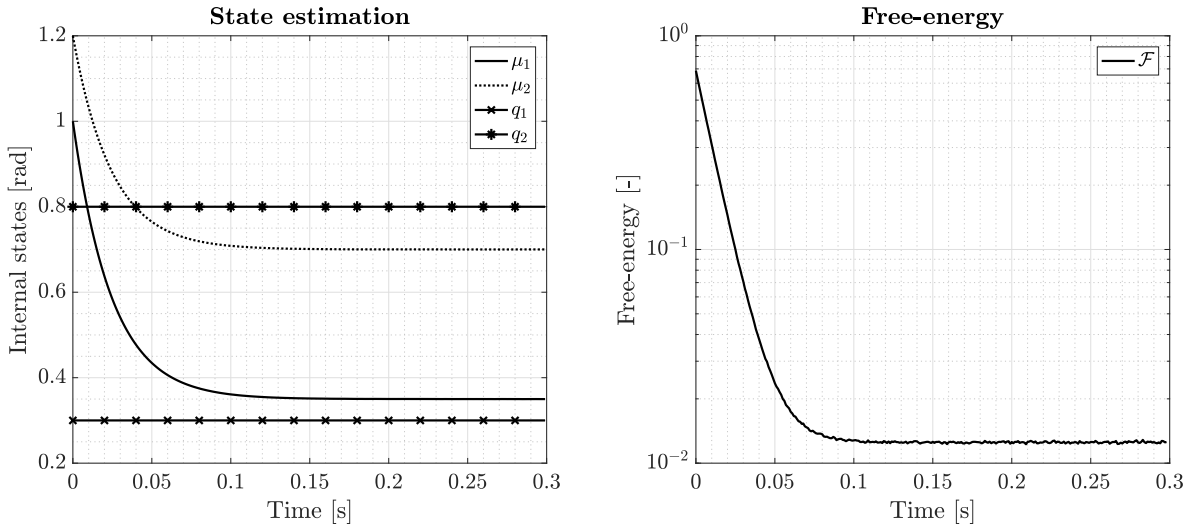



**Figure 3-3:** Free-energy minimisation for state estimation in a static case. $\boldsymbol{\mu}_d \neq \boldsymbol{q}$

As can be seen, the beliefs do not converge to the true states due to the discrepancy between sensed and desired values to be sensed. This situation will be solved allowing the control actions to further minimise the free-energy, and meet the control objective.

## 3-3 Active inference for robot control

In this section, we present a novel active inference controller (AIC) for a generic $n$-DOF manipulator controlled in joint space. The resulting approach represents a general model-free control law, which provides high adaptability against unmodeled dynamics, and high scalability to increasing number of degrees of freedom. Since the controller will steer the robot to a desired set-point, the states will be characterized by a dynamical evolution. To perform dynamic state estimation, the use of the generalised motions is then necessary, in contrast with the previous example.

### Generalised motions

With the generalised motions, we consider that the states are described by increasingly higher-order derivatives, up to the selected order $n_d$. The generalised state vector is represented as:

$$\tilde{\boldsymbol{\mu}} = [\boldsymbol{\mu},\ \boldsymbol{\mu}',\ \boldsymbol{\mu}'',\ \boldsymbol{\mu}''',\ ...,\ \boldsymbol{\mu}^{(n_d)}] \tag{3-17}$$

where $n_d$ is the order to be considered. The generalised motions of the beliefs under local linearity assumption [12] are:

$$\begin{aligned} \boldsymbol{\mu}' &= \boldsymbol{\mu}^{(1)} = \boldsymbol{f}(\boldsymbol{\mu}) + \boldsymbol{w} \\ \boldsymbol{\mu}'' &= \boldsymbol{\mu}^{(2)} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{\mu}}\boldsymbol{\mu}' + \boldsymbol{w}' \\ &\vdots \end{aligned} \tag{3-18}$$

Similarly, the generalised motions of the sensory input are given by:

$$\begin{aligned} \boldsymbol{y} &= \boldsymbol{y}^{(0)} = \boldsymbol{g}(\boldsymbol{\mu}) + \boldsymbol{z} \\ \boldsymbol{y}' &= \boldsymbol{y}^{(1)} = \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{\mu}} \boldsymbol{\mu}' + \boldsymbol{z}' \\ &\vdots \end{aligned} \tag{3-19}$$

The generalised sensory input is also represented as:

$$\tilde{\boldsymbol{y}} = [\boldsymbol{y},\ \boldsymbol{y}',\ \boldsymbol{y}'',\ \boldsymbol{y}''',\ ...,\ \boldsymbol{y}^{(n_d-1)}] \tag{3-20}$$

The generalised motions can extend up to infinite order. However, the noise related to high orders is predominant, allowing to decide on the number of derivatives to consider [15].

### 3-3-1 Free-energy in a dynamic environment

A general expression for the free-energy for a robot manipulator, using generalised motions, is derived starting from:

$$\mathcal{F} = -\ln p(\tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{y}}) = -\ln p(\tilde{\boldsymbol{y}}|\tilde{\boldsymbol{\mu}}) p(\tilde{\boldsymbol{\mu}}) \tag{3-21}$$

The likelihood of the sensory data $p(\tilde{\boldsymbol{y}}|\tilde{\boldsymbol{\mu}})$ and the prior $p(\tilde{\boldsymbol{\mu}})$ have to be specified. To do so, a few considerations have to be made. According to [5] and to the assumption previously given, the noise at each dynamical order is considered uncorrelated. Then, according to Eq. (3-19), the sensory data at a particular order relates only with the states at the same dynamical order. Similarly, for the state dynamics, the state at a certain dynamical order are related only with those which are one order below. This allows us to write:

$$p(\tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{y}}) = \prod_{i=0}^{n_d-1} p(\boldsymbol{y}^{(i)}|\boldsymbol{\mu}^{(i)}) p(\boldsymbol{\mu}^{(i+1)}|\boldsymbol{\mu}^{(i)}) \tag{3-22}$$

Then, taking the logarithm:

$$\mathcal{F} = -\sum_{i=0}^{n_d-1} \left[\ln p(\boldsymbol{y}^{(i)}|\boldsymbol{\mu}^{(i)}) + \ln p(\boldsymbol{\mu}^{(i+1)}|\boldsymbol{\mu}^{(i)})\right] \tag{3-23}$$

Using the Laplace assumption, and thus considering Gaussian distributed probability densities, we can write:

$$p(\boldsymbol{y}^{(i)}|\boldsymbol{\mu}^{(i)}) = \frac{1}{|\Sigma_{y^{(i)}}| \sqrt[n]{2\pi}} exp\left\{-\frac{1}{2}(\boldsymbol{y}^{(i)} - \boldsymbol{g}(\boldsymbol{\mu})^{(i)})^\top \Sigma_{y^{(i)}}^{-1} (\boldsymbol{y}^{(i)} - \boldsymbol{g}(\boldsymbol{\mu})^{(i)})\right\} \tag{3-24}$$

$$p(\boldsymbol{\mu}^{(i+1)}|\boldsymbol{\mu}^{(i)}) = \frac{1}{|\Sigma_{\mu^{(i)}}| \sqrt[n]{2\pi}} \exp\left\{-\frac{1}{2}(\boldsymbol{\mu}^{(i+1)} - \boldsymbol{f}(\boldsymbol{\mu})^{(i)})^\top \Sigma_{\mu^{(i)}}^{-1} (\boldsymbol{\mu}^{(i+1)} - \boldsymbol{f}(\boldsymbol{\mu})^{(i)})\right\} \tag{3-25}$$

### Free-energy expression with generalised motions

Equipped with the extra theoretical knowledge about generalised motions, we can define an expression for the free-energy for a multivariate case in a dynamically changing environment. This is a general expression which holds for any system with generalised sensory input $\tilde{\boldsymbol{y}}$ and states $\tilde{\boldsymbol{\mu}}$, using $n_d$ generalised motions. Substituting Eq. (3-24) and Eq. (3-25) into Eq. (3-23), leads to express $\mathcal{F}$ as a sum of prediction errors:

$$\mathcal{F} = \frac{1}{2}\sum_{i=0}^{n_d-1}\left[(\boldsymbol{y}^{(i)} - \boldsymbol{g}^{(i)})^\top P_{y^{(i)}}(\boldsymbol{y}^{(i)} - \boldsymbol{g}^{(i)}) + (\boldsymbol{\mu}^{(i+1)} - \boldsymbol{f}^{(i)})^\top P_{\mu^{(i)}}(\boldsymbol{\mu}^{(i+1)} - \boldsymbol{f}^{(i)})\right] \tag{3-26}$$

where $n_d$ is the number of generalised motions chosen and the covariance matrices have been substituted with the equivalent precision matrices. Furthermore, it holds the following relation:

$$\boldsymbol{g}^{(i)} = \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{\mu}}\boldsymbol{\mu}^{(i)}, \;\; \boldsymbol{f}^{(i)} = \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{\mu}}\boldsymbol{\mu}^{(i)}, \;\; \boldsymbol{g}^{(0)} = \boldsymbol{g}, \;\; \boldsymbol{f}^{(0)} = \boldsymbol{f} \tag{3-27}$$

The minimisation of this expression can be done by refining the internal beliefs, thus performing state estimation, but also computing the control actions to fulfill the prior expectations and achieve a desired motion.

The next two sub-subsections describe how this general free-energy equation is adapted for any robot manipulator equipped with joint position and velocity sensors. Before giving the details about the active inference controller, however, the initial assumptions underlining the controller structure are here reported.

**Assumption 1** The robot manipulator is equipped with position and velocity sensors, which respectively provide the two variables $\boldsymbol{y}_q, \;\; \boldsymbol{y}_{\dot{q}} \in \mathbb{R}^n$.

**Assumption 2** The states $\boldsymbol{x}$ are chosen to be the joint positions of the robot manipulator. Doing so, we can control the robot arm in joint space through free-energy minimization, and simplify the equations for states update and control actions.

**Assumption 3** Since only the position and velocity measurements are available, we will consider the generalised motions up to order two, so $n_d = 2$. Doing so, Eq. (3-18) and Eq. (3-19) reduce to:

$$\begin{cases} \boldsymbol{\mu}' &= \boldsymbol{f}(\boldsymbol{\mu}) + \boldsymbol{w} \\ \boldsymbol{\mu}'' &= \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{\mu}}\boldsymbol{\mu}' + \boldsymbol{w}' \end{cases} \qquad \begin{cases} \boldsymbol{y} &= \boldsymbol{g}(\boldsymbol{\mu}) + \boldsymbol{z} \\ \boldsymbol{y}' &= \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{\mu}}\boldsymbol{\mu}' + \boldsymbol{z}' \end{cases} \tag{3-28}$$

**Assumption 4** The Gaussian noise affecting the different sensory channels is uncorrelated. Without loss of generality, we consider that the sensors are affected by the same level of noise. The covariance matrices for sensory input and state beliefs are then represented as:

$$P_{y^{(0)}} = \Sigma^{-1}_{y^{(0)}} = I_n/\sigma_q, \quad P_{y^{(1)}} = \Sigma^{-1}_{y^{(1)}} = I_n/\sigma_{\dot{q}}, \tag{3-29}$$

$$P_{\mu^{(0)}} = \Sigma^{-1}_{\mu^{(0)}} = I_n/\sigma_\mu, \quad P_{\mu^{(1)}} = \Sigma^{-1}_{\mu^{(1)}} = I_n/\sigma_{\mu'} \tag{3-30}$$

where we can see that the controller associates four different precision matrices to map its confidence about sensory input and internal beliefs. Note that $I_n \in \mathbb{R}^{n \times n}$ is the identity matrix.

### Generative models

To be able to numerically evaluate $\mathcal{F}$, the generative models for sensory data and state dynamics are needed. In comparison with the static case previously analysed, the number of sensory input is now increased due to the generalised motions, and the static generative model for the state update is not sufficient anymore.

**Generative models of the sensory data** The generalised sensory data vector $\tilde{\boldsymbol{y}} = [\boldsymbol{y},\ \boldsymbol{y}']$ is the internal representation of the sensory input for positions and velocities. Since we chose the states to be the joint positions, and the sensory data provide the noisy values of $\boldsymbol{y}_q$ and $\boldsymbol{y}_{\dot{q}}$, it follows directly from Eq. (3-28):

$$\begin{cases} \boldsymbol{y}_q &= \boldsymbol{\mu} + \boldsymbol{z} \\ \boldsymbol{y}_{\dot{q}} &= \boldsymbol{\mu}' + \boldsymbol{z}' \end{cases} \qquad \begin{cases} \boldsymbol{y}_q &= \boldsymbol{y} \\ \boldsymbol{y}_{\dot{q}} &= \boldsymbol{y}' \end{cases} \qquad \boldsymbol{g}(\boldsymbol{\mu}) = \boldsymbol{\mu} \tag{3-31}$$

**Generative model of the state dynamics** The function $\boldsymbol{f}(\boldsymbol{\mu})$ is defined taking inspiration from what presented in [5]. The definition of the generative model of the state dynamics is crucial. The techniques implemented in past work, overly complicated the selection of $\boldsymbol{f}(\boldsymbol{\mu})$. For instance, [24] used online regressors to estimate this function along with the generative models of the sensory data. Even if this could turn out to be essential for an unknown process, for the case of robot arm control it is most likely unnecessary. In [38] instead, the generative model of the real world process was modeled using Newton's laws. Both of the previous approaches did not include the prior expectation $\boldsymbol{\mu}_d$ into the generative model, to specify the goal of the controlled system. The author considers the inclusion of $\boldsymbol{\mu}_d$ in $\boldsymbol{f}(\boldsymbol{\mu})$ essential in the definition of an active inference controller.

Having said that, to simplify the controller structure and to reduce the computational complexity, the generative model $\boldsymbol{f}(\boldsymbol{\mu})$ is chosen such that the robot is steered to a desired position $\boldsymbol{\mu}_d$. In other words, the controller believes that the states will evolve in such a way that they will reach the goal $\boldsymbol{\mu}_d$:

$$\boldsymbol{f}(\boldsymbol{\mu}) = \boldsymbol{\mu}_d - \boldsymbol{\mu} \tag{3-32}$$

The value $\boldsymbol{\mu}_d$ is a constant $\in \mathbb{R}^n$ corresponding to the desired set-point for the joints of the robot manipulator. This is an adapted version of the idea proposed by Buckley in [5], were a simple one dimensional case of a particle trying to settle to a specific temperature was detailed. Instead of providing a scalar temperature as set-point, we propose to encode the goal position through an $n$-dimensional vector $\boldsymbol{\mu}_d$.

Once defined $\boldsymbol{f}(\boldsymbol{\mu})$, the generalised motions of the states can be further specified. Given Eq. (3-32), we can simplify Eq. (3-28) to:

$$\begin{cases} \boldsymbol{\mu}' &= \boldsymbol{\mu}_d - \boldsymbol{\mu} + \boldsymbol{w} \\ \boldsymbol{\mu}'' &= -\boldsymbol{\mu}' + \boldsymbol{w}' \end{cases} \tag{3-33}$$

#### Free-energy for n-DOF robot manipulator with position and velocity feedback

Substituting Eq. (3-31) and Eq. (3-33) in Eq. (3-26), leads to the free-energy expression for a generic robot manipulator:

$$\begin{aligned}\mathcal{F} &= \frac{1}{2}(\boldsymbol{y}_q - \boldsymbol{\mu})^\top P_{y^{(0)}}(\boldsymbol{y}_q - \boldsymbol{\mu}) + \frac{1}{2}(\boldsymbol{y}_{\dot{q}} - \boldsymbol{\mu}')^\top P_{y^{(1)}}(\boldsymbol{y}_{\dot{q}} - \boldsymbol{\mu}') \\ &+ \frac{1}{2}(\boldsymbol{\mu}' + \boldsymbol{\mu} - \boldsymbol{\mu}_d)^\top P_{\mu^{(0)}}(\boldsymbol{\mu}' + \boldsymbol{\mu} - \boldsymbol{\mu}_d) + \frac{1}{2}(\boldsymbol{\mu}'' + \boldsymbol{\mu}')^\top P_{\mu^{(1)}}(\boldsymbol{\mu}'' + \boldsymbol{\mu}') \end{aligned} \tag{3-34}$$

Note that this equation holds for any $n$-DOF manipulator equipped with position and velocity sensors. To illustrate the high scalability of this approach, a 2-DOF and a 7-DOF examples using the same equations are presented later on.

### 3-3-2 Beliefs update in a dynamic environment

Now that the free-energy is defined, the general beliefs update law for state estimation is determined from the gradient of $\mathcal{F}$, with respect to each of the states in generalised motions. Assuming a learning rate $\kappa_\mu$ to be tuned, it holds:

$$\dot{\tilde{\boldsymbol{\mu}}} = D\tilde{\boldsymbol{\mu}} - \kappa_\mu \frac{\partial \mathcal{F}}{\partial \tilde{\boldsymbol{\mu}}} \tag{3-35}$$

#### State update for an n-DOF robot manipulator with position and velocity feedback

According to the free-energy principle, the states of the robot manipulator can be estimated using a gradient descent scheme. Recalling that $\tilde{\boldsymbol{\mu}} = [\boldsymbol{\mu},\ \boldsymbol{\mu}',\ \boldsymbol{\mu}'']$, and applying Eq. (3-35) having defined $\mathcal{F}$ as in Eq. (3-34), leads to the following state update law:

$$\begin{aligned}\dot{\boldsymbol{\mu}} &= \boldsymbol{\mu}' - \kappa_\mu[-P_{y^{(0)}}(\boldsymbol{y}_q - \boldsymbol{\mu}) + P_{\mu^{(0)}}(\boldsymbol{\mu}' + \boldsymbol{\mu} - \boldsymbol{\mu}_d)] \\ \dot{\boldsymbol{\mu}}' &= \boldsymbol{\mu}'' - \kappa_\mu[-P_{y^{(1)}}(\boldsymbol{y}_{\dot{q}} - \boldsymbol{\mu}') + P_{\mu^{(0)}}(\boldsymbol{\mu}' + \boldsymbol{\mu} - \boldsymbol{\mu}_d) + P_{\mu^{(1)}}(\boldsymbol{\mu}'' + \boldsymbol{\mu}')] \\ \dot{\boldsymbol{\mu}}'' &= -\kappa_\mu \left[P_{\mu^{(1)}}(\boldsymbol{\mu}'' + \boldsymbol{\mu}')\right] \end{aligned} \tag{3-36}$$

Note that $\kappa_\mu$ is the tuning parameter for state estimation. Again, this is a general state update law which can be extended to any number of degrees of freedom.

