Identification et contrôle des systèmes non linéaires : application aux robots humanoïdes

Wael Suleiman

École Doctorale Systèmes

Thèse

en vue de l'obtention du

Doctorat de l'université de Toulouse
délivré par l'université Toulouse III - Paul Sabatier

Spécialité: Systèmes Automatiques

présentée et soutenue publiquement le 18 septembre 2008

Identification et contrôle des systèmes non linéaires : application aux robots humanoïdes

Wael SULEIMAN

Préparée au Laboratoire d'Analyse et d'Architecture des Systèmes
sous la direction de M. André MONIN et M. Jean-Paul LAUMOND

Jury

M. Gérard FAVIER	Rapporteur
M. Bruno SICILIANO	Rapporteur
M. Jean-Louis CALVET	Examinateur
M. Eiichi YOSHIDA	Examinateur
M. André MONIN	Directeur de Thèse
M. Jean-Paul LAUMOND	Co-directeur de Thèse

To My Family

Acknowledgments

I would like to express my gratitude to the individuals who helped to see me through the process and realization of this thesis.

I would firstly like to acknowledge my thesis advisors André Monin and Jean-Paul Laumond. With their enthusiasm, their inspiration, their insights and their good ideas, they helped to make the research problems and even the open problems of them curiously interesting to be investigated. I would like to thank them as well for teaching me signal processing, robotics and how to carry out a research work efficiently.

I would to express my gratitude to the successive directors of the LAAS - CNRS, Malik Ghallab and Raja Chatila for giving me the opportunity to conduct my research at LAAS.

I am grateful to Dr. Bruno Siciliano, Dr. Gérard Favier and Dr. Jean-Louis Calvet for their agreeing to review this dissertation and to be part of my thesis committee. Their valuable feedback helped me to improve the dissertation in many ways.

A special thank to Mme Jackie Som and Mme Brigitte Ducrocq the secretaries of MRS and Gepetto groups for their kindly help to facilitate the administrative procedures.

I wish to thank people with whom I had the chance to collaborate during my thesis: Eiichi Yoshida, Fumio Kanehiro, Florent Lamiraux, Oussama Kanoun, Gibran Etcheverry, ...

I am indebted to my friends for their friendship and for their help: Hassan, Christine, Rémon, Anne-Marie, Meike, Annie, Anis, Paulo, Claudia, Ignacio, Ayman, Abdellatif, Ali, Gustavo and all those who I might be forgetting.

Lastly, and most importantly, I want to express my gratitude to my family, especially to my parents, my brothers *Wassim* and *Mohanad*, and my sister *Ghina*, on whose constant encouragement and love I have relied throughout all the time. To them I dedicate this thesis.

Contents

Introduction

Le premier objectif des chercheurs actuellement est de pousser la performances des systèmes jusqu'à leur limites physiques. Afin d'atteindre cet objectif, on doit généralement répondre à deux questions. Ces questions sont:

1. Quel est le *modèle* du système étudié et si il n'est pas disponible comment peut on l'*identifier*?
2. Quelle est la *fonction de coût* à optimiser pour obtenir la performance désirée et quelle est la *méthode d'optimisation* adaptée qu'il faut appliquer?

Les réponses à ces deux questions impliquent plusieurs domaines de recherches et exigent des compétences pluridisciplinaires. Parmi ces domaines de recherche, nous trouvons l'identification des systèmes non-linéaires et l'optimisation des mouvements. Dans cette thèse nous abordons des problèmes qui sont liés à ces deux domaines de recherche. Plus précisément, nous considérons l'identification de plusieurs modèles classiques des systèmes non-linéaires et l'optimisation des mouvements des robots humanoïdes.

Bien que l'identification des modèles des systèmes non-linéaires abordés dans cette thèse soit beaucoup étudiée durant ces dernières années, nous visons à présenter des nouvelles reformulations de l'identification de ces modèles et à proposer des nouvelles méthodes pour résoudre les problèmes reformulés.

En outre, nous abordons le problème de l'optimisation des mouvements des robots humanoïdes et nous proposons des méthodes nouvelles et efficaces. Ces méthodes sont le résultat de l'application des méthodes de contrôle des systèmes non-linéaires et de la théorie de l'optimisation aux systèmes anthropomorphique ayant un grand nombre de degrés de libertés.

L'histoire de l'identification des systèmes trouve ses racines dans les travaux de Gauss (1809) et Fisher (1912). Cependant, la plupart des chercheurs prétendent que la théorie de l'identification moderne est apparue après l'année 1965 dans laquelle les deux articles séminaux, [Ho and Kalman 1965] et [Åström and Bohlin 1965], ont été publiés. Ces deux articles ont ouvert la voie au développement de deux techniques d'identification que sont: les méthode de sous-espace et les méthodes d'identification basées sur la prédiction de l'erreur de sortie.

Même si l'identification des systèmes dynamiques constitue un domaine de recherche qui ne cesse de prendre de l'importance, la théorie de l'optimisation et ses applications à la plupart des systèmes demeure en pratique un domaine de recherche très actif. À noter que la majorité des méthodes d'identification sont souvent basées sur l'optimisation d'un critère.

La théorie de l'optimisation utilise des méthodes et des techniques mathématiques qui permettent de trouver les valeurs minimales ou maximales d'une fonction coût sur un intervalle donné. L'importance de la théorie de l'optimisation et ses applications ne cesse d'augmenter, l'optimisation jouant un rôle essentiel dans beaucoup de domaines.

Historiquement, la méthode proposée par Lagrange pour trouver le valeur minimale d'une fonction sous contraintes égalité a été publiée en 1788 dans son célèbre livre *Mécanique Analytique*. Le cas des contraintes inégalité a été abordé par Fourrier [Fourier 1798], puis par Gauss [Gauss 1829].

Ce manuscrit est principalement composé de deux parties que sont:

La partie I (chapitres 2–6) aborde des problèmes liés à l'identification des systèmes linéaires et non-linéaires. Dans le chapitre 2 nous présentons un aperçu des méthodes de sous-espaces et un bref historique de leurs développements ces dernières années. L'identification des systèmes linéaires dans le cas des expérimentations multiples sera abordée dans le chapitre 3. La méthode proposée est basée sur une classe de méthodes de sous-espaces. Une nouvelle méthode pour identifier une représentation dans l'espace d'état des séries de Volterra d'ordre fini et à horizon infini sera proposée dans le chapitre 4. L'identification des systèmes quadratiques en l'état fera l'objet du chapitre 5. Enfin, l'application des méthodes d'identifications des systèmes dynamiques, en vue de synthétiser et de modéliser la locomotion humaine en utilisant des données de capture de mouvements, sera discutée dans le chapitre 6.

La partie II (chapitres 7–9) aborde le contrôle des robots humanoïdes. Dans le chapitre 7 nous proposons une méthode efficace qui a pour but d'optimiser les mouvements des robots humanoïdes. L'imitation des mouvements humains par un robot humanoïde sera abordée dans le chapitre 8. Nous montrons que ce problème peut être formulé comme un problème d'optimisation pour lequel nous proposons une méthode robuste et efficace. Cette méthode est capable de prendre en compte les limites physiques du robot humanoïde en tant que contraintes à respecter. Le paramétrage temporel des chemins dans l'espace de configuration pour un robot humanoïde fera l'objet du chapitre 9. Dans ce chapitre nous formalisons le problème du paramétrage temporel comme un problème d'optimisation et nous proposons une méthode basée sur l'approximation par différences finies pour le résoudre. Les méthodes proposées dans cette partie ont été validées à travers des expérimentations sur la plate-forme du robot humanoïde HRP-2 №14.

Les parties I et II sont écrites en telle façon que chaque partie pourrait être lue indépendamment de l'autre.

Chaque chapitre sera précédé par un résumé en français de deux pages.

1

Introduction

The demanding of everyday to push the systems to the limits of their performance is permanently increasing. Pursuing perfection of systems' performance mainly leads to answer two questions in whatever field it is sought. These questions are:

1. What is the *model* of the studied system and if it isn't available how can I *identify* it?
2. What is the *cost function* which should be optimized to obtain the desired performance and what is the adapted *optimization method* which should be employed?

Trying to answer the above two questions draws upon numerous research fields and requires pluridisciplinary skills. Foremost among these research fields are nonlinear system identification and motion optimization. These two fields shall be the subject of investigation in this thesis. We will mainly focus on the identification of some classical nonlinear models and the optimization of humanoid robot motions.

Although the identification of the considered nonlinear models in this thesis have been extremely studied in the literature of system identification, we aim to give new insights into the reformulation of the identification of these models and to propose efficient methods to solve the reformulated problems.

Moreover, we address the problem of humanoid robot motion optimization for which we propose new and efficient methods. These methods are the result of the comprehensive application of nonlinear system control and optimization theory to anthropomorphic systems with large number of degrees of freedom.

First of all, a reliable and accurate model of the system under consideration is crucial in many research areas. These areas include not only engineering applications, but also biology, socio-economics, and ecology, just to mention only a few. The purposes of such a model are many and they mainly depend on the application. For example, in engineering this model can be employed to build a feedback controller for the closed loop. In socio-economics, the model has the main purpose of prediction and thereby the prevention.

Dynamic models describing the system of interest can be constructed using the first principles of physics, biology, ecology, and so forth. This procedure is usually difficult and it requires specialist knowledge. The obtained models are often complex, and need a solid knowledge of the dynamic behavior of the system. However, this procedure is sometimes indispensable and we cannot get away from it. It is the case of some complex systems such as the humanoid robots.

On the other hand, applying the first principles on poorly understood systems leads on to poor and inaccurate models and sometimes is even impossible. Moreover, it is possible that some physical parameters of the system are not given on account of industrial confidentiality. In these cases, an alternative way to get along is system identification methods. System identification theory aims to estimate dynamic models directly from the measured input and output data. In real fact, the connection between the system identification methods and the first principles has a wide range of solidity and colors. It starts from the black-box approach, where no knowledge of the dynamic behavior is supposed, and go through a variety of gray-box approaches, where some knowledge of the dynamic behavior of the system can be derived.

Even though some early work on system identification is relied on the statistical theory of parameter estimation which has its roots in the work of Gauss (1809) and Fisher (1912), many authors claim that 1965 is the birth-year of the modern identification theory. The publication of two seminal papers, [Ho and Kalman 1965] and [Åström and Bohlin 1965], paved the way for the development of the two mainstream identification techniques that still dominate the field today: subspace identification and prediction error identification.

Although system identification is one of the attractive research fields so far, the optimization theory and its applications to most real-life systems and procedures is, indeed, another active research field. Furthermore, many of system identification methods are based in some way on the optimization of specified criteria.

The optimization theory is the mathematical study of problems which ask for minimal or maximal values of an objective function on a given domain. The importance of optimization theory and its applications is continually growing. This is due to the large variety of fields where the optimization comes into play.

Historically, the method of Lagrange for finding minimum value of functions subject to equality constraints was published in 1788 in his famous book *Mécanique Analytique*. The case of inequality constraints was first investigated in 1798 by Fourier [Fourier 1798], then by Gauss [Gauss 1829].

The reminder of this chapter is organized as follows. A brief outline of the thesis is provided in Section 1.1. Section 1.2 contains a summary of the main contributions in the thesis. Some rules for the general notions used in this thesis are given in Section 1.3. Finally, a list of the publications associated to the research work in this thesis is provided in Section 1.4.

1.1 Outline of the Thesis

This thesis mainly consists of two parts:

Part I (Chapters 2–6) deals with the identification of linear and nonlinear systems. It provides an overview of the subspace identification methods and a brief history of their origin in Chapter 2. Chapter 3 describes a method to identify linear systems using multiple short data sets, this method is based on a class of subspace methods. A novel method to identify a finite degree Volterra series is described in Chapter 4. The identification of quadratic in-the-state systems is considered in Chapter 5. Finally, the application of system identification methods to synthesize and model human locomotion using human captured data is explained in Chapter 6.

Part II (Chapters 7–9) deals with the control of humanoid robots. A method for the optimization of humanoid robot motions is described in Chapter 7. In Chapter 8, the imitation of human captured motion by humanoid robot is formulated and a method is proposed to solve it efficiently. The time parameterization of humanoid robot paths is discussed in Chapter 9, and a numerical method to solve this problem is proposed.

In fact, Part I and Part II have been written in such a way that each part can be read independently from the other.

Each chapter will be preceded by a summary of two pages in French.

1.2 Contributions

The main contributions of this thesis are:

- A state-space realization of finite degree Volterra series with infinite horizon is described, and a new method to identify this state-space realization is proposed (Chapter 4). The initial estimation of the realization's parameters is obtained by a sequential projection method that we have developed thanks to the recursive property of the realization structure. Then the realization parameters are optimized using a local gradient search method.
- A method to resolve the problem of synthesizing and modeling human locomotion is provided (Chapter 6). This method is the result of the comprehensive application of the identification theory and its applications to dynamic system.

- An optimization framework for humanoid robot motions is developed (Chapter 7). This framework takes as input a pre-calculated motion, which are provided by motion planning techniques, and the output is an optimized and stable motion. This method has been validated on the humanoid robot HRP2.
- An optimization framework to generate the upper body motion of humanoid robot from human captured motions is described (Chapter 8). The generated motions imitate the original human captured motion and at the same time they respect the physical limits of humanoid robot. This method has been as well validated on the humanoid robot HRP2.
- A numerical method to solve the time parameterization problem of humanoid robot paths is proposed (Chapter 9). This method has the objective of transforming a statically stable path to a minimum time and dynamically stable trajectory which respects the velocity limits of the humanoid robot's joints. This method has been validated through several experiments using the humanoid robot HRP2.

1.3 General Notations

As a general rule in this thesis, selecting elements of matrices is done using MATLAB™ standard matrix operations, e.g. $M(:, i:j)$ stands for the sub-matrix of the matrix M which contains the columns from the i^{th} to j^{th} columns, $M(:, i)$ designs a vector which coincides with the i^{th}column of M, and $V(i)$ denotes the i^{th} element of the vector V.

1.4 List of Publications

The articles associated to this research work are:

1. **W. Suleiman** and A. Monin. New Method for Identifying Finite Degree Volterra Series. *Automatica, Vol. 44,* №2, pp.488-497, February 2008.

2. F. Kanehiro, **W. Suleiman**, F. Lamiraux, E. Yoshida and J-P. Laumond. Integrating Dynamics into Motion Planning for Humanoid Robots. *IEEE/RSJ International Conference on Intelligent RObots and Systems (IROS 2008)*, Nice, France, September 2008.

3. **W. Suleiman**, E. Yoshida, F. Kanehiro, J-P. Laumond and A. Monin. On Human Motion Imitation by Humanoid Robot. *IEEE International Conference on Robotics and Automation*, Pasadena, California (USA), on 19-23 May, 2008.

4. **W. Suleiman**, E. Yoshida, J-P. Laumond and A. Monin. On Humanoid Motion Optimization. *IEEE-RAS 7th International Conference on Humanoid Robots*, Pittsburgh, Pennsylvania (USA), on 29 November - 1 December, 2007.

5. **W. Suleiman** and A. Monin. Identification of Quadratic in-the-State System Using Nonlinear Programming. *The 15th Mediterranean Conference on Control and Automation (MED07)*, Athens (Greece), on 27-29 Jun 2007.

6. **W. Suleiman** and A. Monin. Linear Multivariable System Identification: Multi-experiments Case. *Conference on Systems and Control (CSC2007)*, Marrakech (Morocco), on 16-18 May 2007.

7. **W. Suleiman** , A. Monin and J-P. Laumond. Synthesizing and Modeling Human Locomotion Using System Identification. *IEEE International Conference on Intelligent Robots and Systems (IROS2006)*, Beijing (China), pp. 1972-1977, October 2006.

8. **W. Suleiman** and A. Monin. Identification of Quadratic System by Local Gradient Search. *IEEE International Conference on Control Applications*, Munich (Germany), pp.2565-2570, October 2006,

9. **W. Suleiman**, G. Etcheverry and A. Monin. Nouvelle approche pour l'identification des systèmes non-linéaires. *Conférence Internationale Francophone d'Automatique (CIFA2006)*, Bordeaux (France), on 30 May - 1 Jun 2006.

Part I

System Identification

Résumé du chapitre 2
Méthodes d'identification de sous-espaces

L'identification des systèmes dynamiques, notamment l'identification des systèmes à entrées-multiples/sorties-multiples, est un domaine de recherche qui ne cesse de prendre de l'importance. Le problème d'identification intervient dans plusieurs domaines de recherche et dans beaucoup d'applications en pratique. Citons, par exemple, la simulation des systèmes chimiques ou bio-chimiques et l'identification des modes de vibration pour des systèmes flexibles.

Les premièrese tentatives pour identifier les systèmes à entrées-multiples/sorties-multiples ont été basées sur l'adaptation des méthodes classiques d'identification tel que les méthodes de prédiction d'erreur et les méthodes des variables instrumentales. Les propriétés statistiques de ces méthodes et leur relation étroite avec l'estimateur de minimum de vraisemblance ont rendu ces méthodes très populaires. En outre, ces méthodes ont été appliquées en pratique pour identifier plusieurs systèmes réels.

Cependant, dans le cas des systèmes à entrées-multiples/sorties-multiples, les méthodes classiques présentent divers défauts [Viberg 1995]. Cela est lié au modèle d'équation différentielle utilisé dans les méthodes classiques.

Afin de dépasser ces limites, l'utilisation du modèle d'espace d'état a été proposé [Viberg 1995]. Ce type de modèle se révèle particulièrement utile et robuste pour les systèmes complexes.

Un modèle linéaire d'espace d'état peut être défini comme suit

$$x_t = Ax_{t-1} + Bu_t$$
$$y_t = Cx_t + v_t$$

où $x_t \in \mathbb{R}^n, u_t \in \mathbb{R}^m, y_t \in \mathbb{R}^p$ et $v_t \in \mathbb{R}^p$ sont respectivement l'état interne du système, le signal d'entrée, le signal de sortie et le bruit des mesures. La dimension d'état interne du système n est nommé le degré du système.

Dans ces dernières décennies, l'identification de modèles linéaires d'espace d'état est devenu un thème de recherche très active. De nouvelles méthodes sont également apparues; il s'agit des méthodes de sous-espaces.

Dans le chapitre 2, nous présenterons plusieurs méthodes de sous-espaces et nous donnerons

un bref historique du développement des ces méthodes.

Dans le paragraphe suivant, nous présenteront brièvement la méthode PO (Past Output)-MOESP (Multivariable Output-Error State-sPace) [Viberg 1995; Verhaegen 1994]. Cette méthode de sous-espaces sera principalement utilisée dans la suite du manuscrit.

Méthode PO-MOESP

La méthode consiste à stocker dans un premier temps les entrées et les sorties du système linéaire sous la forme matricielle d' Hankel:

$$\mathbf{U}^{\mathbf{N}}_{\mathbf{1},\alpha} \triangleq \begin{pmatrix} u_1 & u_2 & \cdots & u_{N-\alpha+1} \\ u_2 & u_3 & \cdots & u_{N-\alpha+2} \\ \vdots & \vdots & \ddots & \vdots \\ u_\alpha & u_{\alpha+1} & \cdots & u_N \end{pmatrix}$$

où $(1, \alpha)$ représentent respectivement l'indice du premier échantillon et le nombre de lignes de la matrice. N désigne l'indice du dernier échantillon. Le nombre de lignes α doit être choisi tel qu'il soit supérieur à la dimension du système n [Viberg 1995; Verhaegen 1994]. Les matrices de sortie $\mathbf{Y}^{\mathbf{N}}_{\mathbf{1},\alpha}$ et de bruit $\mathbf{V}^{\mathbf{N}}_{\mathbf{1},\alpha}$ sont définies du même manière. L'équation entrée/sortie suivante est alors facilement déduite de la description du système:

$$\mathbf{Y}^{\mathbf{N}}_{\mathbf{1},\alpha} = \mathbf{\Gamma}_\alpha \mathbf{X}_{\mathbf{1},\mathbf{N}-\alpha+\mathbf{1}} + \mathbf{\Phi}_\alpha \mathbf{U}^{\mathbf{N}}_{\mathbf{1},\alpha} + \mathbf{V}^{\mathbf{N}}_{\mathbf{1},\alpha}$$

où $\mathbf{\Gamma}^{\mathbf{1}}_{\alpha}$ est la matrice étendue d'observabilité du système. $\mathbf{\Phi}_\alpha$ est une matrice triangulaire par blocs. Celles-ci sont définies par :

$$\mathbf{\Gamma}_\alpha = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{(\alpha-1)} \end{bmatrix}$$

$$\mathbf{\Phi}^{\mathbf{1}}_{\alpha} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ CB & 0 & 0 & \cdots & 0 \\ CAB & CB & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ CA^{(\alpha-2)}B & \cdots & \cdots & CB & 0 \end{bmatrix}$$

$\mathbf{X}_{\mathbf{1},\mathbf{N}-\alpha+\mathbf{1}} = \begin{bmatrix} x_1 & x_2 & \cdots x_{N-\alpha+1} \end{bmatrix}$.
Le principe de cette méthode est d'utiliser les mesures passées comme variables instrumentales, ceci afin de minimiser les effets du bruit.

Considérons la factorisation RQ suivante :

$$\begin{bmatrix} \mathbf{U}^{\mathbf{N}}_{\mathbf{1}+\alpha,\alpha} \\ \mathbf{U}^{\mathbf{N}-\alpha}_{\mathbf{1},\alpha} \\ \mathbf{Y}^{\mathbf{N}-\alpha}_{\mathbf{1},\alpha} \\ \mathbf{Y}^{\mathbf{N}}_{\mathbf{1}+\alpha,\alpha} \end{bmatrix} = \begin{bmatrix} R_{11} & 0 & 0 & 0 \\ R_{21} & R_{22} & 0 & 0 \\ R_{31} & R_{32} & R_{33} & 0 \\ R_{41} & R_{42} & R_{43} & R_{44} \end{bmatrix} Q^T$$

les matrices R_{ii} sont des matrices triangulaires-inférieures et la matrice-colonne unitaire Q est divisée conformément aux dimensions des matrices R_{ii} selon :

$$Q = \begin{bmatrix} Q_1 \ Q_2 \ Q_3 \ Q_4 \end{bmatrix}$$

[Overschee and Moor 1994] ont proposé de considérer la quantité suivante :

$$\begin{aligned} \begin{bmatrix} R_{42} \ R_{43} \end{bmatrix} &= \mathbf{Y}^{\mathbf{N}}_{\mathbf{1}+\alpha,\alpha} \begin{bmatrix} Q_2 \ Q_3 \end{bmatrix} \\ &= \left(\mathbf{\Gamma}^{\mathbf{1}}_{\alpha} \mathbf{Z}^{\mathbf{1}}_{\alpha+\mathbf{1},\mathbf{N}-\alpha+\mathbf{1}} + \mathbf{V}^{\mathbf{N}}_{\mathbf{1}+\alpha,\alpha} \right) \begin{bmatrix} Q_2 \ Q_3 \end{bmatrix} \end{aligned}$$

Il est alors aisé de vérifier que :

$$\lim_{N\to\infty} \frac{1}{N} \mathbf{V}^{\mathbf{N}}_{\mathbf{1}+\alpha,\alpha} \begin{bmatrix} Q_2 \ Q_3 \end{bmatrix} = \mathbf{0}$$

et que les effets du bruit assymptotiquement disparaîtront (en pratique, ils sont diminués), $\mathbf{\Gamma}_\alpha$ est alors extraite de la matrice $\begin{bmatrix} R_{42} \ R_{43} \end{bmatrix}$ grâce à la décomposition en valeurs singulières:

$$\begin{bmatrix} R_{42} \ R_{43} \end{bmatrix} = U_n S_n {V_n}^T + {U_n}^{\perp} S_2 {V_n^{\perp}}^T \tag{1.1}$$

où la matrice S_n contient les valeurs singulières principales (supérieures à un seuil). Puis nous estimons la matrice $\mathbf{\Gamma}_\alpha$:

$$\mathbf{\Gamma}_\alpha = U_n S_n^{1/2}$$

Nous déduisons les matrices C et A directement de $\mathbf{\Gamma}_\alpha$ comme suit:

$$\begin{aligned} C =& \mathbf{\Gamma}_\alpha(1:p,:) \\ \mathbf{\Gamma}_\alpha(1:(\alpha-1)p,:)A =& \mathbf{\Gamma}_\alpha(p+1:p\alpha,:) \end{aligned}$$

Rappelons que p est la dimension de y_t.

Afin d'estimer les matrices B et D, [Verhaegen 1994] montre que la solution de moindres carrés de l'équation surdéterminée suivante:

$$\left({U_n}^{\perp}\right)^T \begin{bmatrix} R_{31} \ R_{32} \ R_{41} \end{bmatrix} = \left({U_n}^{\perp}\right)^T \mathbf{\Phi}_\alpha \begin{bmatrix} R_{21} \ R_{22} \ R_{11} \end{bmatrix}$$

conduit à une estimation de la matrice $\mathbf{\Phi}_\alpha$, ce qui permet d'estimer les matrices B and D aisément.

"Il y a des mirages de la clarté."

Yves Bonnefoy

2

Subspace Identification Methods

The problem of identifying Multi-Input Multi-Output (MIMO) systems is one of the most active subject in dynamic systems identification. This problem arises in a variety of applications, ranging from chemical process simulations and control to identifications of vibrational modes in flexible structures.

The necessity of developing methods to deal with the identification of MIMO systems was a real challenge for the researchers. The first attempt was to adapt the traditional system identification techniques, which are the prediction error method (PEM) and the instrumental variable method (IVM). These methods has been applied to identify many real-world problems and found to perform well thanks to their well known statistical properties and their relations to maximum-likelihood estimation.

However, in the case of MIMO systems, they have some shortcoming. This is because the candidate model structure of PEM and IVM methods is primarily linear difference equation. This model relates the outputs to the inputs signals by the following formula

$$\sum_{i=0}^{N_y} \alpha_i y_{t-i} = \sum_{j=0}^{N_u} \beta_j u_{t-j} + \sum_{k=0}^{N_e} \gamma_k e_{t-k} \tag{2.1}$$

where y_t is the output signal, u_t is the input signal and e_t is the measurement noise which is supposed to be white noise and independent of the input signals. This model is really interesting in the case of Single Input-Single Output (SISO) systems where the coefficients $\{\alpha_i, \beta_j, \gamma_k : i = 0, 1, 2, \cdots, N_y; j = 0, 1, 2, \cdots, N_u; k = 0, 1, 2, \cdots, N_e\}$ are scalar.

On the other hand, using such models is quite cumbersome in the general case of MIMO systems, and the numerical reliability may be poor for large system order. This is obvious on account of the effect that the coefficients, in this case, become matrices and without any assumption on the structure of these matrices, they are full parameterized.

To overcome the limitations of difference equation model, the use of state-space model is preferred typically for complex problems [Viberg 1995].

The remainder of this chapter is organized as follows. Section 2.1 gives an overview of state space realization and the notions of controllability, observability and similarity are introduced. The origin and various classes of subspace methods are given in Section 2.2.

2.1 State Space Realization

The state-space realization of a linear time-invariant (LTI) system, in the absence of noise, is the following

$$\begin{aligned} x_t &= Ax_{t-1} + Bu_t \\ y_t &= Cx_t \end{aligned} \tag{2.2}$$

where x_t is the internal state of the system and its dimension is equal to n (the system order), $y_t \in \mathbb{R}^p$ is the output signal , $u_t \in \mathbb{R}^m$ is the input signal. The matrices $\{A, B, C\}$ are constant matrices.

The direct relation between the input and the output of the system is given by the following formula

$$y_t = \sum_{k=0}^{\infty} h_k u_{t-k} \tag{2.3}$$

where

$$h_k = CA^kB \tag{2.4}$$

h_k denotes the $p \times m$ matrix of impulse responses or the Markov parameters.

2.1.1 Controllability, Observability and Minimality

A state-space realization is observable if and only if

$$Rank(\Gamma_n) = n \tag{2.5}$$

where

$$\Gamma_n = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix} \tag{2.6}$$

and *Rank* stands for the number of independent rows or columns in a matrix.
A state-space realization is controllable if and only if

$$Rank(\Omega_n) = n \tag{2.7}$$

where

$$\Omega_n = \begin{bmatrix} B & AB & \cdots & A^{n-1}B \end{bmatrix} \tag{2.8}$$

We call a state-space realization minimal if there exists no other realization of lower degree. As a consequence, the minimal realization are always both observable and controllable.

2.1.2 Stability

The system represented in state space realization is stable if and only if

$$\rho(A) < 1 \tag{2.9}$$

where $\rho(A)$ designs the spectrum radius of the matrix A defined as follows

$$\rho(A) = \max_{i=1,2,\cdots,n} |\lambda_i| \ : \lambda_i \text{ is an eigenvalue of matrix } A \tag{2.10}$$

2.1.3 Similarity and Canonical Forms

Let $T \in \mathbb{R}^{n \times n}$ be a nonsingular matrix. If the internal state x_t is transformed as follows

$$z_t = (T)^{-1} x_t \tag{2.11}$$

It is then easy to see that

$$\begin{aligned} z_t &= \bar{A} z_{t-1} + \bar{B} u_t \\ y_t &= \bar{C} z_t \end{aligned} \tag{2.12}$$

where

$$\begin{bmatrix} \bar{A} & \bar{B} \\ \bar{C} & 0 \end{bmatrix} = \begin{bmatrix} (T)^{-1} AT & (T)^{-1} B \\ CT & 0 \end{bmatrix} \tag{2.13}$$

It is clear that the transformation (2.11) provides an alternative state-space realization. This is because the input/output relation is still invariant, and of course the transfer function also. As

can easily be checked

$$H(z) = C(zI - A)^{-1}B \quad = \bar{C}\left(zI - \bar{A}\right)^{-1}\bar{B} \tag{2.14}$$

In point of fact, the state-space realization can be made unique by choosing an appropriate parameterization. For LTI systems, various canonical forms have been proposed such as

1. Observable canonical form: In this from the matrix A has the following structure

$$A = \begin{bmatrix} -\alpha_1 & 1 & 0 & \cdots & 0 \\ -\alpha_2 & 0 & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ -\alpha_{n-1} & 0 & \cdots & \cdots & 1 \\ -\alpha_n & 0 & 0 & \cdots & 0 \end{bmatrix} \tag{2.15}$$

   and the matrices B and C are full parameterized.
   SISO system case: In this case, the matrices B and C have the following forms

$$\begin{aligned} B^T &= \begin{bmatrix} b_1 & b_2 & \cdots & b_n \end{bmatrix} \\ C &= \begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix} \end{aligned} \tag{2.16}$$

2. Controllable form: In this from the matrix A has the following structure

$$A = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \cdots & 1 \\ -\alpha_1 & -\alpha_2 & -\alpha_3 & \cdots & -\alpha_n \end{bmatrix} \tag{2.17}$$

   and the matrices B and C are full parameterized.
   SISO system case: In this case, the matrices B and C have the following forms

$$\begin{aligned} B^T &= \begin{bmatrix} 0 & \cdots & 0 & 1 \end{bmatrix} \\ C &= \begin{bmatrix} c_1 & c_2 & \cdots & c_n \end{bmatrix} \end{aligned} \tag{2.18}$$

Note that, the popular form in system identification is the observable form. This is because the system is supposed to be observable by definition.

2.2 Subspace Methods: Origin and History

Subspace identification methods are attractive methods for the estimation of multivariable state-space systems from input-output data. The development of these methods, during the last

decade, has offered a reliable and efficient tools for the identification of linear systems.

Many authors suggest that most subspace based state space system identification (4SID) methods have a great deal in common with Ho and Kalman's realization algorithm [Ho and Kalman 1965]. This algorithm is based first on the calculation of the so-called Hankel matrix of dimension $(n+1)\times(n+1)$ of the system defined by

$$H = \begin{bmatrix} h_0 & h_1 & h_2 & \cdots & h_n \\ h_1 & h_2 & h_3 & \cdots & h_{n+1} \\ h_2 & h_3 & h_4 & \cdots & h_{n+2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h_n & h_{n+1} & h_{n+2} & \cdots & h_{2n} \end{bmatrix} \tag{2.19}$$

where h_i is the impulse response calculated using (2.4).

Second, it is easy to verify that the matrix H can be factored as

$$H = \Gamma_{n+1}\Omega_{n+1} \tag{2.20}$$

where Γ_{n+1} and Ω_{n+1} are the extended observability and controllability matrices defined similarly to (2.5) and (2.7) respectively.
In the case of minimal realization, the matrices Γ_{n+1} and Ω_{n+1} have full rank, and hence the matrix H has full rank which is equal to the system order.

As the state-space realization is not unique, any given full-rank factorization of the Hankel matrix H of the following form

$$H = \bar{\Gamma}_{n+1}\bar{\Omega}_{n+1} \tag{2.21}$$

provides an estimation of the extended observability matrix $\bar{\Gamma}_{n+1}$ and the extended controllability matrix $\bar{\Omega}_{n+1}$ for some state-space realization. This is the key point of Ho and Kalman's algorithm.

After the factorization, the extraction of the matrices B and C is directly from the matrices $\bar{\Gamma}_{n+1}$ and $\bar{\Omega}_{n+1}$ as follows

$$\begin{aligned} B &= \bar{\Omega}_{n+1}(:, 1:m) \\ C &= \bar{\Gamma}_{n+1}(1:p, :) \end{aligned} \tag{2.22}$$

The matrix A can be computed by using the following property

$$\bar{\Gamma}_{n+1}(p+1:(n+1)p, :) = \bar{\Gamma}_{n+1}(1:np, :)\, A \tag{2.23}$$

As the system is observable, by definition, the matrix $\bar{\Gamma}_{n+1}(1:np, :)$ has full rank. As a consequence the matrix A can be computed as follows

$$A = \left[\bar{\Gamma}_{n+1}(1:np, :)\right]^{\dagger} \bar{\Gamma}_{n+1}(p+1:(n+1)p, :) \tag{2.24}$$

where $[.]^{\dagger}$ designs the Moore-Penrose pseudo inverse defined as follows

$$X^{\dagger} = \left(X^T X\right)^{-1} X^T \tag{2.25}$$

At this point, the system matrices are effectively computed and the identification algorithm is finished.

This algorithm seems to be attractive, but for many years it does not hold the attention of the researchers and identification community. One could refer that to the difficulty of the calculation of the impulses responses in real practice and the method does not take the presence of noise into account. However, the modern subspace methods are somewhat inspired from Ho and Kalman's algorithm.

2.2.1 Realization Based 4SID Methods

The main challenge of using 4SID methods was the calculation of the Hankel matrix and reducing the effect of noise on the estimation of the matrices A, B and C. To overcome this difficulty, a direct method has proposed to apply impulse input signals in the different input channels, and simply measure the output in each output [Liu and Skelton 1992].

In order to reduce the effect of noise, the experiment may be repeated and the average of the output signals calculated. It is clear that this method is time consuming. Therefore, to reduce the consumed time one could use the correlation analysis.

Suppose that the output of system is lumped into zero-mean additive noise v_t

$$y_t = \sum_{k=0}^{\infty} h_k u_{t-k} + v_t \tag{2.26}$$

where v_t is a stationary, ergodic white noise with zero mean

$$E\left[v_t v_\tau^T\right] = R_v \delta_{t,\tau} \tag{2.27}$$

where R_v is the positive-definite covariance matrix, and $\delta_{t,\tau}$ is the Kronecker delta defined as follows

$$\delta_{t,\tau} = \begin{cases} 1 & if\ t = \tau \\ 0 & otherwise \end{cases} \tag{2.28}$$

We suppose that v_t is independent of the input signal u_t, that means

$$E\left[u_t v_\tau\right] = 0 \quad \forall t, \tau \tag{2.29}$$

For the moment, suppose that the input signal is a stationary white noise, so that

$$E\left[u_t u_\tau\right] = r_{uu} \delta_{t,\tau} \tag{2.30}$$

Provided u_t and u_τ are uncorrelated for all t and τ and the system is stable, we get

$$r_{yu}(\tau) = E\left[y_t u_{t-\tau}^T\right] = h_\tau r_{uu} \tag{2.31}$$

Hence, a finite number of impulse responses parameters can be consistently estimated from (2.31) [Söderström and Stoica 1989; Ljung 1999]. In the general case, the input signal is not white noise. However, by estimating the covariance matrix of the input signal, a prefilter can then be applied to both u_t and y_t, rendering the filtered input signal stationary white noise.