### 3-3-3 Control actions

In the free-energy principle, the control actions play a fundamental role in the minimisation process. In fact, the control input $\boldsymbol{u}$ allows to steer the system to a desired state while minimising the prediction errors. The control input is computed as before using gradient descent. The dynamics of the control actions are then given, in a multivariate case, by:

$$\dot{\boldsymbol{u}} = -\kappa_a \frac{\partial \tilde{\boldsymbol{y}}}{\partial \boldsymbol{u}} \frac{\partial \mathcal{F}}{\partial \tilde{\boldsymbol{y}}} \tag{3-37}$$

where $\kappa_a$ is the chosen learning rate to be tuned. The final step to specify all the components of the active inference controller, is the definition of the partial derivatives of the sensory input with respect to the control actions.

### Control actions for an n-DOF robot manipulator with position and velocity feedback

The general actions update is expressed by Eq. (3-37). The partial derivatives of Eq. (3-34) with respect to the generalised sensory input are given by:

$$\frac{\partial \mathcal{F}}{\partial \boldsymbol{y}_q} = P_{y^{(0)}}(\boldsymbol{y}_q - \boldsymbol{\mu}), \quad \frac{\partial \mathcal{F}}{\partial \boldsymbol{y}_{\dot{q}}} = P_{y^{(1)}}(\dot{\boldsymbol{y}}_q - \boldsymbol{\mu}') \tag{3-38}$$

Having said that, the actions update is expressed as:

$$\dot{\boldsymbol{u}} = -\kappa_a \left[ \frac{\partial \boldsymbol{y}_q}{\partial \boldsymbol{u}} P_{y^{(0)}}(\boldsymbol{y}_q - \boldsymbol{\mu}) + \frac{\partial \boldsymbol{y}_{\dot{q}}}{\partial \boldsymbol{u}} P_{y^{(1)}}(\boldsymbol{y}_{\dot{q}} - \boldsymbol{\mu}') \right] \tag{3-39}$$

Active inference requires then to define the change in the sensory input with respect to the control actions, namely $\partial \boldsymbol{y}_q / \partial \boldsymbol{u}$ and $\partial \boldsymbol{y}_{\dot{q}} / \partial \boldsymbol{u}$. This is usually a hard task, and it can be seen as a forward dynamic problem. One approach to compute these relations, is through online learning using high-dimensional space regressors. However, this increases the complexity of the overall scheme and can produce unreliable results, as shown by the authors in [24]. In this work, we propose to approximate the partial derivatives relying on the high adaptability of the active inference controller against unmodeled dynamics, as suggested in the conclusive remarks in [24].

### Approximation of the true relation between states and sensory input

Let us first analyse the structure of the partial derivative matrices in Eq. (3-39). The control action is a vector of $n$ torques applied to the $n$ joints of the robot manipulator. Each torque has a direct effect only on the corresponding joint to which it is applied. This allows us to conclude that $\partial \boldsymbol{y}_q / \partial \boldsymbol{u}$ and $\partial \boldsymbol{y}_{\dot{q}} / \partial \boldsymbol{u}$ are diagonal matrices.

Furthermore, considering the second Newton's law, the total torque applied to a rotational joint equals the moment of inertia times the angular acceleration. The diagonal terms of the partial derivatives matrices are then time varying positive values which depend on the current robot configuration. In other words, this means that a positive torque applied to a joint will always result in a positive contribution for both position and velocity of that specific joint. In this control scheme, we propose to approximate the true time-varying relation with a positive constant, making use of the learning rate $\kappa_a$ as tuning parameter to achieve a sufficiently fast actions update. The general control update law, is finally given by:

$$\dot{\boldsymbol{u}} = -\kappa_a \left[ C_{y_q} P_{y^{(0)}}(\boldsymbol{y}_q - \boldsymbol{\mu}) + C_{y_{\dot{q}}} P_{y^{(1)}}(\boldsymbol{y}_{\dot{q}} - \boldsymbol{\mu}') \right] \tag{3-40}$$

where:

$$\frac{\partial \boldsymbol{y}_q}{\partial \boldsymbol{u}} \approx C_{y_q}, \ \frac{\partial \boldsymbol{y}_{\dot{q}}}{\partial \boldsymbol{u}} \approx C_{y_{\dot{q}}} \tag{3-41}$$

The positive definite diagonal constant matrices $C_{y_q}$, $C_{y_{\dot{q}}}$ are then set to the identity, meaning that we only encode the sign of the relation between control input and change in the sensory data. In the simulation results for both the 2-DOF and 7-DOF, we will show how this approximation results valid, thanks to the adaptability of the AIC.

### Active inference control scheme

The active inference controller for a generic robot manipulator can be schematized as in Fig. 3-4, where the five main blocks are highlighted. Note that computing the free-energy itself is not strictly necessary for the active inference controller, since the equations for beliefs and actions update are available in closed form.

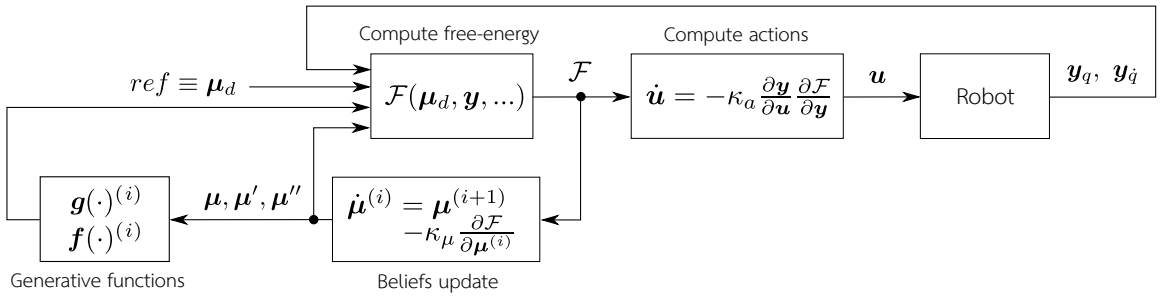



**Figure 3-4:** General active inference control scheme for a robot manipulator

The equations to be implemented for an AIC scheme are the free-energy as in Eq. (3-34), the state update as in Eq. (3-36), and the control actions as in Eq. (3-40). The resulting tuning parameters for the AIC are then:

- $P_{y^{(0)}},\ P_{y^{(1)}},\ P_{\mu^{(0)}},\ P_{\mu^{(1)}}$: the diagonal positive semi-definite precision matrices representing the confidence of the controller regarding its sensory input and internal beliefs about the states;
- $\kappa_\mu$, $\kappa_a$: the learning rates for state update and control actions respectively.

## 3-3-4 2-DOF example

For this simulation, the same 2-DOF robot arm as in Figure 3-1 is used. In contrast with the previous case, the robot is now free to move, and it will be controlled through active inference in the joint space. The control action is the the torque to be applied to each of the two actuators. The dynamic equations for the simulation of the robot arm were presented in Section 3-2-2.

For the simulation, the following scenario has been considered: the goal is specified through $\boldsymbol{\mu}_d$ in the joint space, and the starting position of the robot is set to $\boldsymbol{q} = [-\pi/2,\ 0]^\top\ [rad]$. Doing so, the robot starts from the completely extended down-down position. The goal position is set to:

$$\boldsymbol{\mu}_d = [-0.2,\ 0.3]\ [rad] \tag{3-42}$$

The control actions were allowed after $t_a = 1\ [s]$ of simulation, to appreciate their effect on the free-energy minimisation.

The sensory input is affected, without loss of generality, by normally distributed Gaussian noise with standard deviation 0.001. The noise in the two encoders and tachometers is supposed uncorrelated [5], such that $\boldsymbol{z} \sim (\boldsymbol{0}, 0.001 I_2)$ and $\boldsymbol{z}' \sim (\boldsymbol{0}, 0.001 I_2)$, with $I_2$ two

dimensional identity matrix. The actual simulated measurements of the joint positions are then given by:

$$\boldsymbol{y}_q = \boldsymbol{q} + \boldsymbol{z} \tag{3-43}$$

$$\boldsymbol{y}_{\dot{q}} = \dot{\boldsymbol{q}} + \boldsymbol{z}' \tag{3-44}$$

where $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$ are the ground truth values. The three equations to implement are given by Eq. (3-34), Eq. (3-36) and Eq. (3-40). The precision matrices $P_{y^{(0)}}$, $P_{y^{(1)}}$, $P_{\mu^{(0)}}$, $P_{\mu^{(1)}}$ are chosen to be the identity, to evaluate the state estimation and action computation with equally weighted sensory input and internal beliefs. The learning rate $\kappa_\mu$ for the state update is set to 20, while the learning rate $\kappa_a$ for the actions is set to 500, in order to have a fast convergence. The integration step is set to 1 $[ms]$ as before.

### Simulation results

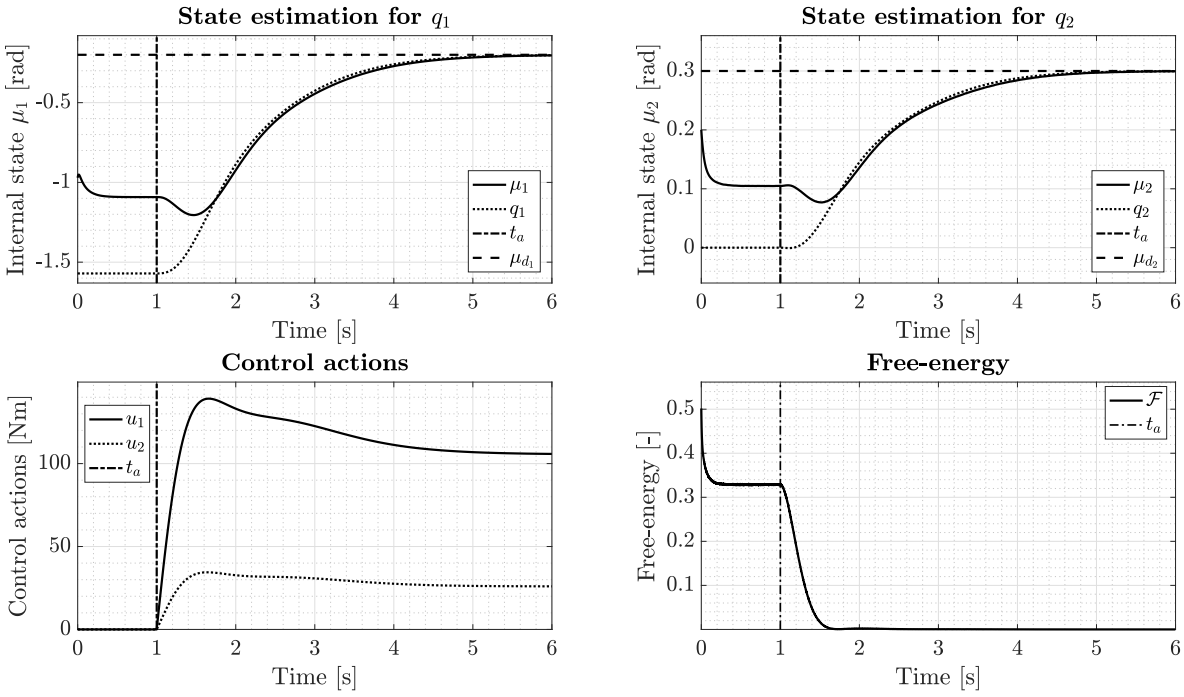



**Figure 3-5:** Active inference for robot arm control. Reaching task with a 2-DOF manipulator

As can be seen, during the first second of simulation the actions are not allowed. In this case, as showed in the previous static example, the beliefs do not converge to the true states due to the discrepancy between sensed and desired values to be sensed. Once the control actions are allowed at $t_a = 1$ $[s]$, the free-energy can be further minimised. This means two things: the beliefs about the states will converge to the real states, and the real states will converge to the imposed goal, thanks to the control actions. In the next section, a more complex simulation is carried out, using a 7-DOF robot manipulator. This is done to show that the derived algorithm is naturally scalable to high degrees of freedom, and that the reduced computational complexity allows real-time control of the robot arm for a pick and place task. Besides, the adaptability performance of the algorithm will be compared with the ones of the model reference adaptive controller.

## 3-4 Simulation results with a 7-DOF robot manipulator

This section presents the performance comparison between the novel AIC and the more established MRAC. The free-energy minimisation through states and actions update is coded in C++[1], to simulate the robot behaviour using the Robot Operating System (ROS) [39] ad Gazebo. To analyse the adaptability of the algorithms against unmodeled dynamics, the following steps have been taken:

- A heavily approximated model of the robot arm to be controlled is defined;
- The AIC and MRAC are tuned using the approximated model;
- The performance of the two controllers using the approximated model is compared;
- The same controllers tuned using the approximated model, are then applied to the system with accurate parameters description;
- The performance degradation of the two schemes against unmodeled dynamics is evaluated.

The tests to be performed are based on a pick and place cycle using the Franka Emika Panda 7-DOF robot manipulator[2], as depicted in Fig. 3-6. The desired joint values to perform the task are chosen such that the arm simulates the pick and place of an object from one bin to the other. More specifically, the following sequence of set-points is given to the robot arm:

1. The goal is set to be $\boldsymbol{q}_A = [1,\ 0.5,\ 0,\ -2,\ 0,\ 2.5,\ 0]\ [rad]$ from the initial position of the robot at $t = 0$, in order to reach the first bin A;
2. The goal is set to $\boldsymbol{q}_B = [0,\ 0.2,\ 0,\ -1,\ 0,\ 1.2,\ 0]\ [rad]$ at $t = 6s$, to move to the central position B;
3. The goal is set to $\boldsymbol{q}_C = [-1,\ 0.5,\ 0,\ -1.2,\ 0,\ 1.6,\ 0]\ [rad]$ at $t = 12s$, to reach the second bin C;
4. At $t = 18s$ the goal is set to $\boldsymbol{q}_B$ to move back to the central position, and at $t = 24s$ the goal is set again to $\boldsymbol{q}_A$ to re-start the cycle.

### 3-4-1 AIC tuning

In the previous sections we introduced the structure and the tuning parameters for both the MRAC and the AIC. We now present how the tuning procedure has been performed for the novel AIC. During the tuning part, the advantages of the AIC over the MRAC began to emerge. In particular, the active inference controller resulted in an overall easier tuning procedure, due to the reduced number of parameters and their clear physical meaning.

[1]The C++ code is freely available at https://github.com/cpezzato/panda_simulation
[2]Franka Emika https://www.franka.de/

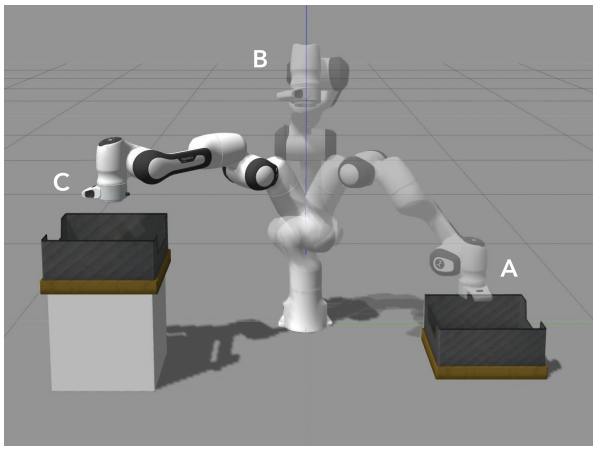



**Figure 3-6:** Pick and place cycle to position the end-effector of the 7-DOF Franka Emika Panda in A, B or C. The set-points are given in the joint space, following the order $\boldsymbol{q}_A$, $\boldsymbol{q}_B$, $\boldsymbol{q}_C$, $\boldsymbol{q}_B$, $\boldsymbol{q}_A$.

### Number of tuning parameters

The number of tuning parameters for the MRAC equals the number of DOF times the number of weighing terms. This results in $17 \times n$ parameters to be tuned. These terms are the elements of the weights described in Eq. (2-38) and Eq. (2-41). Even though the matrices are diagonal and one could use the same weight for each link, to achieve good performance a fine tuning of the individual parameters is essential.

Regarding the AIC, instead, the number of tuning parameters is independent from the DOF and it equals 6, according to Eq. (3-36) and Eq. (3-40). The lower number of parameters resulted in an overall easier tuning procedure for the active inference controller.

### AIC tuning procedure

To obtain a satisfactory response for the AIC, we followed the tuning procedure reported below. As a general approach, the state estimation in a static case has been tuned first, assuming equal weights for sensory input and internal beliefs. Once the state convergence was fast enough, the control actions have been included to steer the robot to the goal. Then, the weights on the confidence of the controller have been increased to reduce the oscillations. We now report the tuning steps in more detail.

- We set the controller confidence about sensory input and internal beliefs to one. That is, we impose the matrices $P_{y^{(0)}},\ P_{y^{(1)}},\ P_{\mu^{(0)}},\ P_{\mu^{(1)}}$ to the identity;
- We disabled the control actions and incremented the learning rate $\kappa_\mu$ until the state estimation in a static situation was fast enough (i.e. $\sim 20 - 30\ [ms]$);
- We included the control actions and increased the learning rate $\kappa_a$ until the robot was steered to the desired position, showing significant oscillations;
- We dampened the oscillatory behaviour decreasing the sensory confidence about the most noisy sensors, and the internal beliefs about the higher order generalised motions. In this way, the controller relies more on the least noisy sensory input.

### 3-4-2 Performance comparison between AIC and MRAC

#### Pick and place cycle with approximated model

The pick and place performance is now presented. The controllers have been tuned using a fairly wrong model of the robot arm on purpose. The links of the robot manipulator have been approximated as cuboids, and 20% random uncertainty in each link's mass has been assumed. This will allow to evaluate later on the adaptability performance, applying the same controllers to the system with accurate description. The joint values, and the computed control actions using AIC and MRAC, are depicted in Fig. 3-7. To keep the presented results as neat as possible, the plots in this section will report the behaviour of only two relevant joints, in terms of position and computed torques. For the complete plots, an interested reader is referred to Appendix B. Note that, for the MRAC, saturation of the control input at $\pm 85 Nm$ is sometimes reached after providing the new goal position.

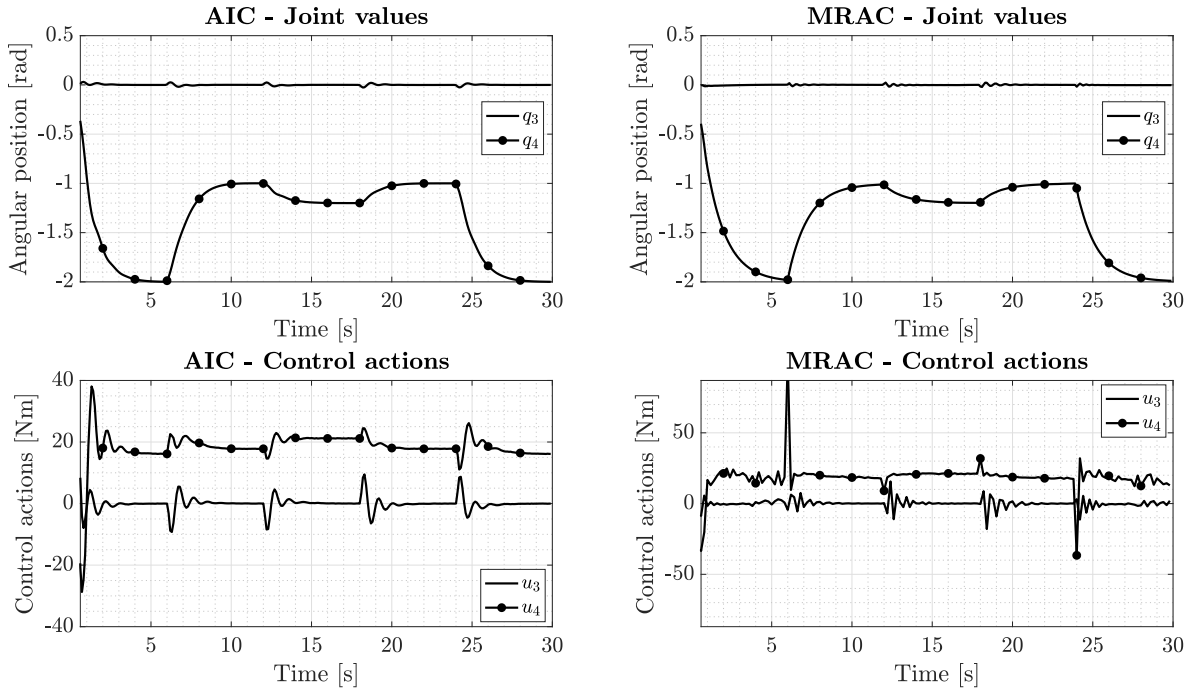



**Figure 3-7:** Response and control actions for $q_3$ and $q_4$. Simulation using ROS and Gazebo.

#### Performance degradation in case of large parameters variations

The same controllers tuned using the approximated model of the 7-DOF robot arm, are now applied to control the manipulator for which accurate dynamics have been specified. For clarity, we present the performance analysing the difference between the responses of the model with approximated and accurate dynamics.

The two control architectures should adapt to the large parameter changes, and keep the difference between the responses limited. The results are presented Fig. 3-8.

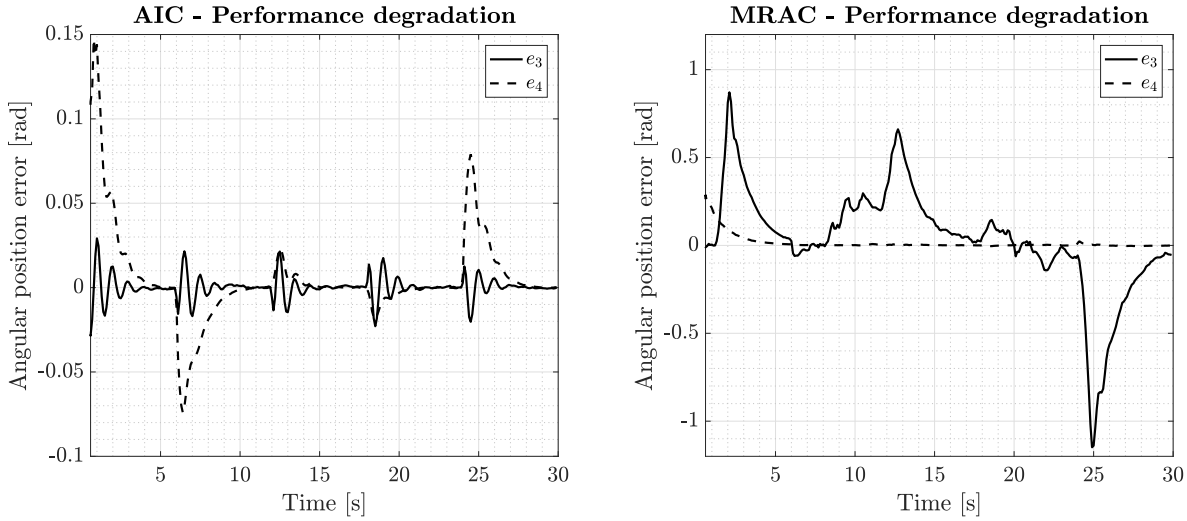



**Figure 3-8:** Performance degradation applying the controllers (tuned with approximated model) to the robot with accurate system description. Simulation using ROS and Gazebo.

As can be seen, there is one order of magnitude between the AIC errors in the joint positions, and the correspondent ones from the MRAC. Besides, the convergence to zero of the errors is also considerably faster in the AIC. The performance degradation becomes more clear if the Cartesian pose of the end-effector is considered, as in Figure 3-9.

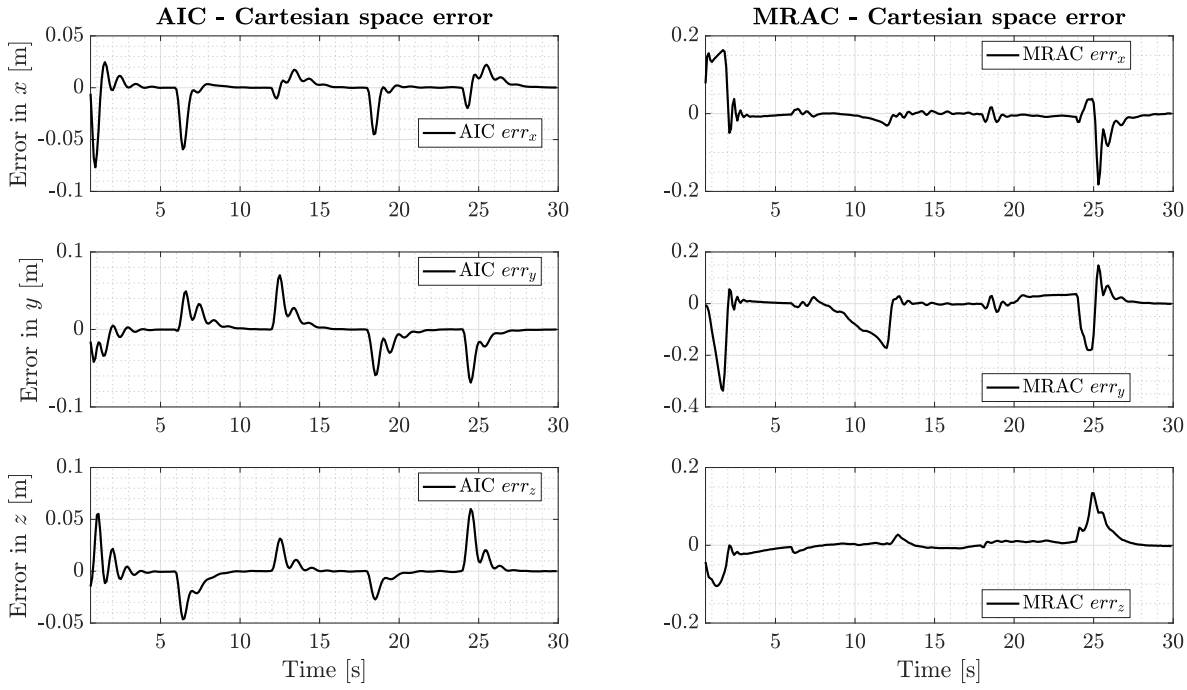



**Figure 3-9:** Performance degradation in Cartesian space. Simulation using ROS and Gazebo.

### Remarks

The active inference controller has shown better adaptability performance with respect to the MRAC. Overall, the performance degradation in joint space due to unmodeled dynamics is more than ten times lower in the AIC. The free-energy minimisation through beliefs and actions update, allowed to simultaneously infer the robot's states and to move to the desired goal. The behaviour of the free-energy during the pick and place cycle, with accurate dynamic description of the robot, is reported in Figure 3-10. It can be noticed how, whenever a new set-point is imposed, the free-energy increases. This is due to the fact that suddenly the new desired position differs from the actual one. In this situation, the controller is trying to minimise this discrepancy through state estimation and actions, and this leads to the minimisation of $\mathcal{F}$.

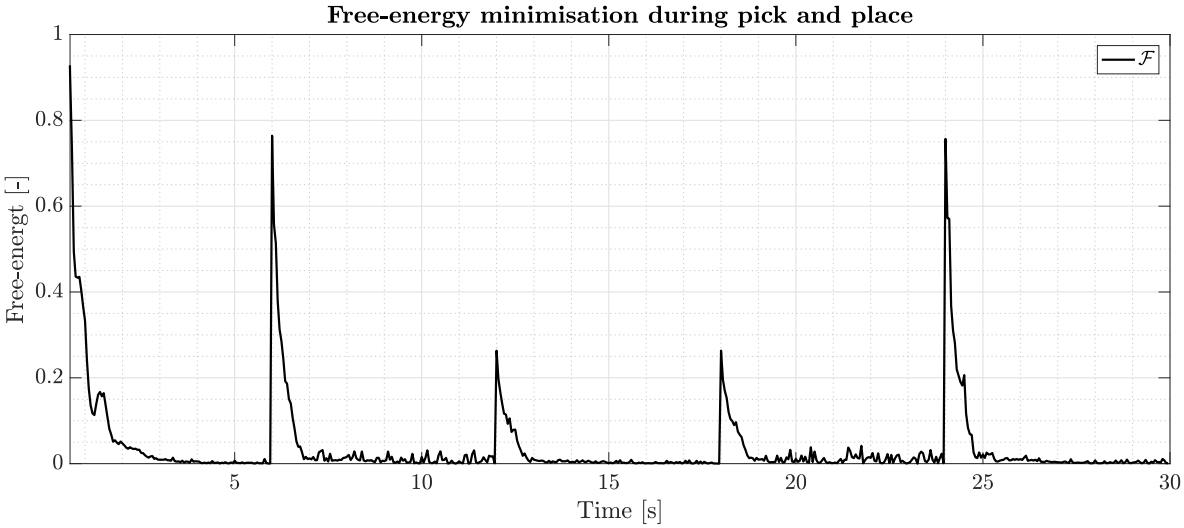



**Figure 3-10:** Free-energy minimisation during pick and place cycle with 7-DOF manipulator.

The AIC showed very high performance. However, one point which the author considers important to mention, is the difficulty encountered while trying to impose a specific response for the robot arm. In fact, the way the robot will reach the target in the joint space, is not imposed by the tuning parameters, but rather by the encoded generative function $\boldsymbol{f}(\boldsymbol{\mu})$. In a sense, the AIC and the MRAC share another similarity, since the behaviour of the robot is imposed a priory through another reference model. In conclusion, the main disadvantage of this approach is that, in order to change the modes of the response, one should act on $\boldsymbol{f}(\boldsymbol{\mu})$, as one would change the reference models on the MRAC.

**Table 3-2:** Summary of AIC versus MRAC

| | **Parameters** | **Control input** | **Performance degradation** | **Response** |
|---|---|---|---|---|
| *MRAC* | Linearly increasing with $n$ | More aggressive, with saturation | Marginally stable behaviour | Imposed by $\boldsymbol{q}_m$ |
| *AIC* | Constant with $n$ | No saturation | Low performance degradation | Imposed by $\boldsymbol{f}(\boldsymbol{\mu})$ |

## 3-5 Concluding remarks

In this chapter, we presented the structure and the performance of a novel active inference controller. The general expressions for the free-energy, the state update, and the control actions, have been derived for a generic $n$-DOF robot manipulator with position and velocity sensors. The derived control law represents a general model-free approach for robot control in joint space.

The performance of the controller has been tested in a 2-DOF toy example, to highlight its peculiarity in terms of interconnection between actions and perception. Besides, the scalability and adaptability performance of the proposed solution have been showed in a complex 7-DOF example. With the latter, we demonstrated that this control architecture is relevant for robotics applications.

The proposed definition of the dynamic generative model, allowed to express the free-energy minimization problem without any kinematic or dynamic knowledge of the system. This, in combination with the model approximations regarding the control actions, resulted in a controller which is less sensitive to large parameters variation. This is an evidence of the capability of the AIC to compensate for unmodeled dynamics, as supposed by previous work regarding active inference.

The comparison with the MRAC showed that the AIC achieves better performance with respect to the adaptive controller. In fact, the AIC showed greater adaptability in case of large parameter uncertainties, and did not achieve saturation of the control input. Furthermore, the AIC has the advantage that the controller complexity does not increase with higher number of DOF, and it is easier to tune. The main disadvantage of the proposed approach is that it is not straightforward to impose a particular response or motion constraints. To achieve so, the dynamic generative model of the evolution of the states should be modified.

# Chapter 4

# Active inference for fault tolerant control

*The fault tolerant properties of active inference foreseen in Chapter 2, are now analysed in detail. In particular, a novel active inference based fault tolerant scheme for sensory faults is devised. The general expression of $\mathcal{F}$ for a n-DOF robot manipulator, is used to include the necessary sensory redundancy for fault recovery. Besides encoders and tachometers, also a camera is implemented to retrieve the end-effector position. The overall algorithm for state estimation and control, extends naturally to accommodate heterogeneous sensors. An upper bound on the sensory prediction errors is mathematically derived, and then used as on-line threshold for fault detection. The recovery actions are implemented reducing the precision matrix associated with the faulty sensor.*

## 4-1 Problem statement

In recent years, industries has become more and more keen on making use of robots for manufacturing and production lines [23]. As a consequence, in order to guarantee high standards of reliability and security, the area of fault tolerant control of robot manipulators gathered more importance [49]. Several approaches have been developed in the past years, focusing on physical joints, actuators and sensors faults [10, 42, 36, 46, 6]. The two main drawbacks in the current approaches, as highlighted also in Chapter 2, are:

- The definition of meaningful residuals to map the effects of the faults affecting the sensors;
- The selection of a robust threshold which qualifies as best compromise between missed and false alarms.

Besides, the recovery from sensory faults requires a fault specific external block, that has to be defined apart from the default controller and the FDI scheme. In the following subsections,

we present how the AIC can be exploited to facilitate the residual generation, the threshold determination and the recovery in case of sensory faults.

### The controlled plant

First of all, let us specify in more detail the system to be controlled. The control problem that we consider, is a reaching task using the 2-DOF robot arm of the previous chapter. However, this time the robot is equipped with one extra sensor, namely a camera to retrieve the end-effector position. Furthermore, during the execution of a trajectory, we suppose the presence of faults either in the encoders or in the camera, such as sensor freezing or camera occlusion. The controlled plant is depicted in Fig. 4-1.

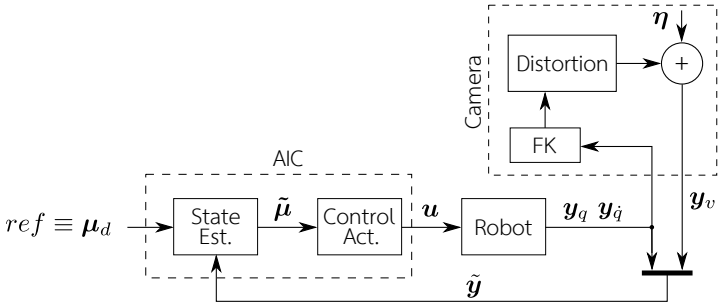



**Figure 4-1:** Control scheme of a robot manipulator using active inference and redundant sensory input

The robot arm is equipped with a camera to retrieve the end effector Cartesian position $\boldsymbol{y}_v = [y_{v_x}, y_{v_z}]^\top$, and with position and velocity sensors $\boldsymbol{y}_q,\ \boldsymbol{y}_{\dot{q}} \in \mathbb{R}^2$ for the two joints. The camera output is simulated in MATLAB using the direct kinematics of the robot arm plus a radial distortion and additive noise. In such a case, the generalised sensory input is then composed by:

$$\tilde{\boldsymbol{y}} = [\boldsymbol{y}^{(0)}, \boldsymbol{y}^{(1)}] = [(\boldsymbol{y}_q,\ \boldsymbol{y}_v),\ \boldsymbol{y}_{\dot{q}}] \tag{4-1}$$

**Assumption 1** The proprioceptive sensors $\boldsymbol{y}_q,\ \boldsymbol{y}_{\dot{q}}$ are affected, without loss of generality, by Gaussian noise $\boldsymbol{z}$ and $\boldsymbol{z}'$ respectively, of zero mean and standard deviation $\sigma_q = 0.001\ [rad]$ and $\sigma_{\dot{q}} = 0.001\ [rad/s]$.

**Assumption 2** The additive noise $\boldsymbol{\eta}$ affecting the simulated camera is supposed, without loss of generality, as Gaussian noise with zero mean and standard deviation $\sigma_\eta$ equal to 0.01 $[m]$ [29, 37, 30]. Furthermore, we inject a barrel distortion in the camera, with coefficients $K_1 = -1.5e^{-3},\ K_2 = 5e^{-6},\ K_3 = 0$ [47], for a more realistic sensory input. Figure 4-2 shows the distortion effect for a field of view of $2 \times 2\ [m]$.