After the estimation of the impulse responses, the following question raises: How determine the system order (n) if the measured output is noised?
Theoretically, the rank of Hankel matrix is equal to n, but in the case of measurement noise this matrix will be full rank.

[Kung 1978] and [Juang and Pappa 1985] have proposed to build the Hankel matrix by choosing the number of rows roughly rather than the expected system order, then they reduce the rank of the Hankel matrix by using Singular Value Decomposition (SVD).

Let $\hat{H}$ be an estimate of the Hankel matrix, and its SVD is given by

$$\hat{H} = \hat{U}\hat{S}\hat{V}^T \tag{2.32}$$

where $\hat{U}$ and $\hat{V}$ are orthogonal matrices, and $\hat{S}$ is diagonal matrix with the singular values in non-increasing order on the diagonal. In the presence of measurement noise all singular values will be positive. In order to estimate the system order a threshold should be chosen. The singular values rather than this threshold are considered as the principal ones and the number of these values designs the system order.

Assuming that the matrices $\hat{U}$ and $\hat{V}$ are partitioned as follows

$$\hat{H} = \hat{U}_n\hat{S}_n\hat{V}_n^T + \hat{U}_n^\perp\hat{S}_2\left(\hat{V}_n^T\right)^\perp \tag{2.33}$$

where $\hat{S}_n$ contains the principals singular values. The extended observability and controllability matrices can be then calculated as follows

$$\begin{aligned}\hat{\Gamma}_n &= \hat{U}_n\hat{S}_n^{\frac{1}{2}} \\ \hat{\Omega}_n &= \hat{S}_n^{\frac{1}{2}}\hat{V}_n^T\end{aligned} \tag{2.34}$$

After that, to extract the matrices A, B and C the algorithm continues analogously to Ho and Kalman's algorithm.

2.2.2 Direct 4SID Methods

The direct 4SID methods are based on the same idea of first estimating the matrices Γ_n and Ω_n, then compute the matrices A, B and C. The main difference with the previous methods is that the input/output data are used directly without going through the calculation of Hankel matrix.

From the input-output relation of linear system, the following equation [Gopinath 1969; DeMoor et al. 1988] can be easily obtained

$$Y_{\alpha,N} = \Gamma_\alpha X_{1,N-\alpha+1} + \Phi_\alpha U_{\alpha,N} \tag{2.35}$$

where the input and output are stocked in the following form

$$U_{\alpha,N} \triangleq \begin{pmatrix} u_1 & u_2 & \cdots & u_{N-\alpha+1} \\ u_2 & u_3 & \cdots & u_{N-\alpha+2} \\ \vdots & \vdots & \ddots & \vdots \\ u_\alpha & u_{\alpha+1} & \cdots & u_N \end{pmatrix}$$

(α, N) refer to the number of rows in the matrix and the data length respectively. The output matrix $Y_{\alpha,N}$ is defined analogously to $U_{\alpha,N}$.
The matrices Γ_α, Φ_α and $X_{1,N-\alpha}$ are defined as follows

$$\begin{aligned} \Gamma_\alpha &= \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{(\alpha-1)} \end{bmatrix}, \Phi_\alpha = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ CB & 0 & 0 & \cdots & 0 \\ CAB & CB & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ CA^{(\alpha-2)}B & \cdots & \cdots & CB & 0 \end{bmatrix} \\ X_{1,N-\alpha+1} &= \begin{bmatrix} x_1 & x_2 & \cdots x_{N-\alpha+1} \end{bmatrix}. \end{aligned} \tag{2.36}$$

Note that the number of rows (α) should be roughly chosen to be greater than the expected linear system order (n) [Viberg 1995]. This condition guarantees that the extended matrix of observability Γ_α has a full rank (n).

Our objective is to estimate the matrices Γ_α and Φ_α. As the known quantities are the matrices $U_{\alpha,N}$ and $Y_{\alpha,N}$, and if we suppose that the matrix Φ_α is known then one could simply subtract the $\Phi_\alpha U_{\alpha,N}$ from $Y_{\alpha,N}$ to obtain an estimation of $\Gamma_\alpha X_{1,N-\alpha+1}$. Once an estimation of $\Gamma_\alpha X_{1,N-\alpha+1}$ is available one could estimate the matrix Γ_α by using SVD.

As Φ_α is unknown, we start by compute an estimation of it. This estimation problem can be formulated as follows

$$\min_{\Phi_\alpha} \quad \| Y_{\alpha,N} - \Phi_\alpha U_{\alpha,N} \|_F^2 \tag{2.37}$$

where $\| \, . \, \|_F$ denotes Frobenius norm, which is defined for a matrix $M \in \mathbb{R}^{m \times n}$ by

$$\| M \|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} |M(i,j)|^2} \tag{2.38}$$

Solving the above equation leads to

$$Y_{\alpha,N} - \hat{\Phi}_\alpha U_{\alpha,N} = Y_{\alpha,N} \Pi^{\perp}_{U^T_{\alpha,N}} \tag{2.39}$$

where $\Pi^{\perp}_{U^T_{\alpha,N}}$ is the orthogonal projection onto the nullspace of $U_{\alpha,N}$

$$\Pi^{\perp}_{U^T_{\alpha,N}} = I - U^T_{\alpha,N} \left(U_{\alpha,N} U^T_{\alpha,N} \right)^{-1} U_{\alpha,N} \tag{2.40}$$

such that

$$U_{\alpha,N} \Pi^{\perp}_{U^T_{\alpha,N}} = \mathbf{0} \tag{2.41}$$

The inverse of the matrix $\left(U_{\alpha,N} U^T_{\alpha,N} \right)$ exists if the input is persistently exciting[1] and $N > m\alpha$. Recall that m is the dimension of the input signal u_t.

By using Eq. (2.39), we can isolate the part of the output y_t which depends on the state x_t as follows

$$\begin{aligned} \Gamma_\alpha X_{1,N-\alpha+1} =& Y_{\alpha,N} - \hat{\Phi}_\alpha U_{\alpha,N} - \left(\Phi_\alpha - \hat{\Phi}_\alpha \right) U_{\alpha,N} \\ =& Y_{\alpha,N} \Pi^{\perp}_{U^T_{\alpha,N}} - \left(\Phi_\alpha - \hat{\Phi}_\alpha \right) U_{\alpha,N} \end{aligned} \tag{2.42}$$

By neglecting the term $\left(\Phi_\alpha - \hat{\Phi}_\alpha \right) U_{\alpha,N}$, the above equation becomes

$$\hat{\Gamma}_\alpha X_{1,N-\alpha+1} = Y_{\alpha,N} \Pi^{\perp}_{U^T_{\alpha,N}} \tag{2.43}$$

where $\hat{\Gamma}_\alpha$ is an estimation of the matrix Γ_α.

As the extended observability matrix (Γ_α) is a full rank matrix, one could estimate the matrix $\hat{\Gamma}_\alpha$ from Eq. (2.43) by using SVD.
Consider the following SVD

$$Y_{\alpha,N} \Pi^{\perp}_{U^T_{\alpha,N}} = \begin{bmatrix} Q_n & Q_n^{\perp} \end{bmatrix} \begin{bmatrix} S_n & 0 \\ 0 & S_2 \end{bmatrix} \begin{bmatrix} V_n^T \\ \left(V_n^{\perp} \right)^T \end{bmatrix} \tag{2.44}$$

where the matrix S_n contains the principals singular values (further than threshold).

[1] Generally speaking, it states that the input has properly excited the controllable and observable modes of the system.

An admissible solution for the matrix $\hat{\Gamma}_\alpha$ can be done by using the following formula

$$\hat{\Gamma}_\alpha = Q_n S_n^{\frac{1}{2}} \tag{2.45}$$

Note that the matrix $S_s^{\frac{1}{2}}$ is scaled the columns of the matrix $\hat{\Gamma}_\alpha$, but this does not change the estimated poles of system. The matrices A and C are calculated from $\hat{\Gamma}_\alpha$ similarly to Ho and Kalman's method as follows

$$\begin{aligned} C =& \hat{\Gamma}_\alpha(1:p,:) \\ A =& \left[\hat{\Gamma}_\alpha(1:(\alpha-1)p,:)\right]^\dagger \hat{\Gamma}_\alpha(p+1:p\alpha,:) \end{aligned} \tag{2.46}$$

The matrix $\hat{\Phi}_\alpha$ can be estimated by considering the least-squares solution to the following equation

$$Q_n^T Y_{\alpha,N} U_{\alpha,N}^T \left(U_{\alpha,N} U_{\alpha,N}^T\right)^{-1} = Q_n^T \hat{\Phi}_\alpha \tag{2.47}$$

Let us define the matrix Υ as follows

$$\Upsilon = Q_n^T Y_{\alpha,N} U_{\alpha,N}^T \left(U_{\alpha,N} U_{\alpha,N}^T\right)^{-1} \tag{2.48}$$

By using the linearity dependence of the matrix Φ_α on the matrix B, and once an estimation of the matrix $\hat{\Gamma}_\alpha$ is available. The estimation of B can be done by solving the following over-determined system of equations

$$\Upsilon(:,1:m) = \hat{\Gamma}_\alpha(1:(\alpha-1)p,:)B \tag{2.49}$$

As $\alpha > n$ and the matrix $\hat{\Gamma}_\alpha(1:np,:)$ is a full rank matrix, the matrix B can be estimated as follows

$$B = \left[\hat{\Gamma}_\alpha(1:(\alpha-1)p,:)\right]^\dagger \Upsilon(:,1:m) \tag{2.50}$$

2.2.3 PO-MOESP Method

The previous methods supposes that if the output signal is corrupted by a additive noise, then the effect of the noise on the estimation of system matrices will be reduced by the calculation of SVD.
In point of fact, these methods lead to a biased estimation and in general an optimization process should be used to improve the results.

Therefore, some methods have been developed in order to take directly the measurement noise into consideration and try to reduce its effect by using the idea of instrumental variables [Aoki 1990; VanOverschee et al. 1991; Verhaegen 1991] and the statical properties of the noise, e.g. its independence from the input signal and its white nature.

Let us consider a more general innovation representation

$$\begin{aligned} x_t &= Ax_{t-1} + Bu_t \\ y_t &= Cx_t + v_t \end{aligned} \tag{2.51}$$

where v_t is a stationary, ergodic white noise.
The PO-MOESP (Past Output-Multivariable Output-Error State Space) method [Verhaegen 1994] is one of these methods. The main idea of PO-MOESP method is to use the past input and output data as an instrumental variable in the objective of removing the effect of the noise.

In order to excite the observable and controllable modes of the system, we suppose that the input signal is sufficiently persisted exciting.
First, the method starts classically by stock the input in the following form defined as follows

$$U_{1,\alpha,N} \triangleq \begin{pmatrix} u_1 & u_2 & \cdots & u_{N-\alpha+1} \\ u_2 & u_3 & \cdots & u_{N-\alpha+2} \\ \vdots & \vdots & \ddots & \vdots \\ u_\alpha & u_{\alpha+1} & \cdots & u_N \end{pmatrix}$$

where the first subscript refers to the index of the first data sample, α stands for the number of rows in the matrix, and N refers to the index of the last data sample in the matrix. The number of rows should be chosen to be greater than the system order n [Viberg 1995]. The output matrix $Y_{1,\alpha,N}$ and the noise matrix $V_{1,\alpha,N}$ are defined analogously.

The following input-output equation is then easily derived from the system description (2.51)

$$Y_{1,\alpha,N} = \Gamma_\alpha X_{1,N-\alpha+1} + \Phi_\alpha U_{1,\alpha,N} + V_{1,\alpha,N} \tag{2.52}$$

where Γ_α is the extended observability matrix of the system, Φ_α is the extended matrix of controllability. The matrices Γ_α, Φ_α and $X_{1,N-\alpha+1}$ are defined similarly to Eq (2.36).
Consider the following QR factorization

$$\begin{bmatrix} U_{1+\alpha,\alpha,N} \\ U_{1,\alpha,N-\alpha} \\ Y_{1,\alpha,N-\alpha} \\ Y_{1+\alpha,\alpha,N} \end{bmatrix} = \begin{bmatrix} R_{11} & 0 & 0 & 0 \\ R_{21} & R_{22} & 0 & 0 \\ R_{31} & R_{32} & R_{33} & 0 \\ R_{41} & R_{42} & R_{43} & R_{44} \end{bmatrix} \begin{bmatrix} Q_1 \\ Q_2 \\ Q_3 \\ Q_4 \end{bmatrix} \tag{2.53}$$

where the R_{ii} are lower-triangular matrices and the orthonormal matrix Q_{ii} is partitioned according to the dimension of lower-triangular matrices R_{ii}.

Theorem 2.1 *If we suppose that the input signal u_t is sufficiently persisted exciting, then*

$$\lim_{N\to\infty}\frac{1}{\sqrt{N}}Y_{1+\alpha,\alpha,N}Q_2^T=\lim_{N\to\infty}\frac{1}{\sqrt{N}}\Gamma_\alpha X_{1+\alpha,N}Q_2^T \tag{2.54}$$

$$\lim_{N\to\infty}\frac{1}{\sqrt{N}}Y_{1+\alpha,\alpha,N}Q_3^T=\lim_{N\to\infty}\frac{1}{\sqrt{N}}\Gamma_\alpha X_{1+\alpha,N}Q_3^T \tag{2.55}$$

$$\lim_{N\to\infty}\frac{1}{\sqrt{N}}Y_{1,\alpha,N-\alpha}Q_1^T=\lim_{N\to\infty}\frac{1}{\sqrt{N}}\left(\Gamma_\alpha X_{1,N-\alpha}Q_1^T+\Phi_\alpha R_{21}\right) \tag{2.56}$$

$$\lim_{N\to\infty}\frac{1}{\sqrt{N}}Y_{1,\alpha,N-\alpha}Q_2^T=\lim_{N\to\infty}\frac{1}{\sqrt{N}}\left(\Gamma_\alpha X_{1,N-\alpha}Q_2^T+\Phi_\alpha R_{22}\right) \tag{2.57}$$

$$\lim_{N\to\infty}\frac{1}{\sqrt{N}}Y_{\alpha+1,\alpha,N}Q_1^T=\lim_{N\to\infty}\frac{1}{\sqrt{N}}\left(\Gamma_\alpha X_{\alpha+1,N}Q_1^T+\Phi_\alpha R_{11}\right) \tag{2.58}$$

where the convergence of the above equations is with the probability of one.

Proof 2.1 *See Appendix A.1.1.*

Using Eqs (2.54- 2.55) of Theorem 2.1, we obtain

$$\begin{aligned}\frac{1}{\sqrt{N}}\left[R_{42}\ R_{43}\right]&=\frac{1}{\sqrt{N}}Y_{\alpha+1,\alpha,N}\left[Q_2^T\ Q_3^T\right]+O_N(\epsilon)\\&=\frac{1}{\sqrt{N}}\Gamma_\alpha X_{\alpha+1,N}\left[Q_2^T\ Q_3^T\right]+O_N(\epsilon)\end{aligned} \tag{2.59}$$

where $O_N(\epsilon)$ is a matrix of appropriate dimension and ϵ-norm for finite N and vanishing when $N\to\infty$.

As the input signal is sufficiently exciting, we obtain that the matrix $X_{\alpha+1,\alpha,N}\left[Q_2^T\ Q_3^T\right]$ would be of rank n, if and only if $\alpha>n$.

Let us consider the SVD of the matrix $\frac{1}{\sqrt{N}}\left[R_{42}\ R_{43}\right]$

$$\frac{1}{\sqrt{N}}\left[R_{42}\ R_{43}\right]=U_nS_nV_n^T+U_n^\perp S_2{V_n^\perp}^T \tag{2.60}$$

where the matrix S_n contains the principals singular values (further than threshold).

Then the matrix Γ_α can be estimated as follows

$$\Gamma_\alpha=U_nS_n^{1/2} \tag{2.61}$$

and so we estimate the matrices C et A directly from Γ_α similarly to Eq. (2.46).

Eqs (2.56 - 2.58) of Theorem 2.1 are denoted compactly as

$$\begin{aligned}\frac{1}{\sqrt{N}}\begin{bmatrix}R_{31} & R_{32} & R_{41}\end{bmatrix} =&\Gamma_\alpha \frac{1}{\sqrt{N}}\begin{bmatrix}X_{1,\alpha,N-\alpha}Q_1^T & X_{1,\alpha,N-\alpha}Q_2^T & X_{1+\alpha,\alpha,N}Q_1^T\end{bmatrix}\\ &+\Phi_\alpha \frac{1}{\sqrt{N}}\begin{bmatrix}R_{21} & R_{22} & R_{11}\end{bmatrix}+O_N(\epsilon)\end{aligned} \tag{2.62}$$

when the column space of the matrix Γ_α and its orthogonal complement are respectively equal to U_n and ${U_n}^\perp$, then one can eliminate the term which depends on Γ_α by multiplying Eq. (2.62) on the left by $\left(U_n^\perp\right)^T$, and obtain

$$\left({U_n}^\perp\right)^T \frac{1}{\sqrt{N}}\begin{bmatrix}R_{31} & R_{32} & R_{41}\end{bmatrix} = \left({U_n}^\perp\right)^T \frac{1}{\sqrt{N}}\Phi_\alpha \begin{bmatrix}R_{21} & R_{22} & R_{11}\end{bmatrix}+O_N(\epsilon) \tag{2.63}$$

Caused by the sufficiently persistence of excitation of the input, the matrix $\begin{bmatrix}R_{21} & R_{22} & R_{11}\end{bmatrix}$ has a right pseudo-inverse. Multiplying Eq (2.63) by this inverse, yields

$$\left({U_n}^\perp\right)^T \frac{1}{\sqrt{N}}\begin{bmatrix}R_{31} & R_{32} & R_{41}\end{bmatrix}\left(\frac{1}{\sqrt{N}}\begin{bmatrix}R_{21} & R_{22} & R_{11}\end{bmatrix}\right)^\dagger = \left({U_n}^\perp\right)^T \Phi_\alpha + O_N(\epsilon) \tag{2.64}$$

If we denote the left side of this equation by Ξ, then this equation can be written as

$$\Xi = \left({U_n}^\perp\right)^T \Phi_\alpha + O_N(\epsilon) \tag{2.65}$$

Then, the matrix B can be estimated by solving the following equation

$$\Xi(:,1:m) = \left({U_n}^\perp\right)^T \times \begin{pmatrix} 0_{p\times n} \\ U_n(1:p,:) \\ U_n(p+1:2p:) \\ \vdots \\ U_n(p(\alpha-2)+1:p(\alpha-1),:)\end{pmatrix} B \tag{2.66}$$

2.3 Conclusion

On account of the limitations of linear difference model to be a reliable model for the complex systems, the attention of researchers has pointed to the linear state space realization as an alternative representation.

In this chapter an overview of subspace identification methods has been given. The research on the improvement of the subspace identification methods is an active research, and many methods have been proposed. However, the main objective in this chapter is explaining some basic subspace methods and pointing out their main features.

The application of subspace methods to identify real world systems has proven their efficiency.

Moreover, the state space realization provided by subspace methods is suitable for the control purposes.

Résumé du chapitre 3
Identification des systèmes : cas d'expérimentations multiples

Bien que l'identification des systèmes linéaires ait été beaucoup étudiée durant ces dernières années, comme nous l'avons montré dans le chapitre 2, le cas des expérimentations multiples n'est pas fréquemment abordé. Pourtant, ce cas est très important en pratique, où on identifie le modèle du système en question en mesurant ses réponses de sorties à plusieurs séquences d'entrées. L'objectif de chaque séquence d'entrée est généralement d'exciter un ou plusieurs modes du système considéré.

Les séquences d'entrée/sortie pour certaines expérimentations réelles sont parfois courtes, soit parce que la fréquence d'échantillonnage est élevée, soit parce que le phénomène étudié est très court. Dans ces cas particuliers, toutes les données d'entrée/sortie devront être exploitées dans le but d'obtenir un unique modèle précis du système.

En outre, dans le cas des expérimentations courtes et multiples, les effets des conditions initiales ne peuvent pas être négligées. Celles-ci devront donc être estimées.

Dans le chapitre 3, nous développons une méthode d'identification des systèmes linéaires en considérant les ensembles de données conjointement afin d'obtenir un modèle précis du système concerné.

Rappelons que le modèle linéaire en espace d'état est décrit par les équations suivantes :

$$x_t = Ax_{t-1} + Bu_t$$
$$y_t = Cx_t + v_t$$

où $x_t \in \mathbb{R}^n, u_t \in \mathbb{R}^m, y_t \in \mathbb{R}^p$ et $v_t \in \mathbb{R}^p$ sont respectivement l'état interne du système, le signal d'entrée, le signal de sortie et le bruit des mesures. L'objectif de l'identification est d'estimer les matrices A, B, C et éventuellement la condition initiale de l'état (x_0).

Supposons qu'on dispose d'une base de données qui contient K expérimentations

$$\left\{u_t^i, y_t^i\right\}_{t=1}^{N_i} : i = 1, 2, \cdots, K$$

Le but dans ce cas sera l'estimation des matrices A, B, C et des conditions initiales de l'état de chaque expérimentation $\{x_0^i : i = 1, 2, \cdots, K\}$. Nous montrons que la méthode sous-espace classique PO-MOESP (section 2.2.3), peut être étendue dans ce but.

Cependant, cette estimation sera biaisée au fait de la courte durée des signaux d'entrée et à cause des effets de bruit sur les données mesurées. L'optimisation du modèle obtenu par l'extension de la méthode PO-MOESP est donc une étape nécessaire afin d'avoir un modèle fiable du système linéaire.

Le modèle obtenu par la méthode PO-MOESP sera seulement utilisé pour donner une valeur initiale des paramètres du système à un algorithme d'optimisation de type moindres carrés.

Définissons le vecteur des paramètres du système comme suit:

$$\theta = \begin{bmatrix} vec(A) \\ vec(B) \\ vec(C) \\ x_0^1 \\ x_0^2 \\ \vdots \\ x_0^K \end{bmatrix}$$

où vec(.) désigne l'opérateur de vectorisation. Celui-ci est défini comme suit:

$$vec : M \in \mathbb{R}^{m \times n} \rightarrow \mathbb{R}^{m \cdot n}$$
$$vec(M) = vec\begin{bmatrix} m_1 & m_2 \cdots m_n \end{bmatrix} = \begin{bmatrix} m_1^T & m_2^T \cdots m_n^T \end{bmatrix}^T$$

Si on définit l'erreur de sortie par la relation $e_k^i(\theta) = y_k^i - \hat{y}_k^i(\theta)$, le problème d'identification revient à trouver le vecteur θ qui minimise la norme de l'erreur de sortie pour toutes les expérimentations conjointement.

Cela nous amène à minimiser la somme suivante:

$$J_K(\theta) = \frac{1}{K} \sum_{j=1}^{K} \frac{1}{N_j} \sum_{k=1}^{N_j} \left\| e_k^i(\theta) \right\|_2^2 \tag{2.67}$$

Pour résoudre ce problème d'optimisation, nous pouvons, par exemple, utiliser la méthode Levenberg-Marquard, .

Nous montrons dans le chapitre 3 l'efficacité de la méthode proposée sur plusieurs exemples académiques ainsi que sur un système réel d'enroulement industriel.

"No amount of experimentation can ever prove me right; a single experiment can prove me wrong."

Albert Einstein

3

System Identification: Multi-experiments Case

The identification of linear systems has been extremely studied in the literature of system identification as discussed in Chapter 2.

On the other hand, the case of short multi-experiments is infrequently addressed. The raison why this is important is that the identification of real systems is usually done by measuring the output responses of the system to various input sequences. The objective of each sequence is exciting one or more modes of the system.

In real experiments, sometimes we obtain short sets of input/output data that due to a large time sampling period or short tracked phenomena. In these cases, we should exploit all the data sets to obtain an accurate model of the system. Moreover, in the case of short experiments, the effect of initial conditions can not be neglected, so we should also estimate them.

The objective in this chapter is to develop an identification method able to treat the data sets simultaneously in order to provide an accurate model of the linear system.

The remainder of the chapter is organized as follows, In Section 3.1, an extension of the classical subspace method PO-MOESP to treat the case of multi-experiments is introduced. The output error identification problem is formulated in Section 3.2 in order to optimize the obtained model. The algorithm of identification is summarized in Section 3.3. Finally, Section 3.4 presents some illustrative examples and comparisons with the classical methods of linear system identification.

3.1 Extension of PO-MOESP Method

Consider the linear state space model, which has the following structure

$$\begin{aligned} x_t &= Ax_{t-1} + Bu_t \\ y_t &= Cx_t + v_t \end{aligned} \tag{3.1}$$

where x_t, u_t, y_t and v_t are the internal state of the system, the input signal, the output signal and the measurement noise.
It is clear that the model representation (2.52) of PO-MOESP method holds for an arbitrary non-zero initial conditions. For that, non-zero initial conditions have no effect at all on the calculations of the triple $[A, B, C]$. As a result, dealing with multiple data sets does not introduce an additional problem [Verhaegen et al. 1994].

Therefore PO-MOESP method can be easily adapted to deal with multiple data sets.
Consider the following data sets

$$\left\{u_t^i, y_t^i\right\}_{t=1}^{N_i} \text{ and } i = 1, 2, \cdots, K \tag{3.2}$$

where each data set corresponds to one experiment.
For each input/output data set, we obtain the following data equation

$$Y_{1,\alpha,N_i}^i = \Gamma_\alpha X_{1,N_i-\alpha+1}^i + \Phi_\alpha U_{1,\alpha,N_i}^i + V_{1,\alpha,N_i}^i \tag{3.3}$$

Then, the data equations can be easily combined as follows

$$[Y_{1,\alpha,N_1} \cdots Y_{1,\alpha,N_K}] = \Gamma_\alpha \left[X_{1,N_1-\alpha+1}^1 \cdots X_{1,N_K-\alpha+1}^K\right] + \Phi_\alpha \left[U_{1,\alpha,N_1}^1 \cdots U_{1,\alpha,N_K}^K\right] + \left[V_{1,\alpha,N_1}^1 \cdots V_{1,\alpha,N_K}^K\right] \tag{3.4}$$

The structure of this equation is similar to that of the original data equation (2.52), for that we still can use the main body of PO-MOSEP algorithm by computing QR factorization of the following matrix

$$\begin{bmatrix} U_{1+\alpha,\alpha,N_1}^1 | U_{1+\alpha,\alpha,N_2}^2 | \cdots | U_{1+\alpha,\alpha,N_K}^K \\ U_{1,\alpha,N_1-\alpha}^1 | U_{1,\alpha,N_2-\alpha}^2 | \cdots | U_{1,\alpha,N_K-\alpha}^K \\ Y_{1,\alpha,N_1-\alpha}^1 | Y_{1,\alpha,N_2-\alpha}^2 | \cdots | Y_{1,\alpha,N_K-\alpha}^K \\ Y_{1+\alpha,\alpha,N_1}^1 | Y_{1+\alpha,\alpha,N_2}^2 | \cdots | Y_{1+\alpha,\alpha,N_K}^K \end{bmatrix} = \begin{bmatrix} \mathcal{R}_{11} & 0 & 0 & 0 \\ \mathcal{R}_{21} & \mathcal{R}_{22} & 0 & 0 \\ \mathcal{R}_{31} & \mathcal{R}_{32} & \mathcal{R}_{33} & 0 \\ \mathcal{R}_{41} & \mathcal{R}_{42} & \mathcal{R}_{43} & \mathcal{R}_{44} \end{bmatrix} \mathcal{Q}^T \tag{3.5}$$

Using a logic analogous to the case of single experiment, we calculate the following SVD

$$\left[\mathcal{R}_{42}\ \mathcal{R}_{43}\right] = \mathcal{U}_n \mathcal{S}_n \mathcal{V}_n^T + \mathcal{U}_n^\perp \diamond \mathcal{S}_2 \mathcal{V}_n^{\perp T} \tag{3.6}$$

where the matrix $\mathcal{S}_n$ contains the principles singular values (further than threshold). Then, the

matrix Γ_α can be estimated as follows

$$\Gamma_\alpha = \mathcal{U}_n \mathcal{S}_n^{1/2} \tag{3.7}$$

The matrix Φ_α can be estimated from the following equation

$$\left(\mathcal{U}_n{}^{\perp}\right)^T \frac{1}{\sqrt{N}} \left[\mathcal{R}_{31}\ \mathcal{R}_{32}\ \mathcal{R}_{41}\right] \left(\frac{1}{\sqrt{N}} \left[\mathcal{R}_{21} \quad \mathcal{R}_{22} \quad \mathcal{R}_{11}\right]\right)^{\dagger} = \left(\mathcal{U}_n{}^{\perp}\right)^T \Phi_\alpha \tag{3.8}$$

The matrices C, A and B can be estimated by using Eqs (2.46) and (2.66).

The initial condition x_0^i can be estimated for a data set $\{u_t^i, y_t^i\}$ from the following equation

$$\begin{bmatrix} y_1^i \\ y_2^i \\ \vdots \\ y_\alpha^i \end{bmatrix} = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{(\alpha)} \end{bmatrix} x_0^i + \begin{bmatrix} CB & 0 & \cdots & 0 \\ CAB & CB & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ CA^{(\alpha)}B & \cdots & \cdots & CB \end{bmatrix} \begin{bmatrix} u_1^i \\ u_2^i \\ \vdots \\ u_\alpha^i \end{bmatrix} \tag{3.9}$$

Let us denote

$$\Xi_\alpha \triangleq \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^{(\alpha)} \end{bmatrix} \tag{3.10}$$

As Γ_α is a full rank matrix and $\alpha > n$, the matrix Ξ_α has also full rank. Recall that n is the order of the linear system.
Eq. (3.9) provides an estimation of x_0^i by using the pseudo-inverse of Ξ_α as follows

$$\hat{x}_0^i = \Xi_\alpha^{\dagger} \left(\begin{bmatrix} y_1^i \\ y_2^i \\ \vdots \\ y_\alpha^i \end{bmatrix} - \begin{bmatrix} CB & 0 & \cdots & 0 \\ CAB & CB & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ CA^{(\alpha-1)}B & \cdots & \cdots & CB \end{bmatrix} \begin{bmatrix} u_1^i \\ u_2^i \\ \vdots \\ u_\alpha^i \end{bmatrix} \right) \tag{3.11}$$

3.2 Output Error Identification

It is well known that the model obtained by the above approach is not optimal. This is because the input signals are short, and as well the effects of noise on the observed output signals. Therefore, the optimization of the obtained model is a necessary step to obtain a reliable model.

The obtained parameters of model with the PO-MOESP technique will just be used as an initial guess of the parameters to be optimized.
If we consider that we have K experiments, then the vector of parameters of the linear model can be chosen as follows

$$\theta = \begin{bmatrix} vec(A) \\ vec(B) \\ vec(C) \\ x_0^1 \\ x_0^2 \\ \vdots \\ x_0^K \end{bmatrix} \tag{3.12}$$

where vec(.) denotes the vectorization operator defined as follows

$$\begin{aligned} &vec : M \in \mathbb{R}^{m\times n} \rightarrow \mathbb{R}^{m\cdot n} \\ &vec(M) = vec\begin{bmatrix} m_1 & m_2 \cdots m_n \end{bmatrix} = \begin{bmatrix} m_1^T & m_2^T \cdots m_n^T \end{bmatrix}^T \end{aligned}$$

Let us define

$$e_k^i(\theta) = y_k^i - \hat{y}_k^i(\theta) \tag{3.13}$$

Our objective is to find the vector of parameters θ which minimizes the output error function for all experiments simultaneously.

Therefore, this leads to minimize the following sum

$$J_K(\theta) = \frac{1}{K}\sum_{j=1}^{K}\frac{1}{N_j}\sum_{k=1}^{N_j}\left\|e_k^i(\theta)\right\|_2^2 = \frac{1}{K}E_K(\theta)^T E_K(\theta) \tag{3.14}$$

where

$$E_K(\theta) = \begin{bmatrix} E_{N_1}^1(\theta)^T & E_{N_2}^2(\theta)^T \cdots E_{N_K}^K(\theta)^T \end{bmatrix}^T \tag{3.15}$$

and

$$E_{N_i}^i(\theta) = \frac{1}{\sqrt{N_i}}\begin{bmatrix} e_1^i(\theta)^T & e_2^i(\theta)^T \cdots e_{N_i}^i(\theta)^T \end{bmatrix}^T \tag{3.16}$$

Assume that we have the sets $\{u_t^i, y_t^i : t = 1, 2, \cdots, N_i \; and \; i = 1, 2, \cdots, K\}$, and the estimated output $\hat{y}_t^j(\hat{\theta})$ of the data set number j is given by the following model

$$\begin{aligned} \hat{x}_t^j &= A(\hat{\theta})\hat{x}_{t-1}^j + B(\hat{\theta})u_t^j \\ \hat{y}_t^j(\hat{\theta}) &= C(\hat{\theta})\hat{x}_t^j \end{aligned} \tag{3.17}$$

The numerical solution of the minimization problem (3.14) can be calculated by different algorithms, e.g. gradient search method (Levenberg-Marquard method is a popular one). This iterative method is based on the updating of the system vector of parameters θ as follows

$$\hat{\theta}^{l+1} = \hat{\theta}^l - (\psi_K^T(\hat{\theta}^l)\psi_K(\hat{\theta}^l) + \lambda^l I)^{-1}\psi_K^T(\hat{\theta}^l)E_K(\hat{\theta}^l) \tag{3.18}$$

where

$$\psi_K(\theta) = \begin{bmatrix} \psi^1_{N_1}(\theta) \\ \psi^2_{N_2}(\theta) \\ \vdots \\ \psi^K_{N_K}(\theta) \end{bmatrix} \tag{3.19}$$

and

$$\psi^i_{N_i}(\theta) \triangleq \frac{\partial E^i_{N_i}(\theta)}{\partial \theta^T} \tag{3.20}$$

is the jacobian of the error vector $E^i_{N_i}$.

3.2.1 Computing the Iterative Parameter Update

In order to compute the update rule (3.18), the following quantities $E_K(\hat{\theta}^l)$ and $\psi_K(\hat{\theta}^l)$ must be computed. For that, we simulate the systems (3.17) that corresponds to $\hat{\theta}^l$.
This simulation brings out the state $\hat{x}^i_t$ and $\hat{y}^i_t$ for $i = 1, 2, \cdots, K$.

In order to compute $\psi_K(\hat{\theta}^l)$ defined by (3.19), we should compute all gradients $\psi^i_{N_i}(\theta)$ (3.20). According to (3.13), these gradients require the following computation

$$\frac{\partial e^i_k(\theta)}{\partial \theta} = -\frac{\partial \hat{y}^i_t(\theta)}{\partial \theta} \tag{3.21}$$

For that, let us define

$$\zeta^{i,j}_t = \frac{\partial \hat{x}^i_t}{\partial \theta_j} \tag{3.22}$$

where θ_j is the j^{th} element of the vector θ.

The computation of $\frac{\partial \hat{y}^i_t}{\partial \theta} = \begin{bmatrix} \frac{\partial \hat{y}^i_t}{\partial \theta_1} & \cdots & \frac{\partial \hat{y}^i_t}{\partial \theta_q} \end{bmatrix}$, where q is the number of parameters in θ, can be made using the following model

$$\begin{aligned} \zeta^{i,j}_t &= A\zeta^{i,j}_{t-1} + \frac{\partial A}{\partial \theta_j}\hat{x}^i_{t-1} + \frac{\partial B}{\partial \theta_j}u^i_t \\ \frac{\partial \hat{y}^i_t}{\partial \theta_j} &= C\zeta^{i,j}_t + \frac{\partial C}{\partial \theta_j}\hat{x}^i_t \end{aligned} \tag{3.23}$$

Note that $\zeta^{i,j}_0 = \frac{\partial \hat{x}^i_0}{\partial \theta_j} \neq \mathbf{0}$. This is because the initial condition is included into the vector of parameters.

Lemma 3.1 *The system (3.23) is stable if and only if the linear system (3.17) is stable.*

Proof 3.1 *See Appendix A.2.1.*

3.3 Summary of the Identification Algorithm

We can resume the algorithm of the proposed identification method as follows

1. The extension of PO-MOESP method explained in Section 3.1 provides an initial estimation of the system vector of parameters $\hat{\theta}^0$, $l \leftarrow 0$.
2. Calculate the state $\hat{x}_t$ and $\hat{y}_t$ by simulating the system (3.17) with $\hat{\theta} = \hat{\theta}^l$.
3. Compute $E_K(\hat{\theta}^l)$ using (3.15).
4. Calculate the matrix ψ_K using (3.19), (3.20), (3.21) and (3.23).
5. Calculate the update rule of the gradient search algorithm using (3.18), $l \leftarrow l+1$.
6. Perform the termination test for minimization. If true, the algorithm stops. Otherwise, return to Step 2. i.e. compute the values $J_L\left(\hat{\theta}^{l-1}\right)$ and $J_L\left(\hat{\theta}^{l}\right)$ using (3.14) and test if $\| J_L\left(\hat{\theta}^{l}\right) - J_L\left(\hat{\theta}^{l-1}\right) \|_2$ is small enough.

3.4 Illustrative Examples

In this section, we compare the results of the proposed method with those obtained by the actual methods for identifying linear systems.

1. We compare with the results obtained by a classic subspace method (N4SID) [Overschee and Moor 1994].
2. We study the effects of the Signal to Noise Ration (SNR) on the obtained results and we compare with the results obtained by N4SID method.
3. We study the effects of the number of data sets on the obtained results.
4. We consider the case of multiple experiments where the input of each experiment is a sum of sinusoidal signals.
5. We consider the case of multiple experiments collected from an industrial winding process.

For the first four study cases, we consider the linear system (3.1), which is characterized by the following matrices

$$A = \begin{bmatrix} 0.5 & 0 & 0 & 0 & 0 & 0 \\ 0 & -0.1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.8 & -0.4 & 0 \\ 0 & 0 & 0 & 0.8 & 0.6 & 0 \\ 0 & 0 & 0 & 0 & 0 & -0.7 \end{bmatrix}, \; B = \begin{bmatrix} -0.4 & 1.2 \\ -1.7 & -0.1 \\ 0.2 & 0.3 \\ 0.3 & 0.2 \\ -1.1 & -0.2 \\ 1.2 & 0.7 \end{bmatrix} \tag{3.24}$$

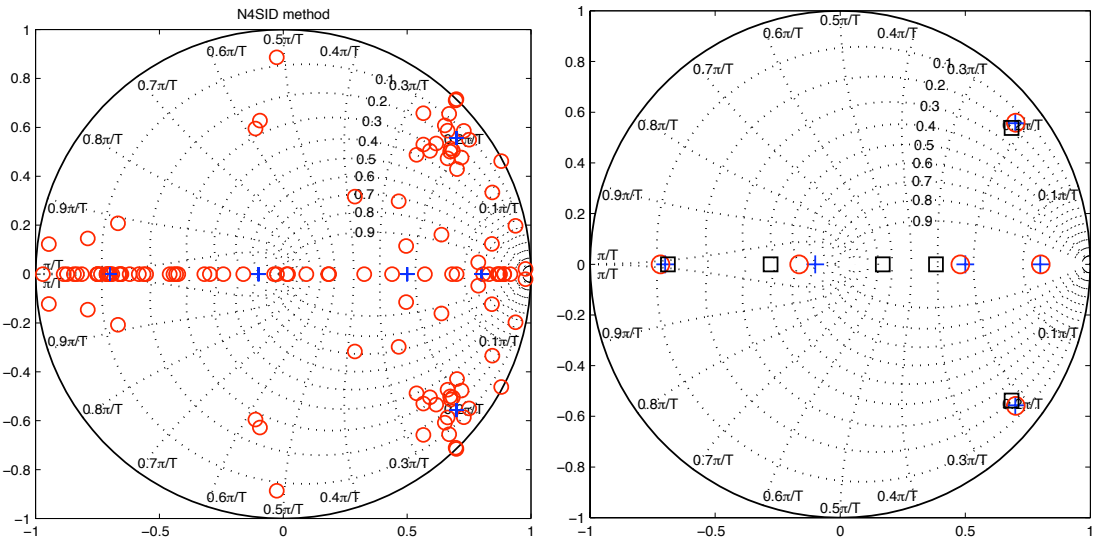
$$C = \begin{bmatrix} -0.6 & -0.1 & 1.1 & -0.1 & 0.3 & 0.7 \\ 2.2 & 0.1 & 0.1 & -0.8 & -1.3 & 1.6 \end{bmatrix}$$

and the measurement noise is a Gaussian white noise.

In all following figures, the poles of the matrix A will be designed by plus marks.

3.4.1 Comparison with N4SID Method

We have simulated the system 20 times with random initial conditions x_0^i, and the length of each data set is equal to 40 samples.



(a) N4SID (circle mark)

(b) PO-MOESP extension (square mark) and optimized model (circle mark)

Figure 3.1: Comparison with N4SID method, the eigenvalues of A are designed by plus marks.

It is well known that the eigenvalues of the matrix A are invariant for all state-space realizations of the system. For that, we compare the performance of the proposed method and the conventional methods by calculating the eigenvalues of the obtained matrix $\hat{A}$, and comparing them with those of A.

The results obtained by N4SID from MATLAB system identification toolbox [Ljung 1995] for each data set are given in Fig. 3.1(a). Recall that N4SID from MATLAB system identification toolbox can only handle the case of single experiment. The SNR is chosen to be equal to 15 dB. The results obtained by considering all the data sets without optimization and the optimized results are given in Fig. 3.1(b).

From Figure 3.1, we conclude that

1. The N4SID method is not able to identify a reliable model of the linear system by treating the data sets separately.

2. The extension of PO-MOESP method gives a good initial estimation of the linear model, but it is not accurate.
3. Using the proposed optimization algorithm improves the accuracy of the estimated linear model.

3.4.2 Sum of Sinusoidal Signals

In this example, we have simulated the system 30 times with random initial conditions x_0^i, and the length of each data set is equal to 50 samples.
The input signals are chosen to be sum of sinusoidal signals

$$u_t = \sum_{k=1}^{l} \beta_k sin(\omega_k t) \tag{3.25}$$

where l is the number of sinusoidal signals; it is chosen to be equal to 5.

For each experiment, the constants $\{\beta_k : k = 1, 2, \cdots, l\}$ are chosen randomly in the interval $[-1, +1]$, and the frequencies $\{\omega_k : k = 1, 2, \cdots, l\}$ are chosen randomly in the interval $[-\pi, +\pi]$. The measurement noise is scaled such that the SNR is equal to 20 dB.
An example of the input/output sequence is given in Fig. 3.2.

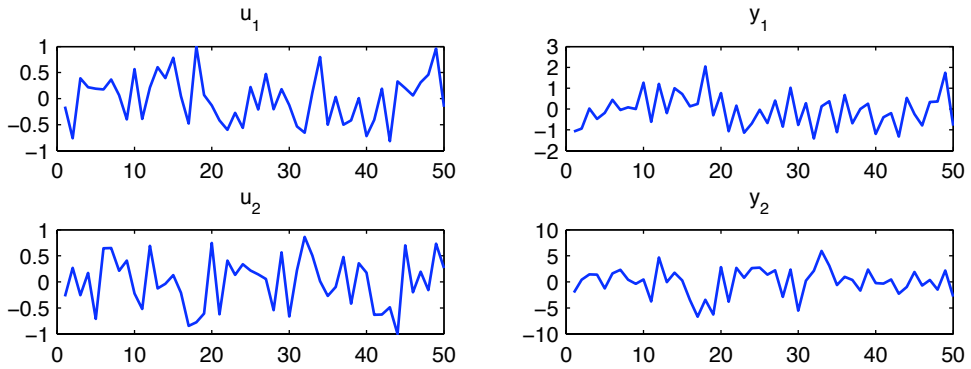
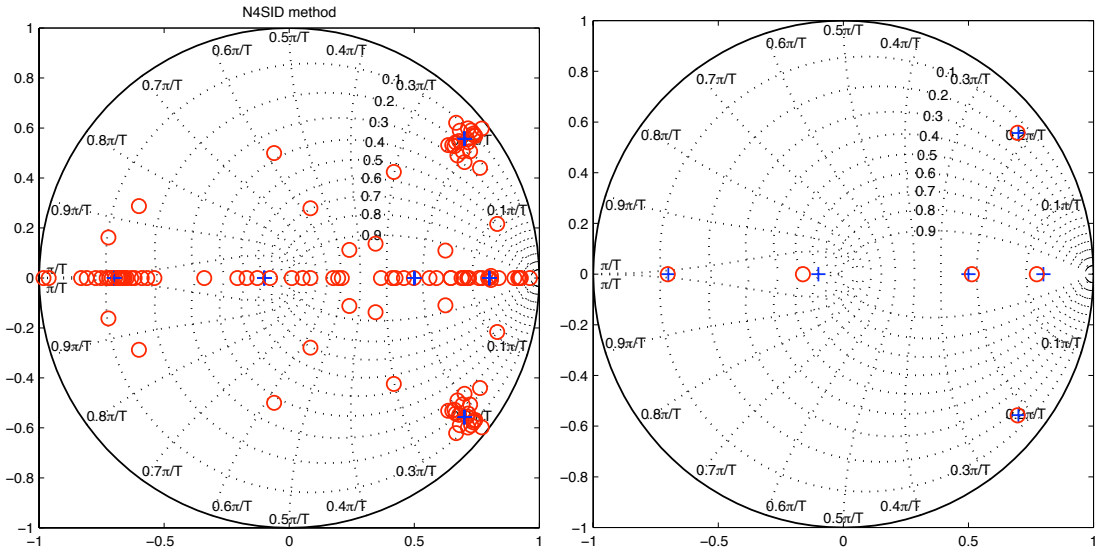


Figure 3.2: The inputs u_1 and u_2 are sum of sinusoidal signals, and the outputs are y_1 and y_2.

From Fig. 3.3, we conclude that the proposed method is able to estimate a reliable model in this case. Although, the N4SID method succeeds some times in finding a good model, it generally fails in achieving this task.

Note that such kind of input signal can be applied on real world systems in order to identify their models. Therefore, considering this case is important from a practical point of view.



(a) N4SID (b) Our method

Figure 3.3: The inputs are sum of sinusoidal signals.

3.4.3 The Effect of the SNR

In this example, we have simulated the system 20 times with random initial conditions x_0^i, and the length of each data set is equal to 40 samples.
In order to study the effect of the SNR, we consider the following values of SNR 20, 10 and 5 dB. The results obtained by N4SID from MATLAB system identification toolbox for each data set are given in Fig. 3.4.

From Fig. 3.4, we conclude that considering all experiments simultaneously reduces the effect of noise on the estimated model. As a consequence, we obtain a model more accurate than those obtained by N4SID method.

3.4.4 The Effect of the Number of Data Sets

In this experience, we consider data sets of small length, which is equal to 30 samples. Note that the method N4SID does not work in this case and gives an error message. This is because the data length is too small. Assume that we have simulated the linear system 3, 5, 7, 10, 15 and 20 times with random initial conditions, and the measurement noise is scaled such that $SNR = 15\ dB$.
The eigenvalues of the obtained matrix $\hat{A}$ by considering all the data sets and minimizing the output error function are given in Fig. 3.5.
It is clear that, the accuracy of the estimated model increases with increasing the number of data sets.

3.4.5 Industrial Winding Process

The process is a test setup of an industrial winding process. A description of the industrial winding system [Bastogne et al. 1998] is presented in Fig. 3.6.

The main part of the plant is composed of a plastic web that is unwinded from first reel (unwinding reel), goes over the traction reel and is finally rewinded on the the rewinding reel. Reel 1 and 3 are coupled with a DC-motor that is controlled with input currents I_1^* and I_3^*. The angular speed of each reel are S_1, S_2 and S_3. The tensions in the web between reel 1 and 2 is T_1, and between reel 2 and 3 is T_3. They are measured by dynamo tachometers and tension meters. The system has five input signals (S_1, S_2, S_3, I_1^* and I_3^*) and two outputs (T_1 and T_2).
The input/output data set is obtained from DaISy (SISTA's Identification Database) [Moor 1997]. The data length is 2500 samples and the sampling time is 0.1 sec.
An example of the input signals used in the experiment is given in Fig. 3.7.

In order to transform this identification into a multi-experiments identification problem, we have divided the data set into two parts. The lengths of these two parts are 1500 and 1000 samples respectively.
Then, we have studied the following two cases

- Case 1: The first part is divided into 42 data sets, which have length $\in [20, 60]$. These data sets are used to identify the model of the industrial winding and the second part is used to validate the obtained model. The obtained results are given in Fig. 3.8(a).
- Case 2: The first part is divided into 21 data sets, which have length $\in [35, 100]$. Similarly to the previous case, these data sets are used to identify the model of the industrial winding and the second part is used to validate the obtained model. The obtained results are given in Fig. 3.8(b).

To deal with a clear representation, we present the data over a window of length is equal to 150 samples in Fig. 3.8.

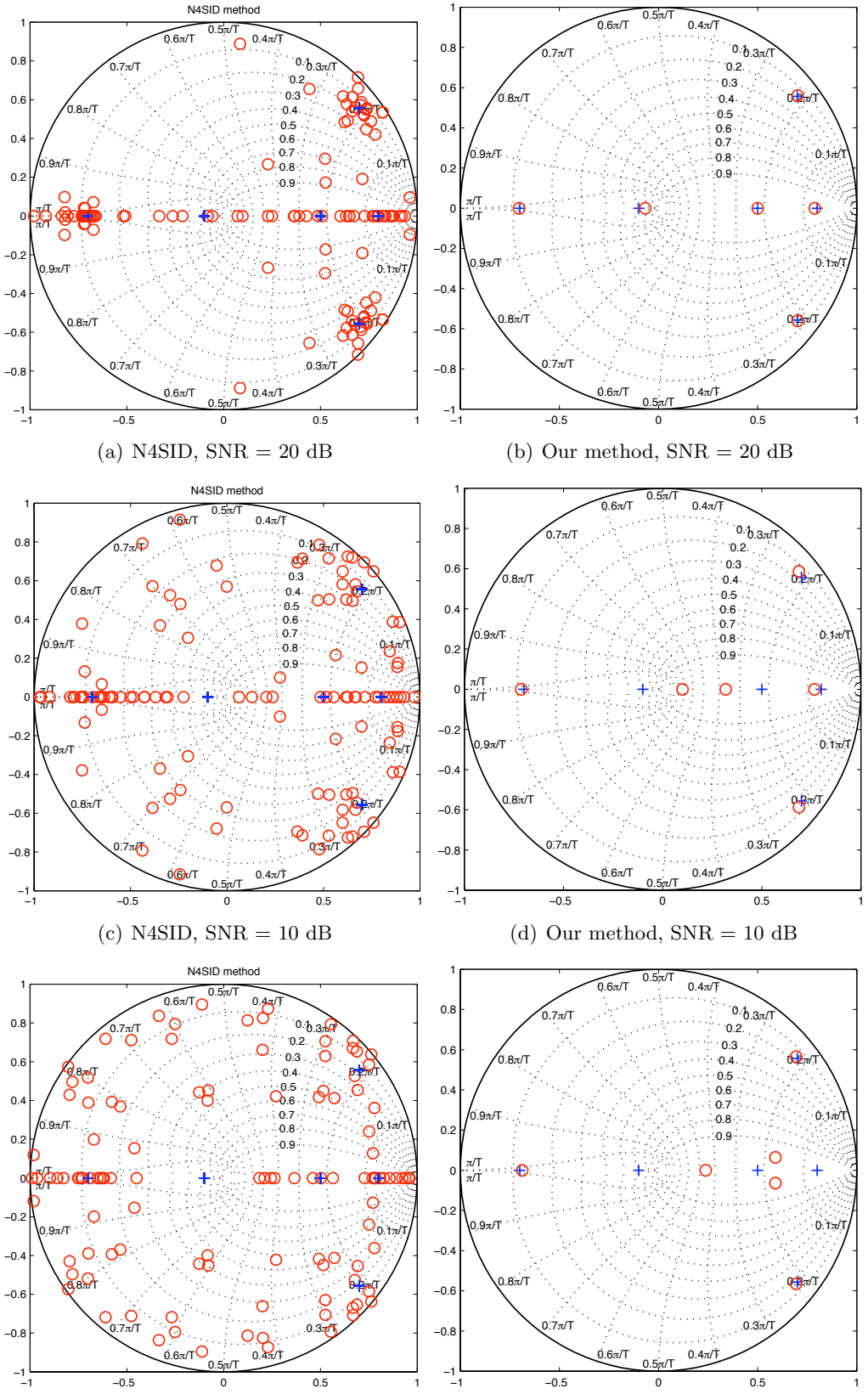
To compare the obtained results with those obtained by N4SID method, we have applied N4SID function from MATLAB identification toolbox to the data set which have the biggest length. Note that the estimation for the initial condition of the validation data set is done offline by using the identified model and Eq. (3.11).

Fig. 3.8 shows that the proposed method is able to estimate a linear approximation model of the industrial winding process, when N4SID method fails to find such a model in the first case because the data sets are short.

3.5 Conclusion

In this chapter, we have proposed a method to deal with the identification of linear multivariable systems in the case of short multi-experiments. The results have pointed out that when the classical methods of subspace methods fail in finding a reliable linear model of the system, the proposed method provides an accurate model of the linear system.

In the literature of system identification theory, the problem of dealing with short multi-experiments is infrequently treated. However, we believe that this problem should be given more attention because it is the case for many real situations in practice.



(a) N4SID, SNR = 20 dB

(b) Our method, SNR = 20 dB

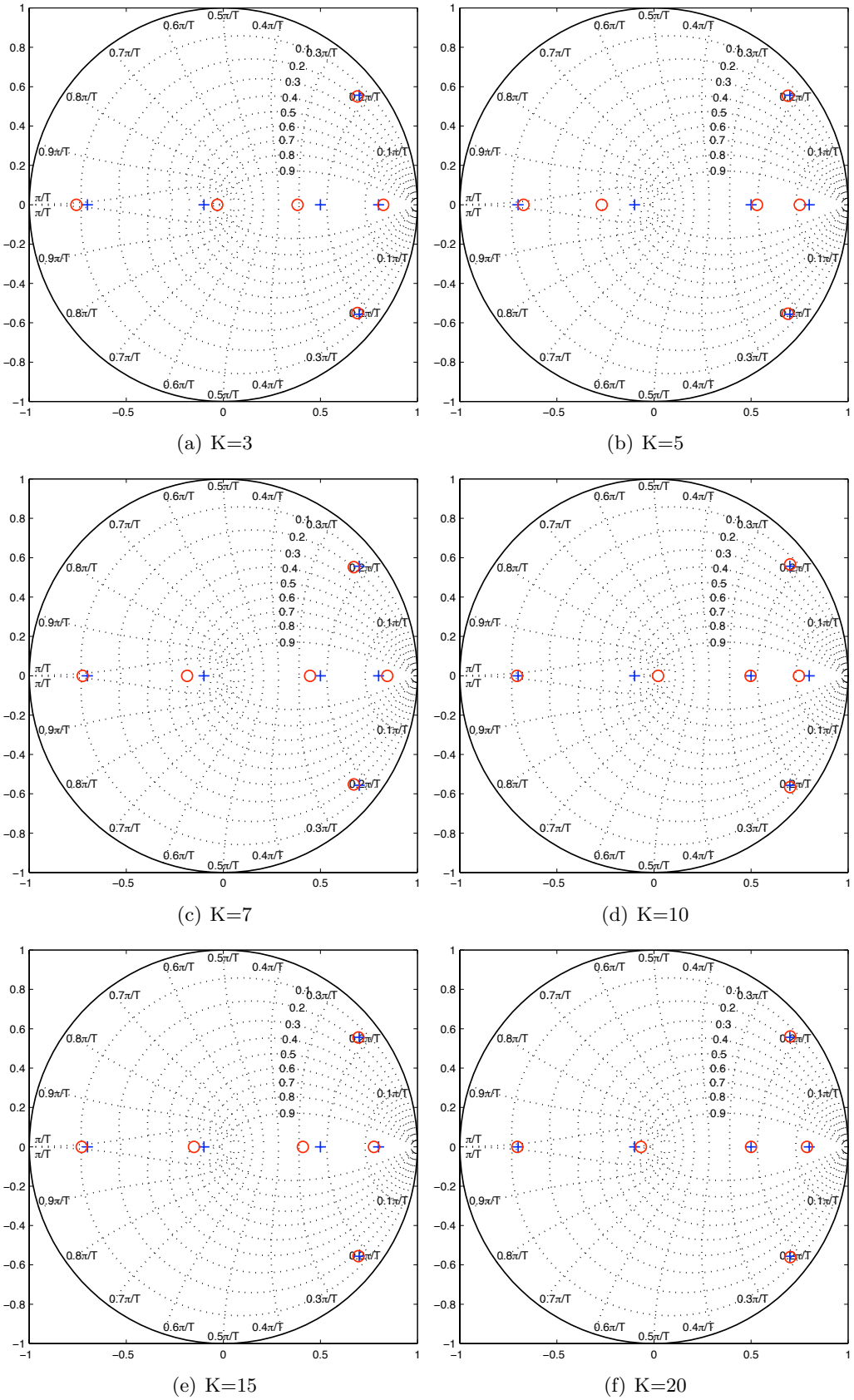
(c) N4SID, SNR = 10 dB

(d) Our method, SNR = 10 dB

(e) N4SID, SNR = 5 dB

(f) Our method, SNR = 5 dB

Figure 3.4: The effect of SNR.



(a) K=3 (b) K=5

(c) K=7 (d) K=10

(e) K=15 (f) K=20

Figure 3.5: The effect of the number of data sets, K denotes the number of data sets.

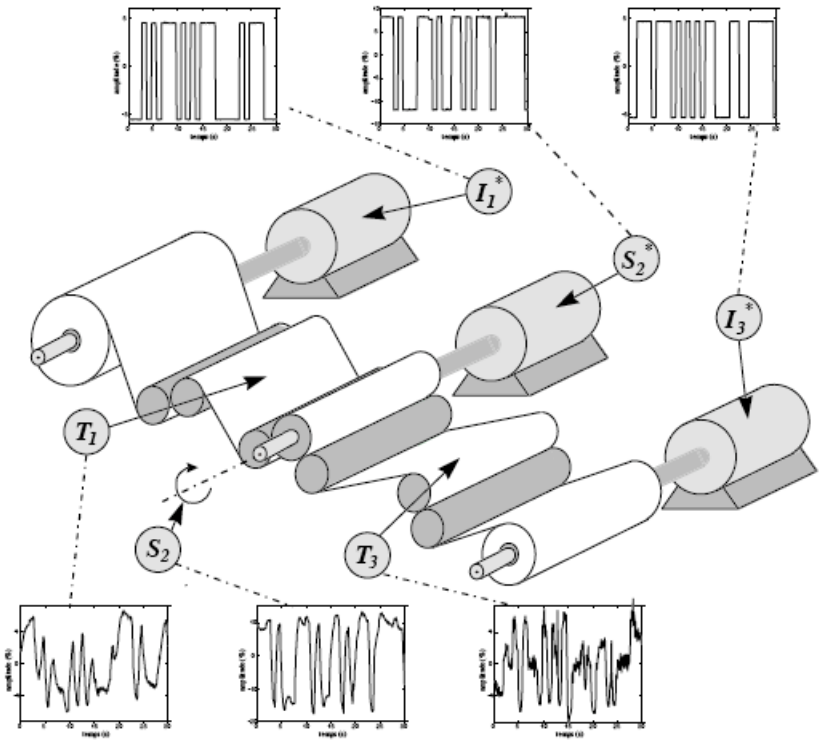


Figure 3.6: A description of the winding system.

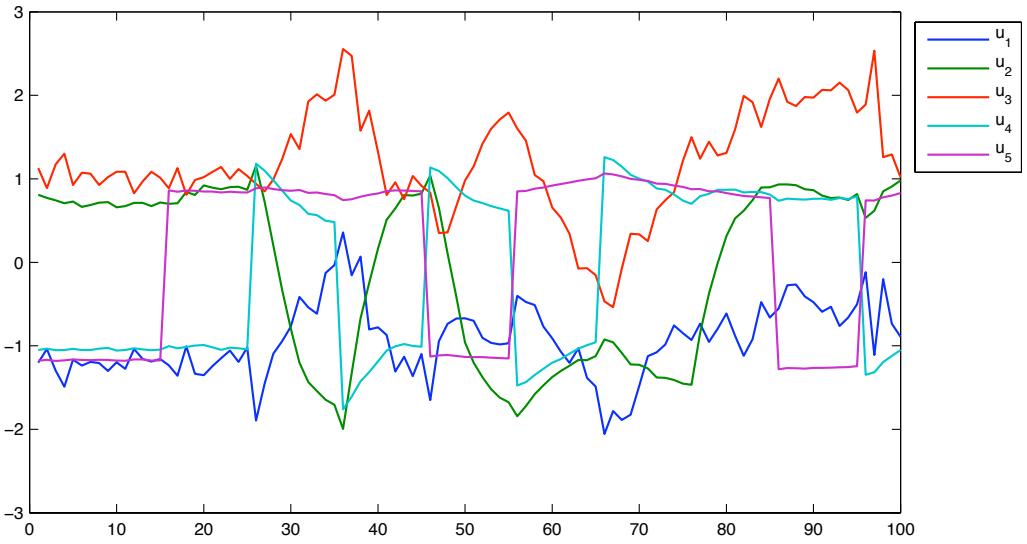
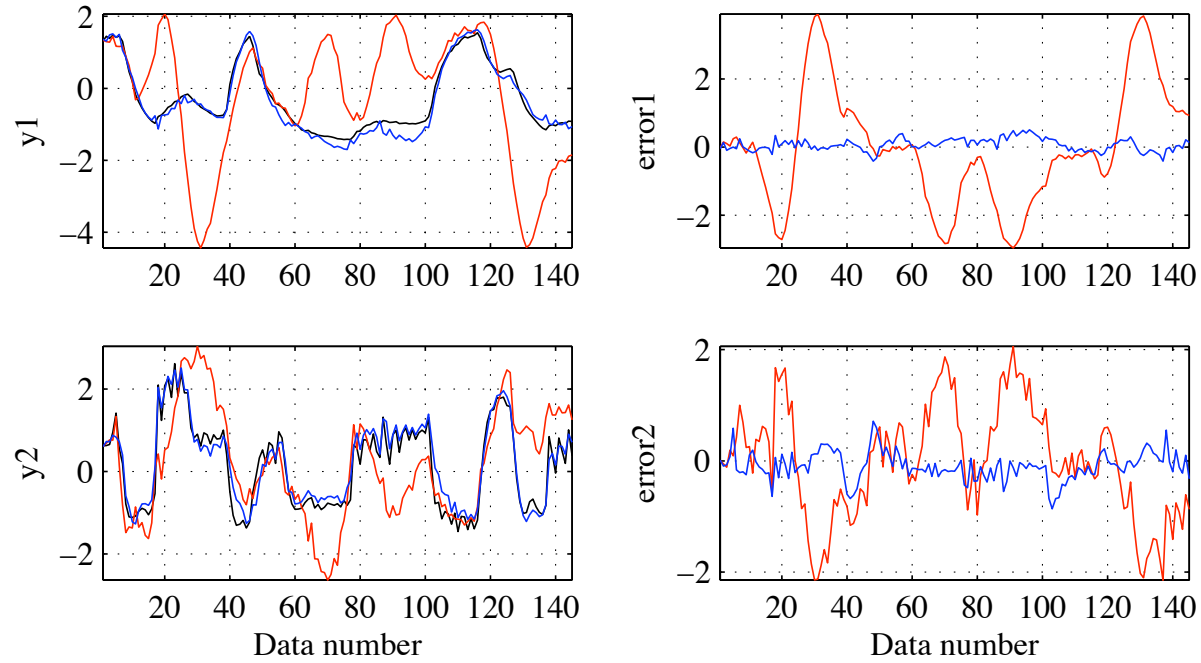
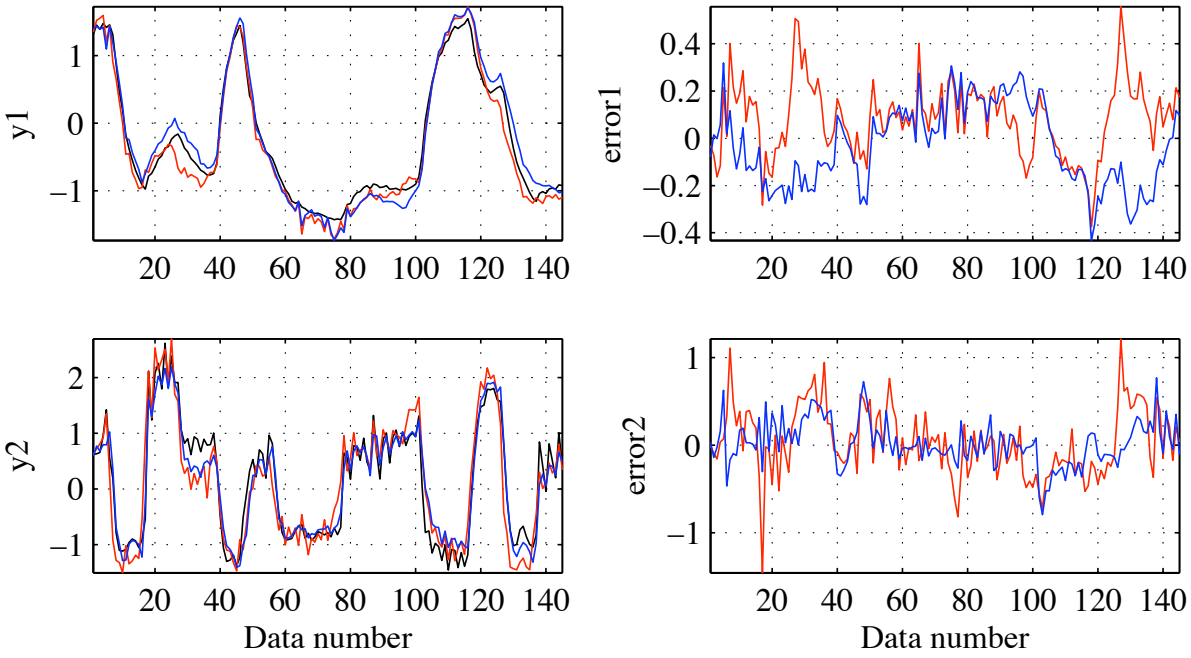


Figure 3.7: The input signals of the industrial winding test setup.



(a) 42 data sets (20 samples ≤ length ≤60 samples).



(b) 21 data sets (35 samples≤ length ≤100 samples).

Figure 3.8: The output of industrial winding process (black line), and the outputs of the linear model obtained by our method (blue line) and N4SID method (red line) are superimposed.

Résumé du chapitre 4
Nouvelle méthode pour l'identification de séries de Volterra d'ordre fini et à horizon infini

L'identification des systèmes linéaires a connu un progrès significatif ces dernières années. De ce fait, on dispose actuellement de plusieurs algorithmes et de méthodes efficaces pour identifier les systèmes linéaires multi-variables [Söderström and Stoica 1989; Moonen et al. 1989; Overschee and Moor 1994; Viberg 1995; Ljung 1999].

L'identification des systèmes non-linéaires est cependant de plus en plus au centre de l'attention des chercheurs, de fait que la plupart des systèmes manifestent en pratique des comportements non-linéiares.

Il est bien connu que la classe des systèmes non-linéaires est une classe très large. Par conséquent, la définition d'un modèle universel pour cette classe est une tâche inaccessible pratiquement. Les chercheurs se sont donc focalisés sur des classes spécifiques des systèmes non-linéaires pour lesquelles ils ont développé des méthodes d'identification appropriées [Billings 1980; Haber and Unbeheauen 1990; Lee 1998; Mathews and Sicuranza 2000; Pearson 1995; 2000; Doyle et al. 2001].

Dans le chapitre 4, nous nous intéressons à l'identification des séries de Volterra d'ordre fini et à horizon infini [Volterra 1930; Brockett 1972; 1976].

Le développement des séries de Volterra d'ordre fini a la forme suivante:

$$y_t = \sum_{n=1}^{l} \sum_{\tau_1=0}^{\infty} \sum_{\tau_2=0}^{\tau_1} \dots \sum_{\tau_n=0}^{\tau_{n-1}} w_n(\tau_1, \tau_2, \dots, \tau_n) \times u_{t-\tau_1} \otimes u_{t-\tau_2} \otimes \dots \otimes u_{t-\tau_n} + v_t$$

où $u_t \in \mathbb{R}^m$ désigne le signal d'entrée, et $y_t \in \mathbb{R}^p$ désigne le signal de sortie. Le bruit des mesures est v_t supposé être un bruit blanc indépendant du signal d'entrée. L'opérateur $\otimes$ désigne le produit de Kronecker.

Les fonctions $w_n(\tau_1, \tau_2, \dots, \tau_n) \in \mathbb{R}^{p \times m^n}$ désignent les noyaux de Volterra. La série de Volterra ci-dessus est appelée une série de Volterra d'ordre l et à horizon infini, car le dernier terme de la série est associé au $l^{\text{ième}}$ noyau de Volterra. Cette classe de séries de Volterra est dense, dans

le sens de L^2, dans la classe des systèmes non-linéaires analytiques par rapport à leurs signaux d'entrée et de sortie.

L'approximation des systèmes non-linéaires par des séries de Volterra s'est révélée utile dans plusieurs domaines de recherches, comme le filtrage [Monin and Salut 1996], la commande prédictive [Allgöwer and Zheng 2000], l'écologie [Takeuchi 1996], en biologie [Thieme 2003] et dans le contrôle des systèmes non-linéaires [Nijmeijer and van der Schaft 1996; Doyle et al. 2001].

Le calcul direct des noyaux de Volterra des séries à horizon infini est infaisable en pratique. La raison en est que les séries de Volterra à horizon infini font appel à l'intégrale du signal d'entrée et qu'en conséquence, les dimensions des matrices manipulées croissent exponentiellement avec l'horizon temporel.

Afin de surmonter cette difficulté, l'horizon des séries de Volterra est souvent artificiellement borné [Rugh 1981; Schetzen 1989; Kibangou et al. 2005]. Ainsi, on obtient la classe des séries de Volterra tronquées d'ordre fini. Elle a la forme suivante:

$$y_t = \sum_{n=1}^{l} \sum_{\tau_1=0}^{T} \sum_{\tau_2=0}^{\tau_1} \dots \sum_{\tau_n=0}^{\tau_{n-1}} w_n(\tau_1, \tau_2, \dots, \tau_n) \times u_{t-\tau_1} \otimes u_{t-\tau_2} \otimes \dots \otimes u_{t-\tau_n} + v_t$$

où T est la dimension de l'horizon. Dans ce cas, l'estimation des noyaux de Volterra devient triviale (par exemple, par les méthodes des moindres carrés).

En revanche, ce problème d'estimation de noyaux devient rapidement un problème mal-conditionné du fait que le nombre de paramètres à estimer augmente exponentiellement en fonction de l et T.

Dans le chapitre 4, nous développons une méthode pour identifier une réalisation sous forme d'un modèle d'état des séries de Volterra d'ordre fini et à horizon infini.

Le modèle d'état est le suivant:

$$\begin{aligned}
Z_t^1 &= A^1 Z_{t-1}^1 + B^1 u_t && , \; Z_0^1 = 0 \\
Z_t^2 &= A^2 Z_{t-1}^2 + B^2 (u_t \otimes Z_t^1) && , \; Z_0^2 = 0 \\
&\vdots \\
Z_t^l &= A^l Z_{t-1}^l + B^l (u_t \otimes Z_t^{l-1}) && , \; Z_0^l = 0 \\
y_t &= \sum_{i=1}^{l} C^i Z_t^i + v_t
\end{aligned}$$

La méthode est basée sur la minimisation de la norme Euclidienne de l'erreur de sortie du modèle ci-dessus. Pour résoudre ce problème d'optimisation, nous utiliserons une méthode de type gradient.

Nous montrons toutefois que la réalisation d'état n'est pas unique. Afin de surmonter le problème de non-unicité du modèle et d'empêcher la méthode de gradient de chercher dans des

directions dans lesquelles la fonction d'objectif ne change pas, nous proposons une méthode de paramétrisation locale de manière à identifier l'espace tangent dans lequel il faut chercher la nouvelle mise à jour des paramètres du modèle.

En outre, nous proposons une méthode de projection (non orthogonale) séquentielle dans le but d'estimer les sous-systèmes homogènes (linéaire, quadratique, cubique, ... etc).

L'algoritme de projection est donc structuré ainsi :

1. Estimation de la meilleure approximation linéaire stable (le sous-système linéaire).
2. A partir de la simulation du sous-système linéaire, nous projetons le résidu sur la classe des sous-systèmes quadratiques.
3. A partir de la simulation du sous-système linéaire + quadratique, nous projetons le résidu sur la classe des sous-systèmes cubique
4. etc.