The robot arm is controlled by means of an active inference controller AIC based on the previous chapter. To define the controller's structure, and in particular the free-energy expression for prediction error minimisation, two generative models for the sensory data and the world dynamics have to be specified. The next subsection presents the controller's equations for state estimation and control actions using sensory redundancy.

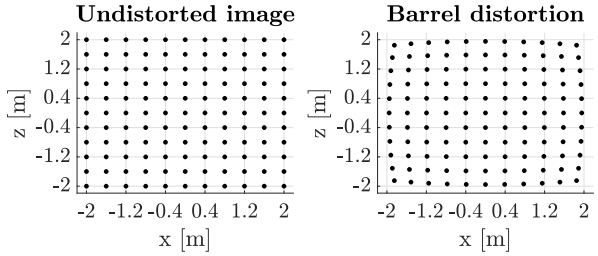



**Figure 4-2:** Effect of simulated barrel distortion

## 4-2 Incorporating sensory redundancy

To incorporate sensory redundancy, the starting point is the general definition of $\mathcal{F}$ for a robot manipulator. We report Eq. (3-26) from Chapter 3 for clarity:

$$\mathcal{F} = \frac{1}{2}\sum_{i=0}^{n_d-1}\left[(\boldsymbol{y}^{(i)} - \boldsymbol{g}^{(i)})^\top P_{y^{(i)}}(\boldsymbol{y}^{(i)} - \boldsymbol{g}^{(i)}) + (\boldsymbol{\mu}^{(i+1)} - \boldsymbol{f}^{(i)})^\top P_{\mu^{(i)}}(\boldsymbol{\mu}^{(i+1)} - \boldsymbol{f}^{(i)})\right] \tag{4-2}$$

Note that, in this definition, there is no assumption on the dimension of the generalised sensory input $\tilde{\boldsymbol{y}}$. The number of available sensory input is not constrained, and the only requirement to evaluate $\mathcal{F}$, is to provide a generative model $\boldsymbol{g}^{(i)}(\boldsymbol{\mu})$ for each generalised order.

One of the great advantages of the free-energy principle, is the possibility to combine an arbitrary number of sensory data into the same prediction error minimisation problem. This is an important property when dealing with sensory faults, since redundancy can easily be implemented for recovery. To define the equations for state estimation and control actions for the AIC, the first step is to determine the generative models of the sensory data $\tilde{\boldsymbol{g}}$, and the state dynamics $\tilde{\boldsymbol{f}}$. As before, we suppose to control the robot arm in joint space, and we assume the states to be the joint positions. This leads to:

$$\boldsymbol{g}_q(\boldsymbol{\mu}) = \boldsymbol{\mu}, \qquad \frac{\partial \boldsymbol{g}_q(\boldsymbol{\mu})}{\partial \boldsymbol{\mu}} = 1 \tag{4-3}$$

To be able to evaluate the free-energy, the definition of the generative model for the visual camera $\boldsymbol{g}_v(\boldsymbol{\mu})$, and its Jacobian are still missing. Note that, since the camera is not giving the joint positions directly but the end-effector one, it is not possible anymore to simplify the generative models as for the encoders and tachometer. The relation between joint values and end-effector has to be encoded. There are at least three ways of doing so:

- Using the direct kinematic of the robot arm;
- Learning the generative model off-line;
- Learning the generative model on-line.

We chose to derive the generative model off-line through Gaussian Process Regression (GPR) [40] as in [25]. In fact, since we injected camera distortion, the estimate of the sensory input using the direct kinematics would result poor. On the other hand, the use of a complex on-line learning technique can lead to unreliable estimates [24]. The use of GPR is a good compromise, and the main advantage is that the Jacobian of the process can be determined in closed form.

### 4-2-1 Gaussian process regression for the camera generative model

The problem that we want to solve is a prediction problem. We basically try to determine the non-linear relation between the measured end-effector position $\boldsymbol{y}_v$, and the measured joint position $\boldsymbol{y}_q$, trough $\boldsymbol{g}_v$. More formally, a GPR generates data in a domain such that any finite subset follows a multivariate Gaussian distribution. The training data is composed by a set of observations of the camera output $[\bar{\boldsymbol{y}}_{v_x},\ \bar{\boldsymbol{y}}_{v_z}]^\top$ in several robot configurations $\bar{\boldsymbol{y}}_q$. The core of this technique is the relation between one observation and another, which is encoded in the covariance matrix $K$, and especially in the covariance kernel function $k(\boldsymbol{y}_{q_i}, \boldsymbol{y}_{q_j})$. We chose to use a squared exponential kernel of the form:

$$k(\boldsymbol{y}_{q_i}, \boldsymbol{y}_{q_j}) = \sigma_f^2 \exp\left(-\tfrac{1}{2}(\boldsymbol{y}_{q_i} - \boldsymbol{y}_{q_j})^\top \Lambda (\boldsymbol{y}_{q_i} - \boldsymbol{y}_{q_j})\right) + \sigma_n^2 d_{ij} \tag{4-4}$$

where $\boldsymbol{y}_{q_i},\ \boldsymbol{y}_{q_j} \in \bar{\boldsymbol{y}}_q$, and $d_{ij}$ is the Kronecker delta function. $\Lambda$ is a diagonal matrix of hyperparameters. The prediction of the camera output, once given the current state $\boldsymbol{y}_{q_*}$, is determined by the posterior mean:

$$\boldsymbol{g}_v(\boldsymbol{y}_{\boldsymbol{q}_*}) = \begin{bmatrix} K_* K^{-1} \bar{\boldsymbol{y}}_{v_x} \\ K_* K^{-1} \bar{\boldsymbol{y}}_{v_z} \end{bmatrix} \tag{4-5}$$

Since we chose to use a squared exponential kernel, we can determine a closed form for the derivative of the estimated process [25]. This is indeed equal to:

$$\boldsymbol{g}_v(\boldsymbol{y}_{\boldsymbol{q}_*})' = \frac{\partial k(\boldsymbol{y}_{\boldsymbol{q}_*}, \bar{\boldsymbol{y}}_q)}{\partial \boldsymbol{y}_{\boldsymbol{q}_*}} \begin{bmatrix} \boldsymbol{\alpha}_x \\ \boldsymbol{\alpha}_z \end{bmatrix} \quad \Rightarrow \quad \boldsymbol{g}_v(\boldsymbol{y}_{\boldsymbol{q}_*})' = \begin{bmatrix} -\Lambda^{-1}(\boldsymbol{y}_{\boldsymbol{q}_*} - \bar{\boldsymbol{y}}_q)^\top [k(\boldsymbol{y}_{\boldsymbol{q}_*}, \bar{\boldsymbol{y}}_q)^\top \cdot \boldsymbol{\alpha}_x] \\ -\Lambda^{-1}(\boldsymbol{y}_{\boldsymbol{q}_*} - \bar{\boldsymbol{y}}_q)^\top [k(\boldsymbol{y}_{\boldsymbol{q}_*}, \bar{\boldsymbol{y}}_q)^\top \cdot \boldsymbol{\alpha}_z] \end{bmatrix} \tag{4-6}$$

where $\cdot$ means element-wise multiplication, $\boldsymbol{\alpha}_x = K^{-1}\bar{\boldsymbol{y}}_{v_x}$ and $\boldsymbol{\alpha}_z = K^{-1}\bar{\boldsymbol{y}}_{v_z}$. Note that, the computational demand of the GPR increases with the number of DOF, due to the need or an augmented training set, and the higher size of the matrices in play. For a more in detail explanation regarding the GPR, an interested reader is referred to Appendix C-1.

### 4-2-2 Free-energy with sensory redundancy

Now that all the generative models have been specified, the free-energy can be expressed, starting from Eq. (3-26), as:

$$\begin{aligned} \mathcal{F} &= \frac{1}{2}(\boldsymbol{y}_q - \boldsymbol{\mu})^\top P_{y_q} (\boldsymbol{y}_q - \boldsymbol{\mu}) + \frac{1}{2}(\boldsymbol{y}_{\dot{q}} - \boldsymbol{\mu}')^\top P_{y_{\dot{q}}} (\boldsymbol{y}_{\dot{q}} - \boldsymbol{\mu}') \\ &+ \frac{1}{2}(\boldsymbol{y}_v - \boldsymbol{g}_{\boldsymbol{v}}(\boldsymbol{\mu}))^\top P_{y_v} (\boldsymbol{y}_v - \boldsymbol{g}_{\boldsymbol{v}}(\boldsymbol{\mu})) \\ &+ \frac{1}{2}(\boldsymbol{\mu}' + \boldsymbol{\mu} - \boldsymbol{\mu}_d)^\top P_\mu (\boldsymbol{\mu}' + \boldsymbol{\mu} - \boldsymbol{\mu}_d) \\ &+ \frac{1}{2}(\boldsymbol{\mu}'' + \boldsymbol{\mu}')^\top P_{\mu'} (\boldsymbol{\mu}'' + \boldsymbol{\mu}') \end{aligned} \tag{4-7}$$

In the expression above, we considered the generalised motions up to order two, since measurements up to the velocity are available. To distinguish between the different sensory inputs, the precision matrices have been named accordingly to the relative sensor. The matrices

$P_{y_q}$, $P_{y_{\dot{q}}}$, $P_{y_v}$ are diagonal positive semi-definite precision matrices containing the confidence about specific sensory inputs. The higher these values, the higher the relevance given to one specific sensor. Very low values for the precision associated with certain inputs would exclude the specific sensor from both the state estimation and the control action parts. It is convenient, also, to rename the precision matrices $P_{\mu^{(0)}}$, $P_{\mu^{(1)}}$ as $P_\mu$, $P_{\mu'}$. Again, they express the confidence about the estimated generalised states $\boldsymbol{\mu}$ and $\boldsymbol{\mu}'$.

### 4-2-3 Beliefs update with sensory redundancy

As can be seen from Fig. 4-1, the AIC is composed by two main blocks. The first one is the state estimation block, which continuously minimises the free-energy according to a gradient descent scheme with respect to $\tilde{\boldsymbol{\mu}}$. In this block, the controller is computing the most probable posterior states estimate, according to the current sensory input. The equations for the beliefs update are given in general by:

$$\dot{\tilde{\boldsymbol{\mu}}} = D\tilde{\boldsymbol{\mu}} - \kappa_\mu \frac{\partial \mathcal{F}}{\partial \tilde{\boldsymbol{\mu}}} \tag{4-8}$$

Considering $\mathcal{F}$ as in Eq. (4-7) leads to:

$$\begin{aligned}
\dot{\boldsymbol{\mu}} &= \boldsymbol{\mu}' - \kappa_\mu[-P_{y_q}(\boldsymbol{y}_q - \boldsymbol{\mu}) - P_v(\boldsymbol{y}_v - \boldsymbol{g}_v(\boldsymbol{\mu}))\frac{\partial \boldsymbol{g}_v}{\partial \boldsymbol{\mu}} + P_\mu(\boldsymbol{\mu}' + \boldsymbol{\mu} - \boldsymbol{\mu}_d)] \\
\dot{\boldsymbol{\mu}}' &= \boldsymbol{\mu}'' - \kappa_\mu[-P_{y_{\dot{q}}}(\boldsymbol{y}_{\dot{q}} - \boldsymbol{\mu}') + P_\mu(\boldsymbol{\mu}' + \boldsymbol{\mu} - \boldsymbol{\mu}_d) + P_{\mu'}(\boldsymbol{\mu}'' + \boldsymbol{\mu}')] \\
\dot{\boldsymbol{\mu}}'' &= -\kappa_\mu \left[P_{\mu'}(\boldsymbol{\mu}'' + \boldsymbol{\mu}')\right]
\end{aligned} \tag{4-9}$$

Note that the two non-linear functions $\boldsymbol{g}_v(\boldsymbol{\mu})$ and $\frac{\partial \boldsymbol{g}_v}{\partial \boldsymbol{\mu}}$ are given by Eq. (4-5) and Eq. (4-6) respectively.

### 4-2-4 Control actions with sensory redundancy

In an AIC the control actions are defined through the gradient descent of $\mathcal{F}$ with respect to the control input $\boldsymbol{u}$:

$$\dot{\boldsymbol{u}} = -\kappa_a \frac{\partial \tilde{\boldsymbol{y}}}{\partial \boldsymbol{u}} \frac{\partial \mathcal{F}}{\partial \tilde{\boldsymbol{y}}} \tag{4-10}$$

Using $\mathcal{F}$ as in Eq. (4-7), the control actions for the 2-DOF example with sensory redundancy are defined as:

$$\dot{\boldsymbol{u}} = -\kappa_a \left[\frac{\partial \boldsymbol{y}_q}{\partial \boldsymbol{u}} P_{y_q}(\boldsymbol{y}_q - \boldsymbol{\mu}) + \frac{\partial \boldsymbol{y}_{\dot{q}}}{\partial \boldsymbol{u}} P_{y_{\dot{q}}}(\boldsymbol{y}_{\dot{q}} - \boldsymbol{\mu}') + \frac{\partial \boldsymbol{y}_v}{\partial \boldsymbol{u}} P_v(\boldsymbol{y}_v - \boldsymbol{g}_v(\boldsymbol{\mu}))\right]$$

where the only terms still to be defined are:

$$\frac{\partial \boldsymbol{y}_q}{\partial \boldsymbol{u}} = \begin{pmatrix} \frac{\partial y_{q_1}}{\partial u_1} & \frac{\partial q_1}{\partial u_2} \\ \frac{\partial y_{q_2}}{\partial u_1} & \frac{\partial q_2}{\partial u_2} \end{pmatrix}, \quad \frac{\partial \boldsymbol{y}_{\dot{q}}}{\partial \boldsymbol{u}} = \begin{pmatrix} \frac{\partial \dot{q}_1}{\partial u_1} & \frac{\partial \dot{q}_1}{\partial u_2} \\ \frac{\partial \dot{q}_2}{\partial u_1} & \frac{\partial \dot{q}_2}{\partial u_2} \end{pmatrix} \tag{4-11}$$

and:

$$\frac{\partial \boldsymbol{y}_v}{\partial \boldsymbol{u}} = \begin{pmatrix} \frac{\partial v_x}{\partial u_1} & \frac{\partial v_x}{\partial u_2} \\ \frac{\partial v_y}{\partial u_1} & \frac{\partial v_y}{\partial u_2} \end{pmatrix} \tag{4-12}$$

As explained for the AIC with position and velocity sensors, the partial derivatives of $\boldsymbol{y}_q$ and $\boldsymbol{y}_{\dot{q}}$ can be set to positive definite diagonal matrices. We chose again to impose them to be the identity matrix, using the learning rate $\kappa_a$ to have a sufficiently fast response.

Regarding the relation between $\boldsymbol{u}$ and $\boldsymbol{y}_v$, it is possible to do similar geometric considerations. The partial derivatives describe how the control actions influence the $x$ and $z$ position of the end-effector. These terms are also time varying parameters which, however, can assume both positive and negative values depending on the current robot configuration. As for the other two partial derivatives, we approximate the true relation only with its sign, and we rely on $\kappa_a$ for tuning.

To define the partial derivatives $\partial \boldsymbol{y}_v / \partial \boldsymbol{u}$, we suppose to operate the robot arm in a specific region, such that we can properly define the sign of the relations. The matrix $\partial \boldsymbol{y}_v / \partial \boldsymbol{u}$ can assume a limited set of values, depending on the combination of the two joint angles $q_1$ and $q_2$. For the presented example, since the reaching task will be carried out in the fourth quadrant of a Cartesian reference frame, (i.e. $-\pi/2 \leq q_1 \leq 0$) then positive values of $u_1$ will lead to positive increments of both $x$ and $z$.

For what concerns the torque for the second joint $u_2$, a bit more articulated reasoning has to be carried out. The effect of the second torque depends also on the sum of the first and second joint angles. Always considering to operate in the fourth quadrant, we have:

**Table 4-1:** Sign of $\frac{\partial v_x}{\partial u_2}$, $\frac{\partial v_y}{\partial u_2}$ in the 4th quadrant

| **Joint values** | $\text{sign}\left(\frac{\partial v_x}{\partial u_2}\right)$ | $\text{sign}\left(\frac{\partial v_y}{\partial u_2}\right)$ |
|---|---|---|
| $-\pi/2 \leq q_1 + q_2 < 0$ | +1 | +1 |
| $0 \leq q_1 + q_2 < \pi/2$ | -1 | +1 |
| $\pi/2 \leq q_1 + q_2 < 3\pi/2$ | -1 | -1 |
| $3\pi/2 \leq q_1 + q_2 < 2\pi$ | +1 | -1 |

The full fourth quadrant have then been characterized. Note that, the geometric considerations above can be extended for the whole Cartesian workspace.

## 4-3 An active inference based fault tolerant controller

*The AIC controller structure have been fully specified for the case of redundant sensory input. We now present how the above controller can be exploited to obtain an intrinsically fault tolerant scheme.*

The main idea behind the active inference based fault tolerant scheme, is to use the sensory prediction errors for fault detection, and the sensory redundancy for fault recovery. The first step is to determine an on-line threshold from the sensory prediction errors through an observer. Subsequently, when a fault is detected and isolated, the confidence of the controller about the faulty sensor can be consistently reduced. This allows to recover from sensory faults, exploiting the structure of the controller itself, and avoiding the implementation of a switching procedure between sensors. The general scheme is depicted in Fig. 4-3.

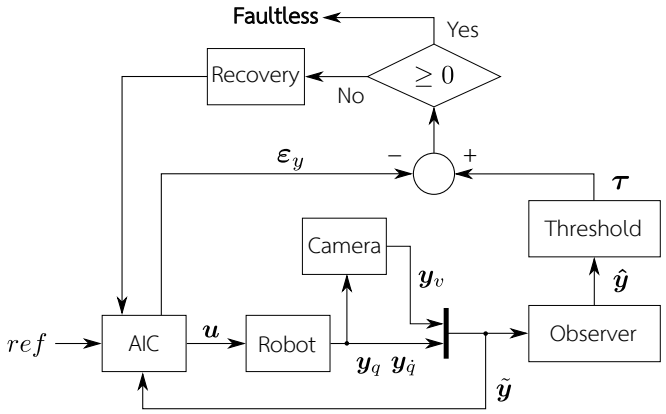



**Figure 4-3:** Active inference based fault tolerant scheme

### Definition of the supposed faults

We now rigorously define the sensory faults that will be later on injected in the system.