De fait que ces sous-systèmes ne sont pas orthogonaux, la projection séquentielle présentée ci-dessus n'est pas optimale. En revanche, la solution obtenue par cette projection est une bonne solution initiale pour une méthode d'optimisation de type gradient.

Nous avons comparé les performances de la méthode d'identification proposée avec des méthodes conventionnelles classiques pour l'identification des systèmes non-linéaires à travers deux exemples de ces systèmes. Cette comparaison a révélé l'efficacité de notre méthode. Ces précisions des résultats obtenus par notre méthode ont largement dépassé ceux des méthodes classiques.

De plus, une comparaison avec les séries de Volterra tronquées a montré que non seulement le nombre de paramètres à estimer pour le modèle d'état est largement inférieur à celui des séries de Volterra tronqués, mais qu'aussi la précision du modèle obtenu par notre méthode est considérablement supérieur à celle des séries de Volterra tronquées.

"Empires die, but Euclid's theorems keep their youth forever."

Vito Volterra, 1930

4

New Method for Identifying Finite Degree Volterra Series

The developments of linear system identification methods have recently offered a significantly practical tool to deal with the identification of multivariable linear or pseudo linear system [Söderström and Stoica 1989; Moonen et al. 1989; Overschee and Moor 1994; Viberg 1995; Ljung 1999].

However, in practice, the identification of nonlinear multivariable systems is becoming crucial since many systems in nature are nonlinear. In the theory of nonlinear systems, the term "nonlinear" defines a class of systems for which the linear approximation fails to be an efficient model able to capture the dynamic of the system. The class of nonlinear systems is a complex class, and defining a universal represented model of this class is a very complicated task. Therefore, anyone interested in nonlinear modeling is compelled to focus on specific model classes [Billings 1980; Haber and Unbeheauen 1990; Lee 1998; Mathews and Sicuranza 2000; Pearson 1995; 2000; Doyle et al. 2001].

In this chapter, we focus on the representation of nonlinear system by Volterra series with infinite horizon [Volterra 1930; Brockett 1972; 1976].
The expansion of finite degree Volterra series has the following form

$$y_t = \sum_{n=1}^{l} \sum_{\tau_1=0}^{\infty} \sum_{\tau_2=0}^{\tau_1} \dots \sum_{\tau_n=0}^{\tau_{n-1}} w_n(\tau_1, \tau_2, \dots, \tau_n) \times u_{t-\tau_1} \otimes u_{t-\tau_2} \otimes \dots \otimes u_{t-\tau_n} + v_t \tag{4.1}$$

where $u_t \in \mathbb{R}^m$ is the input signals, and $y_t \in \mathbb{R}^p$ is the output signals. The measurement noise v_t is assumed to be a white-noise that is independent of the input signal and with zero mean, and $\otimes$ denotes the Kronecker product. The functions $w_n(\tau_1, \tau_2, \ldots, \tau_n) \in \mathbb{R}^{p \times m^n}$ denote the Volterra kernels. As the last term in the series involves the l^{th} kernel, they will be called Volterra series of degree l with infinite horizon. This class of Volterra series is dense, in the L^2 sense, in nonlinear analytic input-output systems [Brockett 1976].

Approximating nonlinear systems by Volterra series has been used in many research areas, such as filtering [Monin and Salut 1996], predictive control [Allgöwer and Zheng 2000], ecology [Takeuchi 1996], biology [Thieme 2003], and system control [Nijmeijer and van der Schaft 1996; Doyle et al. 2001].

In reality, the direct calculation of the Volterra kernels from Eq. (4.1) is not feasible in practice. This is because it involves the integral of input signal and the size of the matrices which should be manipulated increases exponentially.

For that, the horizon of finite degree Volterra series is generally fixed [Rugh 1981; Schetzen 1989; Kibangou et al. 2005]. The obtained class is called truncated finite degree Volterra series of degree l.

It has the following form

$$y_t = \sum_{n=1}^{l} \sum_{\tau_1=0}^{T} \sum_{\tau_2=0}^{\tau_1} \cdots \sum_{\tau_n=0}^{\tau_{n-1}} w_n(\tau_1, \tau_2, \ldots, \tau_n) \times u_{t-\tau_1} \otimes u_{t-\tau_2} \otimes \ldots \otimes u_{t-\tau_n} + v_t \tag{4.2}$$

where T is the length of the considered horizon. In this case, the estimation of Volterra kernels becomes trivial and can be obtained by least-square methods.

Lemma 4.1 *If we consider that the Volterra kernels are fully parameterized, then the number of parameters which should be estimated is the following*

$$C(l,T) = p \left\{ \sum_{n=1}^{l} \frac{(T+n)!}{n!T!} m^n \right\} \tag{4.3}$$

where p and m are the dimensions of the output and input signals respectively.

Proof 4.1 *See Appendix A.3.1.*

It is clear that $C(l,T)$ increases exponentially with respect to l and T. For that, in the case of multivariable systems, we are obligated to operate high dimensional matrices, which are often ill-conditioned [Nowak and Veen 1994].

The main contribution in this chapter is developing a new method to identify a state-space realization of finite degree Volterra series with infinite horizon (4.1). The initial estimation of the realization's parameters is obtained by a sequential projection method that we have developed

thanks to the recursive property of the realization structure. Then, the realization parameters are optimized using a local gradient search method.

The chapter is organized as follows: In Section 4.1, a realization of finite degree Volterra series in finite dimension state space representation is defined. In Section 4.2, the output error identification problem is formulated and the structure's parameters are chosen. A local parameterization of the realization of finite degree Volterra series is developed in Section 4.3. In Section 4.4, we propose a sequential projection method to calculate an initial estimation of the structure's parameters. In Section 4.5, we summarize the algorithm of identification. Finally, Section 4.6 presents some illustrative examples and a comparison with some system identification methods for nonlinear systems.

4.1 Realization of Finite Degree Volterra Series

Brokett [Brockett 1976] has proved that any observable realization of finite Volterra series of degree l can be approximated by an recursive realization of the form

$$\begin{aligned} Z_t^1 &= A^1 Z_{t-1}^1 + B^1 u_t \quad , \; Z_0^1 = 0 \\ Z_t^2 &= A^2 Z_{t-1}^2 + B^2 (u_t \otimes Z_t^1) \quad , \; Z_0^2 = 0 \\ &\vdots \\ Z_t^l &= A^l Z_{t-1}^l + B^l (u_t \otimes Z_t^{l-1}) \quad , \; Z_0^l = 0 \\ y_t &= \sum_{i=1}^{l} C^i Z_t^i + v_t \end{aligned} \tag{4.4}$$

The direct relation between y_t and u_t, can be obtained by using the property $FG \otimes HJ = (F \otimes H)(G \otimes J)$ and a simple development of (4.4) leads to

$$y_t = \sum_{i=1}^{l} \sum_{\tau_1=0}^{t-1} \sum_{\tau_2=0}^{\tau_1} \cdots \sum_{\tau_i=0}^{\tau_{i-1}} C^i \Phi^i(\tau_1, \ldots, \tau_i) \times u_{t-\tau_1} \otimes u_{t-\tau_2} \ldots \otimes u_{t-\tau_i} + v_t \tag{4.5}$$

where the transition matrices are defined by

$$\begin{aligned} &\Phi^1(\tau_1) = (A^1)^{\tau_1} B^1 \\ &\text{and for i} \geq 2\text{:} \\ &\Phi^i(\tau_1, \tau_2, \ldots, \tau_i) = (A^i)^{\tau_1} B^i \left[I_m \otimes \Phi^{i-1}(\tau_2, \ldots, \tau_i) \right] \end{aligned} \tag{4.6}$$

By comparing (4.1) and (4.5), we observe that Realization (4.4) leads to Volterra series similar to (4.1), in which we suppose that $u_t = 0$ for $t \leq 0$, and the Volterra kernels are approximated by a sum of matrix products.

In fact, this approximation depends on the dimensions of the states $\{Z_t^i : i = 1, 2, \cdots, l\}$, and

it is clear that it becomes better if we increase the dimensions of states. On the other hand, for practical purposes these dimensions should be chosen as lower as possible in order to find a compromise between the approximation and numerical complexity issues.
Obviously, Realization (4.4) is not unique, assume that the states of Structure (4.4) transform into $X_t^i = (T^i)^{-1} Z_t^i$ where $T^i \in \mathbb{R}^{n_i \times n_i}$ are nonsingular matrices, and n_i is the dimension of the state Z_t^i. The structure becomes

$$\begin{aligned} X_t^1 &= \bar{A}^1 X_{t-1}^1 + \bar{B}^1 u_t \\ X_t^2 &= \bar{A}^2 X_{t-1}^2 + \bar{B}^2 (u_t \otimes X_t^1) \\ &\vdots \\ X_t^l &= \bar{A}^l X_{t-1}^l + \bar{B}^l (u_t \otimes X_t^{l-1}) \\ y_t &= \sum_{i=1}^{l} \bar{C}^i X_t^i + v_t \end{aligned}$$

where

$$\begin{bmatrix} \bar{A}^i & \bar{B}^i \\ \bar{C}^i & 0 \end{bmatrix} = \begin{bmatrix} (T^i)^{-1} A^i T^i & (T^i)^{-1} B^i T_m^{i-1} \\ C^i T^i & 0 \end{bmatrix} \tag{4.7}$$

and $\{T_m^j : j = 0, 1, 2, \cdots, l-1\}$ are defined as follows

$$T_m^j = I_m \otimes T^j, \;\; T_m^0 = I_m : I_m \text{ is the identity matrix}$$

Lemma 4.2 *The realization of finite degree Volterra series (4.4) is asymptotically stable if and only if:*

$$\text{For } i = 1, 2, \cdots, l : \rho(A^i) < 1 \tag{4.8}$$

where $\rho(A^i)$ stands for the spectral radius of A^i.

Proof 4.2 *See Appendix A.3.2.*

4.2 Output Error Identification

Our goal is to determine the coefficient matrices of Structure (4.4). Assume that all matrices are fully parameterized, so the vector of parameters of Structure (4.4) can be given by

$$\theta = \begin{bmatrix} vec(A^1) \\ vec(B^1) \\ vec(C^1) \\ \vdots \\ vec(A^l) \\ vec(B^l) \\ vec(C^l) \end{bmatrix} \tag{4.9}$$

recall that vec(.) denotes the vectorization operator defined as follows

$$\begin{aligned} &vec : M \in \mathbb{R}_{m\times n} \to \mathbb{R}_{m\cdot n} \\ &vec(M) = vec\begin{bmatrix} m_1 & m_2 \cdots m_n \end{bmatrix} = \begin{bmatrix} m_1^T & m_2^T \cdots m_n^T \end{bmatrix}^T \end{aligned}$$

Given the input u_t and output y_t of the real system, the model corresponding to θ can be given as follows

$$\begin{aligned} \hat{Z}_t^1 &= A^1(\theta)\hat{Z}_{t-1}^1 + B^1(\theta)u_t \\ \hat{Z}_t^2 &= A^2(\theta)\hat{Z}_{t-1}^2 + B^2(\theta)(u_t \otimes \hat{Z}_t^1) \\ &\vdots \\ \hat{Z}_t^l &= A^l(\theta)\hat{Z}_{t-1}^l + B^l(\theta)(u_t \otimes \hat{Z}_t^{l-1}) \\ \hat{y}_t(\theta) &= \sum_{i=1}^{l} C^i(\theta)\hat{Z}_t^i \end{aligned} \tag{4.10}$$

Note that as Z_t^i depends on θ, the mapping $\hat{y}_t(\theta)$ is nonlinear with θ. Our goal is achieved if the output $\hat{y}_t(\theta)$ approximates the output of the real system accurately. This criterion can be transformed into the minimization of the output error with respect to the parameters θ. Considering a data of length N, the output-error cost function is given by

$$J_N(\theta) = \frac{1}{N}\sum_{k=1}^{N} \|y_k - \hat{y}_k(\theta)\|_2^2 = \frac{1}{N}\|E_N(\theta)\|^2 \tag{4.11}$$

where

$$E_N(\theta) = \begin{bmatrix} e_1(\theta)^T & e_2(\theta)^T \cdots e_N(\theta)^T \end{bmatrix}^T \tag{4.12}$$

is the error vector in which $e_k(\theta) = y_k - \hat{y}_k(\theta)$. The minimization of (4.11) is clearly a nonlinear, nonconvex optimization problem.

The numerical solution of this problem can be calculated by gradient search method

(Levenberg-Marquard method for instance).
This iterative method is based on the updating of the system parameters θ as follows

$$\hat{\theta}^{i+1} = \hat{\theta}^i - (\psi_N^T(\hat{\theta}^i)\psi_N(\hat{\theta}^i) + \lambda^i I)^{-1}\psi_N^T(\hat{\theta}^i)E_N(\hat{\theta}^i) \tag{4.13}$$

Where λ^i is the regularization parameter and

$$\psi_N(\theta) \triangleq \frac{\partial E_N(\theta)}{\partial \theta^T} \tag{4.14}$$

is the jacobian of the error vector $E_N(\theta)$.

As we mentioned in Section 4.1, Structure (4.4) is not unique. As a consequence, the minimization of $J_N(\theta)$ does not have a unique solution. The nonuniqueness solution of θ is the consequence of the full parameterization of the matrices of the realization.
However, the optimal solution can be made unique by choosing a suitable canonical parameterization. By observing that each subsystem of the realization is a linear system, one could use a classical parameterization of the various parameterization proposed for the linear systems.

Unfortunately, these parameterizations are not numerically robust [McKelvey and Helmersson 1997]. To overcome the nonuniqueness problem of the optimal θ and keep the full parameterization of the matrices, Ribarits *et al* [Ribarits et al. 2004] have proposed a method for the linear systems, in which the directions that do not change the cost of output error function are identified and projected out at each iteration.

For that only the active parameters are updated. In analogues way, we will define a local parameterization of the realization of finite degree Volterra series (4.4) in order to define the directions in which the cost function $J_N(\theta)$ does not change.

4.3 Local Parameterization

Recall that two realizations of finite degree Volterra series (4.4) are similar if their coefficient matrices are related by Equation (4.7), where the transformation matrices $\{T^i : i = 1, 2, \cdots, l\}$ parameterize the subset of equivalent models. The obtained subset defines a manifold.

Let us define the *similarity map* S_θ as follows

$$S_\theta : \left\{T^1, T^2, \cdots, T^l\right\} \in \left\{\mathbb{R}^{n_1 \times n_1}, \mathbb{R}^{n_2 \times n_2}, \cdots, \mathbb{R}^{n_l \times n_l}\right\}, det(T^i) \neq 0 \rightarrow \mathbb{R}^{\Sigma_{k=1}^l n_k^2}$$

$$S_\theta(T^1,T^2,\cdots,T^l)=\begin{bmatrix} vec\left(\left(T^1\right)^{-1}A^1T^1\right)\\ vec\left(\left(T^1\right)^{-1}B^1\right)\\ vec\left(C^1T^1\right)\\ \vdots\\ vec\left(\left(T^l\right)^{-1}A^lT^l\right)\\ vec\left(\left(T^l\right)^{-1}B^lT_m^{l-1}\right)\\ vec\left(C^lT^l\right)\end{bmatrix} \tag{4.15}$$

Two models will be called similar if they lead to the same input/output map.

Let us define the set

$$\mathfrak{I}_\theta=\left\{\bar{\theta}|\bar{\theta}=S_\theta(T^1,T^2,\cdots,T^l),det(T^i)\neq 0\right\} \tag{4.16}$$

This equivalence class is called the *indistinguishable set at* θ, because it contains all the models that cannot be distinguished from θ by looking at their input-output behavior.

In order to identify the tangent plane of this manifold at $(A^i,B^i,C^i:i=1,2,\cdots,l)$, we linearize Relation (4.7) around the identity matrices.

Considering a small perturbation $T^i=I_{n_i}+\Delta T^i$, we suppose that $\rho(\Delta T^i)\ll 1$ then by using the approximation $\left(I_{n_i}+\Delta T^i\right)^{-1}\simeq I_{n_i}-\Delta T^i$ and neglecting all second order terms, we obtain

$$\begin{aligned}\begin{bmatrix}\bar{A}^i & \bar{B}^i\\ \bar{C}^i & 0\end{bmatrix}=&\begin{bmatrix}A^i & B^i\\ C^i & 0\end{bmatrix}+\begin{bmatrix}-\Delta T^iA^i+A^i\Delta T^i & -\Delta T^iB^i\\ C^i\Delta T^i & 0\end{bmatrix}\\ &+\begin{bmatrix}0 & B^i\Delta T_m^{i-1}\\ 0 & 0\end{bmatrix}\end{aligned} \tag{4.17}$$

By defining

$$\Lambda_i^j\triangleq\begin{bmatrix}0_{n_j\times(i-1)n_j} & I_{n_j} & 0_{n_j\times(m-i)n_j}\end{bmatrix}$$

it is possible to write ΔT_m^j as follows

$$\Delta T_m^j=\sum_{i=1}^{m}{\Lambda_i^j}^T\Delta T^j\Lambda_i^j$$

If we consider the following vectors of parameters

$$\theta = \begin{bmatrix} vec(A^1) \\ vec(B^1) \\ vec(C^1) \\ \vdots \\ vec(A^l) \\ vec(B^l) \\ vec(C^l) \end{bmatrix} \quad and \quad \bar{\theta} = \begin{bmatrix} vec(\bar{A}^1) \\ vec(\bar{B}^1) \\ vec(\bar{C}^1) \\ \vdots \\ vec(\bar{A}^l) \\ vec(\bar{B}^l) \\ vec(\bar{C}^l) \end{bmatrix} \tag{4.18}$$

the relation between θ and $\bar{\theta}$ can be obtained using the property $vec(ABC) = \left(C^T \otimes A\right) vec(B)$

$$\bar{\theta} = \theta + M_\theta \begin{bmatrix} vec(\Delta T^1) \\ \vdots \\ vec(\Delta T^l) \end{bmatrix} \tag{4.19}$$

where for $1 \leq j \leq l-1$

$$M_\theta \left(:, 1 + \sum_{k=1}^{j-1} n_k^2 : \sum_{k=1}^{j} n_k^2 \right) = \begin{bmatrix} \mathbf{0}_{\alpha_j \times n_j^2} \\ -\left(A^j\right)^T \otimes I_{n_j} + I_{n_j} \otimes A^j \\ -\left(B^j\right)^T \otimes I_{n_j} \\ I_{n_j} \otimes C^j \\ \mathbf{0}_{n_{j+1}^2 \times n_j^2} \\ \sum_{i=1}^{m} {\Lambda_i^j}^T \otimes \left(B^{j+1} {\Lambda_i^j}^T \right) \\ \mathbf{0}_{\gamma_j \times n_j^2} \end{bmatrix}$$

and

$$M_\theta \left(:, 1 + \sum_{k=1}^{l-1} n_k^2 : \sum_{k=1}^{l} n_k^2 \right) = \begin{bmatrix} \mathbf{0}_{\alpha_l \times n_l^2} \\ -\left(A^l\right)^T \otimes I_{n_l} + I_{n_l} \otimes A^l \\ -\left(B^l\right)^T \otimes I_{n_l} \\ I_{n_l} \otimes C^l \end{bmatrix} \tag{4.20}$$

with

$$\begin{aligned} \alpha_j = & \quad \textstyle\sum_{i=1}^{j-1} n_i (n_i + m\, n_{i-1} + p) \\ \gamma_j = & \quad p\, n_{j+1} + \textstyle\sum_{i=j+2}^{l} n_i (n_i + m\, n_{i-1} + p) \end{aligned}$$

Lemma 4.3 *The left null space of M_θ (4.20) defines a basis of the directions in which the parameters should be modified to lead a change in the value of cost function $J_N(\theta)$.*

Proof 4.3 *See Appendix A.3.3.*

Let the QR decomposition of M_θ be given by

$$M_\theta = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix} \tag{4.21}$$

then a basis of the null space of M_θ is Q_2.

Thus, the update rule should be modified such that we project out the directions in which the cost function does not change. The new update rule becomes[1]

$$\hat{\theta}^i = \hat{\theta}^{i-1} - Q_2 \left(Q_2^T \psi_N^T \psi_N Q_2 + \lambda^i I \right)^{-1} Q_2^T \psi_N^T E_N \tag{4.22}$$

where Q_2, ψ_N and E_N depend on $\hat{\theta}^{i-1}$.
Since Q_2 depends on the past parameter $\hat{\theta}^{i-1}$, the QR decomposition (4.21) must be computed at each iteration.

4.3.1 Computing the Iterative Parameter Update

In order to compute the update rule (4.22), the quantities $E_N(\hat{\theta})$ and $\psi_N(\hat{\theta})$ should be computed. Computing the vector $E_N(\hat{\theta})$ can be done by simulating the system (4.10) with $\hat{\theta} = \hat{\theta}^{i-1}$. At the same time, this simulation brings out the states $\{\hat{Z}_t^i : i = 1, 2, \cdots, l\}$.
As it is mentioned in Section 3.2.1 of Chapter 3, computing the gradient $\psi_N(\theta)$ requires the computation of the derivative of $\hat{y}_t$ with respect to θ.

Let us define

$$\zeta_{t,k}^j = \frac{\partial \hat{Z}_t^j}{\partial \theta_k} \tag{4.23}$$

Then, the computation of $\frac{\partial \hat{y}_t}{\partial \theta^T} = \begin{bmatrix} \frac{\partial \hat{y}_t}{\partial \theta_1} & \cdots & \frac{\partial \hat{y}_t}{\partial \theta_q} \end{bmatrix}$, where q is the number of parameters in θ, can be made as follows

$$\begin{aligned}
\zeta_{t,k}^1 =& A^1 \zeta_{t-1,k}^1 + \frac{\partial A^1}{\partial \theta_k} \hat{Z}_{t-1}^1 + \frac{\partial B^1}{\partial \theta_k} u_t \\
\zeta_{t,k}^2 =& A^2 \zeta_{t-1,k}^2 + \frac{\partial A^2}{\partial \theta_k} \hat{Z}_{t-1}^2 + \frac{\partial B^2}{\partial \theta_k} (u_t \otimes \hat{Z}_t^1) + B^2 (u_t \otimes \zeta_{t,k}^1) \\
& \vdots \\
\zeta_{t,k}^l =& A^l \zeta_{t-1,k}^l + \frac{\partial A^l}{\partial \theta_k} \hat{Z}_{t-1}^l + \frac{\partial B^l}{\partial \theta_k} (u_t \otimes \hat{Z}_t^{l-1}) + B^l (u_t \otimes \zeta_{t,k}^{l-1}) \\
\frac{\partial \hat{y}_t}{\partial \theta_k} =& \sum_{j=1}^{l} C^j \zeta_{t,k}^j + \frac{\partial C^j}{\partial \theta_k} \hat{Z}_t^j
\end{aligned} \tag{4.24}$$

In fact, if we consider the local parameterization of finite degree Volterra realization, then computing the update rule (4.22) can be done without calculating all the elements of matrix

[1] See Appendix A.3.4 for more details.

ψ_N. Indeed, let us define

$$\Omega_N \triangleq \psi_N Q_2 = \frac{\partial E_N}{\partial \theta^T} Q_2 \tag{4.25}$$

The update rule can be expressed as follows

$$\hat{\theta}^i = \hat{\theta}^{i-1} - Q_2 \left(\Omega_N^T \Omega_N + \lambda^i I \right)^{-1} \Omega_N^T E_N \tag{4.26}$$

Consider the tp elements of the s^{th} column of Ω_N

$$\Omega_N((t-1)p+1 : tp, s) = \sum_{k=1}^{q} \frac{\partial \hat{y}_t}{\partial \theta_k} Q_2(k,s) \tag{4.27}$$

In order to calculate this sum, let us define

$$\zeta_t^j = \sum_{k=1}^{q} \frac{\partial \hat{Z}_t^j}{\partial \theta_k} Q_2(k,s) \tag{4.28}$$

Using (4.24), the computation of (4.27) can be done as follows

$$\begin{aligned}
\zeta_t^1 &= A^1 \zeta_{t-1}^1 + \Delta A^1 \hat{Z}_{t-1}^1 + \Delta B^1 u_t \\
\zeta_t^2 &= A^2 \zeta_{t-1}^2 + \Delta A^2 \hat{Z}_{t-1}^2 + \Delta B^2 (u_t \otimes \hat{Z}_t^1) + B^2 (u_t \otimes \zeta_t^1) \\
&\vdots \\
\zeta_t^l &= A^l \zeta_{t-1}^l + \Delta A^l \hat{Z}_{t-1}^l + \Delta B^l (u_t \otimes \hat{Z}_t^{l-1}) + B^l (u_t \otimes \zeta_t^{l-1}) \\
\sum_{k=1}^{q} \frac{\partial \hat{y}_t}{\partial \theta_k} Q_2(k,s) &= \sum_{j=1}^{l} C^j \zeta_t^j + \Delta C^j \hat{Z}_t^j
\end{aligned} \tag{4.29}$$

where

$$\Delta A^j = \sum_{k=1}^{q} \frac{\partial A^j}{\partial \theta_k} Q_2(k,s) \quad : j = 1, 2, \cdots, l \tag{4.30}$$

and $\Delta B^j, \Delta C^j$ are defined analogously. These matrices can be obtained from

$$\begin{aligned}
\begin{bmatrix} vec(\Delta A^1) \\ vec(\Delta B^1) \\ vec(\Delta C^1) \\ \vdots \\ vec(\Delta A^l) \\ vec(\Delta B^l) \\ vec(\Delta C^l) \end{bmatrix} &= \sum_{k=1}^{q} \frac{\partial \theta}{\partial \theta_k} Q_2(k,s) \\
&= \sum_{k=1}^{q} e^k Q_2(k,s)
\end{aligned} \tag{4.31}$$

where

$$e^k = \begin{bmatrix} 0 & \dots & 0 & \underset{\substack{\uparrow \\ k}}{1} & 0 & \dots & 0 \end{bmatrix}^T$$

Note that the number of columns of Ω_N is smaller than that of ψ_N because the first one is the number of active parameters and the second one is the total number of parameters.

4.4 Computing an Initial Estimation

The reason why a good initial estimation is crucial is that the local gradient search method converges to the nearest local optimum in the neighborhood of initial guess of solution. Moreover, the initial estimation should also provide a stable realization that verifies the constraint (4.8). In this section, we propose a sequential projection method in order to provide an initial good estimation of the vector of parameters (θ).

The basic idea is to use a decomposition of finite degree Volterra series realization as sum of homogeneous systems (linear, quadratic, cubic, ... etc). Unfortunately, the homogeneous subprocesses are not orthogonal and a sequential projection over each subprocess leads to error. Nevertheless, this approximation may give a good initial guess of parameters. .

The algorithm of projection can be resumed as follows

1. Estimate the best linear approximation (the linear subsystem) of the nonlinear system.
2. From the simulation of the linear subsystem, we project the residual on the class of 2-degree subsystems.
3. From the simulation of linear + 2-degree subsystems, we project the residual on the class of 3-degree subsystem.
4. Etc.

Moreover, this projection procedure yields an estimation of the dimensions of the states $\{Z_t^i : i = 1,2,\cdots,l\}$, which means the set $\{n_i : i = 1,2,\cdots,l\}$. Furthermore, it yields the degree of the realization (l) by defining a precision criterion as it will appear in the sequel.

4.4.1 Identification of Linear Subsystem

Estimating the best linear approximation of nonlinear system can be formulated as a minimization problem

$$\begin{aligned} & min\ \frac{1}{N}\sum_{t=1}^{N} \parallel y_t - y_t^1 \parallel_2^2 \\ & \text{where} \\ & Z_t^1 = A^1 Z_{t-1}^1 + B^1 u_t \\ & y_t^1 = C^1 Z_t^1 \end{aligned} \tag{4.32}$$

The parameters of the optimization problem, which should be estimated, are the triple (A^1, B^1, C^1). The above minimization problem can be transformed into the following matricial form [Gopinath 1969; DeMoor et al. 1988]

$$\min_{\Gamma_\alpha^1, \Phi_\alpha^1} \quad \parallel Y_{\alpha,N} - \Gamma_\alpha^1 Z_{0,N-\alpha}^1 - \Phi_\alpha^1 U_{\alpha,N} \parallel_F^2 \tag{4.33}$$

where $\parallel . \parallel_F$ denotes Frobenius norm. The input and output are stocked in Hankel matrices form

$$U_{\alpha,N} \triangleq \begin{pmatrix} u_1 & u_2 & \cdots & u_{N-\alpha+1} \\ u_2 & u_3 & \cdots & u_{N-\alpha+2} \\ \vdots & \vdots & \ddots & \vdots \\ u_\alpha & u_{\alpha+1} & \cdots & u_N \end{pmatrix}$$

where α and N refer to the number of rows in the matrix and data length respectively. The output Hankel matrix $Y_{\alpha,N}$ is defined analogously to $U_{\alpha,N}$. The matrices Γ_α^1, Φ_α^1 and $Z_{0,N-\alpha}^1$ are defined as follows

$$\Gamma_\alpha^1 = \begin{bmatrix} C^1 \\ C^1 A^1 \\ \vdots \\ C^1 (A^1)^{(\alpha-1)} \end{bmatrix}$$

$$\Phi_\alpha^1 = \begin{bmatrix} C^1 B^1 & 0 & 0 & \cdots & 0 \\ C^1 A^1 B^1 & C^1 B^1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ C^1 (A^1)^{(\alpha-1)} B^1 & \cdots & \cdots & \cdots & C^1 B^1 \end{bmatrix}$$

$$Z_{0,N-\alpha}^1 = \begin{bmatrix} Z_0^1 & Z_1^1 & \cdots Z_{N-\alpha}^1 \end{bmatrix}.$$

Note that the number of rows (α) should be roughly chosen to be greater than the expected linear system order n_1. This condition guarantees that the extended matrix of observability Γ_α^1 has a full rank. Our objective is to estimate the matrices Γ_α^1 and Φ_α^1.

Since only the data matrices $U_{\alpha,N}$ and $Y_{\alpha,N}$ are known, and instead of solving the

minimization problem (4.33) with respect to Γ_α^1 and Φ_α^1, one can first start by estimating the matrix Φ_α^1 by solving the following optimization problem

$$\min_{\Phi_\alpha} \quad \| Y_{\alpha,N} - \Phi_\alpha^1 U_{\alpha,N} \|_F^2 \tag{4.34}$$

Solving the above equation leads to

$$Y_{\alpha,N} - \hat{\Phi}_\alpha U_{\alpha,N} = Y_{\alpha,N} \Pi_{U_{\alpha,N}^T}^{\perp} \tag{4.35}$$

where $\Pi_{U_{\alpha,N}^T}^{\perp}$ is the orthogonal projection onto the nullspace of $U_{\alpha,N}$

$$\Pi_{U_{\alpha,N}^T}^{\perp} = I - U_{\alpha,N}^T \left(U_{\alpha,N} U_{\alpha,N}^T \right)^{-1} U_{\alpha,N} \tag{4.36}$$

such that

$$U_{\alpha,N} \Pi_{U_{\alpha,N}^T}^{\perp} = \mathbf{0} \tag{4.37}$$

The inverse of the matrix $\left(U_{\alpha,N} U_{\alpha,N}^T \right)$ exists if the input is persistently exciting, and $N > m\alpha$.

By replacing Φ_α by its estimation using (4.35) in (4.33), an estimation of the matrix Γ_α^1 can be obtained by solving the following minimization problem

$$\min_{\Gamma_\alpha^1} \quad \| Y_{\alpha,N} \Pi_{U_{\alpha,N}^T}^{\perp} - \Gamma_\alpha^1 Z_{0,N-\alpha}^1 \|_F^2 \tag{4.38}$$

In fact, the origin of the idea of subtracting the term that involves the input signal belongs to the direct 4SID method, which is a classical subspace method [DeMoor et al. 1988] (for more details refer to chapter 2).

To solve the problem (4.38), one could use the Singular Value Decomposition (SVD). Recall that, by hypothesis, the extended observability matrix (Γ_α^1) is a full rank matrix. Consider the following SVD

$$Y_{\alpha,N} \Pi_{U_{\alpha,N}^T}^{\perp} = \begin{bmatrix} \mathcal{Q}_s & \mathcal{Q}_n \end{bmatrix} \begin{bmatrix} \mathcal{S}_s & 0 \\ 0 & \mathcal{S}_n \end{bmatrix} \begin{bmatrix} \mathcal{V}_s^T \\ \mathcal{V}_n^T \end{bmatrix} \tag{4.39}$$

where the matrix $\mathcal{S}_s$ contains the principals singular values (further than threshold).
Note that the dimension of this matrix yields n_1; the dimension of Z_t^1.

Then, the matrix $\hat{\Gamma}_\alpha^1$ can be obtained by the following formula

$$\hat{\Gamma}_\alpha^1 = \mathcal{Q}_s \tag{4.40}$$

Although $\hat{A}^1$ can be obtained directly from the matrix $\hat{\Gamma}_\alpha^1$, the matrix $\hat{A}^1$ might be unstable. However, a stable $\hat{A}^1$ can be obtained by solving the following minimization problem

$$\begin{aligned} &\min_{\hat{A}^1} \quad \| \hat{\Gamma}^1_\alpha - \mathcal{Q}_s \|_F^2 \\ &\text{subject to} \\ &\rho\left(\hat{A}^1\right) < 1 \end{aligned} \tag{4.41}$$

The above problem can be converted[2] to an optimization problem that involves minimizing a linear function over symmetric cones. To solve this optimization problem, one might use a semidefinite solver, e.g. SeDuMi [Sturm 1999] which is a MATLAB™ package.

Solving the minimization problem (4.41) provides a stable matrix $\hat{A}^1$. The matrix C^1 can be estimated using the following formula

$$C^1 = \left(\sum_{k=0}^{\alpha-1} \mathcal{Q}_s \left(kp+1 : (k+1)p, :\right) \right) \Lambda_{A^1}^T \left(\Lambda_{A^1} \Lambda_{A^1}^T \right)^{-1} \tag{4.42}$$

where $\Lambda_{A^1} \triangleq I_{n_1} + \sum_{i=1}^{\alpha-1} \left(A^1\right)^i$.
As A^1 verifies that $\rho(A) < 1$, Λ_{A^1} is a full rank matrix its inverse exists.

The matrix Φ^1_α can be estimated by considering the least-squares solution to the overdetermined system of equations

$$\mathcal{Q}_n^T Y_{\alpha,N} U_{\alpha,N}^T \left(U_{\alpha,N} U_{\alpha,N}^T \right)^{-1} = \mathcal{Q}_n^T \hat{\Phi}^1_\alpha \tag{4.43}$$

Finally, the matrix B^1 can be easily calculated from the estimation of $\hat{\Phi}^1_\alpha$.

4.4.2 Identification of Higher Order Subsystems

After using the procedure described in the previous section, an estimation of (A^1, B^1, C^1) is available. Calculating the state Z^1_t and the output y^1_t for $t = 1, .., N$ of the linear subsystem can be done by evaluating the following model

$$\begin{aligned} Z^1_t &= A^1 Z^1_{t-1} + B^1 u_t \\ y^1_t &= C^1 Z^1_t \end{aligned}$$

The next task is to project the residual on the class of 2-degree subsystem. We define the residual as follows

$$\tilde{y}_t = y_t - y^1_t \tag{4.44}$$

[2] See Appendix A.3.5 for more details.

The estimation of the best 2-degree subsystem can be done by solving the following optimization problem

$$\begin{aligned} &min \quad \frac{1}{N}\sum_{t=1}^{N} \parallel \tilde{y}_t - y_t^2 \parallel_2^2 \\ &\text{where} \\ &Z_t^2 = A^2 Z_{t-1}^2 + B^2 \left(u_t \otimes Z_t^1\right) \\ &y_t^2 = C^2 Z_t^2 \end{aligned} \tag{4.45}$$

As $u_t \otimes Z_t^1$ is computed thanks to the previous step, this minimization problem is similar to (4.32). By using an analogous logic, we obtain an estimation of the matrices A^2, B^2, C^2.

Then, this procedure is reiterated until we get an appreciated precision. Defining this precision criterion is the objective of the following section.