**Assumption 3** No fault occurs before the fault time $t_f$, such that the controlled system is supposed in nominal working conditions for $0 \leq t < t_f$.

**Assumption 4** We will investigate the effect of faults in the proprioceptive sensors and in the visual camera, for which the sensor redundancy is available. In particular, we assume the following:

1. *Encoder fault*: the fault in one of the encoders is simulated freezing the sensor's output to the value at $t_f$ for 200 $[ms]$, that is $\boldsymbol{y}_q(t) = [q_1(t_f),\ q_2(t)]^\top + \boldsymbol{z}$ for $t_f \leq t < t_f + 0.2$. Remember that $\boldsymbol{z}$ is Gaussian noise;
2. *Camera fault*: the fault in the camera is supposed to be occlusion of the field of view, so complete lack of information from the visual sensors. This is simulated with an abrupt change in the camera output which is bigger than the usual noise supposed in the sensor. For instance, at $t_f$ we inject an additive term equal to 0.075 $[m]$. Remember that the standard deviation of the noise in the camera is 0.01 $[m]$.

## 4-3-1 On-line threshold determination

To define an on-line threshold for the sensory prediction errors, we first have to prove the existence of an upper bound for the free-energy in the faultless case. To do so, we exploit the fact that $\mathcal{F}$ is a quadratic function. Let us consider the free-energy for a generic system, using generalised motions up to order $n_d$:

$$\mathcal{F} = \sum_{i=0}^{n_d-1} \frac{1}{2} \left[ \boldsymbol{\varepsilon}_y^{(i)\top} P_{y^{(i)}} \boldsymbol{\varepsilon}_y^{(i)} + \boldsymbol{\varepsilon}_\mu^{(i)\top} P_{\mu^{(i)}} \boldsymbol{\varepsilon}_\mu^{(i)} \right] \tag{4-13}$$

where $\boldsymbol{\varepsilon}_y^{(i)}$ and $\boldsymbol{\varepsilon}_\mu^{(i)}$ are respectively the sensory and model prediction errors $(\boldsymbol{y}^{(i)} - \boldsymbol{g}^{(i)}(\boldsymbol{\mu}))$ and $(\boldsymbol{\mu}^{(i+1)} - \boldsymbol{f}^{(i)}(\boldsymbol{\mu}))$.

**Assumption 5** The generative model $\boldsymbol{g}^{(i)}(\boldsymbol{\mu})$, $i = 1, 2$ expresses the relation between the states of the system and the sensory input, up to some uncertainties $\boldsymbol{\delta}_y^{(i)}$.

**Assumption 6** The uncertainties $\boldsymbol{\delta}_y^{(i)}$ affecting the generative models are bounded by a maximum value $\boldsymbol{\delta}_M \in \mathbb{R}^+$, which includes unmodeled dynamics and measurement noise. It holds then:

$$-\boldsymbol{\delta}_M \leq \boldsymbol{\delta}_y^{(i)} \leq \boldsymbol{\delta}_M, \ \forall\ t > 0 \tag{4-14}$$

**Theorem 1** *Let us consider the free-energy for a generic system with generalised sensory input $\tilde{\boldsymbol{y}}$, states $\tilde{\boldsymbol{\mu}}$ and generative models $\boldsymbol{g}^{(i)}(\boldsymbol{\mu})$, $\boldsymbol{f}^{(i)}(\boldsymbol{\mu})$. The system is subject to uncertainties $\boldsymbol{\delta}_y^{(i)}$ in the generative functions for sensory prediction errors. In such a case, using generalised motions up to order $n_d$, $\mathcal{F}$ can be expressed as:*

$$\mathcal{F} = \sum_{i=0}^{n_d-1} \frac{1}{2}\left[(\boldsymbol{\varepsilon}_y^{(i)} + \boldsymbol{\delta}_y^{(i)})^\top P_{y^{(i)}}(\boldsymbol{\varepsilon}_y^{(i)} + \boldsymbol{\delta}_y^{(i)}) + \boldsymbol{\varepsilon}_\mu^{(i)\top} P_{\mu^{(i)}} \boldsymbol{\varepsilon}_\mu^{(i)}\right] \tag{4-15}$$

*If Assumptions 5 and 6 hold then:*

$$\mathcal{F} \leq \sum_{i=0}^{n_d-1} \frac{1}{2}\left[\boldsymbol{\varepsilon}_y^{(i)\top} P_{y^{(i)}} \boldsymbol{\varepsilon}_y^{(i)} + \boldsymbol{\varepsilon}_\mu^{(i)\top} P_{\mu^{(i)}} \boldsymbol{\varepsilon}_\mu^{(i)} + \boldsymbol{\delta}_M^\top P_{y^{(i)}} \boldsymbol{\delta}_M + 2||\boldsymbol{\varepsilon}_y^{(i)\top} P_{y^{(i)}} \boldsymbol{\delta}_M||_2^2\right] \tag{4-16}$$

**Proof of Theorem 1** *Let us consider the order zero sensory prediction error $\boldsymbol{\varepsilon}_y$, the corresponding uncertainty $\boldsymbol{\delta}_y$, and the diagonal positive definite precision matrix $P_y$. Using Assumptions 5 and 6 we can write:*

$$\begin{aligned}
(\boldsymbol{\varepsilon}_y + \boldsymbol{\delta}_y)^\top P_y (\boldsymbol{\varepsilon}_y + \boldsymbol{\delta}_y) &= (\boldsymbol{\varepsilon}_y^\top + \boldsymbol{\delta}_y^\top)(P_y \boldsymbol{\varepsilon}_y + P_y \boldsymbol{\delta}_y) \\
&= \boldsymbol{\varepsilon}_y^\top P_y \boldsymbol{\varepsilon}_y + \boldsymbol{\varepsilon}_y^\top P_y \boldsymbol{\delta}_y + \boldsymbol{\delta}_y^\top P_y \boldsymbol{\varepsilon}_y + \boldsymbol{\delta}_y^\top P_y \boldsymbol{\delta}_y \\
&= \boldsymbol{\varepsilon}_y^\top P_y \boldsymbol{\varepsilon}_y + 2\boldsymbol{\varepsilon}_y^\top P_y \boldsymbol{\delta}_y + \boldsymbol{\delta}_y^\top P_y \boldsymbol{\delta}_y \\
&\leq \boldsymbol{\varepsilon}_y^\top P_y \boldsymbol{\varepsilon}_y + 2||\boldsymbol{\varepsilon}_y^\top P_y \boldsymbol{\delta}_y||_2^2 + \boldsymbol{\delta}_y^\top P_y \boldsymbol{\delta}_y \\
&\leq \boldsymbol{\varepsilon}_y^\top P_y \boldsymbol{\varepsilon}_y + 2||\boldsymbol{\varepsilon}_y^\top P_y \boldsymbol{\delta}_M||_2^2 + \boldsymbol{\delta}_M^\top P_y \boldsymbol{\delta}_M
\end{aligned} \tag{4-17}$$

*The steps above can be repeated for all the prediction errors and for each order of the generalised motions, leading to the result of* **Theorem 1**. *This concludes the proof.* □

Using **Theorem 1**, we can define an upper bound on the free-energy $\mathcal{F}$ once the maximum uncertainty $\boldsymbol{\delta}_M$ is specified. Furthermore, since $\mathcal{F}$ is a sum of squared prediction errors, the upper bound on $\mathcal{F}$ will always be an upper bound of the single sensory prediction errors.

***Corollary 1*** *Let us consider a generic sensory prediction error $(y_s^{(i)} - g_s^{(i)}(\boldsymbol{\mu}))$ for a sensor $s$, and let us call it $\varepsilon_s^{(i)}$. Assuming that different values of maximum uncertainties are available for each sensory prediction error, such that:*

$$-\delta_{M_s}^{(i)} \leq \delta_s^{(i)} \leq \delta_{M_s}^{(i)}, \quad i = 1...n_d - 1, \quad \delta_{M_s}^{(i)} \in \mathbb{R}^+ \tag{4-18}$$

*Then, each sensory prediction error can be bounded by a specific threshold as:*

$$(\varepsilon_s^{(i)} + \delta_s^{(i)})^\top \sigma_{y^{(i)}}^{-1} (\varepsilon_s^{(i)} + \delta_s^{(i)}) \leq \varepsilon_s^{(i)\top} \sigma_{y^{(i)}}^{-1} \varepsilon_s^{(i)} + \delta_{M_s}^{(i)\top} \sigma_{y^{(i)}}^{-1} \delta_{M_s}^{(i)} + 2||\varepsilon_s^{(i)\top} \sigma_{y^{(i)}}^{-1} \delta_{M_s}^{(i)}||_2^2 \tag{4-19}$$

**Proof of Corollary 1** *The proof directly follows from the* ***Proof of Theorem 1****, if different maximum values for the uncertainties* $\delta^{(i)}_{M_s}$ *are considered instead of a single* $\boldsymbol{\delta}_M$. □

## 4-3-2 Fault detection and isolation

As explained in Section 2-3, in conventional model based FDI, there is in general the need of generating the residual signals for fault detection. Another advantage of using the free-energy, is that $\mathcal{F}$ already provides the normalised residuals as sensory prediction errors. Thus, we can limit our effort to the computation of the threshold. The latter is done according to *Corollary 1*, using the estimated sensory input through an observer. In particular, the sensory prediction errors for each group of sensors (i.e. encoders, velocity sensors and visual camera), are compared with the thresholds generated using Eq. (4-19), where the real sensory data is substituted with the observed ones. Using different thresholds for different groups of sensors, will lead to directly perform fault isolation, besides giving a tighter threshold with respect to the one over the whole $\mathcal{F}$ as expressed in Eq. (4-16).

### State observer

A simple observer is here introduced to determine the estimated sensory input for the on-line threshold. We can express the following state-space model defining $[x_1\ x_3]^\top = \boldsymbol{q}$ and $[x_2\ x_4]^\top = \dot{\boldsymbol{q}}$ and thus $\boldsymbol{x} = [x_1\ x_2\ x_3\ x_4]$. A Luenberger observer for $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$ is given by:

$$\begin{cases} \dot{\hat{\boldsymbol{x}}} = A\hat{\boldsymbol{x}} + L(\boldsymbol{y} - \hat{\boldsymbol{y}}) \\ \hat{\boldsymbol{y}} = \hat{\boldsymbol{x}} \end{cases} \tag{4-20}$$

where $L \in \mathbb{R}^{n\times n}$ is the observer gain and:

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix} \text{ and } L > 0 \tag{4-21}$$

The estimation error dynamics are given by:

$$\dot{\bar{\boldsymbol{x}}} = \dot{\boldsymbol{x}} - \dot{\hat{\boldsymbol{x}}} \tag{4-22}$$

which leads to:

$$\dot{\bar{\boldsymbol{x}}} = (A - L)\bar{\boldsymbol{x}} \tag{4-23}$$

Therefore, $L$ should make $(A - L)$ Hurwitz such that $\lim_{t\to\infty} \bar{\boldsymbol{x}}(t) = 0$. The modes of the response are determined by the eigenvalues of $(A - L)$. Since the system is observable, the eigenvalues of $(A - L)$ can be placed arbitrarily. Choosing $L$ as diagonal positive definite, will asymptotically bring the estimation error to zero. Note that the full state is available through measurements. The use of the observer is necessary for two main reasons: the first is to obtain a filtered version of the sensory input, and the second is to avoid drifts of the estimated quantities due to unmodeled dynamics. In this way, the threshold generated using the observed sensory input will follow any abrupt change, but with a certain transient imposed by $L$. This will make the real sensory prediction errors exceed the threshold in case of detectable sensory faults, at least for a certain amount of time.

#### Threshold selection

According to Eq. (4-19) and Eq. (4-20), an on-line threshold for a generic sensor is given by:

$$\begin{aligned} \tau_s &= (\hat{y}_s^{(i)} - g_s^{(i)})^\top P_{y_s^{(i)}} (\hat{y}_s^{(i)} - g_s^{(i)}) + \delta_{M_s}^{(i)\top} P_{y_s^{(i)}} \delta_{M_s}^{(i)} \\ &+ 2||(\hat{y}_s^{(i)} - g_s^{(i)})^\top P_{y_s^{(i)}} \delta_{M_s}^{(i)}||_2^2 \end{aligned} \tag{4-24}$$

where $s$ indicates the type of the sensor, i.e. for a 2-DOF robot arm with proprioceptive sensors for position and velocity of the joints and a visual camera, $s = \{q,\ \dot{q},\ v\}$. Note that the camera output is not directly estimated from the observer, but this can be reconstructed using the already available generative model $\boldsymbol{g}_v$ and the estimated $\hat{\boldsymbol{q}}$. Finally, when the threshold $\tau_s$ is chosen, we have to compare the sensory prediction errors against it. If we assume that $y_s^{(i)} - g_s^{(i)}(\boldsymbol{\mu})$ is a generic sensory prediction error for $i = 0, ..., n_d - 1$, it holds:

$$\begin{cases} y_s^{(i)} - g_s^{(i)}(\boldsymbol{\mu}) & \leq \tau_s \quad \text{Faultless} \\ y_s^{(i)} - g_s^{(i)}(\boldsymbol{\mu}) & > \tau_s \quad \text{Fault in sensor } s \end{cases} \tag{4-25}$$

### 4-3-3 Fault recovery

The final part to be detailed, is the definition of the recovery actions after a fault has been detected and isolated. The fault recovery in case of sensory faults is simplified by means of the free-energy principle. We exploit indeed the fact that the controller encodes the precision matrices $P_{y_q}$, $P_{y_{\dot{q}}}$ and $P_{y_v}$ to define the confidence about each sensory input. The framework allows to include in a natural way multiple sensors for the state estimation and control actions. Every sensor contributes to the posterior prediction of the most plausible state of the robot arm. If a sensor is marked as faulty, it is sufficient to decrease the confidence about that specific input to perform recovery. The controller can thus exploit the sensory redundancy for two reasons: the first one is to have a better a posteriori approximation of the states, and the second is to compensate for missing or wrong sensory data. Formally, once a fault is detected and isolated using the approach explained before, the precision matrix of the faulty sensor $P_{fs}$ is reduced to small values or to zero:

$$P_{fs} = \mathbf{0} \tag{4-26}$$

Note that there is no need to design a switching procedure between different sensors, but the recovery is embedded in the controller itself.

### 4-3-4 Analysis of false and missed alarms rate

We present now an analysis of the detectability of the faults, as well as the false alarm rate (FAR) associated with the active inference based fault tolerant scheme. The detectability of a fault depends on the entity of the fault itself, and on the selection of the maximum uncertainties $\delta_{M_s}^{(i)}$ for the definition of the threshold. In particular, according to Eq. (4-24), a fault which will produce an abrupt anomaly in the sensory input bigger than $\delta_{M_s}^{(i)}$ will be detected. For what concerns the FAR, this is determined considering again $\delta_{M_s}^{(i)}$. In practice

this value can be chosen according to the standard deviation of the noise present in the sensory data. The maximum uncertainty can then be set, for example, as $5\sigma_s^{(i)}$ or $7\sigma_s^{(i)}$ where $\eta_s \sim (0, \sigma_s^{(i)})$ is the Gaussian noise affecting the sensor $s$. Choosing $5\sigma_s^{(i)}$, for instance, the probability of having a false alarm due to an abrupt change in the sensory input related to the noise, and not to a fault, is less that $10^{-6}$. Having said that, also the dynamics of the observer influences the FAR. However, if these dynamics are imposed to be fast enough through the observer gain $L$, the effect results negligible, and the FAR remains limited.

## 4-4 Simulation results

The simulations are carried out using the 2-DOF robot specified in Chapter 3. The simulation consists in controlling the robot arm from the initial position $[-\pi/2,\ 0]$ to the desired position $\boldsymbol{\mu}_d = [-0.6,\ 0.5]$. A fault is supposed during the trajectory towards the final position, thus we set $t_f = 2s$. For encoders and tachometers, the maximum uncertainties $\delta_{M_q}$ and $\delta_{M_{\dot{q}}}$ are both set to $5\sigma_q = 5\sigma_{\dot{q}} = 0.005$. For the more noisy camera data, we chose $\delta_{M_v} = 7\sigma_\eta = 0.07$.

### Fault in the encoder: sensor's output freezing

In this scenario, a fault is injected in the first encoder, freezing its output for $200ms$ at $t_f = 2s$. Fig. 4-4 depicts the behaviour of the normalised sensory prediction errors. As can be seen, the detection and recovery happens at $t_{DR}$, as soon as the threshold is exceeded, so when the ratio $\varepsilon_{q_1}/\tau_q$ is bigger than one.