4.4.3 Determining the Realization Degree

Consider that the subsystems of order inferior to j are available. In order to verify that the nonlinear system is accurately fitted by the finite degree Volterra series realization, we evaluate the following criterion

$$\begin{aligned} &\hat{y}_t = \sum_{i=1}^{j} y_t^i \\ &\left(1 - \frac{\sum_{t=1}^{N} \left(y_t - \hat{y}_t\right)^T \left(y_t - \hat{y}_t\right)}{\sum_{t=1}^{N} \left(y_t - \bar{y}\right)^T \left(y_t - \bar{y}\right)}\right) \times 100 \geq \eta \end{aligned} \tag{4.46}$$

where η is a user-defined constant and $\bar{y}$ denotes the mean of the output signals defined as follows

$$\bar{y} = \frac{1}{N}\sum_{t=1}^{N} y_t \tag{4.47}$$

If the criterion (4.46) is true, the algorithm stops and the realization's degree is j. Otherwise, we estimate the subsystem of order $j+1$.
Note that the measurement noise v_t should be considered when we define η as follows

$$0 < \eta < \left(1 - \frac{\sum_{t=1}^{N} v_t^T v_t}{\sum_{t=1}^{N} \left(y_t - \bar{y}\right)^T \left(y_t - \bar{y}\right)}\right) \times 100 \tag{4.48}$$

The right hand side of the above inequality corresponds to the ideal case of perfect identification $(y_t - \hat{y}_t = v_t)$.

4.5 Identification Algorithm

We can resume the algorithm of identification as follows

1. Calculate an initial estimation of the realization's parameters using the sequential projection described in Section 4.4. Recall that this procedure yields the realization degree (l), the dimensions of the states Z_t^i $\{n_i : i = 1,2,\cdots,l\}$, and the matrices $\{A^i, B^i, C^i : i = 1,2,\cdots,l\}$. The estimated vector of parameters $\hat{\theta}^0$ is used as an initial guess for the optimization process and $k \leftarrow 0$.
2. Calculate the states $\{Z_t^i : i = 1,2,\cdots,l\}$ and $\hat{y}_t$ by simulating the system (4.10) with $\theta = \hat{\theta}^k$.
3. Compute $E_N(\hat{\theta}^k)$ using (4.12).
4. Calculate the matrix $M_{\hat{\theta}^k}$ using (4.20).
5. Calculate the QR decomposition of $M_{\hat{\theta}}$ (4.21), from which we obtain Q_2.
6. Calculate $\Delta A^j, \Delta B^j$ and ΔC^j $: j = 1,2,\cdots,l$ using (4.31).
7. Calculate the matrix Ω_N using (4.27) and (4.29), we suppose that $\{\zeta_0^j = \mathbf{0} : j = 1,2,\cdots,l\}$.
8. Calculate the update rule of the gradient search algorithm using (4.26) and $k \leftarrow k+1$.
9. Perform the termination test for minimization, If true, the algorithm stops. Otherwise, return to step (2), i.e. compute the values $J_N\left(\hat{\theta}^{k-1}\right)$ and $J_N\left(\hat{\theta}^k\right)$ using (4.11) and test if $\left\| J_N\left(\hat{\theta}^k\right) - J_N\left(\hat{\theta}^{k-1}\right) \right\|_2$ is small enough.

4.6 Illustrative Examples

In this section, first we compare the computational complexity of the direct gradient search and the gradient search in the local parameterization of finite degree Volterra realization. This comparison is expressed as a function of the realization degree. At the same time, we corroborate that the method is able to identify the realization of finite degree Volterra series.

Second, we consider two examples of nonlinear system, and we compare the capability of finite degree Volterra realization to approximate the considered examples with some of state of the art system identification methods for nonlinear systems.

In order to validate the obtained models, for a data of N samples, we have used $T_{id} = \frac{N}{2}$ samples for the identification purpose, and the rest of them $T_{val} = T - T_{id} = \frac{N}{2}$ samples for the validation purpose. The validation step, can be viewed as an evaluation of the prediction accuracy of the models.

To verify that the models are able to extract the real output signal from the measured noised one, we consider the output signal *without* the measurement noise in the validation step.

The model accuracy is defined as the Percent Variance Accounted For ($\%VAF$)

$$\%VAF \triangleq \left(1 - \frac{\sum_{t=1}^{N}(y_t - \hat{y}_t)^T(y_t - \hat{y}_t)}{\sum_{t=1}^{N}(y_t - \bar{y})^T(y_t - \bar{y})}\right) \times 100$$

where $\hat{y}_t$ denotes the estimated output signal and $\bar{y}$ is the mean of the output signals.

4.6.1 Computational Complexity

In this section, we figure out that using a local parameterization accelerates the convergence of gradient search method. For that, we consider a finite degree Volterra series realization (4.4), where the states $\{Z_t^i : \ i = 1, 2, \cdots, l\}$ have all the same dimension which is equal to $n = 10$.
To obtain the optimization time as a function of the degree of Volterra series realization, we consider the realizations of degrees $1, 2, \cdots, 8$.

The identification experiment for each value of the realization degree is repeated 10 times, and the average of optimization time is then computed.
The optimization time is the required time to reach a specific precision (local optimum).

Each experiment has three inputs ($u_t \in \mathbb{R}^3$) which have been chosen to be uniform white noise and three outputs ($y_t \in \mathbb{R}^3$). The length of the input/output data is equal to 4000 samples. The measurement noise v_t is a Gaussian white noise scaled such that the signal to noise ratio $SNR = 10\ dB$.

The initial estimation of the realization's vector of parameters (θ) is computed using the sequential projection method explained in Section 4.4. The computation time of sequential projection method is calculated and given in Fig. 4.2.

The optimization time as a function of realization degree (l) for the direct gradient search method and the gradient search in local parameterization space is given in Fig. 4.1. The implementation of two approaches is done using MATLAB™ programming environment.

It is clear from Fig. 4.1 that using local parameterization reduces the time of computation. This reducing becomes advantageous when the realization degree increases. In order to

Table 4.1: Accuracy of the identified models

Identification	Initial estimated model	Optimized model
Accuracy ($\%VAF$)	82.3 ± 2.7	89.1± 1.8
Validation		
Accuracy ($\%VAF$)	86.9 ± 1.3	98.6 ± 0.7

corroborate that the proposed method is able to identify the finite Volterra series efficiently, we calculate the average and the standard deviation of the accuracies of the obtained models.

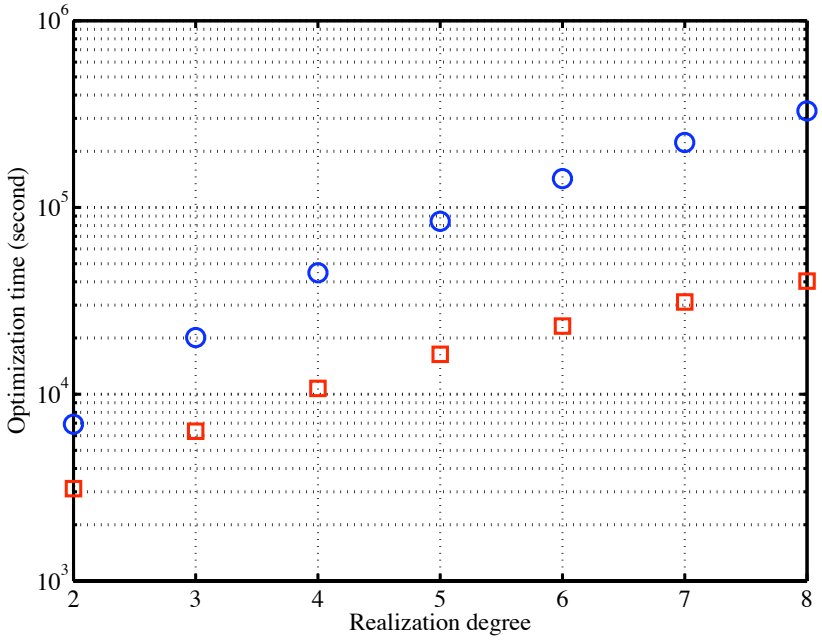


Figure 4.1: Optimization time for the direct gradient search method (circle mark) and the gradient search in local parameterization space (square mark).

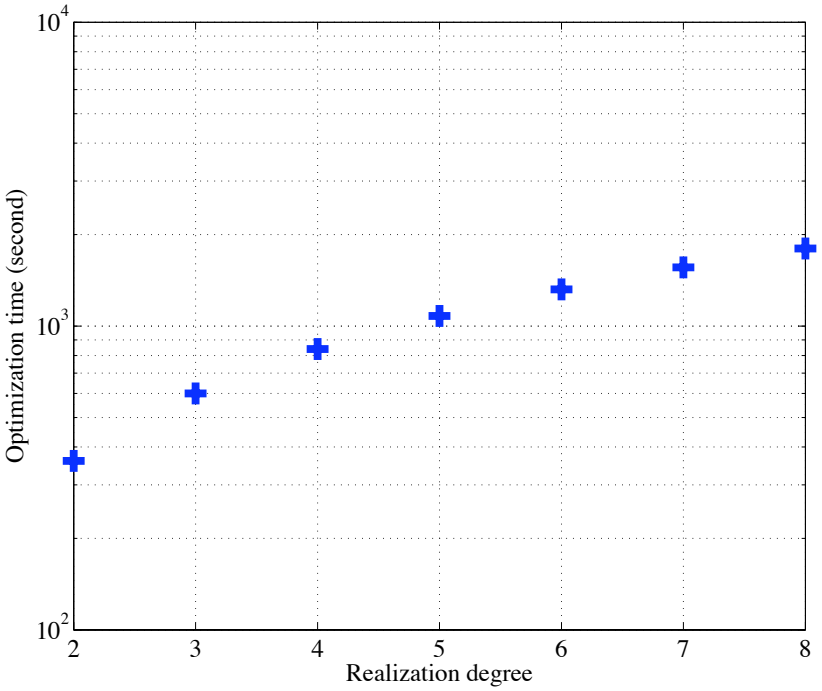


Figure 4.2: Computation time of the sequential projection method.

The accuracies are reported in Table 4.1 where the initial estimated model is the model obtained by the sequential projection method explained in Section 4.4 and the optimized model is that obtained by local gradient search in local parameterization space. These accuracies demonstrate

that the proposed method has efficiently identified the finite degree Volterra realization and the sequential projection method provides a good initial guess for the optimization process.

4.6.2 First Example of Nonlinear System

Consider the following example

$$\begin{aligned} x_t^1 =& 0.7\ x_{t-1}^1 + u_t && , x_0^1 = 0 \\ x_t^2 =& 0.7\ x_{t-1}^2 + \left(x_t^1\right)^2 + x_t^1 + u_t && , x_0^2 = 0 \\ y_t =& \left(x_t^1\right)^2 + x_t^2 + v_t \end{aligned} \tag{4.49}$$

The input signal $u_t \in \mathbb{R}$ is chosen to be uniform white noise with zero mean and standard deviation equals to 0.5, and its length is equal to 1000 samples. The measurement noise v_t is a Gaussian white noise scaled such that the signal to noise ratio $SNR = 10\ dB$. We compare the performance of finite Volterra series realization with neural network based nonlinear system identification, which is a popular approach [Billings et al. 1992; Sjoberg et al. 1994; Nørgaard et al. 2000]. For comparison, we have chosen two nonlinear model structures

1. Neural Network Output Error (NNOE) model, Fig. 4.3. The output function generated by the neural network can be calculated as follows

$$\begin{aligned} \varphi_t &= \begin{bmatrix} \hat{y}_{t-1}(\theta) & \hat{y}_{t-2}(\theta) & \hat{y}_{t-k}(\theta) & u_t & u_{t-1} & u_{t-j} \end{bmatrix}^T \\ \hat{y}_t(\theta) &= g\left(\varphi_t, \theta\right) \end{aligned} \tag{4.50}$$

   where φ_t is a vector containing the regressors, θ is a vector containing the weights and g is the function realized by the neural network. the best performance was obtained with $k = 3$, $j = 2$ and neural network with 10 hidden hyperbolic tangent units in the hidden layer.

2. Neural Network State Space Innovation Form (NNSSIF) model, Fig. 4.4, which determines a nonlinear state space model of the dynamic system

$$\begin{aligned} \hat{x}_t(\theta) &= g\left(\hat{x}_{t-1}(\theta), u_t\right) \\ \hat{y}_t &= C\hat{x}_t(\theta) \end{aligned} \tag{4.51}$$

   where $\hat{x}_t(\theta)$ is the state vector and its dimension is n_x, g is the function realized by the neural network and C is a constant matrix.

   In this example, the best performance was obtained by $n_x = 6$ and neural network with 10 hidden hyperbolic tangent units in the hidden layer. The neural network is trained with Levenberg-Marquardt method.

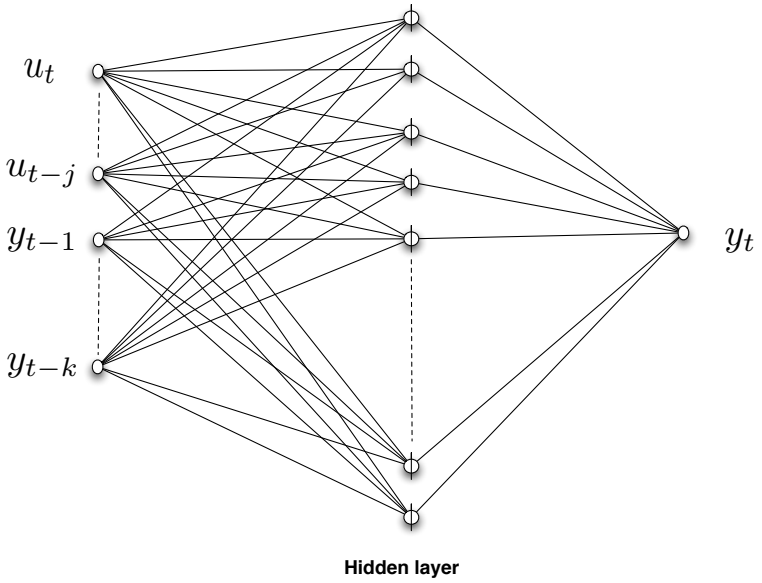


Figure 4.3: Neural Network Output Error model.

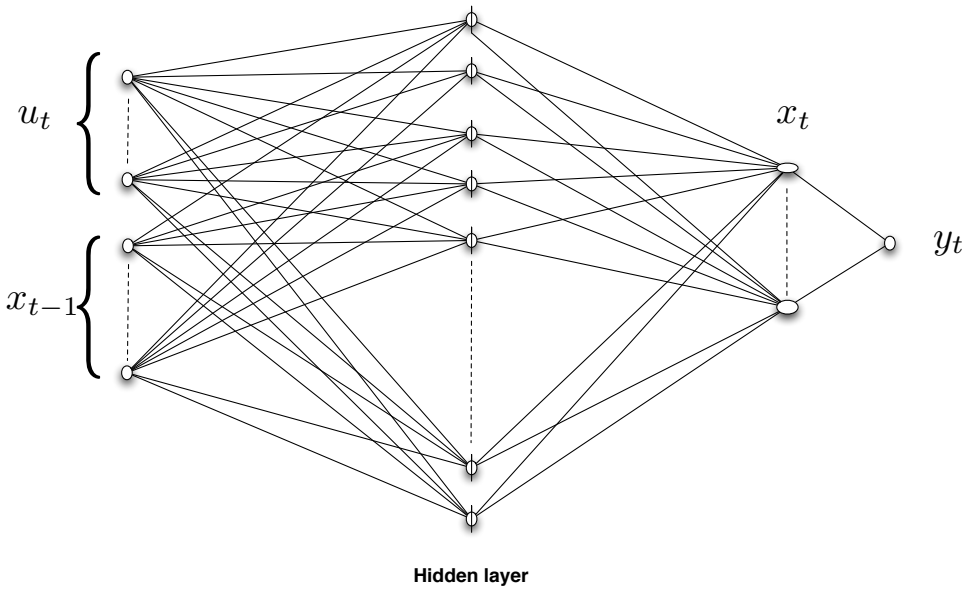
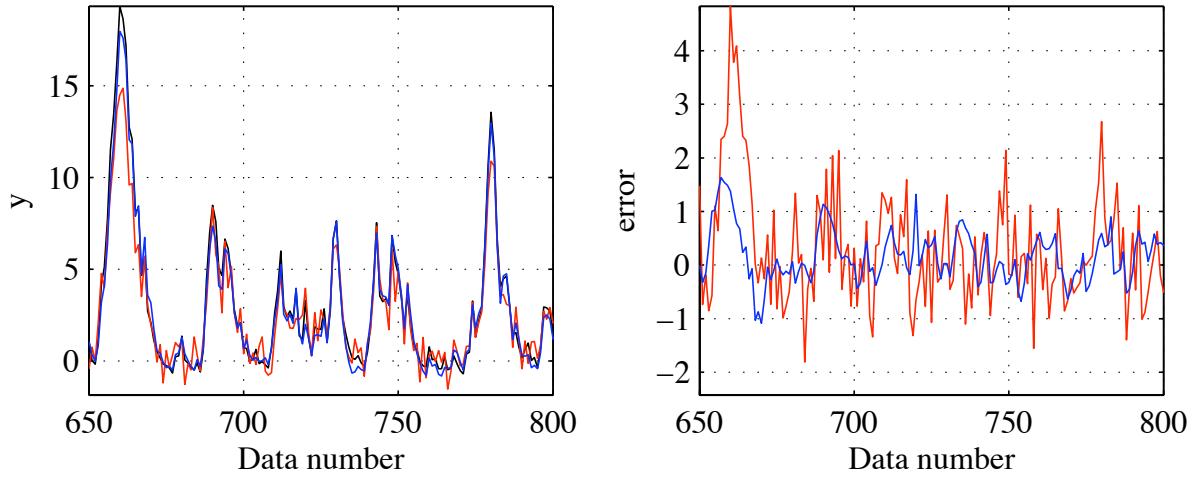
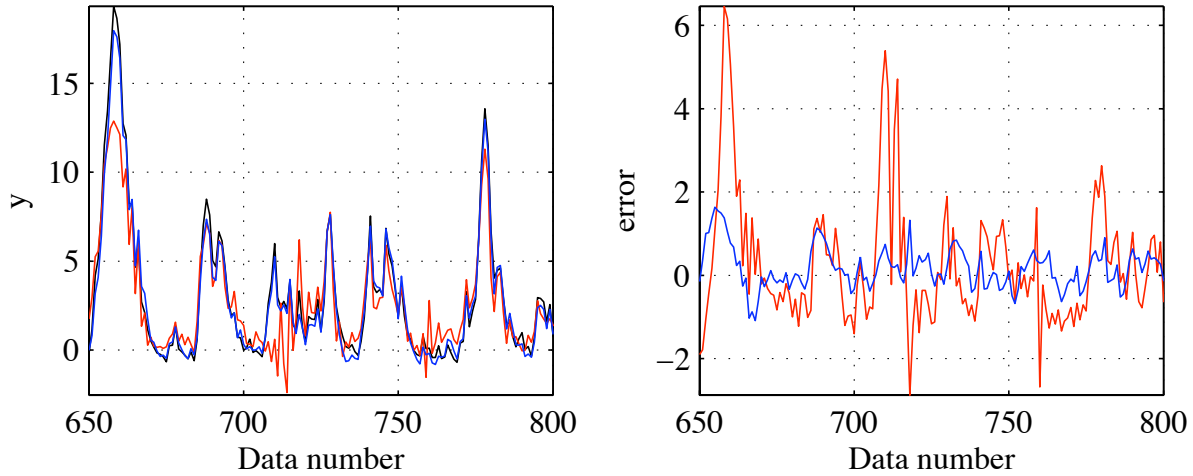


Figure 4.4: Neural Network State Space Innovation Form model.

The implementation of two models (NNOE and NNSSIF) is done using Neural Network Based System Identification toolbox [Nørgaard 2000].



(a) The real output *without* the measurement noise (black line). The outputs of quadratic system (blue line) and NNOE model (red line), and their errors.



(b) The real output *without* the measurement noise (black line). The outputs of quadratic system (blue line) and NNSSIF model (red line), and their errors.

Figure 4.5: Comparison of the results of quadratic system, NNOE and NNSSIF models.

Table 4.2: Accuracy of the identified models

Identification	Quadratic system	NNOE model	NNSSIF model
Accuracy (%VAF)	85.7	76.5	63.9
Validation			
Accuracy (%VAF)	98.0	81.9	73.4

Note that the dimensions of the states ($Z_t^i:\ i=1,2$) of the quadratic system (Volterra series realization of degree two) are 2 and 3 respectively.
Recall that these dimensions are obtained by using the sequential projection method explained in Section 4.4. The accuracies reported in Table 4.2 show the outperformance of quadratic system in comparison with NNOE and NNSSIF models.

To deal with a clear graphical representation, we represent the results in the validation interval over a window of 150 samples in Fig. 4.5.

4.6.3 Second Example of Nonlinear System

Consider the following example of nonlinear system

$$\begin{aligned}
x_t^1 =& 0.8x_{t-1}^1 + 0.8u_t(1) + 0.6u_t(2) + 0.8u_t(3)\\
x_t^2 =& 0.5\left(x_{t-1}^1\right)^2 + 0.5x_{t-1}^2 + 0.9\left(u_t(2)\right)^3 + u_t(1)u_t(2)x_{t-1}^1\\
x_t^3 =& 0.7x_{t-1}^2 + 0.7x_{t-1}^3 + 0.8u_t(2)u_t(3)x_{t-1}^1 + 0.9\left(u_t(1)\right)^3 + 0.9\left(u_t(2)\right)^3\\
y_t =& \begin{pmatrix} 0.6 & 0.3 & 0.6\\ 0.2 & 0.7 & 0.6\\ 0.3 & 0.0 & 0.5 \end{pmatrix}\begin{pmatrix} x_t^1\\ x_t^2\\ x_t^3 \end{pmatrix} + v_t
\end{aligned} \tag{4.52}$$

where the non observed nonlinear state is $x_t = \begin{bmatrix} x_t^1, & x_t^2, & x_t^3 \end{bmatrix}^T$, and $u_t(i)$ denotes the i^{th} element of the input signal $u_t \in \mathbb{R}^3$. The system is simulated with three dimensional uniform white noise with standard deviation equals to 1 as input. The length of the input data is equal to 2000 samples. The measurement noise v_t is a Gaussian white noise is scaled such that the $SNR = 15\,dB$. Finally, the initial conditions are $\begin{bmatrix} x_0^1, & x_0^2, & x_0^3 \end{bmatrix}^T = \begin{bmatrix} 0, & 0, & 0 \end{bmatrix}^T$.

We compare the performance of finite degree Volterra series realization with NNSSIF model which can handle Multiple Inputs - Multiple Outputs (MIMO) systems.
The best performance was obtained by $n_x = 10$ and 15 hidden hyperbolic tangent units in the hidden layer.

Note that the dimensions of the states ($Z_t^i:\ i=1,2,3$) of the cubic system (Volterra series realization of degree three) are 6, 9 and 13 respectively. The accuracies reported in Table 4.3 show the outperformance of a cubic system in comparison with NNSSIF model. To deal with a clear graphical representation, we represent the results in the validation interval over a window of 150 samples in Fig. 4.6.

It will be of interest to compare the performance of finite Volterra series realization with truncated Volterra series which has the same degree. Therefore, the accuracies and the number of parameters of truncated Volterra series of degree three with various horizon lengths T are calculated and reported in Table 4.4.
Note that the accuracy reported in Table 4.4 is the total accuracy of the output signal $y_t \in \mathbb{R}^3$.

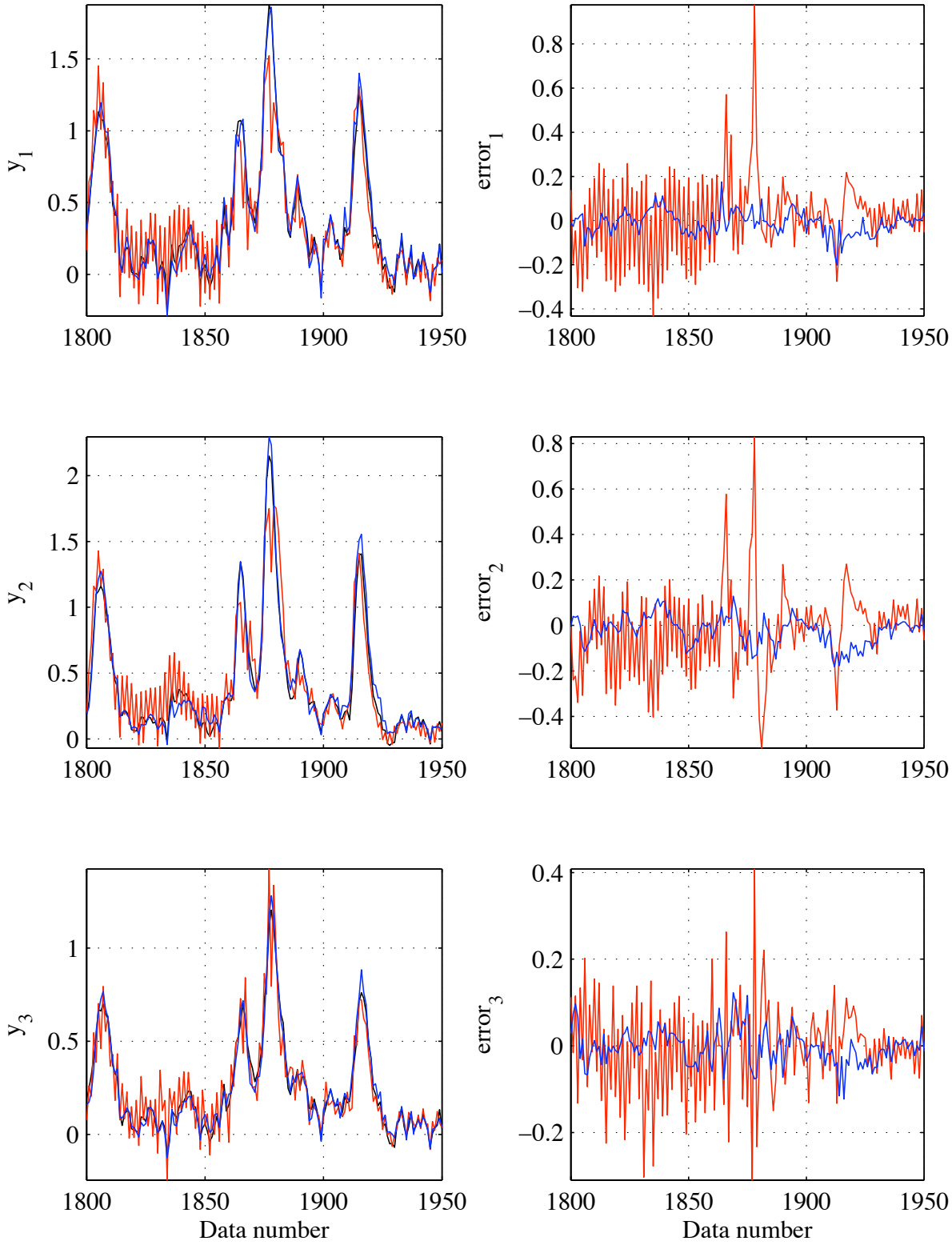


Figure 4.6: The real outputs *without* the measurement noise (black line) . The outputs of cubic system (blue line) and NNSSIF model (red line), and their errors are superimposed.

Table 4.3: Accuracy of the identified models

Identification	Accuracy ($\%VAF$)	
	Cubic system	NNSSIF model
First output	96.8	79.2
Second output	96.9	82.4
Third output	96.0	86.5
Validation		
First output	98.0	82.3
Second output	97.6	83.8
Third output	97.3	88.5

Table 4.4: Accuracy comparison between cubic system and truncated Volterra series of degree three

	Number of parameters	Identification Accuracy	Validation Accuracy
Cubic system	901	96.7	97.9
	Truncated Volterra series of degree three and horizon T		
$T = 7$	10764	99.1	38.4
$T = 9$	19395	99.2	59.9
$T = 10$	25047	99.4	61.6
$T = 11$	31698	99.3	61.4
$T = 12$	39429	99.0	58.8
$T = 15$	69912	99.4	48.4

From Table 4.4, three remarks can be concluded

1. The number of parameters of cubic system is much smaller than that of truncated Volterra series of degree three .
2. The validation accuracy of cubic system is much higher than that of truncated Volterra series of degree three. This result is particularly interesting.
   The reason why the prediction of cubic system is more accurate than truncated Volterra series is that the cubic system consider the integral past of the input signal and its parameters are optimized such that the output of the model accurately approximates the real output of system.
   On the contrary, the truncated Volterra series consider a fixed horizon and calculate the Volterra kernels which correspond to this horizon.
3. On the contrary of theoretical expectations, the validation accuracy of truncated Volterra series decreases with increasing the horizon T. This is because the number of parameters increases and as well the size of matrices which become ill-conditioned.

4.7 Conclusion

In this chapter, we have presented a new method to identify a recursive state-space realization of finite degree Volterra series with infinite horizon. The method is based on a local parameterization of the state-space representation of the realization and subsequent gradient search in the resulting local parameter space.

Furthermore, we have proposed a sequential projection procedure to calculate an initial estimation of the realization parameters.
The method has successfully applied to identify various illustrative examples, and a comparison with some methods from the state of the art nonlinear system identification methods have pointed out the outperformance of finite degree Volterra series realization.

Moreover, a comparison with truncated Volterra series of the same degree has borne out that not only the number of parameters of finite Volterra series realization is much smaller than that of truncated Volterra series, but also the prediction accuracy of finite degree Volterra series realization is superior to that of truncated Volterra series.

It is known that the main cumbersome of gradient search method is the convergence to a local optimum. However, since the sequential projection method considers the whole trajectory of input/output to estimate an initial guess of θ, the gradient search method has a good chance to converge to the global optimum.

Résumé du chapitre 5
Identification des systèmes quadratiques en l'état

Dans la littérature concernant l'identification des systèmes dynamiques, on constate que les systèmes bilinéaires font partie des modèles classiques souvent proposés pour approcher les systèmes non-linéaires [Rugh 1981; Mohler 1991; Isidori 1995]. L'avantage de cette classe est la linéarité par rapport à l'état du système ($x_t \in \mathbb{R}^n$) et au signal d'entrée ($u_t \in \mathbb{R}^m$) séparément.

Un système bilinéaire peut être représenté par le modèle suivant:

$$\begin{aligned} x_{t+1} &= A_b x_t + B_b u_t + N_b (u_t \otimes x_t) \\ y_t &= C_b x_t + v_t \end{aligned}$$

où $\otimes$ désigne l'opérateur de produit de Kronecker, et $y_t \in \mathbb{R}^p$ est le signal de sortie du système.

Il a été établi que cette classe de système est dense, dans le sens de L^2, dans la classe des systèmes non-linéaires analytiques. De ce fait, analyser les propriétés des systèmes bilinéaires et proposer des méthodes d'identification de ceux-ci ont été des sujets actifs de recherche pendant ces dernières années [Monin and Salut 1996; Chen and Maciejowski 2000c; 2000a].

Considérons un système bilinéaire en boucle fermée comme illustré sur la fig. 1.

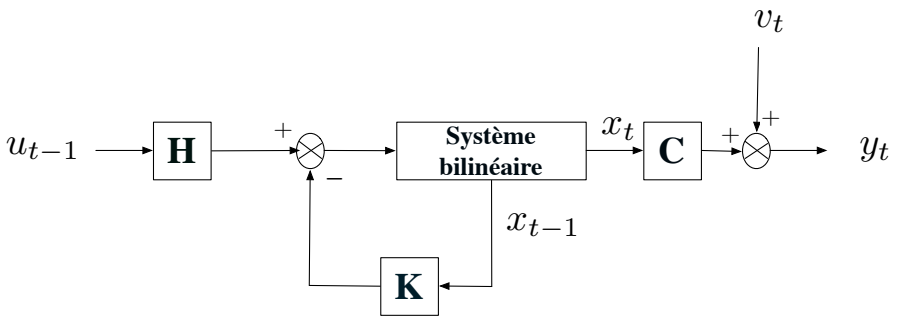


Figure 1 : Système quadratique en l'état vu comme système bilinéaire en boucle fermée.

Le modèle équivalent du système présenté par le schéma sur la fig. 1 peut être calculé en utilisant la propriété suivante $FG \otimes HJ = (F \otimes H)(G \otimes J)$.

Ainsi, on obtient

$$x_{t+1} = Ax_t + Bu_t + N(u_t \otimes x_t) + Q(x_t \otimes x_t)$$
$$y_t = Cx_t + v_t$$

où

$$A = A_b - B_b K$$
$$B = B_b H$$
$$N = N_b(H \otimes I_n)$$
$$Q = -N_b(K \otimes I_n)$$

Le système ainsi obtenu est nommé système quadratique en l'état du fait qu'il contient le terme $x_t \otimes x_t$. À noter que ce modèle s'est avéré très utile dans plusieurs domaines de recherche. Par exemple, en écologie et en biologie on utilise couramment le "modèle proie-prédateur" [Volterra 1931; Thau 1972] qui peut être formulé comme un système quadratique en l'état. Dans la domaine d'ingénierie, on trouve plusieurs exemples de systèmes quadratiques, comme par exemple l'équation de la dynamique d'un moteur à courant alternant ou bien la représentation de la saturation d'amplificateurs de puissances.

Nous montrons dans le chapitre 5 que ce système quadratique en l'état apparaît comme un modèle approximatif de deuxième degré d'un système non-linéaire plus général ayant la forme suivante:

$$x_{t+1} = f(x_t, u_t)$$
$$y_t = Cx_t + v_t$$

Dans le but d'identifier ce modèle, nous proposons une méthode basée sur la minimisation de la norme Euclidienne de l'erreur de sortie. Nous résoudrons alors ce problème d'optimisation en faisant appel à nouveau à une méthode de type gradient.

Nous montrons par ailleurs que la structure du système quadratique en l'état n'est pas unique. Afin de surmonter le problème de non-unicité du modèle, nous proposons une méthode de paramétrisation locale de manière à identifier les directions dans lesquelles la fonction d'objectif ne change pas. Ces directions seront par la suite rejetées dans la méthode d'optimisation de type gradient.

Le cas des plusieurs expériences de courte durée est également envisagé. Nous proposons une méthode pour traiter ce cas qui peut être vu comme une extension de la méthode proposée dans le chapitre 3 dans le cas des systèmes linéaires.

L'efficacité de la méthode proposée pour identifier les systèmes quadratiques en l'état, dans le cas de simple expérience ou le cas d'expériences multiples, a été illustrée par des résultats de simulation d'exemples académiques.

"Science has proof without any certainty. Creationists have certainty without any proof."

Ashley Montague

5

Identification of Quadratic in-the-State System

In the literature, we find that the bilinear systems is one of the classic models which is proposed to approximate a general nonlinear system [Rugh 1981; Mohler 1991; Isidori 1995]. This class of systems is linear separately with respect to the state ($x_t \in \mathbb{R}^n$) and the input signal ($u_t \in \mathbb{R}^m$), and can be characterized by the following model

$$\begin{aligned} x_{t+1} &= A_b x_t + B_b u_t + N_b (u_t \otimes x_t) \\ y_t &= C_b x_t + v_t \end{aligned} \tag{5.1}$$

where $\otimes$ denotes the Kronecker product, and $y_t \in \mathbb{R}^p$ is the output.

It is proven that this class of systems is dense, in L^2 sense, in the class of analytical nonlinear systems. For that, studying the properties and the identification of bilinear systems has been subject of active research during the last decade [Monin and Salut 1996; Chen and Maciejowski 2000c; 2000a].

Let us consider a bilinear system in a feedback loop which is presented in Fig. 5.1.
The representation of the equivalent model can be calculated by using the property $FG \otimes HJ = (F \otimes H)(G \otimes J)$ as follows

$$\begin{aligned} x_{t+1} &= A x_t + B u_t + N (u_t \otimes x_t) + Q (x_t \otimes x_t) \\ y_t &= C x_t + v_t \end{aligned} \tag{5.2}$$

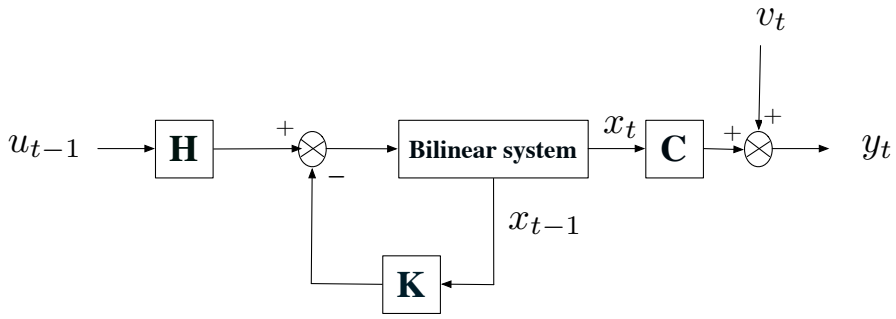


Figure 5.1: Quadratic system as a bilinear system in a feedback loop.

where

$$\begin{aligned} A &= A_b - B_b K \\ B &= B_b H \\ N &= N_b (H \otimes I_n) \\ Q &= -N_b (K \otimes I_n) \end{aligned} \tag{5.3}$$

The class of systems given by the model (5.2) is called Quadratic in-the-State Systems (QSS). In fact, This class enjoys a useful model more than a representation of a bilinear system in a feedback loop. For example, in ecology and biology where the population dynamics of two interacting species may be described by the pray-predator Volterra equation [Volterra 1931; Thau 1972], which has the exactly structure of QSS. In the engineering area, there are also several examples of QSS, e.g. the dynamics of a general AC machine [A. De Carli and Ruberti 1966].

Moreover, the quadratic system arises in the approximation of the nonlinear system which has the following structure

$$\begin{aligned} x_{t+1} &= f(x_t, u_t) \\ y_t &= C x_t + v_t \end{aligned} \tag{5.4}$$

Without loss of generality, we assume that the equilibrium of the system is $f(x_0, u_0) = 0$ and $(x_0, u_0) = (0, 0)$. Let us define a new vector z_t as follows

$$z_t = \begin{bmatrix} x_t \\ u_t \end{bmatrix} \tag{5.5}$$

The Taylor approximation of system (5.4) around the equilibrium point z_0 can be written as

follows

$$x_{t+1} = \left[f(z_t), \frac{\partial f(z_t)}{\partial z_t}, \frac{\partial^2 f(z_t)}{\partial z_t^2}, \cdots \right]_{z_0} \times \begin{bmatrix} I_n \\ (z_t - z_0) \\ (z_t - z_0) \otimes (z_t - z_0) \\ \vdots \end{bmatrix} \tag{5.6}$$

By neglecting the terms of order superior to two, we obtain

$$x_{t+1} = Ax_t + Bu_t + N(u_t \otimes x_t) + Q(x_t \otimes x_t) + M(u_t \otimes u_t) \tag{5.7}$$

It is clear that the above structure is similar to the structure of quadratic system. In order to have the same structure, we define a new input signal $\tilde{u}_t$ as follows

$$\tilde{u}_t = \begin{bmatrix} u_t \\ u_t \otimes u_t \end{bmatrix} \tag{5.8}$$

Thus, the system (5.7) becomes

$$x_{t+1} = Ax_t + \begin{bmatrix} B & M \end{bmatrix} \tilde{u}_t + \begin{bmatrix} N & 0 \end{bmatrix} (\tilde{u}_t \otimes x_t) + Q(x_t \otimes x_t) \tag{5.9}$$

which has exactly the structure of quadratic system (5.2).

This chapter is organized as follows: In Section 5.1 the identification problem of QSS is formulated and a solution using gradient search method is given. The local parameterization of QSS is developed in order to define the directions in which the output error cost function does not change, and a summary of the identification algorithm is explained in Section 5.2. In Section 5.3 the case of multiple data sets case is treated. Estimating an initial guess of the vector of parameters is discussed in Section 5.4. Some simulations examples are given in Section 5.5. Finally, Section 5.6 concludes this chapter.

5.1 Identification Procedure

The identification of Structure (5.2) relies upon the estimation of the matrices A, B, C, N, Q. Without any assumption on the structure of these matrices, we assume that all matrices are fully parameterized. Let us define

$$\theta = \begin{bmatrix} vec(A) \\ vec(B) \\ vec(N) \\ vec(Q) \\ vec(C) \end{bmatrix} \tag{5.10}$$

Suppose that we have an estimation $\hat{\theta}$ of θ, and a set of input/output data from the real system $\{u_t, y_t : t = 0, 1, \cdots, L\}$. The estimated output $\hat{y}_t(\hat{\theta})$ can be given as follows

$$\begin{aligned}\hat{x}_{t+1} &= A(\hat{\theta})\hat{x}_t + B(\hat{\theta})u_t + N(\hat{\theta})\,(u_t \otimes \hat{x}_t) + Q(\hat{\theta})\,(\hat{x}_t \otimes \hat{x}_t)\\ \hat{y}_t(\hat{\theta}) &= C(\hat{\theta})\hat{x}_t\end{aligned} \tag{5.11}$$

Our goal is achieved if $\hat{y}_t(\hat{\theta})$ approximates the output y_t accurately enough. This criterion can be transformed into the minimization of the output error with respect to the parameter θ which leads to a output-error identification problem similar to (4.11).

Considering the following output-error cost function

$$J_L(\theta) = \frac{1}{L}\sum_{k=1}^{L} \|y_k - \hat{y}_k(\theta)\|_2^2 = \frac{1}{L} E_L(\theta)^T E_L(\theta) \tag{5.12}$$

where

$$E_L(\theta) = \left[e_1(\theta)^T \; e_2(\theta)^T \cdots e_L(\theta)^T\right]^T \tag{5.13}$$

is the error vector in which $e_k(\theta) = y_k - \hat{y}_k(\theta)$.

Similarly to the case of finite degree Volterra series realization, we use gradient search method to solve this identification problem. This iterative method is based on the updating of the system parameters θ as follows

$$\hat{\theta}^{i+1} = \hat{\theta}^i - (\psi_L^T(\hat{\theta}^i)\psi_L(\hat{\theta}^i) + \lambda^{i+1} I)^{-1}\psi_L^T(\hat{\theta}^i) E_L(\hat{\theta}^i) \tag{5.14}$$

Where λ^{i+1} is the regularization parameter and

$$\psi_L(\theta) \triangleq \frac{\partial E_L(\theta)}{\partial \theta^T} \tag{5.15}$$

is the Jacobian of the error vector $E_L(\theta)$.

5.1.1 Computing the Iterative Parameter Update

In order to compute the update rule (5.14), the following quantities $E_L(\theta)$ and $\psi_L(\theta)$ must be computed. Computing the vector $E_L(\theta)$ can be done by simulating the system (5.11) that corresponds to θ^{i-1}. Note that this simulation brings out the state $\hat{x}_t$ and $\hat{y}_t$.

In order to simulate $\psi_L(\theta^{i-1})$, we should compute the derivative of $\hat{y}_t$ with respect to θ^{i-1}. Let us define

$$\zeta_t^j = \frac{\partial \hat{x}_t}{\partial \theta_j} \tag{5.16}$$

where θ_j is the j^{th} element of the vector θ.

The computation of $\frac{\partial \hat{y}_t}{\partial \theta} = \left[\begin{array}{ccc} \frac{\partial \hat{y}_t}{\partial \theta_1} & \cdots & \frac{\partial \hat{y}_t}{\partial \theta_l} \end{array}\right]$, where l is the number of parameters in θ, can be made using the following model

$$\begin{aligned}\zeta_{t+1}^j =& A\zeta_t^j + \frac{\partial A}{\partial \theta_j}\hat{x}_t + \frac{\partial B}{\partial \theta_j}u_t + N(u_t \otimes \zeta_t^j) + \frac{\partial N}{\partial \theta_j}(u_t \otimes \hat{x}_t) + \\ & Q\left(\zeta_t^j \otimes \hat{x}_t\right) + Q\left(\hat{x}_t \otimes \zeta_t^j\right) + \frac{\partial Q}{\partial \theta_j}\left(\hat{x}_t \otimes \hat{x}_t\right) \\ \frac{\partial \hat{y}_t}{\partial \theta_j} =& C\zeta_t^j + \frac{\partial C}{\partial \theta_j}\hat{x}_t\end{aligned} \tag{5.17}$$

5.2 Local Parameterization

Assume the following transformation of the state

$$z_t = T^{-1}x_t \tag{5.18}$$

where $T \in \mathbb{R}^{n\times n}$ is a nonsingular matrix. Then, we obtain the following structure

$$\begin{aligned}z_{t+1} &= \bar{A}z_t + \bar{B}u_t + \bar{N}\left(u_t \otimes z_t\right) + \bar{Q}\left(z_t \otimes z_t\right) \\ y_t &= \bar{C}z_t + v_t\end{aligned} \tag{5.19}$$

where

$$\begin{aligned}\bar{A} =& \quad T^{-1}AT \\ \bar{B} =& \quad T^{-1}B \\ \bar{N} =& \quad T^{-1}N\left(I_m \otimes T\right) \\ \bar{Q} =& \quad T^{-1}Q\left(T \otimes T\right) \\ \bar{C} =& \quad CT\end{aligned} \tag{5.20}$$

As a consequence, the relation input/output does not change by transforming the state x_t according to (5.18), where the transformation matrix T parameterizes the subset of equivalent models. Note that this subset defines a manifold.

As Structure (5.2) is not unique, the minimization of $J_L(\theta)$ does not have a unique solution. Indeed, the optimal solution can be made unique by choosing a suitable parameterization. Such parameterization for QSS does not exist up to our knowledge. Therefore, we will define a local parameterization of QSS analogously to the case of finite degree Volterra series (Chapter 4).

The objective of the local parameterization is identifying the directions that do not change the cost function $J_L(\theta)$ and projecting them out at each iteration, for that only the active parameters are updated.

In order to identify the tangent plane of the manifold, we linearize the relation (5.20) around the identity matrix $T = I_n$.

Considering a small perturbation $T = I_n + \Delta T$, by using the approximation $(I_n + \Delta T)^{-1} \simeq I_n - \Delta T$ and neglecting all second-order terms, we obtain

$$\begin{aligned}
\bar{A} &= A - \Delta T A + A \Delta T \\
\bar{B} &= B - \Delta T B \\
\bar{N} &= N - \Delta T N + N\left(I_m \otimes \Delta T\right) \\
\bar{Q} &= Q - \Delta T Q + Q\left(I_n \otimes \Delta T\right) + Q\left(\Delta T \otimes I_n\right) \\
\bar{C} &= C + C \Delta T
\end{aligned} \tag{5.21}$$

If we consider the following vectors of parameters

$$\theta = \begin{bmatrix} vec(A) \\ vec(B) \\ vec(N) \\ vec(Q) \\ vec(C) \end{bmatrix} \quad and \quad \bar{\theta} = \begin{bmatrix} vec(\bar{A}) \\ vec(\bar{B}) \\ vec(\bar{N}) \\ vec(\bar{Q}) \\ vec(\bar{C}) \end{bmatrix} \tag{5.22}$$

Then

$$\bar{\theta} = \theta + \begin{bmatrix} vec(-\Delta T A + A \Delta T) \\ vec(-\Delta T B) \\ vec(-\Delta T N + N\left(I_m \otimes \Delta T\right)) \\ vec(-\Delta T Q + Q\left(I_n \otimes \Delta T\right) + Q\left(\Delta T \otimes I_n\right)) \\ vec(C \Delta T) \end{bmatrix} \tag{5.23}$$

By defining

$$\Pi_{i,K} \triangleq \begin{bmatrix} 0_{n \times (i-1)n} & I_n & 0_{n \times (K-i)n} \end{bmatrix} \tag{5.24}$$

it is possible to write $I_m \otimes \Delta T$ as follows

$$I_m \otimes \Delta T = \sum_{i=1}^{m} \Pi_{i,m}^T \Delta T \Pi_{i,m} \tag{5.25}$$

On the other hand, $\Delta T \otimes I_n$ can be written as follows

$$\Delta T \otimes I_n = \sum_{i=1}^{n} \sum_{j=1}^{n} \Pi_{i,n}^T \Pi_{j,n} \Delta T\left(i,j\right) \tag{5.26}$$

recall that $\Delta T\left(i,j\right)$ is the element of the row i and column j of ΔT.

Using the property $vec(ABC) = \left(C^T \otimes A\right) vec(B)$, we obtain

$$\begin{aligned}
vec\left(Q\left(\Delta T \otimes I_n\right)\right) =& \sum_{i=1}^{n} \sum_{j=1}^{n} \left(\Pi_{j,n}^T \otimes Q\right) vec\left(\Pi_{i,n}^T\right) \Delta T\left(i,j\right) \\
=& \Gamma_Q \, vec(\Delta T)
\end{aligned} \tag{5.27}$$

where

$$\begin{aligned}&\Gamma_Q = [\gamma_{1,1}\,\gamma_{2,1}\cdots\gamma_{n,1}\,\gamma_{1,2}\,\gamma_{2,2}\cdots\gamma_{n,2}\cdots\gamma_{1,n}\cdots\gamma_{n,n}]\\&\text{and}\\&\gamma_{i,j} = \left(\Pi_{j,n}^T \otimes Q\right) vec\left(\Pi_{i,n}^T\right)\end{aligned} \tag{5.28}$$

Using analogous logic, the relation between θ and $\bar{\theta}$ becomes

$$\bar{\theta} = \theta + M_\theta vec(\Delta T) \tag{5.29}$$

where

$$M_\theta = \begin{bmatrix} -A^T \otimes I_n + I_n \otimes A \\ -B^T \otimes I_n \\ -N^T \otimes I_n + \sum_{i=1}^{m} \Pi_{i,m}^T \otimes \left(N\Pi_{i,m}^T\right) \\ -Q^T \otimes I_n + \sum_{i=1}^{n} \Pi_{i,n}^T \otimes \left(Q\Pi_{i,n}^T\right) + \Gamma_Q \\ I_n \otimes C \end{bmatrix} \tag{5.30}$$

Using Lemma 4.3, we obtain that the left null space of M_θ (5.30) contains the directions in which the parameters should be modified to lead a change in the value of the cost function $J_N(\theta)$ (5.12).

Recall that, the left null space of M_θ can be efficiently obtained by a QR decomposition

$$M_\theta = \begin{bmatrix} \mathcal{Q}_1 & \mathcal{Q}_2 \end{bmatrix} \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix} \tag{5.31}$$

Then a basis of the left null space of M_θ is $\mathcal{Q}_2$. The new update rule becomes

$$\hat{\theta}^i = \hat{\theta}^{i-1} - \mathcal{Q}_2 \left(\mathcal{Q}_2^T \psi_L^T \psi_L \mathcal{Q}_2 + \lambda^i I\right)^{-1} \mathcal{Q}_2^T \psi_L^T E_L \tag{5.32}$$

where $\mathcal{Q}_2$ and ψ_L depend on $\hat{\theta}^{i-1}$. Since $\mathcal{Q}_2$ depends on the past parameter $\hat{\theta}^{i-1}$ the QR (5.31) must be computed at each iteration.

5.2.1 Summary of the Implementation Algorithm

We can resume the algorithm of implementation as follows

1. Calculate the state $\hat{x}_t$ and $\hat{y}_t$ by simulating the system (5.11) with $\theta = \hat{\theta}^{i-1}$.
2. Compute $E_L(\hat{\theta}^{i-1})$ using (5.13).
3. Calculate the matrix $M_{\hat{\theta}^{i-1}}$ using (5.30).
4. Calculate the QR decomposition of $M_{\hat{\theta}^{i-1}}$ (5.31), from which we obtain $\mathcal{Q}_2$.
5. Calculate the matrix ψ_L using (5.17). We suppose that $\{\zeta_0^j = \mathbf{0},\ j = 1,2,\cdots,l\}$.

6. Calculate the update rule of the gradient search algorithm using (5.32).
7. Perform the termination test for minimization. If true, the algorithm stops. Otherwise, return to Step 1. i.e. compute the values $J_L\left(\hat{\theta}^{i-1}\right)$ and $J_L\left(\hat{\theta}^{i}\right)$ using (5.12) and test if $\|J_L\left(\hat{\theta}^{i}\right) - J_L\left(\hat{\theta}^{i-1}\right)\|_2$ is small enough.

5.3 Dealing with Multiple Short Data Sets

In this section, we show that the method proposed in Chapter 3 can be extended to treat the quadratic system case. Recall that all possessed data sets should be exploited to obtain an accurate model of the system. Moreover as the experiments are short, the effect of initial conditions can not be neglected, so we should estimate them.

Assume that we have the sets $\{u_t^i, y_t^i : t = 1, 2, \cdots, L_i \text{ and } i = 1, 2, \cdots, q\}$. By taking into account the initial conditions, the vector of parameters θ becomes

$$\theta = \begin{bmatrix} vec(A) \\ vec(B) \\ vec(N) \\ vec(Q) \\ vec(C) \\ x_0^1 \\ x_0^2 \\ \vdots \\ x_0^q \end{bmatrix} \tag{5.33}$$

where x_0^j is the initial value of the state which corresponds to the j^{th} data set. The estimated output $\hat{y}_t^j(\hat{\theta})$ of the data set number j is given by the following model

$$\begin{aligned} \hat{x}_{t+1}^j &= A(\hat{\theta})\hat{x}_t^j + B(\hat{\theta})u_t^j + N(\hat{\theta})\left(u_t^j \otimes \hat{x}_t^j\right) + Q(\hat{\theta})\left(\hat{x}_t^j \otimes \hat{x}_t^j\right) \\ \hat{y}_t^j(\hat{\theta}) &= C(\hat{\theta})\hat{x}_t^j \end{aligned} \tag{5.34}$$

The output error function for all data sets can be calculated as follows

$$J_q(\theta) = \frac{1}{q}\sum_{j=1}^{q}\frac{1}{L_j}\sum_{k=1}^{L_j}\left\|y_k^j - \hat{y}_k^j(\theta)\right\|_2^2 = \frac{1}{q}E_q(\theta)^T E_q(\theta) \tag{5.35}$$

where

$$E_q(\theta) = \left[E_{L_1}^1(\theta)^T \; E_{L_2}^2(\theta)^T \cdots E_{L_q}^q(\theta)^T\right]^T \tag{5.36}$$

and

$$E_{L_i}^i(\theta) = \frac{1}{\sqrt{L_i}}\left[e_1^i(\theta)^T \; e_2^i(\theta)^T \cdots e_{L_i}^i(\theta)^T\right]^T \tag{5.37}$$

is the error vector in which $e_k^i(\theta) = y_k^i - \hat{y}_k^i(\theta)$. The minimization of (5.35) can be calculated by using the gradient search method as follows

$$\hat{\theta}^{i+1} = \hat{\theta}^i - (\psi_q^T(\hat{\theta}^i)\psi_q(\hat{\theta}^i) + \lambda^{i+1} I)^{-1}\psi_q^T(\hat{\theta}^i)E_q(\hat{\theta}^i) \tag{5.38}$$

where

$$\psi_q(\theta) = \begin{bmatrix} \psi_{L_1}^1(\theta) \\ \psi_{L_2}^2(\theta) \\ \vdots \\ \psi_{L_q}^q(\theta) \end{bmatrix} \tag{5.39}$$

and

$$\psi_{L_i}^i(\theta) \triangleq \frac{\partial E_{L_i}^i(\theta)}{\partial \theta^T} \tag{5.40}$$

5.3.1 Local Parameterization (multiple data sets)

Similar to the case of a single experiment, we will look for a local parameterization in order to define the direction in which the cost function (5.35) is invariant.

Let us denote the vectors of parameters of two similar realization of QSS by θ and $\bar{\theta}$.

$$\theta = \begin{bmatrix} vec(A) \\ vec(B) \\ vec(N) \\ vec(Q) \\ vec(C) \\ x_0^1 \\ x_0^2 \\ \vdots \\ x_0^q \end{bmatrix} \quad and \quad \bar{\theta} = \begin{bmatrix} vec(\bar{A}) \\ vec(\bar{B}) \\ vec(\bar{N}) \\ vec(\bar{Q}) \\ vec(\bar{C}) \\ z_0^1 \\ z_0^2 \\ \vdots \\ z_0^q \end{bmatrix} \tag{5.41}$$

By considering a small perturbation $T = I_n + \Delta T$ and analogously to the case of a single experiment we obtain

$$\bar{\theta} = \theta + M_\theta vec(\Delta T) \tag{5.42}$$

where

$$M_\theta = \begin{bmatrix} -A^T \otimes I_n + I_n \otimes A \\ -B^T \otimes I_n \\ -N^T \otimes I_n + \sum_{i=1}^{m} \Pi_{i,m}^T \otimes \left(N \Pi_{i,m}^T \right) \\ -Q^T \otimes I_n + \sum_{i=1}^{n} \Pi_{i,n}^T \otimes \left(Q \Pi_{i,n}^T \right) + \Gamma_Q \\ -\left(x_0^1\right)^T \otimes I_n \\ \vdots \\ -\left(x_0^q\right)^T \otimes I_n \end{bmatrix} \tag{5.43}$$

then, we calculate the QR decomposition of M_θ similar to (5.31), from which we obtain $\mathcal{Q}_2$. The new update rule becomes

$$\hat{\theta}^k = \hat{\theta}^{k-1} - \mathcal{Q}_2 \left(\mathcal{Q}_2^T \psi_q^T \psi_q \mathcal{Q}_2 + \lambda^k I \right)^{-1} \mathcal{Q}_2^T \psi_q^T E_q \tag{5.44}$$

5.4 Computing an Initial Estimation

The complexity of the iterative gradient search algorithm and the stability of the obtained structure depends on the choice of an initial estimation of the vector of parameters θ. One could first estimate a stable bilinear model. For example by using CUEDSID toolbox [Chen and Maciejowski 2000b], which is a MATLAB™ package. Once the matrices A, B, N, C are available, they can be used as initial guesses, and the matrix Q can be initialed as zero matrix. In the case of multiple experiments, we use only a single data set to calculate an initial estimation of the vector of parameters θ.

5.4.1 Estimating initial conditions

In the case of multiple data sets, we need to estimate the initial conditions of QSS $(x_0^i : i = 1, 2, \cdots, q)$. Once the bilinear model is available, we can estimate an initial guess of the initial conditions.

Suppose that the matrix N is divided as follows

$$N = \left[N^1 | N^2 | \cdots | N^m \right] \tag{5.45}$$

where $N^k \in \mathbb{R}^{n \times n}$. Each matrix N^k is related to the k^{th} element of the input.

The initial condition x_0^i for the i^{th} data set can be estimated using the following formula

$$\hat{x}_0^i = \Xi_\alpha^\dagger \left(\begin{bmatrix} y_1^i \\ y_2^i \\ \vdots \\ y_\alpha^i \end{bmatrix} - \begin{bmatrix} CB & 0 & \cdots & 0 \\ C\left(A+\sum_{k=1}^m N^k u_1^i(k)\right)B & CB & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ C\left\{\prod_{j=1}^{\alpha-1}\left(A+\sum_{k=1}^m N^k u_j^i(k)\right)\right\}B & \cdots & \cdots & CB \end{bmatrix} \begin{bmatrix} u_0^i \\ u_2^i \\ \vdots \\ u_{\alpha-1}^i \end{bmatrix} \right) \tag{5.46}$$

where $u_t^i(k)$ denotes the k^{th} element of the vector u_t^i (the input for i^{th} experiment).

$\Xi_\alpha^\dagger$ denotes the pseudo inverse of the matrix Ξ_α defined as follows

$$\Xi_\alpha = \begin{bmatrix} C\left(A+\sum_{k=1}^m N^k u_0^i(k)\right) \\ C\left(A+\sum_{k=1}^m N^k u_1^i(k)\right)\left(A+\sum_{k=1}^m N^k u_0^i(k)\right) \\ \vdots \\ C\prod_{j=0}^{\alpha-1}\left(A+\sum_{k=1}^m N^k u_j^i(k)\right) \end{bmatrix} \tag{5.47}$$

The pseudo inverse $\Xi_\alpha^\dagger$ exists if and only if the bilinear system is observable (for the conditions of controllability and observability of bilinear systems refer to [Rugh 1981]).

5.5 Illustrative Examples

In this section,we first identify a QSS by using a single experiment. Second, we consider an example of QSS identification in the case of multiple short data sets.

We define the model accuracy as the Percent Variance Accounted For ($\%VAF$)

$$\%VAF \triangleq \left(1 - \frac{\sum_{t=1}^L (y_t-\hat{y}_t)^T (y_t-\hat{y}_t)}{\sum_{t=1}^L (y_t-\bar{y}_t)^T (y_t-\bar{y}_t)}\right) \times 100$$

where $\hat{y}_t$ denotes the estimated output signal, and $\bar{y}_t$ is the mean value of y_t.

5.5.1 Identifying QSS using Single Experiment

In this case, we consider a data set of length $L = 2000$ samples. The first $\frac{L}{2}$ samples will be used for the identification purpose, and the last $\frac{L}{2}$ samples for the validation purpose.

The QSS under consideration is characterized by the following matrices

$$\begin{aligned} A &= \begin{pmatrix} 0.6 & 0 \\ 0 & 0.4 \end{pmatrix}, B = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}, C = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \\ N &= \begin{pmatrix} 0.7 & 0 & 0.3 & 0 \\ 0 & 0.2 & 0 & 0.4 \end{pmatrix} \\ Q &= \begin{pmatrix} -0.4 & 0 & -0.4 & 0 \\ 0 & -0.5 & 0 & -0.3 \end{pmatrix} \end{aligned} \tag{5.48}$$

The input is two dimensional uniform white noise, and the standard deviation of independent noise (v_t) is equal to 0.1. The initial estimated model is a bilinear system obtained by using *CUEDSID* package [Chen and Maciejowski 2000b].

Table 5.1: Accuracy of the initial estimated bilinear system and the optimized QSS

Identification task	Accuracy ($\%VAF$)		
	Bilinear system	Optimized QQS	NNSSIF model
First output	64.4	95.2	52.7
Second output	67.6	96.3	48.5
Validation task			
First output	66.2	97.1	46.9
Second output	66.7	98.4	44.6

We compare the obtained results with a classical model for nonlinear system identification, which is the Neural Network State Space Innovation Form (NNSSIF) Fig. 4.4. The state space model of NNSSIF is formulated as follows

$$\begin{aligned} \hat{x}_t(\theta) &= g\left(\hat{x}_{t-1}(\theta), u_t\right) \\ \hat{y}_t &= C\hat{x}_t(\theta) \end{aligned} \tag{5.49}$$

where $\hat{x}_t(\theta)$ is the state vector and its dimension is n_x, g is the function realized by the neural network and C is a constant matrix.
For this example, the best performance was obtained by $n_x = 10$ and neural network with 16 hidden hyperbolic tangent units in the hidden layer.

To point out the efficiency of the proposed method, we calculate the accuracy of the identified models in the validation task by considering the output signal *without* the measurement noise.

The accuracies reported in Table 5.1 show that the proposed method has efficiently identified the given QSS, and the outperformance of the identified QSS in comparison with NNSSIF model.

To deal with a clear graphical representation, we represent only the data over a window of 150 samples in Fig. 5.2.

5.5.2 Identifying QSS Using Multiple Experiments

In this case, we have simulated QSS (5.48) 10 times ($q = 10$) with random initial conditions. That means the obtained data sets are $\{u_t^i, y_t^i, x_0^i : t = 1, 2, \cdots, L_i \text{ and } i = 1, 2, \cdots, q\}$. We suppose that $L_1 = L_2 = L_3 = L_4 = L_5 = 200$, and $L_6 = L_7 = L_8 = L_9 = L_{10} = 100$. The input for each experiment is a uniform white noise, and the standard deviation of the independent noise (v_t) is equal to 0.1.

In this case, we tried to identify the QSS system by using NNSSIF model, but unfortunately the obtained NNSSIF models were unstable. This is because the data length is short.

In order to show that the obtained model by considering all data sets is more accurate than that one obtained by using only a single experiment, we have proceeded as follows: first, we have used a single experiment (the first one) to identify QSS by using the proposed method for single experiment (without estimating the initial conditions x_0). Second, we have used $q - 1 = 9$ data sets to identify QSS and the initial conditions.

Table 5.2: Accuracy of the identified QSS using a single data set and the identified QSS using multiple data sets

Identification task	Accuracy ($\%VAF$)	
	Using single data set	using multiple data sets
First output	71.6	86.4
Second output	72.5	88.1
Validation task		
First output	72.1	87.9
Second output	73.8	90.6

The validation of two models has been done on the last data set (Fig. 5.3). We suppose that the initial condition of the validation data set is known.

It is clear from Table 5.2 that, in this case of short experiments, using multiple experiments leads to a more accurate model.

5.6 Conclusion

In this chapter, we have proposed a method to identify Quadratic in-the-State Systems (QSS). This method is based on a local parameterization of the state space representation of QSS, and subsequent gradient search in the resulting local parameter space.

A bilinear model has been used to initialize the optimization task. In the reported examples this procedure seems to work properly.

Furthermore, the algorithm has successfully applied to identify QSS in the case of single experiment as well as the case of multiple short experiments.

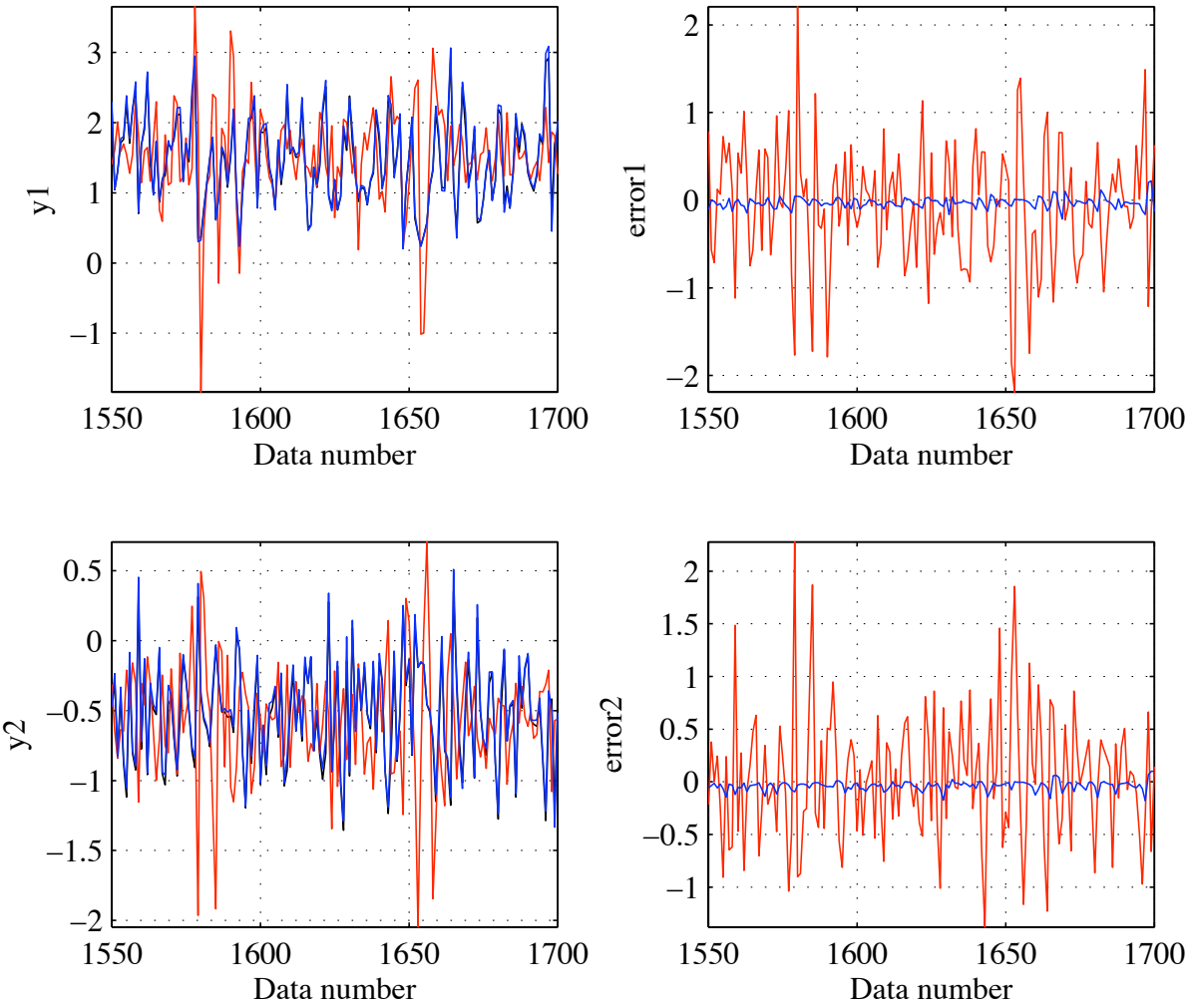


Figure 5.2: The real outputs of QSS *without noise* (black line). The outputs of the identified QSS (blue line) and NNSSIF model (red line) and their errors are superimposed.

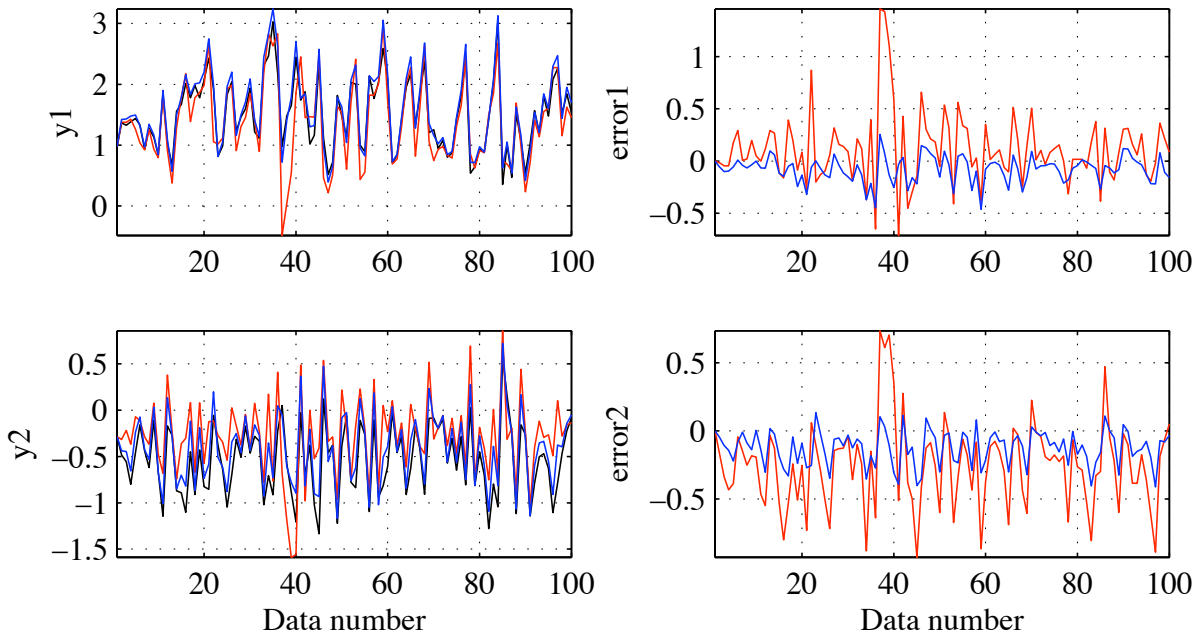


Figure 5.3: The real outputs of QSS *without noise* (black line). The outputs of the identified QSS using all data sets (blue line) and the identified QSS using a single data set (red line) and their errors are superimposed.

Résumé du chapitre 6
Modélisation et synthèse de la locomotion humaine

La locomotion est considérée comme un mouvement essentiel parmi les mouvements humains. Cependant, ce mouvement qui nous apparait simple et instinctif, est un sujet ambitieux dans plusieurs domaines de recherche comme la robotique, les neurosciences, la bio-mécanique et l'animation virtuelle. À noter que la plupart des approches de synthése de mouvements humains s'appuie sur un traitement d'une base de données, obtenue par capture de mouvement, dans l'objectif de fournir un modéle de contrôle qui permet de réaliser un nouveau mouvement non enregistré.

Durant ces dernières années, synthétiser la locomotion humaine a été un domaine actif de recherche. [Wiley and Hahn 1997] utilisent une base de données de capture de mouvement à partir de laquelle ils produisent un nouveau mouvement par les méthodes classiques d'interpolation de l'animation graphique. [Pettre and Laumond 2006] utilisent une technique inspirée de la méthode précédente, la différence principale étant que leur espace d'interpolation est l'espace des vitesses instantanées (l'espace de contrôle) et l'espace des positions.

À la différence des approches précédentes, nous proposons une nouvelle méthode pour résoudre le problème de la synthèse et de la modélisation de la locomotion humaine. Cette méthode est le résultat de l'application de la théorie de l'identification des systèmes dynamiques. Dans notre cas, les signaux d'entrée sont des mouvements enregistrés par un systéme de capture de mouvements.