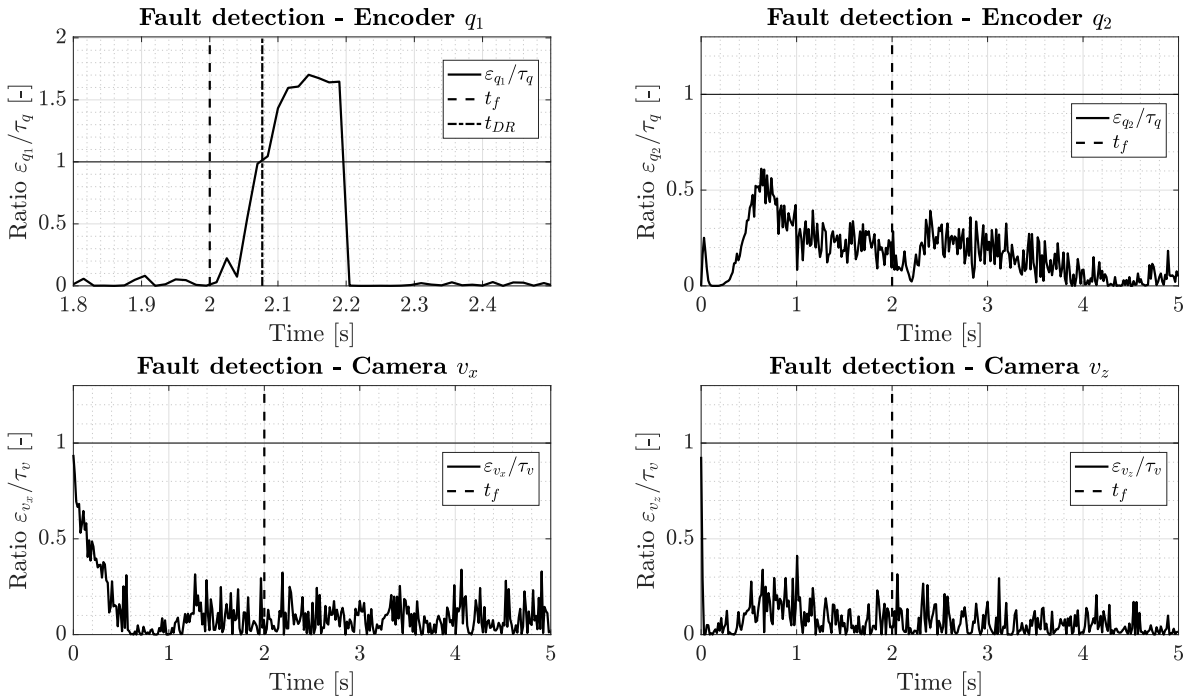



**Figure 4-4:** Normalised sensory prediction errors in case of fault in the encoder for $q_1$

### Fault in the camera: occlusion

Camera occlusion is injected at $t_f = 2s$. The detection for this case is faster due to the bigger entity of the fault. In fact, the entity of this anomaly, which equals 0.075, is slightly higher than the supposed maximum uncertainty $\delta_{M_v} = 0.07$, thus the FDI is fast. The detection and recovery in such a scenario are presented in Fig. 4-5. The actual response of the robot is not influenced significantly, as shown in Fig. 4-6.

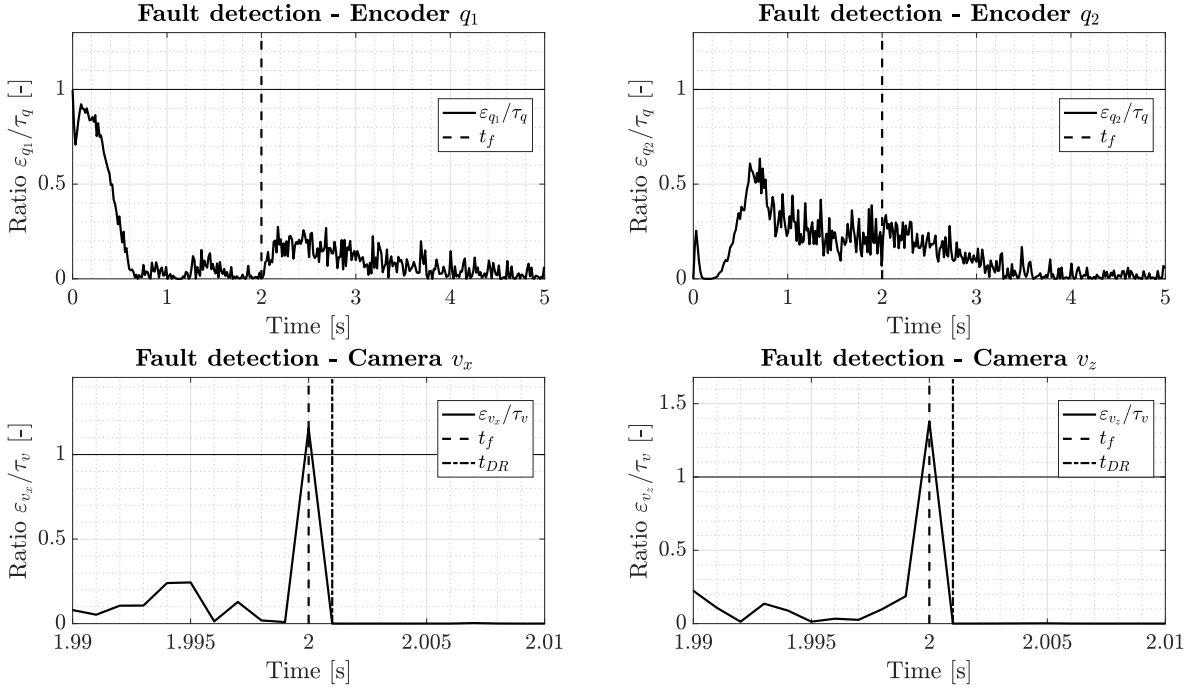



**Figure 4-5:** Normalised sensory prediction errors for fault detection and recovery in case of camera occlusion

For an interested reader, the behaviour of the normalised prediction errors in a faultless case can be found in Appendix B-3. The proposed algorithm showed high performance for fault detection, isolation and recovery. Since the single sensory prediction errors can be bounded by a time varying threshold separately, fault detection and isolation are achieved simultaneously. The smaller the effect of the fault on the system, the greater the time taken for detection. For the first case analysed, the encoder fault is detected isolated and recovered within 82 $[ms]$. The camera occlusion, instead, is detected isolated and recovered in just one time steps (i.e. 1 $[ms]$). The entity of this fault is indeed considerably higher than the previous one.

### Recovery performance and error in joint space

We report now the recovery performance, analysing the trajectory of the robot arm in the joint space. Since the fault detection and recovery is fast, the effects of the faults in the overall execution of the task is limited, as depicted in Figure 4-6. In this case, after the camera occlusion is detected and recovered, the system relies only on the noisy encoders to reach the desired goal. It can be noticed how the internal beliefs converge again to the true

states after the fault, due to the combined effect of state and actions update. The resulting trajectory of the joints is marginally affected.

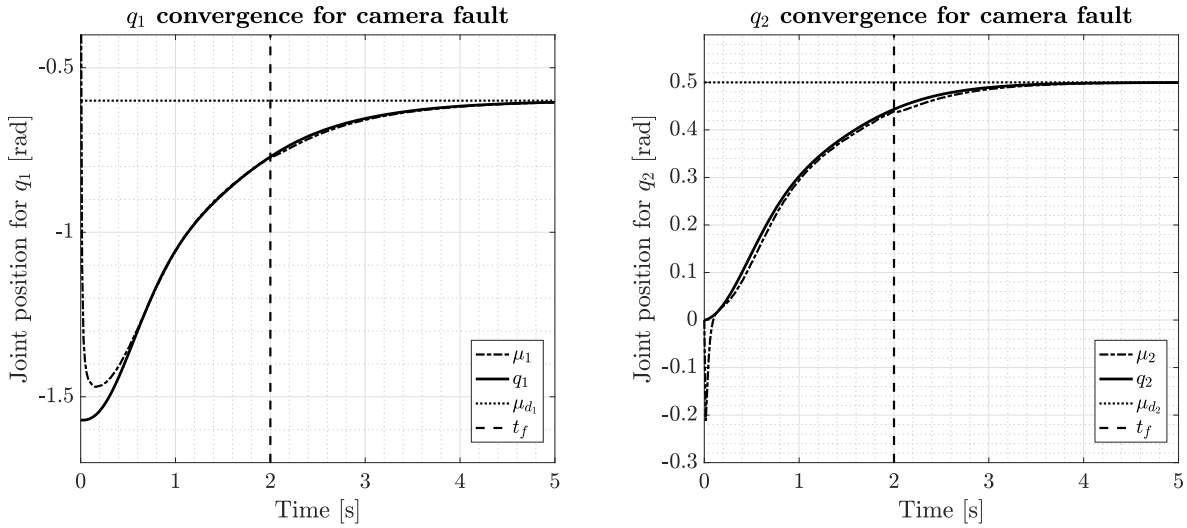



**Figure 4-6:** Joint space trajectory with recovery from camera occlusion

## 4-5 Concluding remarks

In this chapter, we presented a novel approach to obtain a fault tolerant controller using the free-energy principle and active inference. The core contribution of this work is the derivation of an on-line threshold for FDI, based on the sensory prediction errors defined in the free-energy. This has been possible after proving the existence of an upper bound on the free-energy for a generic $n$-DOF robot manipulator. The overall solution only requires an observer besides the default controller to be able to perform FDI and recovery from sensory faults. The simultaneous detection and isolation of faults is possible since the single sensory prediction errors are compared against a dedicated threshold. Furthermore, the free-energy facilitates the inclusion of redundant sensors, and it allows for recovery actions without resort to switching procedures. The main limitation of the proposed approach is the fact that only sensory faults can be detected and isolated using the current formulation. An extension to actuator faults could be achieved adding supplementary sensory data, for instance current sensors, and extending the observer. However, the definition of the generative models for the supplementary sensors could result more challenging. Simulation results were included demonstrating the effectiveness of the proposed solution applied to a 2-DOF robot arm subject to severe faults in the proprioceptive and visual sensors.

# Chapter 5

# Conclusions

*This conclusive chapter summarises the content presented in this work, and the answers to the initial research questions. Besides, some guidelines for future work and current challenges to be solved are reported.*

## 5-1 Summary

In this work we proposed an adaptive and fault tolerant control scheme based on active inference. The starting point was the analysis of the limitations of past work, focusing on the reasons that prevented the development of active inference controllers for robotics. From the preliminary study of the state-of-the-art, the dynamic generative model described in [5], and the static state estimation using the free-energy from [25], appeared promising for reducing the computational complexity of the algorithm.

Combining these ideas with the standard definition of the free-energy principle and active inference, a control architecture for a generic $n$-DOF robot manipulator has been derived. The main challenges in the definition of the generative models, for state estimation and joint space control, have been solved introducing model approximations and defining the states to be controlled as the joint values of the manipulator. More in detail, to be able to compute the expression for the control actions update, an approximation of the true dynamics between control input and change is the sensory data was proposed. Instead of providing the full relation, in fact, only the direction of change was encoded, relying on the high adaptability of the active inference itself. This approximation allowed to overcome the computational limitations highlighted by previous work, leading to a novel application of active inference for an on-line robotic application. The validity of the assumptions taken, and the performance of the novel control architecture, have been tested against another adaptive scheme. The simulation results showed that active inference intrinsically possesses high adaptability to unmodeled dynamics, outperforming the model reference adaptive controller used for comparison.

The adaptability properties, and the capability of the free-energy to include different sensory input in a natural way, suggested the study of the framework from a fault tolerant point of

view. In the fourth chapter we derived a novel fault tolerant active inference based control scheme. We described how the sensory redundancy can be included to recover from sensory faults, without the need of external supervision systems besides the default controller. After proving the existence of an upper bound on the sensory prediction errors, an on-line threshold has been derived to simultaneously detect and isolate sensory faults. A distinguishing feature of this work, is that the fault detection, isolation and recovery are performed within the controller itself, while other traditional solutions need ad-hoc external architectures. The recovery actions were implemented exploiting two things: the fact that the free-energy easily allows for sensory redundancy, and the fact that the precision matrices in the controller can be tuned online to exclude the faulty sensory input. The simulation results confirmed the suitability of the framework for fault detection isolation and recovery in case of sensory faults.

To conclude, a diagram summarising the work-flow of the thesis is depicted at the end of this chapter, in Figure 5-1. The scheme indicates the main references, theoretical concepts and equations taken as starting point for this study. The evolution of the proposed solution from promising existing theory is clearly depicted, highlighting the main underlining ideas which brought to the adaptive and fault tolerant active inference controller.

## 5-2 The answers to the research questions

The research goal of this thesis was to investigate the relevance of active inference for robot control since, at the time this research has been conducted, there was no clear evidence of its applicability for real-time applications. Besides, the adaptability performance of the framework was only claimed in past work, but not verified against other adaptive control architectures. The outcome of this work led to the following answers to the initial research questions.

***Is active inference suitable for robot control?***
The control architecture derived in Section 3-3 showed that the active inference algorithm is effective in a real-time application for robot control. The approximations introduced during the definition of the control scheme, allowed to reduce the computational complexity, relying on the adaptability of active inference to compensate for unmodeled dynamics. The control scheme resulted performing and easily scalable to high degrees of freedom. The solution is particularly suitable for joint space control, for tasks in which the robot's dynamic are uncertain or subject to relevant changes during the operations.

***What are the adaptability properties of active inference compared to other adaptive solutions?***
The comparison of the AIC with the MRAC in Section 3-4 showed that the adaptability performance of this novel control architecture is comparable and even superior to the other adaptive scheme in the proposed control problem. This thesis confirmed what predicted in previous work about the adaptability of the framework against noise and model uncertainties.

***Can the free-energy principle structure be exploited to obtain an intrinsically fault tolerant controller?***
Many advantages regarding active inference emerged during this study. In Chapter 4 we explained that one of these was the ability of extending in a natural way the set of sensory input for state estimation, using heterogeneous sources. This is crucial when considering a system which is subject to sensory faults, since it allows to implement sensory redundancy. We also showed that the recovery actions could directly be implemented within the controller itself, changing the confidence about the faulty sensory input. Overall, then, the structure of active inference can be seen as an intrinsically fault tolerant scheme, which facilitates the detection, isolation and recovery from sensory faults.

To conclude, the novelties presented in Chapter 3 and 4, which allowed to answer the research questions above, will constitute the main content of the two publications that will follow this work. In particular, the first paper will contain the novel active inference controller, and the performance comparison with the MRAC. Then, a second publication will explain the active inference based fault tolerant algorithm, as in Chapter 4.

## 5-3 Future challenges and recommendations

Active inference undoubtedly bares a considerable potential for adaptive and fault tolerant robot control. This work represents only the first step towards a new category of control architectures, thus we provide some guidelines to keep exploring this interesting area. One of the main points which deserves attention, is the stability proof of the closed-loop active inference scheme.

Future work may also focus on a different definition of the generative functions for the state evolution, in order to include motion constraints. In addition, since in this work we only focused on joint space control, a relevant research direction would be the extension to control in Cartesian space, without using inverse models.

Another interesting topic regarding fault tolerance, would be to investigate the effect of learning the variances associated with the prediction errors. This could allow the AIC to automatically perform the fault recovery without any further actions.

To conclude, what we understood from this study, is that the free-energy principle is as complicated as it is powerful. Thus, the author feels to encourage the further development of this fascinating brain inspired controller for robotic applications.

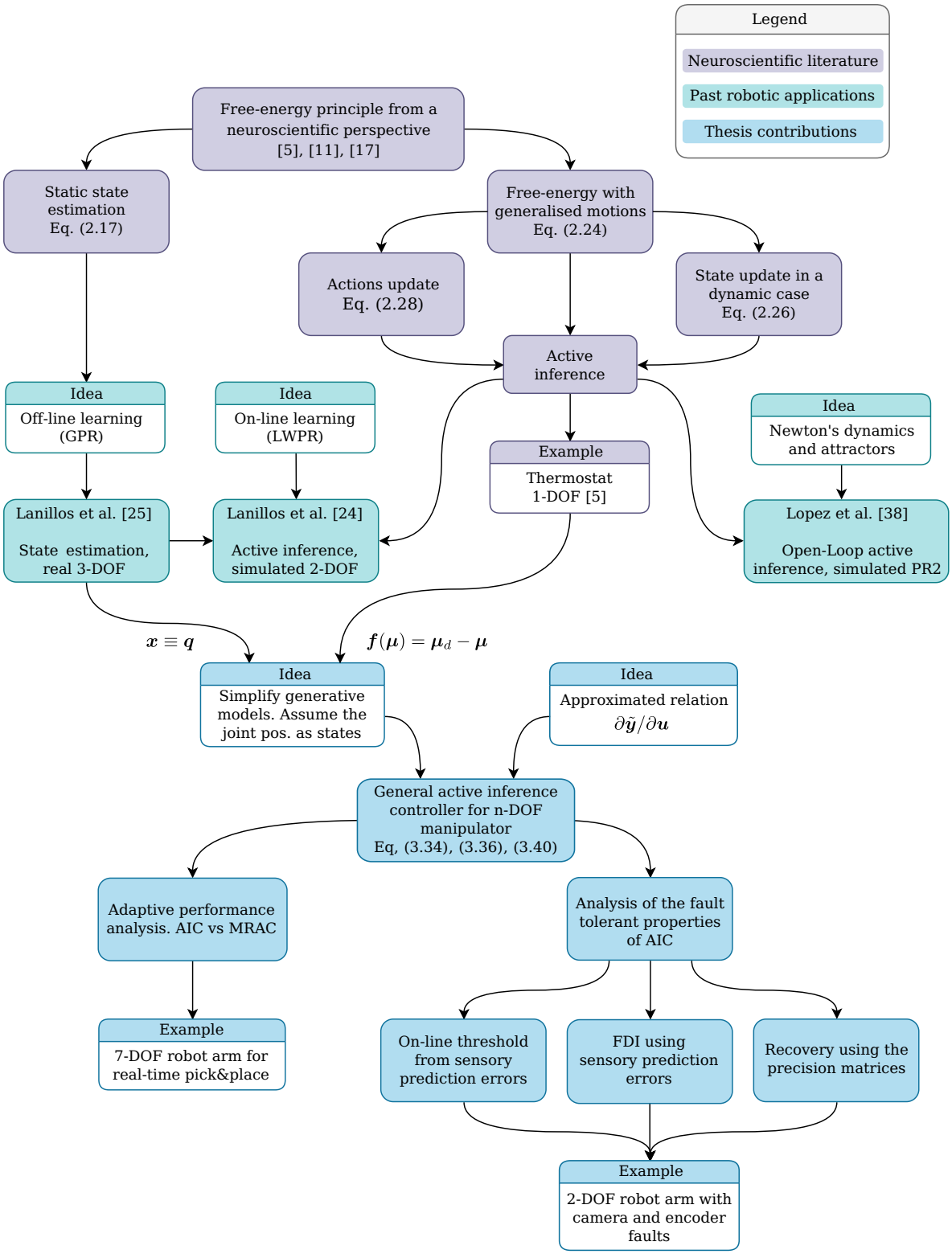



**Figure 5-1:** Thesis work-flow and contributions

# Appendix A

# Coding active inference

*The scripts here reported, and the C++ code for active inference applied to a 7-DOF manipulator using ROS and Gazebo, are freely available at* https://github.com/cpezzato/panda_simulation.

## A-1 Free-energy minimisation in a static environment

```
%% State estimation using free-energy minimisation - Static case

%% Set-up variables for simulation
t = 0.3;        % [s] Simulation time
h = 0.001;      % [s] Integration step

% 2DOF robot parameters
a1 = 1;         % [m] Length link 1
a2 = 1;         % [m] Length link 2

%% Set-up variables for perception
% Real state of the robot, fixed joint position, no control actions
q = [0.3 0.8]'; % [rad]

% Prior belief about the states of the robot arm
mu_d = q;       % [rad]

% Precision matrix for the prior belief
P_mu = eye(2);
% Precision matrix for the proprioceptive sensory data
P_y = eye(2);

% Learning rate
k_mu = 20;

%% Initialization
```

```
% Initialize the vector of beliefs abot the states
mu = zeros(2,t/h);

% Initialize vector for collecting the free-energy values
F = zeros(1,t/h-1);

% Random initial guess about the states of the robot, initial conditions
mu(:,1) = [1 1.2]';

%% Free-energy minimization using gradient descent

for i=1:t/h-1

    % Simulate noisy sensory input from encoders
    % Noise
    z = random('norm', 0, 0.001, length(q), 1);
    % Sensory input
    y_q = q + z;

    % Free-energy computation
    F(i) =  1/2*((y_q-mu(:,i))'*P_y*(y_q-mu(:,i)) + ...
        (mu(:,i)-mu_d)'*P_mu*(mu(:,i)-mu_d));

    % Define free-energy gradient
    gradF = -((y_q-mu(:,i))'*P_y - (mu(:,i)-mu_d)'*P_mu);

    % State update using gradient descent on the free-energy
    mu(:,i+1) = mu(:,i)-k_mu*h*gradF';
end
```

## A-2 Active inference

```
%% Active inference for robot control - Dynamic case

%% Set-up variables for simulation

t = 6;          % [s] Simulation time
h = 0.001;      % [s] Integration step
actionsTime = 1;

% 2DOF robot parameters
a1 = 1;         % [m] Length link 1
a2 = 1;         % [m] Length link 2

%% Set-up variables for perception

% Initialize generative process (so the sensors' output)
q = zeros(2,t/h);   % [rad]
dq = zeros(2,t/h);  % [rad/s]
ddq = zeros(2,t/h); % [rad/s^2]

% Initial state of the robot
q(:,1) = [-pi/2, 0]';
```

```
22
23 % Prior belief about the states of the robot arm, desired position
24 mu_d = [-0.2 0.3]';
25
26 %% Tuning parameters
27
28 % Confidence in the prior belief about the states
29 P_mu0 = eye(2);
30 % Confidence in the prior belief about the derivative of the states
31 P_mu1 = eye(2);
32 % Confidence in the proprioceptive sensory data (position)
33 P_y0 = eye(2);
34 % Confidence in the proprioceptive sensory data (velocity)
35 P_y1 = eye(2);
36
37 % Learning rate for beliefs update
38 k_mu = 20;
39 % Learning rate for actions
40 k_a = 500;
41
42 %% Initialize vectors
43
44 % Initialize actions vector
45 u = zeros(2,t/h);
46 % Initial control action
47 u(:,1) = [0 0]';
48
49 % Initialize the vector of beliefs about the states and their derivatives
50 mu = zeros(2,t/h);
51 mu_p = zeros(2,t/h);
52 mu_pp = zeros(2,t/h);
53
54 % Initial guess about the states of the robot, initial belief
55 mu(:,1) = q(:,1)+[+0.6 +0.2]';   % mu    position + random constant to ...
       appreciate the convergence
56 mu_p(:,1) = [0 0]';              % mu'   velocity
57 mu_pp(:,1) = [0 0]';             % mu''  acceleration
58
59 % Initialize vector for collecting the free-energy values
60 F = zeros(1,t/h-1);
61
62 % Initialize vectors for sensory input (these will be noisy)
63 y_q = zeros(2,t/h);
64 y_dq = zeros(2,t/h);
65
66 %% Active Inference loop
67
68 for i=1:t/h-1
69
70     %% Simulate noisy sensory input from encoders and tachometers
71     z = random('norm', 0, 0.001, size(q,1), 1);
72     z_prime = random('norm', 0, 0.001, size(q,1), 1);
73     y_q(:,i) = q(:,i) + z;
74     y_dq(:,i) = dq(:,i) + z_prime;
75
76     %% Compute free-energy in generalised motions
```

```
77      F(i) = 0.5*(y_q(:,i)-mu(:,i))'*P_y0*(y_q(:,i)-mu(:,i))+... ...
                    % Proprioceptive position for joint 1 and 2
78          + 0.5*(y_dq(:,i)-mu_p(:,i))'*P_y1*(y_dq(:,i)-mu_p(:,i))+... ...
                    % Proprioceptive velocity for joint 1 and 2
79          + ...
              0.5*(mu_p(:,i)+mu(:,i)-mu_d)'*P_mu0*(mu_p(:,i)+mu(:,i)-mu_d)+... ...
              % Model prediction errors mu_p
80          + 0.5*(mu_pp(:,i)+mu_p(:,i))'*P_mu1*(mu_pp(:,i)+mu_p(:,i)); ...
                    % Model prediction errors mu_pp
81
82      %% Belifs update
83      % Support variables for beliefs update
84      mu_dot = mu_p(:,i) - k_mu*(-P_y0*(y_q(:,i)-mu(:,i)) + ...
            P_mu0*(mu_p(:,i)+mu(:,i)-mu_d));
85      mu_dot_p = mu_pp(:,i) - k_mu*(-P_y1*(y_dq(:,i)-mu_p(:,i)) ...
86                            +P_mu0*(mu_p(:,i)+mu(:,i)-mu_d) ...
87                            +P_mu1*(mu_pp(:,i)+mu_p(:,i)));
88      mu_dot_pp = - k_mu*(P_mu1)*(mu_pp(:,i)+mu_p(:,i));
89
90      % State estimation
91      mu(:,i+1) = mu(:,i) + h*mu_dot;              % Belief about the position
92      mu_p(:,i+1) = mu_p(:,i) + h*mu_dot_p;        % Belief about motion of mu
93      mu_pp(:,i+1) = mu_pp(:,i) + h*mu_dot_pp;     % Belief about motion of mu'
94
95      %% Control actions
96      if i > actionsTime/h
97          % Active inference
98          u(:,i+1) = u(:,i)-h*k_a*(P_y1*(y_dq(:,i)-mu_p(:,i)) + ...
                P_y0*(y_q(:,i)-mu(:,i)));
99      else
100         u(:,i+1) = [0,0]';  % Set control torques to zero during the ...
                initial part
101     end
102
103     %% Update sensory input according to the actions taken
104     ddq(:,i) = RealrobotDynamics(q(1,i),q(2,i),dq(1,i),dq(2,i),u(1,i),u(2,i));
105     dq(:,i+1) = dq(:,i)+h*ddq(:,i);
106     q(:,i+1) = q(:,i)+h*dq(:,i);
107
108 end
```

# Appendix B

# Additional simulation results

## B-1 Pick and place with a 7-DOF robot manipulator - Approximated dynamical model

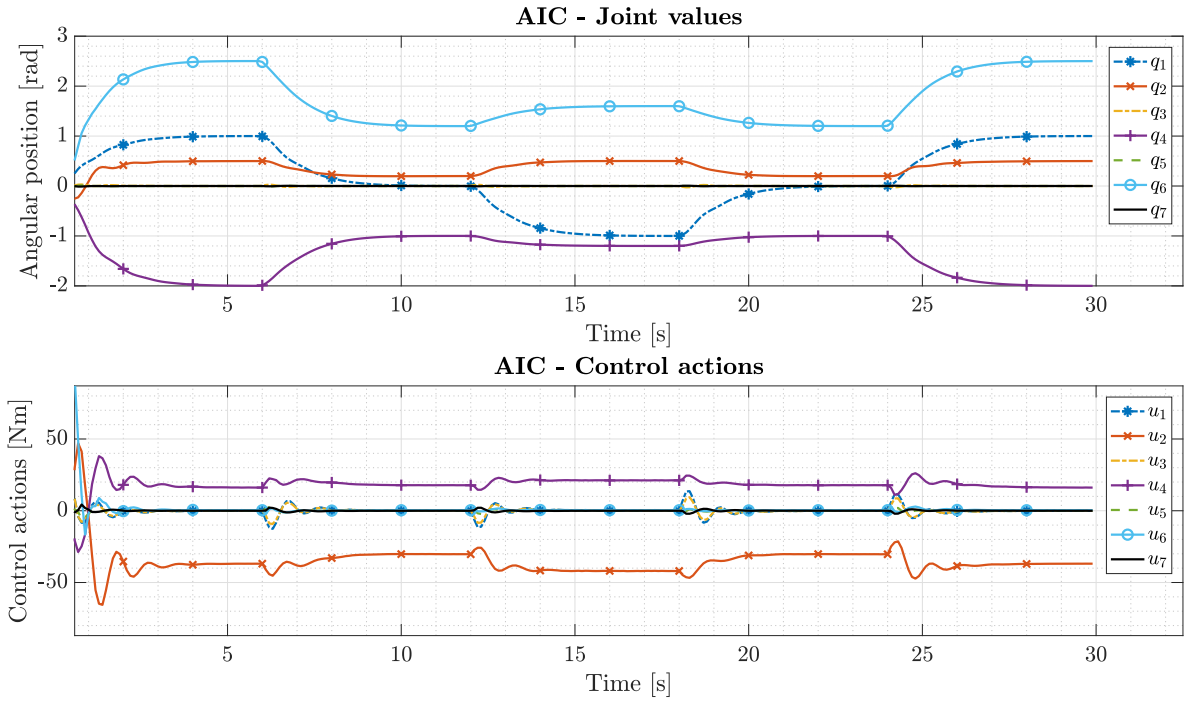



**Figure B-1:** Full response and control actions using AIC. Simulation using ROS and Gazebo.

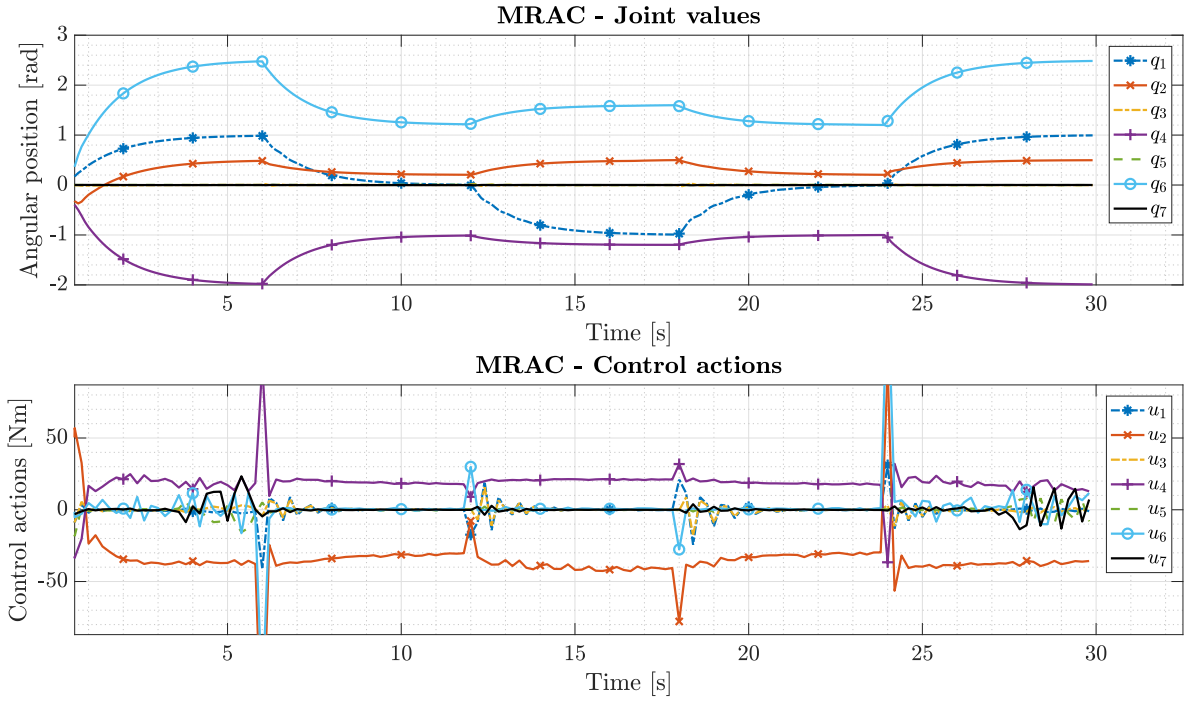



**Figure B-2:** Full response and control actions using MRAC. Simulation using ROS and Gazebo.

## B-2 Performance degradation due to large unmodeled dynamics

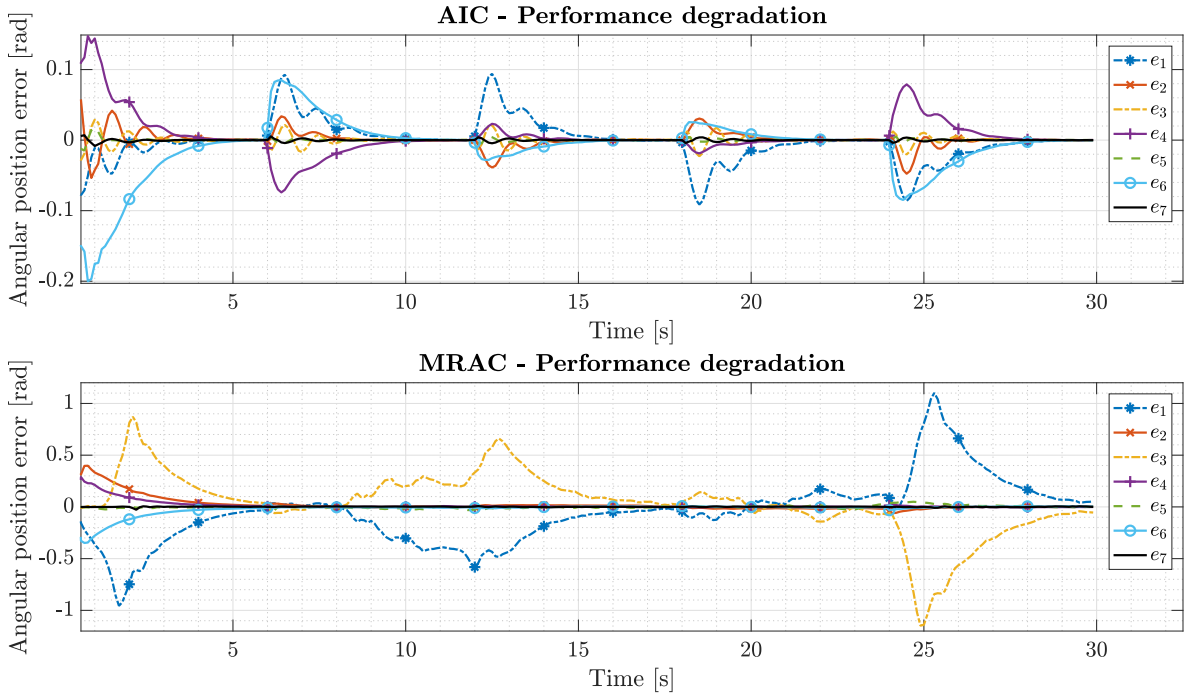



**Figure B-3:** Full performance degradation for the 7-DOF example. The controllers tuned using the approximated model are applied to the accurate system. Simulation using ROS and Gazebo.

## B-3 Normalised sensory prediction errors in faultess case for 2-DOF manipulator

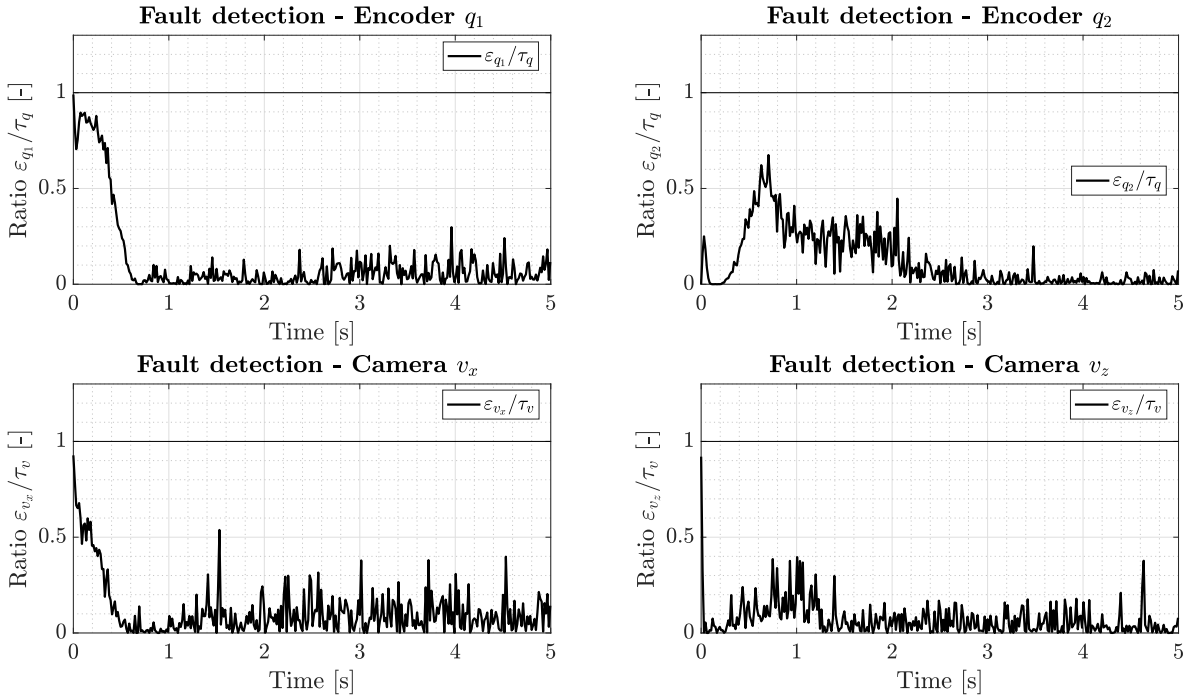



**Figure B-4:** Normalised sensory prediction errors for fault detection and recovery in a faultless case

# Appendix C

# Additional background knowledge

## C-1 Gaussian process regression

For the general mathematical definition of the GPR based on [40], we will make use of the following notation:

**Table C-1:** General notation for Gaussian process regression

| Symbol | Meaning |
|---|---|
| $\boldsymbol{x}$ | Training sample $\in \mathbb{R}^m$ |
| $y$ | Target value $\in \mathbb{R}$ corresponding to $\boldsymbol{x}$ |
| $\bar{\boldsymbol{x}}$ | Collection of training samples $\in \mathbb{R}^{m \times n}$ |
| $\bar{y}$ | Collection of target values $\in \mathbb{R}^n$ |
| $\boldsymbol{l}$ | Length scale for the kernel $\in \mathbb{R}^m$ where $m$ is the number dimension of a generic training sample $\boldsymbol{x}$ |
| $\boldsymbol{\theta}$ | Hyperparameters, in case of squared exponential kernel with noisy measurements $\boldsymbol{\theta} = \{\boldsymbol{l},\ \sigma_f,\ \sigma_n\}$ |

In a Gaussian process, the inference involves only multivariate Gaussian distributions. Let us assume that we have a set of points defined as follows:

$$D = \{(\boldsymbol{x_1}, y_1), (\boldsymbol{x_2}, y_2), ..., (\boldsymbol{x_n}, y_n)\}\ \boldsymbol{x_i} \in \mathbb{R}^m,\ y_i \in \mathbb{R},\ i = 1...n \tag{C-1}$$

In Gaussian regression we are always looking at a finite set of data, and we are trying to determine the distribution of the pairs $(\boldsymbol{x_i}, y_i)$. To do so, we assume that this distribution can be described as a sum of two independent distributions:

$$Y = Z_x + \boldsymbol{\varepsilon} \tag{C-2}$$

where $Z_x \sim (\boldsymbol{\mu}, K_z)$ is a multivariate Gaussian with mean $\boldsymbol{\mu}$ (often set to zero) and covariance matrix $K_z$. Finally $\boldsymbol{\varepsilon} \sim (0, \sigma_n^2 I)$. Since $Y$ is a sum of two independent distributions it holds that (assuming $\boldsymbol{\mu} = 0$):

$$Y \sim \mathcal{N}(0, \underbrace{K_z + \sigma_n^2 I}_{K}) \tag{C-3}$$

Now, we describe the crucial point in order to be able to make predictions about a new expected output $y_*$ given an input $\boldsymbol{x_*}$, a set of training inputs $\bar{\boldsymbol{x}}$, and training targets $\bar{y}$. In particular, let us assume that we have a set of $n$ observations of the input and output variables.