Notre objectif est d'obtenir un modèle de contrôle qui considère comme entrée la trajectoire du bassin $\in \mathbb{R}^3$ et comme sortie les signaux correspondants à toutes les articulations du corps humain.
Ce probléme d'identification est très complexe, non seulement à cause des non-linéarités inhérentes à ce système, mais aussi à cause de la nature multivariable de ce genre de problèmes. La contribution majeure de notre approche est de montrer que ce systéme peut être modélisé par des systèmes linéaires multivariables grâce à une décomposition de la structure du corps humain en cinq chaînes cinématiques simples et à l'utilisation de la paramétrisation de "Exponential-map" [Grassia 1998]. L'avantage de ce modéle linéaire est qu'il est facile à manipuler et que sa complexité est modérée. Comme les méthodes classiques de l'identification prennent en compte

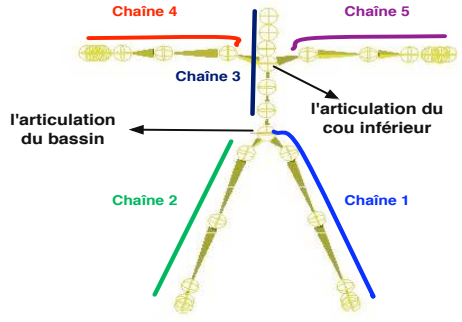


Figure 1 : Une description des cinq chaînes cinématiques de la structure 3D articulée du corps humain.

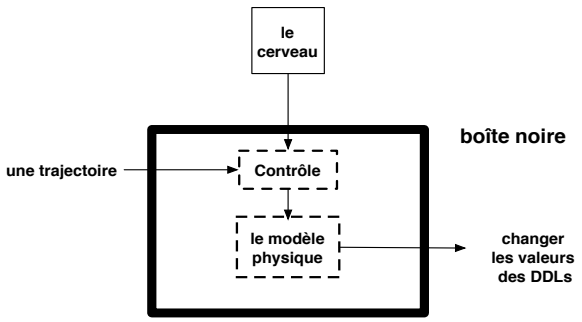


Figure 2 : Une description du modéle du sous-systéme S_i qui est lié à la chaine numéro i.

une seule expérience (pour notre cas une seule trajectoire), nous faisons appel à la méthode d'identification des systémes linéaires dans le cas des expériences multiples développée dans le chapitre 3.

Nous considérons la structure 3D articulée du corps humain présentés Fig. 1. Elle est constituée de 23 articulations. Chaque articulation a trois degrés de liberté (DDL). Ces degrés de liberté sont représentés par les angles d'Euler.

Nous décomposons cette structure en cinq chaînes cinématiques dans le but de les modéliser séparément. La Fig. 2, présente une description du systéme S_i lié à la chaine numéro i. En réalité, ce système est constitué du modèle physique et de son unité de contrôle gérée par le cerveau humain. Le modèle physique est un modèle complexe non-linéaire et multivariable qui pourrait être obtenu à partir d'études biomécaniques. Cependant, en raison du manque d'informations dont nous disposons sur l'unité de contrôle, nous proposons de considérer le modèle physique et son unité de contrôle comme un tout. Le modèle équivalent de l'ensemble peut alors être obtenu par une approche "boite noire".

Afin d'obtenir d'un modéle linéaire d'entrée/sortie, on utilise la paramétrisation "Exponential-map". Cette paramétrisation assure une manipulation propre de la non-linéarité de la dynamique des articulations par un modéle linéaire multivariable. Ainsi, le modéle devient:

$$x_t = Ax_{t-1} + Bu_t$$
$$y_t = Cx_t + v_t$$

où x_t, u_t, y_t et v_t sont respectivement l'état interne du système, le signal d'entrée, le signal de sortie et le bruit des mesures. Ce modèle est alors identifié grâce aux données de capture de mouvement et à la méthode d'identification exposée dans le chapitre 3.

Les premiers résultats obtenus sont très encourageants, et on peut penser que combiner les méthodes d'identification et les méthodes conventionnelles d'analyse des mouvements humains a un avenir prometteur.

"All truly great thoughts are conceived by walking."

Friedrich Nietzsche

6

Synthesizing and Modeling Human Locomotion

Locomotion is considered as the basic human motion. However, this motion, which appears simple and naive to us, is a challenging topic in many sciences domains such as robotics [McMahon 1984; Laumond et al. 2007], neuroscience [Bernstein 1967; Pham et al. 2007], biomechanics [Beckett and Chang 1968] and computer graphics [Multon et al. 1999]. Most of the approaches for synthesizing human locomotion consist in processing a set of motion capture data. Their objective is to provide a control model allowing to perform new not recorded motion.

During the last years, synthesizing human locomotion is an active research area. [Wiley and Hahn 1997] use a database of motion capture to synthesize a new motion by classical interpolation motion editing techniques. [Pettre and Laumond 2006] use a technique inspired from the previous method, the main difference is that their space of interpolation is the instantaneous velocities space (control space). Another method for automating gait generation was proposed in [Sun and Metaxas 2001]. This method represents the locomotion into sagittal-plane, which becomes the space of interpolation. [Kwon and Shin 2005] have proposed a method based on motion modeling in the objective of synthesizing on-line locomotion. Their approach consists of two parts: motion analysis and motion synthesizing. Such a method can be considered in the games and virtual reality applications.

Unlike these previous approaches, in this chapter we propose a new method to resolve the problem of synthesizing and modeling human locomotion.
This method is the result of the comprehensive application of the identification theory and its

applications to dynamic system. In our case, the input signals are the human captured data using a computer vision techniques. These signals are used to train the model. Our objective is to obtain an input-output control model. The inputs of this model are the trajectory of pelvis in $\mathbb{R}^3$ and the outputs are the motion signals of the whole human body.

As such the general identification problem is very challenging because of its nonlinearity and multi-dimensional nature, the core contribution in this chapter is to show that it can be modeled by linear multivariable systems thanks to a decomposition of the human body structure into simple kinematic chains and using exponential-map parameterization.

The advantage of this model is that it is easy to manipulate and his computation complexity is very small. Therefore the locomotion controller can be used for interactive applications acting in real time.

6.1 Black-box Model

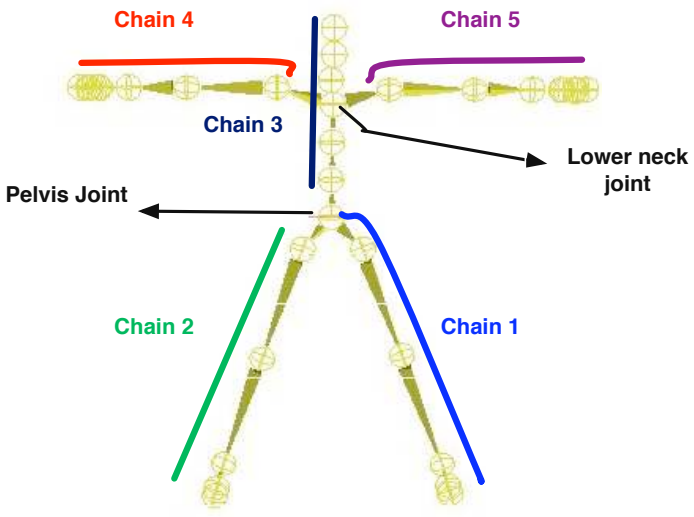


Figure 6.1: A description of the five open kinematic chains of 3D human articulated structure

We consider the 3D human articulated structure presented in Fig. 6.1. It has 23 joints, each joint has three degree of freedoms (DOF), and they are represented by Euler angles.

This structure is decomposed into five kinematics chains. Each chain is considered as subsystem, which might be modeled separately.

Fig. 6.2 shows that the subsystem S_i, in reality, contains the physical model with its control unit managed by human's brain. The physical model is a complex nonlinear multivariable one. It can be obtained from biomechanics studies. However, we do not have any information on the control unit. Therefore we propose to consider the physical model and its control unit aggregate. Such a model can be obtained by black-box approach [Juditsky et al. 1995; Sjoberg et al. 1995].

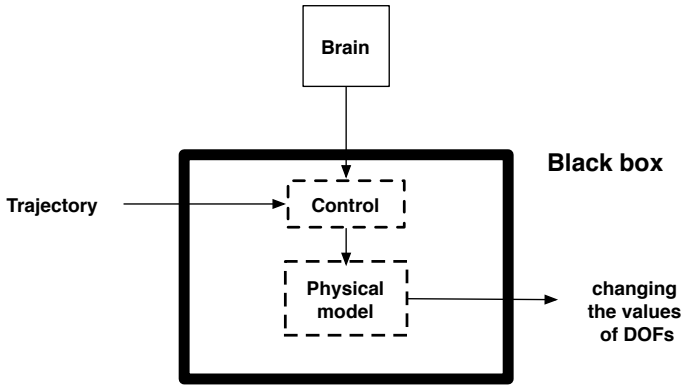


Figure 6.2: A description of the model of the subsystem S_i which is related to the chain number i

As the structure of the model is unknown, it can be modeled by the following general form

$$\begin{aligned} x_t &= f(x_{t-1}, u_t) \\ y_t &= h(x_t, u_t) + v_t \end{aligned} \tag{6.1}$$

where f and h are unknown nonlinear functions, $x_t \in \mathbb{R}^n$ is the internal state of the system, $u_t \in \mathbb{R}^m$ is the input signals, $y_t \in \mathbb{R}^p$ is the output signals and v_t is the measurement noise. The noise v_t is considered to be independent of the input signal u_t.

In general, to identify the model (6.1), one defines a subclass of parametric nonlinear system [Sjoberg et al. 1995]. In our case, such task is difficult due to the multivariable nature of the problem and the lack of informations about the model. To solve this difficulty, we use the exponential-map parameterization [Grassia 1998]. This mapping ensures proper manipulation of nonlinear joint-angles quantities by linear multivariable model [Hsu et al. 2005].
Therefore the model becomes

$$\begin{aligned} x_t &= Ax_{t-1} + Bu_t \\ y_t &= Cx_t + v_t \end{aligned} \tag{6.2}$$

where A, B, C are the constant system matrices. The identification of this linear model should be done by considering multiple trajectories. This procedure will provide a model which takes into account various shapes of trajectories (straight line, left verge, right verge, free navigation, ... , etc).

Therefore, we use the method explained in Chapter 3 to find the linear system (6.2).

6.2 Exponential-map Parameterization

The exponential-map maps a vector in $\mathbb{R}^3$ describing the axis and magnitude of a three DOF rotations to the corresponding rotation. Among the various formulations of the Exp-map [Richard et al. 1994], we have chosen the classical one proposed in [Grassia 1998].

Let $\mathbb{S}^3$ be the set of unit-length quaternion and $SO(3)$ is the subgroup of orthogonal matrices with determinant +1 (rotation matrices). In this formulation, we use first a map from $\mathbb{R}^3$ to $\mathbb{S}^3$, then the standard quaternion map for conversion to $SO(3)$.

We can formulate an exp-map from $\mathbb{R}^3$ to $\mathbb{S}^3$ as follows

$$\begin{aligned} e^{[0,0,0]^T} &= [0,0,0,1]^T \\ \text{and for } v \neq 0 \; e^v &= [q_v, q_w]^T \end{aligned} \tag{6.3}$$

where

$$\begin{aligned} q_v &= \sin(\tfrac{1}{2}\theta)\underline{v} \\ q_w &= \cos(\tfrac{1}{2}\theta) \end{aligned} \tag{6.4}$$

and $\theta = |v|$, $\underline{v} = \frac{v}{|v|}$. This transformation maps v to a unit quaternion representing a rotation of θ about $\underline{v}$.

As most of the motion captured data are represented by Euler angles, a conversion from and to Euler angles is necessary. This conversion can be obtained through the matrix of rotation [Shoemake 1985].
Let us consider the rotation vector $\begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T$ of the Euler rotations angles about the axis X, Y and Z respectively (Roll, Pitch, Yaw). The transformation from Euler angles to Exp-map can be obtain by the following formula

$$\begin{bmatrix} q_v(1) \\ q_v(2) \\ q_v(3) \\ q_w \end{bmatrix} = \begin{bmatrix} \cos(\phi/2)\cos(\theta/2)\cos(\psi/2) + \sin(\phi/2)\sin(\theta/2)\sin(\psi/2) \\ \sin(\phi/2)\cos(\theta/2)\cos(\psi/2) - \cos(\phi/2)\sin(\theta/2)\sin(\psi/2) \\ \cos(\phi/2)\sin(\theta/2)\cos(\psi/2) + \sin(\phi/2)\cos(\theta/2)\sin(\psi/2) \\ \cos(\phi/2)\cos(\theta/2)\sin(\psi/2) - \sin(\phi/2)\sin(\theta/2)\cos(\psi/2) \end{bmatrix} \tag{6.5}$$

The Transformation from Exp-map to Euler angles can be obtain by using the following formula

$$\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} = \begin{bmatrix} \arctan\left(\frac{2(q_w q_v(1) + q_v(2)q_v(3))}{1 - 2\left((q_v(1))^2 + (q_v(2))^2\right)}\right) \\ \arcsin\left(2\left(q_w q_v(2) - q_v(3)q_v(1)\right)\right) \\ \arctan\left(\frac{2(q_w q_v(3) + q_v(1)q_v(2))}{1 - 2\left((q_v(2))^2 + (q_v(3))^2\right)}\right) \end{bmatrix} \tag{6.6}$$

In the sequel of this chapter, the transformation to Exp-map is done using Eq. (6.5) and the Log transformation is done using Eq. (6.6).

6.3 Input/Output Choice

It is well known that to achieve a good identification, the inputs signals should be persistently excited. Therefore considering the cartesian positions of pelvis during the locomotion directly as input signal hands out a poor model.

However, by analyzing the motion signals, we have observed that the three rotations of the pelvis during the locomotion can be assumed as input signals candidates for the chains 1, 2 and 3 and the three rotations of lower neck can be assumed as input signals candidates for the chains 4 and 5 (Fig. 6.1).

This choice is based on the kinematic structure and the frequency spectrum of these signals. Some examples of the pelvis trajectories during locomotion are shown in Fig. 6.3.

The schema of identification becomes as illustrated in Fig. 6.4, where PS denotes preliminary subsystem. This subsystem allows to generate the three rotations of pelvis as outputs, then they are transformed into exp-map representation to play the role of input signals to the subsystems $S1$, $S2$ and $S3$. The input signals of $S4$ and $S5$ are the three rotations of lower neck in exp-map representation. These signals can be obtained from the outputs of identified model of $S3$.

To validate the above assumptions, we should be able to identify the models which approximate the outputs of each subsystem (chain) accurately enough.

6.4 Identification Procedure

To identify the model of locomotion, we should identify the models of subsystems PS and $(S_i : i = 1, \cdots, 5)$. The schema of identification process is illustrated in Fig. 6.4. *Recall that all data sets should be considered in the identification task*, therefore we use the method explained in Chapter 3 to deal with multiple data sets. We summarize the identification algorithm as follows

1. **Identifying the model of PS:** The inputs of this model are the cartesian positions of pelvis $X = [x_t, y_t, z_t]^T$ and the outputs are the three rotations of Pelvis $\Theta = [\theta_t^x, \theta_t^y, \theta_t^z]^T$. Estimating Θ can be directly done from the trajectory of pelvis (i.e θ_t^x is the tangent angle of the trajectory in the plane {y,z}).
2. **Identifying the models of S_i:** To solve this problem, we use PO-MOESP method with the corresponding inputs-outputs signals as illustrated in Fig. 6.4.
   For example, to identify the subsystem S_1, we should first define the input-output signals. In fact, the input signals of S_1 are the output signals of the previous identified model (PS) transformed to exp-map representation.
   The output signals S_1 of are the exp-map representation of the Euler angles of chain 1 (Fig. 6.1). Second, we apply the identification method explained in Chapter 3 to estimate the system matrices (A, B, C).

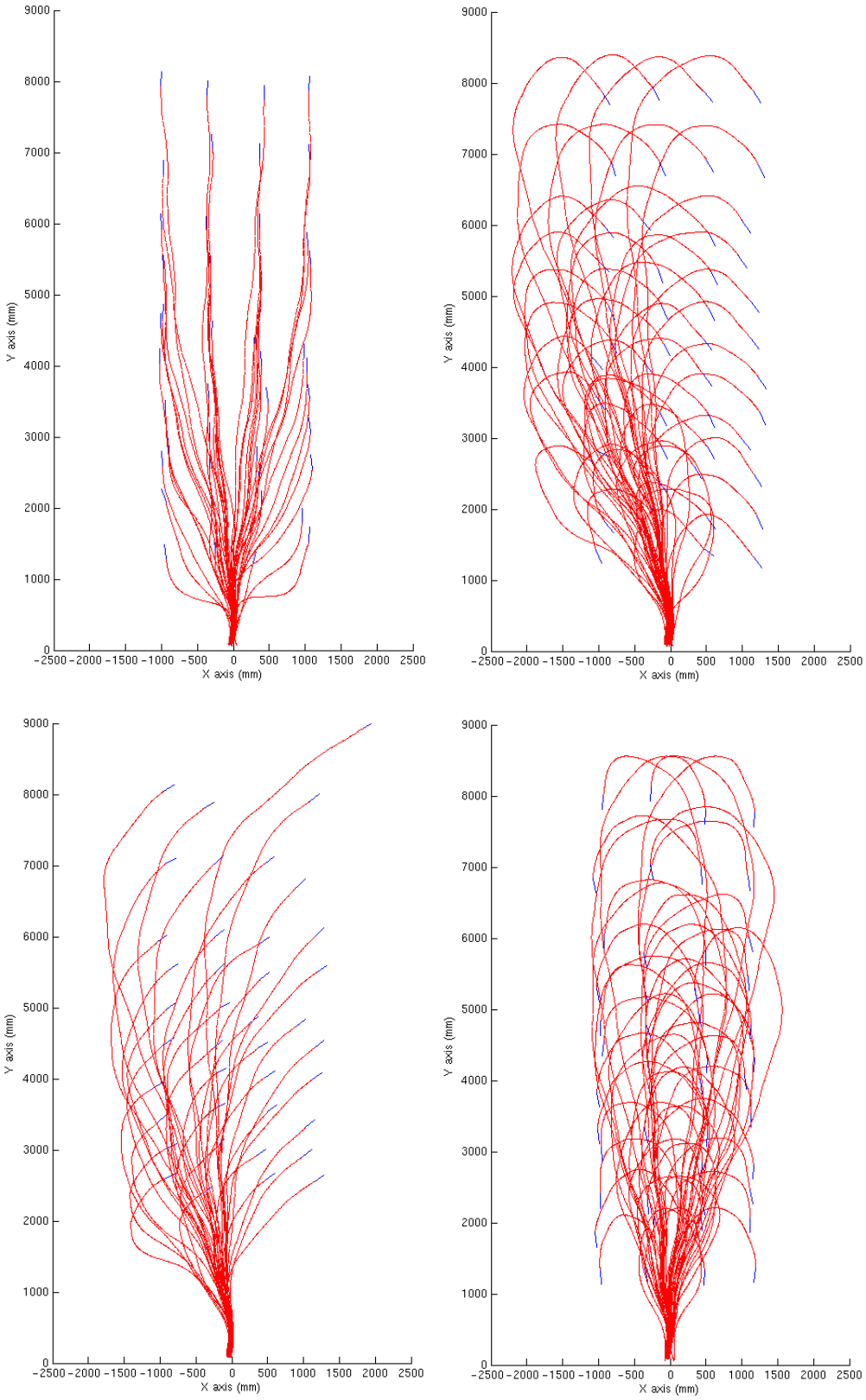


Figure 6.3: Examples of the pelvis trajectories during locomotion in the plane $\{X, Y\}$.

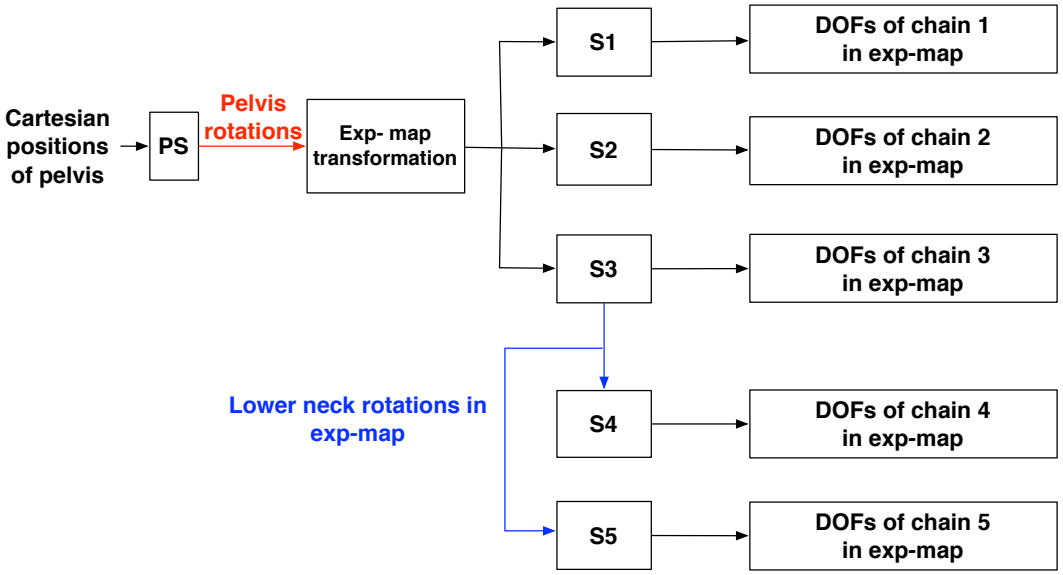


Figure 6.4: An overview of the identification schema

The above explained algorithm of identification provides a model, which takes as input a real trajectory (a trajectory obtained from motion capture) of pelvis and gives the motion signals of whole human body as outputs. As the implementation of our model will be done on an artificial trajectory $\bar{X}_t = [\bar{x}_t, \bar{y}_t, \bar{z}_t]^T$ (e.g. a composition of Bézier curves), a preprocessing of the trajectory is needed.

In fact, a real trajectory can be decomposed into two parts as follows

$$X_t = \begin{bmatrix} x_t \\ y_t \\ z_t \end{bmatrix} = \begin{bmatrix} \bar{x}_t \\ \bar{y}_t \\ \bar{z}_t \end{bmatrix} + \begin{bmatrix} \tilde{x}_t \\ \tilde{y}_t \\ \tilde{z}_t \end{bmatrix} = \bar{X}_t + \tilde{X}_t \tag{6.7}$$

where $\bar{X}_t$ is the main trajectory of pelvis and $\tilde{X}_t$ is related to character zig-zags relative to his trajectory [Gleicher 2001]. For that, transforming an artificial trajectory into a real one can be done by identifying $\tilde{X}_t$. Extracting $\bar{X}_t$ from the real trajectories can be done by filtering x_t, y_t and z_t through a low pass filter, so we obtain $\tilde{X}_t$.

As $\tilde{X}_t$ is related to character zig-zags relative to his trajectory, $\tilde{X}_t$ consists of oscillated signals. Such signals can be modeled by a subspace representation without input sequence. This subclass of subspace representation has the following form

$$\begin{aligned} z_t &= A z_{t-1} \\ \tilde{X}_t &= C z_t + v_t \end{aligned} \tag{6.8}$$

where z_t is the internal state. To estimate the matrices A and C we can use the same identification method explained in Chapter 3.

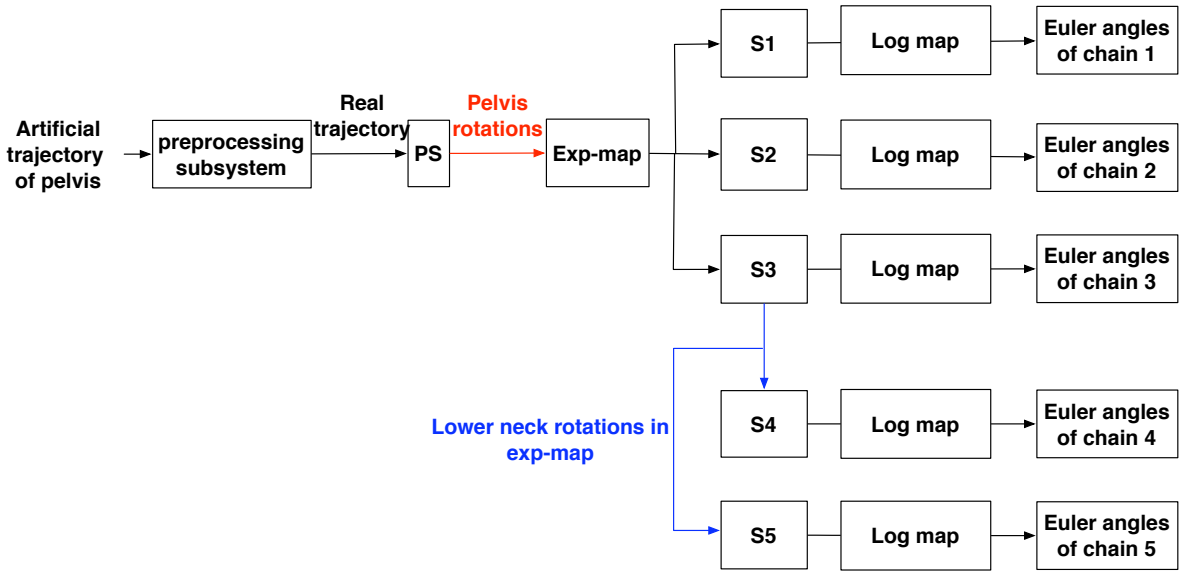


Figure 6.5: An overview of the application of the locomotion controller

6.5 Experimental Results

In this section, we represent some results obtained by our method. We have used a database of motion capture. This database corresponds to one individual character. Fig. 6.5 represents a description of the application of locomotion controller using the identified models. We note that the dimensions of models $(S_i : i = 1, 2, \cdots, 5)$ subsystems are 10, 10, 15, 15, 10 respectively. The dimension of the linear system (6.8) is 20.

The computing time for modeling and synthesizing was 1250 sec using a PC with 1.5 GHz processor and 512 MByte of RAM.

We have validated the obtained model of locomotion on two examples

1. First example: In this example, the trajectory of pelvis has been generated artificially (straight line and arc of circle). Fig. 6.6 illustrates screenshots of the obtained result.
2. Second example: In order to verify the generality of the identified model, we consider a real trajectory corresponding to another character. This trajectory has been normalized such as the height of pelvis of this character becomes equal to that one of the identified character. Note that, in this example the preprocessing subsystem is not used.
   Fig. 6.7 illustrates screenshots of the obtained result.

The most observed visible artifact is the footskate to account for the kinematic constraints imposed by the environment are not included in the identified model. However, such an artifact can be corrected by existing methods if the footplants are annotated [Kovar and Gleicher 2002; Hsu et al. 2005] as postprocessing.

6.6 Conclusion

Synthesizing human motion signals is difficult because of its multi-dimensional and nonlinear nature. However, the locomotion is a synchronized motion, which means that these signals are related. Using this property, we have proposed a new method for modeling the human locomotion and identifying this model. The input signals of our model is the trajectory of pelvis and the outputs are the corresponding motion signals of whole human body. To identify this system, we considered it as black-box for which we proposed an adapted method of identification using motion capture.

The main advantages of our method comparing to other methods are

1. The model takes as input the trajectory of pelvis $\in \mathbb{R}^3$ and generates the whole motion of the 3D human articulated structure. As a consequence, the complicity of controlling the degree of freedoms of virtual actor is reduced to the control of the three elements of the Cartesian position of pelvis. One application is to control the trajectory of pelvis using a joystick, for instance, and the virtual actor is then animated using our algorithm.
2. As the identified models are linear, their computation complexity is very small. Therefore, the proposed algorithm can be used for interactive graphic applications.

We showed that the identified model allows to generate various human locomotion in a fast and efficient way.

The first obtained results are encouraging, and we believe that combining the techniques of system identification and proven techniques of motion editing has a bright future.

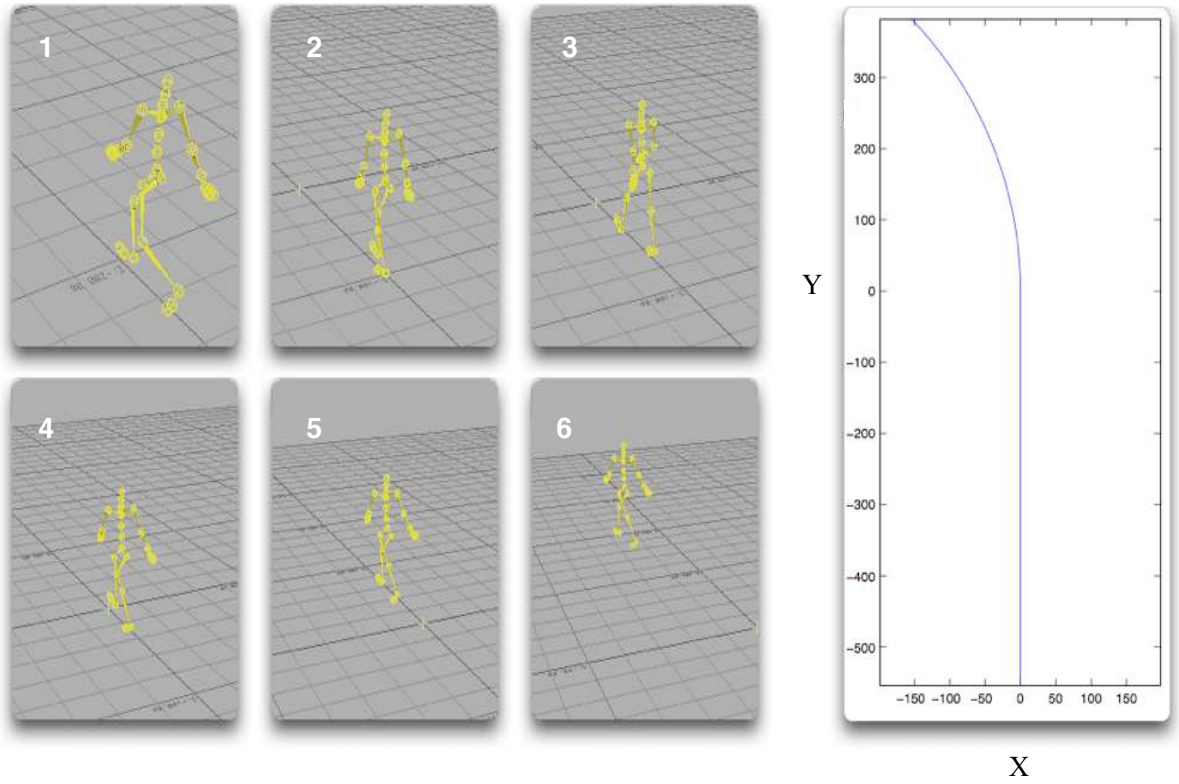


Figure 6.6: First example: screenshots of the application of our locomotion model on an artificial trajectory of pelvis (straight line and arc of circle)

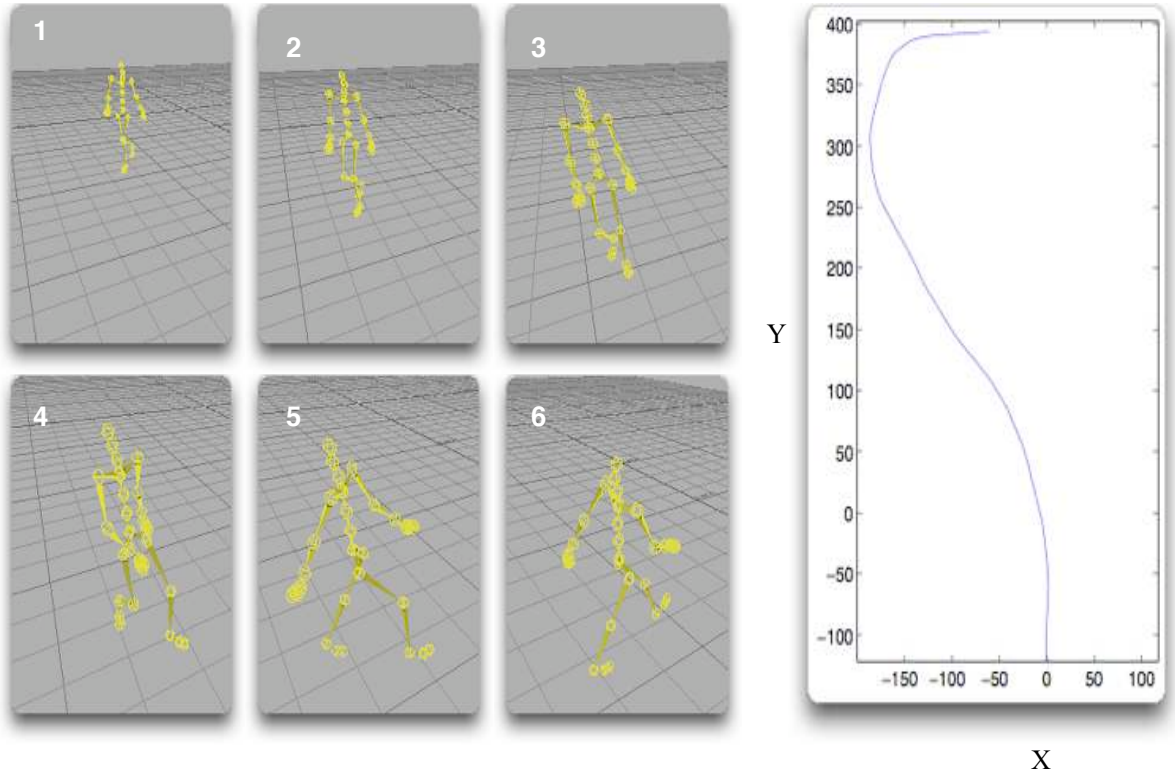


Figure 6.7: Second example: screenshots of the application of our locomotion model on a real trajectory of pelvis

Part II

Humanoid Robot Control

Résumé du chapitre 7
Optimisation des mouvements des robots humanoïdes

Les robots humanoïdes sont sortis du cadre des histoires de science fiction et de cinéma. L'avancée technologique actuelle rend les robots humanoïdes plus robustes et de plus en plus autonomes. Les robots humanoïdes sont non seulement des sujets des recherche très avancés, mais aussi des nécessités pour les pays développés où le vieillissement de la population réduit considérablement la main-d'oeuvre.

Dans le chapitre 7, nous nous intéressons à l'optimisation des mouvements des robots humanoïdes. Ce domaine de recherche est un domaine très attractif du fait que l'optimisation des mouvements peut prendre en compte plusieurs critères comme la stabilité, la minimisation d'énergie, la priorité des tâches, ..., etc. Du fait de cette optimisation, on peut penser que les mouvements optimisés seront en particulier plus robustes.

L'approche utilisée ici s'alimente de mouvements initiaux obtenus par des méthodes de planification cinématique. Ces méthodes peuvent prendre en compte les limites articulaires comme contraintes supplémentaires. En revanche, elles ne sont pas capable de garantir que les mouvements obtenus respectent les limites des couples.

D'après l'analyse des mouvements humains [Winter 1990; Yamaguchi 1990], on constate que ceux-ci minimisent l'énergie consumée ce qui leurs donnent leur régularité.

Par ailleurs, il a été mis en évidence qu'une approximation fiable de l'énergie métabolique humaine est obtenus en considérant l'ensemble des valeurs des couples appliqués sur les articulations. Ainsi, en minimisant la norme Euclidienne des couples appliqués sur les articulations du robot humanoïde, nous avons pu rendre ces mouvements plus réguliers.

Les contributions majeurs du chapitre 7 sont :

1. Un algorithme d'optimisation des mouvements des robots humanoïdes a été développé. Cet algorithme prend comme entrée les mouvements calculés par des méthodes de planification cinématique et donne en sortie des mouvements optimisés et dynamiquement stables (le robot ne chute pas).
2. Les paramètres à optimiser dans la méthode proposée sont les angles des articulations. De ce fait, la méthode est adaptée pour les robots qui sont contrôlés à travers directement par

ces angles (asservissement en position). Notons que c'est le cas pour le robot humanoïde HRP-2.

3. Le gradient de la fonction d'objectif est obtenu analytiquement grâce à un algorithme efficace pour le calcul de la dynamique du robot humanoïde. Cet algorithme permet de calculer la dérivée des couples appliqués sur les articulations par rapport à la position, à la vitesse et à l'accélération des articulations.