Since the GPR assumes that the data is represented by multivariate Gaussian distributions, we can write the following partition:

$$Y = \begin{bmatrix} \bar{y} \\ y_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} K & K_*^\top \\ K_* & K_{**} \end{bmatrix}\right) \tag{C-4}$$

We are interested in the conditional probability $p(y_*|\bar{y})$, in other words we want to know how likely a certain prediction for $y_*$ is. The probability follows a Gaussian distribution:

$$y_*|\bar{y} \sim \mathcal{N}(K_* K^{-1}\bar{y}, K_{**} - K_* K^{-1} K_*^\top). \tag{C-5}$$

The best guess of the value of $y_*$ given a new observation $\boldsymbol{x_*}$, is given by the mean of the above distribution, that is:

$$\mathbb{E}(y_*) = K_* K^{-1}\bar{y} \tag{C-6}$$

Finally, the variance of the best guess about $y_*$ is given by:

$$var(y_*) = K_{**} - K_* K^{-1} K_*^\top \tag{C-7}$$

To be able to compute these quantities, we have to define the matrices $K$, $K_*$ and $K_{**}$. The relation between one observation and another is encoded in the beating heart of this process, the covariance matrix $K$, and especially the covariance function $k(\boldsymbol{x_i}, \boldsymbol{x_j})$. In literature there are different ways of choosing this covariance function or *kernel* [40]. The most common choice is to use a *squared exponential kernel* of the form:

$$k(\boldsymbol{x_i}, \boldsymbol{x_j}) = \sigma_f^2 \exp\left(-\tfrac{1}{2}(\boldsymbol{x_i} - \boldsymbol{x_j})^\top \Lambda (\boldsymbol{x_i} - \boldsymbol{x_j})\right) + \sigma_n^2 d_{ij} \tag{C-8}$$

where $d_{ij}$ is the Kronecker delta, $d_{ij} = 1$ when $i = j$ and 0 otherwise. Furthermore, one possible choice of $\Lambda$ is $\Lambda = diag\{l_p^2\}$, $p = 1, ..., m$. The hyperparameters in this expression are $hyp = \{\boldsymbol{l},\ \sigma_f,\ \sigma_n\}$, with $\boldsymbol{l} = \{l_1,\ l_2, ...,\ l_m\}$. The choice of these parameter highly influences the predictions of $y_*$. They can be initially set to a value and subsequently optimized to maximize the marginal likelihood. This part will be explained later on. For the covariance matrix $K$ it holds:

$$K = \begin{bmatrix} k(\boldsymbol{x_1}, \boldsymbol{x_1}) & k(\boldsymbol{x_1}, \boldsymbol{x_2}) & \cdots & k(\boldsymbol{x_1}, \boldsymbol{x_n}) \\ k(\boldsymbol{x_2}, \boldsymbol{x_1}) & k(\boldsymbol{x_2}, \boldsymbol{x_2}) & \cdots & k(\boldsymbol{x_2}, \boldsymbol{x_n}) \\ \vdots & \vdots & \ddots & \vdots \\ k(\boldsymbol{x_n}, \boldsymbol{x_1}) & k(\boldsymbol{x_n}, \boldsymbol{x_2}) & \cdots & k(\boldsymbol{x_n}, \boldsymbol{x_n}) \end{bmatrix} \tag{C-9}$$

The other two matrices are defined by:

$$K_* = \begin{bmatrix} k(\boldsymbol{x_*}, \boldsymbol{x_1}) & k(\boldsymbol{x_*}, \boldsymbol{x_2}) & \cdots & k(\boldsymbol{x_*}, \boldsymbol{x_n}) \end{bmatrix} \tag{C-10}$$

$$K_{**} = k(\boldsymbol{x_*}, \boldsymbol{x_*}) = \sigma_f^2 + \sigma_n^2 \tag{C-11}$$

### Definition of the hyperparameters

The outcome of our guess is strongly dependant on the form of the kernel chosen for the regression, as well as its hyperparameters $\boldsymbol{\theta}$. If the hyperparameters are chosen wrongly, the result will be incorrect. So how should the parameters (in this case $\boldsymbol{l}$, $\sigma_f$ and $\sigma_n$) be selected? To answer this question let us consider the fact that the maximum a posteriori estimate of $\boldsymbol{\theta}$ occurs when $p(\boldsymbol{\theta}|\bar{\boldsymbol{x}}, \bar{y})$ is maximized. To achieve so, it is necessary to maximize the marginal likelihood given by the equation below:

$$\log p(\bar{y}|\bar{\boldsymbol{x}}, \boldsymbol{\theta}) = -\frac{1}{2}\bar{y}^\top K^{-1}\bar{y} - \frac{1}{2}\log|K| - \frac{n}{2}\log 2\pi \tag{C-12}$$

where $|K|$ is the determinant of $K$. Provided an initial guess of the parameters, after optimization we get the values which maximize the marginal likelihood. In other words, the hyperparameters are fit according to the training data to obtain accurate predictions.

### Gaussian process derivative

Let us call the posterior mean $\mathbb{E}(y_*)$ as $g(\boldsymbol{x}_*)$ and its derivative as $g(\boldsymbol{x}_*)'$. The problem is now to define the derivative $g(\boldsymbol{x}_*)'$. In particular, there is a closed form for the first order derivative of the posterior mean. Remember that the posterior mean is given by:

$$g(\boldsymbol{x}_*) = \mathbb{E}(y_*) = K_* K^{-1}\bar{y} \triangleq K_*\boldsymbol{\alpha} \tag{C-13}$$

The only term in the expression above which is dependant on the test point $\boldsymbol{x}_*$ is $k(\boldsymbol{x}_*, \bar{\boldsymbol{x}})$. To calculate the derivative of the posterior with respect to $\boldsymbol{x}_*$, then, it is sufficient to differentiate the kernel. Since we chose the squared exponential covariance function, the derivative of the kernel with respect to $\boldsymbol{x}_*$ is:

$$g(\boldsymbol{x}_*)' = \frac{\partial k(\boldsymbol{x}_*, \bar{\boldsymbol{x}})}{\partial \boldsymbol{x}_*}\boldsymbol{\alpha} = \frac{\partial}{\partial \boldsymbol{x}_*}\left\{\sigma_f^2 \exp\left(-\tfrac{1}{2}(\boldsymbol{x}_* - \bar{\boldsymbol{x}})^\top \Lambda(\boldsymbol{x}_* - \bar{\boldsymbol{x}})\right) + \sigma_n^2 d_{ij}\right\}\boldsymbol{\alpha} \tag{C-14}$$

Taking advantage of the squared exponential kernel, the partial derivatives are defined by:

$$\begin{aligned} g(\boldsymbol{x}_*)' &= -\Lambda^{-1}(\boldsymbol{x}_* - \bar{\boldsymbol{x}})^\top \sigma_f^2 \exp\left(-\tfrac{1}{2}(\boldsymbol{x}_* - \bar{\boldsymbol{x}})^\top \Lambda(\boldsymbol{x}_* - \bar{\boldsymbol{x}})\right)\boldsymbol{\alpha} \\ &= -\Lambda^{-1}(\boldsymbol{x}_* - \bar{\boldsymbol{x}})^\top [k(\boldsymbol{x}_*, \bar{\boldsymbol{x}})^\top \cdot \boldsymbol{\alpha}] \end{aligned} \tag{C-15}$$

where $\cdot$ means element-wise multiplication.

# Bibliography


[1] K. J. Astrom, "Theory and applications of adaptive control - a survey," *Automatica Vol. 19, No. 5, pp. 471-486*, 1983.

[2] M. Blanke, M. K. J. Lunze, and M. Staroswiecki, *Diagnosis and Fault-Tolerant Control.* Springer, Berlin, 2006, vol. 2.

[3] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American Statistical Association 112*, 2017.

[4] R. Bogacz, "A tutorial on the free-energy framework for modelling perception and learning," *Journal of Mathematical Psychology*, 2017.

[5] C. L. Buckley, C. S. Kim, S. McGregor, and A. K. Seth, "The free energy principle for action and perception: A mathematical review," *Journal of Mathematical Psychology*, 2017.

[6] L. Capisani, A. Ferrara, and P. Pisu, "Sliding mode observers for vision-based fault detection, isolation and identification in robot manipulators," *American Control Conference, Marriott Waterfront, Baltimore, MD, USA*, 2010.

[7] J. Chen and R. J. Patton, *Robust model-based fault diagnosis for dynamic systems.* Springer Science & Business Media, LLC, 1999.

[8] T. M. Cover and J. A. Thomas, *Elements of Information Theory.* Wiley-Interscience, 2006, vol. 2.

[9] S. X. Ding, "Model-based fault diagnosis techniques. design schemes, algorithms, and tools," *Springer*, 2008.

[10] W. Dixon, I. Walker, D. Dawson, and J. Hartranft, "Fault detection for robot manipulators with parametric uncertainty: A prediction-error-based approach," *IEEE Trans. Robototics and Automomation, vol. 16, no. 6, pp. 689-699*, 2000.

[11] K. J. Frinston, "The free-energy principle: a unified brain theory?" *Nature Reviews Neuroscience*, 2010.

[12] K. J. Frinston, J. Mattout, N. Trujillo-Barreto, J. ashburner, and W. Penny, "Variaional free energy and laplace approximation," *NeuroImage 34*, 2006.

[13] K. J. Friston, "Learning and inference in the brain," *Neural Networks 16.9, pp. 1325-1352*, 2003.

[14] ——, "A theory of cortical responses," *Philosophical Transactions of the Royal Society of London B: Biological Sciences 360.1456 pp. 815-836*, 2005.

[15] ——, "Hierarchical models in the brain," *PLoS Comput Biol 4*, 2008.

[16] K. J. Friston, J. Daunizeau, and S. J. Kiebel, "Reinforcement learning or active inference?" *PloS ONE 4*, 2009.

[17] K. J. Friston, J. Daunizeau, J. Kilner, and S. J. Kiebel, "Action and behavior: a free-energy formulatione," *Biol Cybern 102*, 2010.

[18] S. Grimbergen, "Active inference for state space models: A tutorial," *(Unpublished MSc dissertation), TU Delft, The Netherlands*, 2019.

[19] R. Isermann, *Fault-diagnosis systems: an introduction from fault detection to fault tolerance*. Springer Science & Business Media, 2006.

[20] L. Jamone, B. Damas, and J. Santos-Victor, "Incremental learning of context-dependent dynamic internal models for robot control," *Proc. of the IEEE International Symposium on Intelligent Control (ISIC)*, 2014.

[21] D. Kappler, F. Meier, N. Ratliff, and S. Schaal, "A new data source for inverse dynamics learning," *Proc. of the IEEE/RSJ Conference on Intelligent Robots and Systems*, 2017.

[22] A. J. Koivo and T. M. Guo, "Adaptive linear controller for robotic manipulators," *IEEE Transaction Vol. AC-28, pp162-171*, 1983.

[23] J. Krüger, T. Lien, and A. Verl, "Cooperation of human and machines in assembly lines," *CIRP Annals - Manufacturing Technology*, 2009.

[24] P. Lanillos and G. Cheng, "Active inference with function learning for robot body perception," *IEEE Developmental Learning and Epigenetic Robotics (ICDL-Epirob)*, 2018.

[25] ——, "Adaptive robot body learning and estimation through predictive coding," *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2018.

[26] S. Lawrence, C. L. Giles, and A. C. Tsoi, "What size neural network gives optimal generalization? convergence properties of backpropagation," *Tech. Rep. UMIACS-TR-96-22 and CS-TR-3617, Institute for Advanced Computer Studies, Univ. of Maryland*, 1996.

[27] F. Ledezma and S. Haddadin, "First-order-principles-based constructive network topologies: An application to robot inverse dynamics," *IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, 2017.

[28] D. V. Lindley, *Bayesian statistics, a review.* SIAM, 1972, vol. 2.

[29] E. Malis, F. Chaumette, and S. Boudet, "2 1/2 d visual servoing," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 2, pp. 238–250, April 1999.

[30] M. Marshall and H. Lipkin, "Kalman filter visual servoing control law," *IEEE Proceedings of International Conference on Mechatronics and Automation*, 2014.

[31] M. Matteucci, "Elearnt: Evolutionary learning of rich neural network topologies," *Carnegie Mellon University, Tech. Rep.*, 2006.

[32] A. C. Mercadé, "Robot manipulator control under the active inference framework," *(Unpublished MSc dissertation), TU Delft, The Netherlands*, 2018.

[33] K. S. Narendra and J. Balakrishnan, "Adaptive control using multiple models," *IEEE Transaction on Automatic Control, Book reviews*, 1997.

[34] D. Nguyen-Tuong, J. R. Peters, and M. Seeger, "Local gaussian process regression for real time online model learning," *Proc of. Neural Information Processing Systems (NIPS), pp 1193-1200*, 2008.

[35] G. Oliver, P. Lanillos, and C. Gordon, "Active inference body perception and action for humanoid robots," *arXiv:1906.03022 [cs.RO]*, 2019.

[36] G. Paviglianti, F. Pierri, F. Caccavale, and M. Mattei, "Robust fault detection and isolation for proprioceptive sensors of robot manipulators," *Mechatronics, 20(1):162-70*, 2010.

[37] J. A. Piepmeier, G. V. McMurray, and H. Lipkin, "Uncalibrated dynamic visual servoing," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 1, pp. 143–147, Feb 2004.

[38] L. Pio-Lopez, A. Nizard, K. J. Friston, and G. Pezzullo, "Active inference and robot control: a cases study," *Journal of The Royal Society Interface*, 2016.

[39] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Ng, "Ros: An open-source robot operating system," *Proc. ICRA Workshop Open Source Software, pp. 1-6*, 2009.

[40] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning.* MIT Press, 2006.

[41] M. Rocha, P. Cortez, and J. Neves, "Simultaneous evolution of neural network topologies and weights for classification and regression," *International work-conference on artificial neural networks, Springer, pp. 59-66*, 2005.

[42] J. Shin and J. Lee, "Fault detection and robust fault recovery control for robot manipulators with actuator failures," *Proc. IEEE Int. Conf. Robotics and Automation, Detroit, MI, pp. 861-866*, 1999.

[43] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control.* Springer Science & Business Media, 2010.

[44] G. Tao, "Multivariable adaptive control: A survey," *Automatica*, 2014.

[45] M. Tarokh, "Hyperstability approach to the synthesis of adaptive controllers for robot manipulators," *Proceedings of 1991 IEEE international conference on robotics and automation*, 1991.

[46] M. Van, D. Wu, S. Ge, and H. Ren, "Robust fault detection and isolation for proprioceptive sensors of robot manipulators," *IEEE Transactions Industrial Electronics, vol. 12, no. 6, pp. 1998-2007*, 2016.

[47] G. Vass and T. Perlaki, "Applying and removing lens distortion in post production," *Proceedings of 2nd Hungarian Conference on Computer Graphics and Geometry*, 2003.

[48] S. Vijayakumar and S. Schaal, "Locally weighted projection regression: Incremental real time learning in high dimensional space," *Proc of. the International Conference on Machine Learning (ICML), pp. 1079-1086*, 2000.

[49] M. Visinsky, J. Cavallaro, and I. Walker, "Robotic fault detection and fault tolerance: A survey," *Rel. Eng. Syst. Safety, vol. 46, pp. 139-158*, 1994.

[50] R. Walters and M. M. Bayoumi, "Application of a self-tuning pole-placement regulator to an industrial manipulator," *21st IEEE Conference on Decision and Control, Orlando, FL, USA, pp. 323-329*, 1982.

[51] D. D. Zhang and B. Wei, "A review on model reference adaptive control of robotic manipulators," *Annual Reviews in Control 43, pp.188-198*, 2017.

[52] R. Zwanzig, *Nonequilibrium statistical mechanics*. Oxford University Press, 2001.