La minimisation des couples appliqués sur les articulations du robot humanoïde nous conduit au problème d'optimisation suivant:

$$\min_{\mathbf{q_t},\dot{\mathbf{q}}_\mathbf{t},\ddot{\mathbf{q}}_\mathbf{t}} \int_{t_0}^{t_f} {\tau_t}^T \tau_t dt$$

sous les contraints

Contraintes permanentes

$$M(\mathbf{q_t})\ddot{\mathbf{q}}_\mathbf{t} + C(\mathbf{q_t},\dot{\mathbf{q}}_\mathbf{t}) = \tau_t \qquad \text{(Équation de la dynamique)}$$
$$\mathbf{q_{t_0}} = \mathbf{q_0}, \dot{\mathbf{q}}_{\mathbf{t_0}} = \mathbf{0}, \ddot{\mathbf{q}}_{\mathbf{t_0}} = \mathbf{0} \qquad \text{(Configuration initiale)}$$
$$\mathbf{q_{t_f}} = \mathbf{q_f}, \dot{\mathbf{q}}_{\mathbf{t_f}} = \mathbf{0}, \ddot{\mathbf{q}}_{\mathbf{t_f}} = \mathbf{0} \qquad \text{(Configuration finale)}$$
$$\tau^- \leq \tau_t \leq \tau^+ \qquad \text{(Limites des couples)}$$
$$\dot{\mathbf{q}}^- \leq \dot{\mathbf{q}}_\mathbf{t} \leq \dot{\mathbf{q}}^+ \qquad \text{(Limites des vitesses angulaires)}$$
$$\mathbf{q}^- \leq \mathbf{q_t} \leq \mathbf{q}^+ \qquad \text{(Limites angulaires)}$$

Contraintes temporaires

$$V_{\text{pied en contact}} = 0$$
$$\dot{V}_{\text{pied en contact}} = 0$$
$$\mathcal{G}_{pied} = \mathcal{G}^{ref}_{pied} : \ \mathrm{t_{c^1}}, \cdots, \mathrm{t_{c^p}} \qquad \text{(Empreintes des pieds)}$$

où τ_t désigne le vecteur des couples appliqués. $\mathbf{q_t}$, $\dot{\mathbf{q}}_\mathbf{t}$ et $\ddot{\mathbf{q}}_\mathbf{t}$ sont respectivement les vecteurs des angles, des vitesses et des accélérations des articulations du robot.

Les contraintes permanentes sont composées des contraintes physiques du robot humanoïde, de la contrainte d'évolution dans le monde réel (l'équation de la dynamique) et également des contraintes imposées par les configurations initiales et finales. Les contraintes temporaires sont les contraintes de la stabilité dynamique (vitesse et accélération du pied en contact sont nulles) et les contraintes imposées par le planificateur (empreintes des pieds).

Dans le chapitre 7, nous proposons une méthode efficace pour résoudre ce problème d'optimisation.

Enfin, la méthode a été validé sur la plate-forme HRP-2 № 14 mettant ainsi en exergue son efficacité.

"All that is human must retrograde if it does not advance."

Edward Gibbon

7

Humanoid Motion Optimization

Few years ago, talking about humanoid robots was some kind of science fiction. The recent technological advancement has made this dream a reality. Actually, the ability of humanoid robots to execute complex tasks increases rapidly.
The latest trends in humanoid research are to increase their autonomous behavior as well as improving the stability and smoothness of the planned motions.

Optimizing motions to improve their performance is an active research subject in recent years. In virtual reality, [Lo and Metaxas 1999] have proposed a method based on optimal control theory within a recursive dynamics framework. The objective of their work is to simulate dynamically-correct astronaut motions by minimizing joint torques.

In robotic research, [Guilamo et al. 2006] consider the optimization of manipulability trajectories. The optimal solution is in kinematic sense and does not take into account the dynamic constraints.

[Steinbach 1997] gives an overview of the optimization of motions in robotic using inverse dynamic model. [Lee et al. 2005] proposed to use the Newton and quasi-Newton optimization algorithms for dynamics-based robot movement generation. An analytical formulation of the dynamic equation is proposed. The dynamic equation makes use of Lie group and Lie algebra.

[Sentis and Khatib 2006] have proposed a whole-body control framework for humanoids. This framework integrates task-oriented dynamic control while complying with humanoid physical constraints. The controllers are calculated in the operational space [Khatib 1987] at multiple levels. As it is known, these controllers provide the torques which should be applied on each

joint, that means the humanoid robot should be controlled by joint torques. On the contrary, HRP-2 platform and many humanoid robots are controlled by joint positions. Although the use of forward dynamics methods can give the associated joint positions with the calculated joint torques, this method is time consuming and is not numerically efficient.

In this chapter, our objective is to smooth pre-calculated humanoid motions. These motions can be provided by kinematic planning methods, which can take into account the limits of joint angles. On the other hand, they can not guarantee that the calculated motions do not violate the torque limits.

Studying human movements [Winter 1990; Yamaguchi 1990] has brought out a connection between minimizing energy dissipation and forces, and the smoothness of human movements. On account of the complexity of calculating the energy dissipation, one can use an approximative prediction of it. A good predictor of human's metabolic energy is proven to be the joint torques [Skrinar et al. 1983]. As the humanoid robot is supposed to realize human-like motions, our goal can be achieved by minimizing the joints torques during the planned motion.

The contributions in this chapter are:

- Developing an optimization framework for humanoid robot motions. This framework takes as input a pre-calculated motion, which is provided by motion planning techniques. The output is an optimized and stable motion.
- The proposed method uses the inverse dynamic formulation and the quantities to be optimized are the joint positions. As a consequence, the humanoid robot can be controlled directly in the joint space and not in torque control space. Therefore the method is well adapted for a position controlled humanoid like HRP-2 platform.
- By using an efficient dynamic algorithm, we can calculate the derivative of joint torques with respect to joint position, velocity and acceleration $(q, \dot{q}, \ddot{q})$ analytically. This procedure is similar to the procedure used in [Lo and Metaxas 1999; Lee et al. 2005]. However, the problem of virtual human stability and ground reaction forces modeling are not considered in [Lo and Metaxas 1999; Lee et al. 2005]. Moreover, in [Lee et al. 2005] the optimization problem is solved by the classical Newton algorithm which transforms the inequality constraints into equality constraints by adding slack variables. This procedures increases the number of variables rapidly and the optimization algorithm converges very slowly.
- The validation of the proposed method pointed out that the optimized motion is smoother than the pre-calculated one.

The remainder of this chapter is organized as follows. In Section 7.1 the kinematic structure of HRP-2 humanoid robot, and the definition of active and passive parts are introduced. An overview of the algorithm of recursive multibody dynamics is given in Section 7.2. In Section 7.3 the optimization problem is formulated. In Section 7.4 the discretization of configuration space and solving the optimization problem are pointed out. Experimental results are given in Section 7.5 and Section 7.6 concludes the chapter.

7.1 Humanoid Robot: Kinematic Structure

The kinematic structure of the humanoid robot HRP-2 [Kaneko et al. 2004] is given in the Fig. 7.1. In this structure the degree of freedoms are presented by cylinders. The structure contains 30 degree of freedoms.

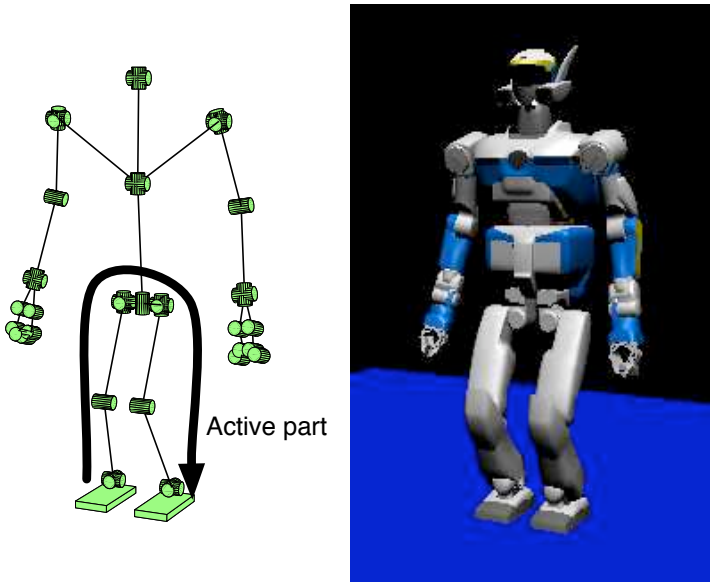


Figure 7.1: Description of HRP2 kinematic structure.

Our objective is to optimize pre-calculated humanoid motions and improve their stability. The pre-calculated motions are the output of motion planning algorithms [Yoshida et al. 2005; Yoshida et al. 2006]. These motions are generally complex, e.g. transporting an object and guaranteeing the collision avoidance [Yoshida et al. 2006]. Such motions usually use the upper part of humanoid robot to execute the desired task and the lower part is mainly responsible of the locomotion and maintaining the stability of humanoid robot.

For that, we divide the kinematic structure into two parts. The first part is the active part, which is the lower part. The upper body is regarded as passive part.
In order to achieve our objective, we optimize only the values of the degree of freedoms of the active part. The passive part will be taken into account in the calculation of the dynamic equations without modifying their angular values. The active part consists of 12 degree of freedoms.

The joints of the active part are called the active joints, and the joints of the passive part are called the passive joints.

7.2 Recursive Multibody Dynamics

Even though the fundamental mathematical frameworks for rigid body dynamics have been established since the 18^{th} century (Lagrange-Euler and Newton-Euler formulations), studying and developing new and efficient implementation and algorithms to control and calculate the dynamic equation of many types of classical robots (rigid, flexible, mobile and humanoid) is still nowadays one of most active research fields [Nakamura 1991; Sciavicco and Siciliano 2000a; Wit et al. 1996].

In this section, we consider the method presented in [Park et al. 1995]. This method proposes to write the recursive multibody dynamics for serial open or branched kinematic chains using Lie group and Lie algebra. The main advantage of this formulation is to relate the joint torques and joint angles explicitly. Therefore the differentiation of joint torques with respect to joint angles can be done analytically.

Let us define the Lie groups $SO(3)$ and $SE(3)$, which denote the orthonormal matrix Θ in $R^{3\times3}$ and the homogeneous transformation group respectively. The Lie algebra of $SO(3)$ and $SE(3)$ are denoted $so(3)$ and $se(3)$ respectively. The operators defined on these groups are: skew, matrix exponential, adjoint map $Ad_G(.)$, dual adjoint $Ad^*_G(.)$, Lie bracket $ad_g(.)$ and dual Lie bracket $ad^*_g(.)$. For more details on Lie group, Lie algebra and the operators definitions see Appendix B.

7.2.1 Forward Kinematics

The kinematics of an open chain can be modeled as a sequence of homogeneous transformation between consecutive joint frames. Let $T_{i-1,i} \in SE(3)$ be the transformation matrix between the frame of link i and the frame of link $i-1$.
The matrix $T_{i-1,i}$ can be written using matrix exponential notation as follows

$$T_{i-1,i} = M_i e^{S_i q_i} \tag{7.1}$$

where $S_i \in se(3)$ is the joint screw written in the coordinate of link $i-1$, q_i is the current position of joint i and M_i is the coordinate transformation between link i and link $i-1$.
Using the above definition of transformation matrix, the end-effector of a kinematic chain can be calculated by the product

$$\begin{aligned} T_{0,n} &= T_{0,1}T_{1,2}\cdots T_{n-1,n} \\ &= M_1 e^{S_1 q_1} M_2 e^{S_2 q_2} \cdots M_n e^{S_n q_n} \end{aligned} \tag{7.2}$$

Note that by expressing the matrix of transformation in exponential form, we can calculate its derivative with respect to q_i analytically.

7.2.2 Recursive Inverse Dynamics of Branched Chains

Branched chains are serial open chains with two or more branches leading to two or more tip links [Park et al. 1995; Sohl and Bobrow 2000]. In the branched chains two definitions arise :

- *Parent link*: the link inward (towards the base) from a given link.
- *Child link*: the link or links which are outward (towards the tips) from a given link.

Let us start by introducing some definitions

Definition 7.1 *The spatial velocity* ($V_i \in \mathbb{R}^6$) *of the link i is defined as follows*

$$V_i = \begin{bmatrix} \boldsymbol{v}_i \\ \\ \boldsymbol{\omega}_i \end{bmatrix} = \begin{bmatrix} \frac{d\boldsymbol{X}_i}{dt} \\ \\ \frac{d\boldsymbol{\theta}_i}{dt} \end{bmatrix} \tag{7.3}$$

where $\boldsymbol{X}_t \in \mathbb{R}^3$ *is the Cartesian position of the link i and* $\boldsymbol{\theta}_i$ *is its orientation.*

Definition 7.2 *The spatial acceleration* ($a_i \in \mathbb{R}^6$) *of the link i is defined as follows*

$$a_i = \begin{bmatrix} \frac{d\boldsymbol{v}_t}{dt} \\ \\ \frac{d\boldsymbol{\omega}_t}{dt} \end{bmatrix} \tag{7.4}$$

Definition 7.3 *The spatial inertia matrix* (J_i) *of the link i is defined as follows*

$$J_i = \begin{bmatrix} I_i - m_i[r_i]^2 & m[r_i] \\ -m[r_i] & m.\mathbf{1} \end{bmatrix} \tag{7.5}$$

where I_i *is the inertia of the link i about its center of mass and m is its mass.* r_i *is the vector from the point of application of the force and the center of mass of the link i. Recall that* $[r_i]$ *is the skew operator (see Appendix B for more details).*

⋆ **Spatial velocity of branched chains**:

- **Initialization**: Given V_0.
- **Outward recursion**: loop over all links in depth manner:

$$\begin{aligned} T_{P,i} &= M_i e^{S_i q_i} \\ V_i &= Ad_{T_{P,i}^{-1}}(V_P) + S_i \dot{q}_i \\ a_i &= -ad_{S_i \dot{q}_i}(V_i) \\ b_i &= -ad^*_{V_i}(J_i V_i) \end{aligned} \tag{7.6}$$

where the index P denotes the parent link of link i, $T_{P,i}$ designs the mapping from the link i to its parent P and V_P denotes the spatial velocity of link P.

⋆ **Applied torques on the branched chains**: In order to calculate the inward recursion of forces (F_i) and torques (τ_i), we define the external forces applied on a link j by $\hat{F}_j$.

- **Initialization:** Given the external applied forces on each link $\hat{F}_j$, $\dot{V}_0$ and $\hat{J}_j = 0$ for each tip link.
- **Inward recursion**: loop over all links in reversed breadth

$$\begin{aligned}
\dot{V}_i =& Ad_{T_{P,i}^{-1}}\left(\dot{V}_P\right) + S_i\ddot{q}_i + a_i \\
\hat{J}_i =& J_i + \sum_{j\in C_i} Ad^*_{T_{i,j}^{-1}} \hat{J}_j Ad_{T_{i,j}^{-1}} \\
B_i =& b_i + \sum_{j\in C_i} Ad^*_{T_{i,j}^{-1}} z_j \\
z_i =& \hat{J}_i\left(S_i\ddot{q}_i + a_i\right) + B_i + \sum_{j\in C_i} Ad^*_{T_{i,j}^{-1}} \hat{F}_j \\
F_i =& \hat{J}_i\, Ad_{T_{P,i}^{-1}}\left(\dot{V}_P\right) + z_i \\
\tau_i =& S_i^T F_i
\end{aligned} \tag{7.7}$$

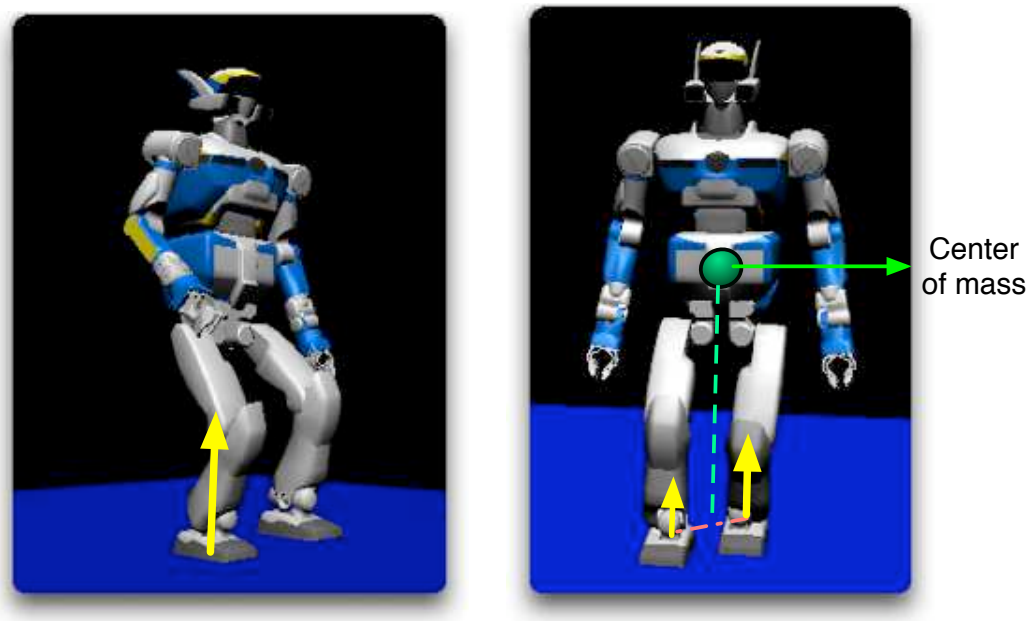
where C_i denotes the child links for link i, and τ_i is the torque applied on the joint i.

7.2.3 Ground Reaction Forces

In the absence of external forces applied on the robot, the only forces are the ground reactions forces. In order to estimate the ground reaction forces, we distinguish two cases

1. Single support: The ground reaction force is applied on the support foot and its magnitude is equal to $M(\boldsymbol{g} - \boldsymbol{a}_G)$, where M is the total mass of the humanoid robot, $\boldsymbol{g}$ is the gravitational acceleration, and $\boldsymbol{a}_G$ is the acceleration of the humanoid robot center of mass.
2. Double support: The ground reaction forces are applied on the two feet. Their magnitudes are proportional to the distance from the projection of the center of mass on the floor and the centers of the support feet.

An illustration of the ground reaction forces in single and double support is shown in Fig. 7.2.



(a) Single support. (b) Double support.

Figure 7.2: Ground reaction forces represented by the yellow arrows.

7.3 Optimization Problem Formulation

The cost function to be minimized is the integral of the Euclidean norm of joint torques. The stability of the humanoid robot can be assured by guaranteeing that the foot, which is in contact with the ground, will be immobile. Such condition can be satisfied by assuring that the spatial velocity and acceleration of the support foot are null.

Let $\mathcal{Q}_t$ be the vector of angular values of joints in the configuration space defined as follows

$$\begin{aligned} \mathcal{Q}_t &= \begin{bmatrix} \mathbf{q_t} \\ \mathbf{q_t^P} \end{bmatrix} \\ &= \begin{bmatrix} q_{1,t} & q_{2,t} & \cdots & q_{n,t} & q^p_{1,t} & q^p_{2,t} & \cdots & q^p_{n_p,t} \end{bmatrix}^T \end{aligned} \tag{7.8}$$

where $\mathbf{q_t}$ and $\mathbf{q_t^P}$ denote the vectors of angular values of active and passive joints respectively.

The optimization problem can be formulated as follows

$$\min_{\mathbf{q_t},\dot{\mathbf{q}}_\mathbf{t},\ddot{\mathbf{q}}_\mathbf{t}} \int_{t_0}^{t_f} {\tau_t}^T \tau_t dt \tag{7.9}$$

subject to

Permanent constraints

$$\begin{aligned}
&S^T F_t = \tau_t && \text{(Dynamic equation)}\\
&\mathbf{q_{t_0}} = \mathbf{q_0}, \dot{\mathbf{q}}_{\mathbf{t_0}} = \mathbf{0}, \ddot{\mathbf{q}}_{\mathbf{t_0}} = \mathbf{0} && \text{(Initial configuration constraints)}\\
&\mathbf{q_{t_f}} = \mathbf{q_f}, \dot{\mathbf{q}}_{\mathbf{t_f}} = \mathbf{0}, \ddot{\mathbf{q}}_{\mathbf{t_f}} = \mathbf{0} && \text{(Final configuration constraints)}\\
&\tau^- \leq \tau_t \leq \tau^+ && \text{(Torque limits)}\\
&\dot{\mathbf{q}}^- \leq \dot{\mathbf{q}}_\mathbf{t} \leq \dot{\mathbf{q}}^+ && \text{(Velocity limits)}\\
&\mathbf{q}^- \leq \mathbf{q_t} \leq \mathbf{q}^+ && \text{(Joint limits)}
\end{aligned}$$

Temporary constraints

$$\begin{aligned}
&V_{support\ foot} = 0\\
&\dot{V}_{support\ foot} = 0\\
&\mathcal{G}_{foot} = \mathcal{G}^{ref}_{foot} \text{ for } \mathrm{t_{c^1}}, \cdots, \mathrm{t_{c^p}} && \text{(Footprint placement)}
\end{aligned}$$

where τ_t, F_t and S are defined as follows

$$\tau_t = \begin{bmatrix} \tau_{1,t} \\ \tau_{2,t} \\ \vdots \\ \tau_{n,t} \end{bmatrix}, F_t = \begin{bmatrix} F_{1,t} \\ F_{2,t} \\ \vdots \\ F_{n,t} \end{bmatrix}, S = \begin{bmatrix} S_1 & 0 & \cdots & 0 \\ 0 & S_2 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & S_n \end{bmatrix} \tag{7.11}$$

τ_t and F_t are the vectors of the applied torques and forces on the joints of active part respectively.
$\tau_{i,t}$ and $F_{i,t}$ denote the value of the applied torque and force on the joint i respectively.
x^- and x^+ denote the minimal and the maximal values of vector x respectively.

$\mathcal{G}_{foot}$ denotes the configuration of the foot in the euclidean space as shown in Fig. 7.3. Using Eq. (7.2), the $\mathcal{G}_{foot}$ can be written as a function of $\mathbf{q_t}$ as follows

$$\mathcal{G}_{foot} = T_{0,foot} \tag{7.12}$$

where $T_{0,foot} = M_1 e^{S_1 q_1} M_2 e^{S_2 q_2} \cdots M_n e^{S_n q_n}$ denotes the transformation matrix of the foot expressed in the global fixed frame. q_1 is the angular value of the ankle associated to the

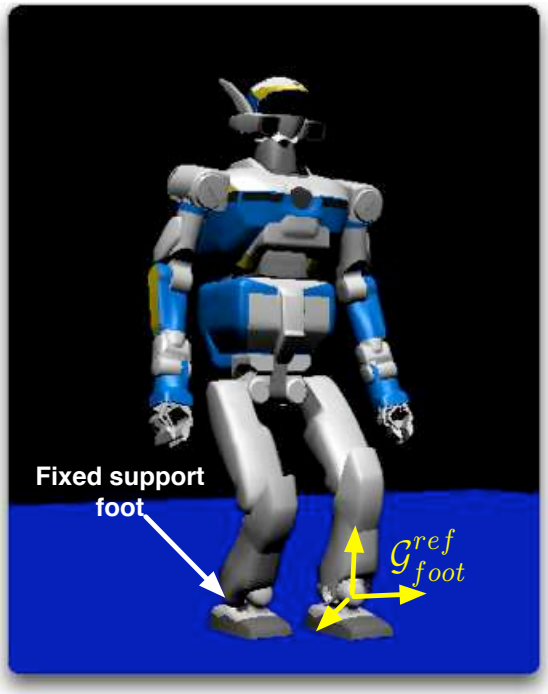


Figure 7.3: Foot configuration.

immobile support foot, and the matrix M_1 is given as follows

$$M_1 = \begin{bmatrix} \mathbf{1} & p_1 \\ \mathbf{0} & 1 \end{bmatrix} \tag{7.13}$$

where $p_1 \in \mathbb{R}^3$ designs the Cartesian position of the ankle joint of the fixed support foot in the global frame. $\mathcal{G}_{foot}^{ref}$ is the planned reference configuration. t_{c^k} denotes the instant of foot contact with the ground.

In order to transform this optimization problem into a classical optimization problem, let us

define

$$X_t = \begin{bmatrix} \mathbf{q_t}^T & \dot{\mathbf{q}}_\mathbf{t}^T & \ddot{\mathbf{q}}_\mathbf{t}^T \end{bmatrix}^T, \quad L(X_t) = \int_{t_0}^{t_f} {\tau_t}^T \tau_t dt$$

$$G(X_t) = \begin{bmatrix} \tau_t - \tau^+ \\ -\tau_t + \tau^- \\ \dot{\mathbf{q}}_\mathbf{t} - \dot{\mathbf{q}}^+ \\ -\dot{\mathbf{q}}_\mathbf{t} + \dot{\mathbf{q}}^- \\ \mathbf{q_t} - \mathbf{q}^+ \\ -\mathbf{q_t} + \mathbf{q}^- \end{bmatrix}, \quad H(X_t) = \begin{bmatrix} \tau_t - S^T F_t \\ \mathbf{q_{t_0}} - \mathbf{q_0} \\ \dot{\mathbf{q}}_{\mathbf{t_0}} \\ \ddot{\mathbf{q}}_{\mathbf{t_0}} \\ \mathbf{q_{t_f}} - \mathbf{q_f} \\ \dot{\mathbf{q}}_{\mathbf{t_f}} \\ \ddot{\mathbf{q}}_{\mathbf{t_f}} \\ V_{support\ foot} \\ \dot{V}_{support\ foot} \\ \mathcal{G}_{foot} - \mathcal{G}^{ref}_{foot} \end{bmatrix} \tag{7.14}$$

Thus the optimization problem (7.9) can be transformed to the following classical form

$$\begin{aligned} &\min_{X_t} \ L(X_t) \\ \text{subject to} \quad & \\ &H(X_t) = 0 \\ &G(X_t) \leq 0 \end{aligned} \tag{7.15}$$

The above optimization problem has been extremely studied in the literature of optimization theory. To solve this optimization problem, one can use the augmented Lagrange multiplier method, which is a very efficient and reliable method [Rockafellar 1974; Lo and Metaxas 1999]. Using the augmented Lagrange multiplier method transforms the optimization problem (7.15) to the minimization of the following function

$$\min_{X_t,\lambda} \tilde{L}(X_t, \lambda) = L(X_t) + \lambda_\psi^T \psi + \frac{1}{2}\sigma \psi^T \psi + \lambda_H^T H + \frac{1}{2}\sigma H^T H \tag{7.16}$$

where $\lambda = \begin{bmatrix} \lambda_\psi^T & \lambda_H^T \end{bmatrix}^T$, $\psi = \max\left\{G(X_t), -\frac{1}{\sigma}\lambda_\psi\right\}$. Then there exist λ^* such that X_t^* is an unconstrained local minimum of $\tilde{L}(X_t, \lambda^*)$ for all σ smaller than some finite $\bar{\sigma}$.
To solve the unconstrained optimization problem of $\tilde{L}(X_t, \lambda)$ with respect to X_t, one can use Gauss-Newton method. Note that the function $\tilde{L}(X_t, \lambda)$ is differentiable in X_t if and only if $L(X_t)$, $H(X_t)$ and $G(X_t)$ are differentiable in X_t. In this case we can write

$$\begin{aligned} \frac{\partial \tilde{L}(X_t, \lambda)}{\partial X_t} =& \frac{\partial L(X_t)}{\partial X_t} + (\lambda_H + \sigma H)^T \frac{\partial H(X_t)}{\partial X_t} + \\ & \max\{0, \lambda_\psi + \sigma G(X_t)\}^T \frac{\partial G(X_t)}{\partial X_t} \end{aligned} \tag{7.17}$$

As λ^* is unknown, an update rule is used

$$\begin{aligned} \lambda_H^{k+1} &= \lambda_H^k + \sigma H(X_t^k) \\ \lambda_\psi^{k+1} &= \lambda_\psi^k + \sigma \psi(X_t^k) \end{aligned} \tag{7.18}$$

where X_t^k is the unconstrained minimum of $\tilde{L}(X_t, \lambda^k)$. Such updating rule will generate a sequence λ^k that will converge to λ^* [Bertsekas 1995]. In practice, a good schedule is to choose a moderate σ^0 and increase it as follows

$$\sigma^{k+1} = \alpha \sigma^k \tag{7.19}$$

where α is between 5 and 10. A threshold $\bar{\sigma}$ is chosen and the update rule of σ stops when σ^k becomes higher than $\bar{\sigma}$.

For more details on the algorithm of augmented Lagrange multiplier method see [Rockafellar 1973; 1974; Bertsekas 1995].

Approximating the gradient function $\frac{\partial \tilde{L}(X_t,\lambda)}{\partial X_t}$ by a numerical difference method is usually used in practice. However, this approach is not only a time consuming method on account of the evaluation of the gradient calculation, but also may not converge well because of the approximation.

As we have mentioned, the main advantage of using the recursive dynamic algorithm explained in Section 7.2.2 is calculating the gradient function analytically in a recursive way.

7.3.1 Gradient Calculation

The objective is to calculate the gradient of the dynamic quantities, such as τ_t, V_t and $\dot{V}_t$.

By considering the vector of parameters $X_t = \begin{bmatrix} \mathbf{q_t}^T & \dot{\mathbf{q}}_\mathbf{t}^T & \ddot{\mathbf{q}}_\mathbf{t}^T \end{bmatrix}^T$, we will start by calculating the derivatives of the operators with respect to any element x of X_t which are q_i, $\dot{q}_i$ or $\ddot{q}_i$.

In order to simplify the expressions of the derivatives, we introduce the symbol δ_{x,q_i} defined as follows

$$\delta_{x,q_i} = \begin{cases} 1 & \text{if } x = q_i \\ 0 & \text{otherwise} \end{cases} \tag{7.20}$$

Thus

$$\begin{aligned} \frac{\partial T_{0,n}}{\partial x} &= T_{0,i}\, S_i \delta_{x,q_i}\, T_{i,n} \\ \frac{\partial Ad_{T_{i-1,i}^{-1}}(Y)}{\partial x} &= Ad_{T_{i-1,i}^{-1}} \left(\frac{\partial Y}{\partial x} \right) + ad_{Ad_{T_{i-1,i}^{-1}}(Y)} (S_i)\, \delta_{x,q_i} \end{aligned}$$

$$\begin{aligned}
\frac{\partial Ad^*_{T^{-1}_{i,i+1}}(Y)}{\partial x} =& ad^*_{Ad_{M_{i+1}}(S_{i+1}\delta_{x,q_{i+1}})}\left(Ad^*_{T^{-1}_{i,i+1}}(Y)\right) + Ad^*_{T^{-1}_{i,i+1}}\left(\frac{\partial Y}{\partial x}\right) \\
\frac{\partial ad_Z(Y)}{\partial x} =& ad_{\frac{\partial Z}{\partial x}}(Y) + ad_Z\left(\frac{\partial Y}{\partial x}\right) \\
\frac{\partial ad^*_Z(Y)}{\partial x} =& ad^*_{\frac{\partial Z}{\partial x}}(Y) + ad^*_Z\left(\frac{\partial Y}{\partial x}\right)
\end{aligned} \tag{7.21}$$

The calculation of the gradient with respect to X_t can be done in a recursive way analogously to the recursive dynamic calculation.

Forward recursion:

- **Initialization**: Given $\frac{\partial V_0}{\partial X_t}$.
- loop over all links in depth manner:

$$\begin{aligned}
\frac{\partial V_i}{\partial X_t} &= \frac{\partial Ad_{T^{-1}_{P,i}}(V_P)}{\partial X_t} + S_i\frac{\partial \dot{q}_i}{\partial X_t} \\
\frac{\partial a_i}{\partial X_t} &= -\frac{\partial ad_{S_i\dot{q}_i}(V_i)}{\partial X_t} \\
\frac{\partial b_i}{\partial X_t} &= -\frac{\partial ad^*_{V_i}(J_iV_i)}{\partial X_t}
\end{aligned} \tag{7.22}$$

Backward recursion:

- **Initialization:** Given $\frac{\partial \hat{F}_j}{\partial X_t}$, $\frac{\partial \dot{V}_0}{\partial X_t}$.
- loop over all links in reversed breadth

$$\begin{aligned}
\frac{\partial \dot{V}_i}{\partial X_t} =& \frac{\partial Ad_{T^{-1}_{P,i}}\left(\dot{V}_P\right)}{\partial X_t} + S_i\frac{\partial \ddot{q}_i}{\partial X_t} + \frac{\partial a_i}{\partial X_t} \\
\frac{\partial \hat{J}_i}{\partial X} =& \sum_{j\in C_i}\frac{\partial Ad^*_{T^{-1}_{i,j}}}{\partial X_t}\hat{J}_jAd_{T^{-1}_{i,j}} + Ad^*_{T^{-1}_{i,j}}\hat{J}_j\frac{\partial Ad^*_{T^{-1}_{i,j}}}{\partial X_t} + Ad^*_{T^{-1}_{i,j}}\frac{\partial \hat{J}_j}{\partial X_t}Ad_{T^{-1}_{i,j}} \\
\frac{\partial B_i}{\partial X_t} =& \frac{\partial b_i}{\partial X_t} + \sum_{j\in C_i}\frac{\partial Ad^*_{T^{-1}_{i,j}}z_j}{\partial X_t} \\
\frac{\partial z_i}{\partial X_t} =& \frac{\partial \hat{J}_i}{\partial X_t}(S_i\ddot{q}_i + a_i) + \hat{J}_i\left(S_i\frac{\partial \ddot{q}_i}{\partial X_t} + \frac{\partial a_i}{\partial X_t}\right) + + \frac{\partial B_i}{\partial X_t} + \sum_{j\in C_i}\frac{\partial Ad^*_{T^{-1}_{i,j}}\hat{F}_j}{\partial X_t} \\
\frac{\partial F_i}{\partial X_t} =& \frac{\partial \hat{J}_i}{\partial X_t}Ad_{T^{-1}_{P,i}}\left(\dot{V}_P\right) + \hat{J}_i\frac{\partial Ad_{T^{-1}_{P,i}}\left(\dot{V}_P\right)}{\partial X_t} + \frac{\partial z_i}{\partial X_t} \\
\frac{\partial \tau_i}{\partial X_t} =& S_i^T\frac{\partial F_i}{\partial X_t}
\end{aligned} \tag{7.23}$$

where as we mentioned, earlier, C_i denotes the child links for link i.

7.4 Discretization of Configuration Space

It is well known that the space of the admissible solutions of the minimization problem (7.9) is very large. In order to transform this infinite dimensional space to a finite one, we can use a basis of shape functions.
Let us consider a basis of shape functions $\mathbf{B}_t$ that is defined as follows

$$\mathbf{B}_t = \begin{bmatrix} B_t^1 & B_t^2 & \cdots & B_t^l \end{bmatrix}^T \tag{7.24}$$

where B_t^i denotes the value of shape function number i at the instant t. The dimension of $\mathbf{B}_t$ is l that defines the dimension of the basis of shape functions.

The projection of the vector of angular values $\mathbf{q_t}$ into the basis of shape functions B_t can be given by the following formula

$$\mathbf{q_t} = Q_B \boldsymbol{B}_t \tag{7.25}$$

where Q_B is a constant matrix.
The derivative $\dot{\mathbf{q}}_\mathbf{t}$ and $\ddot{\mathbf{q}}_\mathbf{t}$ can be then written as follows

$$\begin{aligned} \dot{\mathbf{q}}_\mathbf{t} &= Q_B \dot{\boldsymbol{B}}_t \\ \ddot{\mathbf{q}}_\mathbf{t} &= Q_B \ddot{\boldsymbol{B}}_t \end{aligned} \tag{7.26}$$

In this case, the derivative with respect to each element $Q_B(i,j)$ of the matrix Q_B can be computed using the following formula

$$\begin{aligned} \frac{\partial Y_t}{\partial Q_B(i,j)} &= \frac{\partial Y_t}{\partial X_t} \times \frac{\partial X_t}{\partial Q_B(i,j)} \\ &= \frac{\partial Y}{\partial X_t} \times \left(e_i \otimes \begin{bmatrix} B_t^j \\ \dot{B}_t^j \\ \ddot{B}_t^j \end{bmatrix} \right) \end{aligned} \tag{7.27}$$

where $e_i \in \mathbb{R}^n$,

$$e_i = \begin{matrix} [0 & \ldots & 0 & 1 & 0 & \ldots & 0]^T \\ & & & \uparrow & & & \\ & & & i & & & \end{matrix}$$

and $\otimes$ denotes Kronecker's product operator.

By using the discretization of the configuration space, the optimization problem transforms into the problem of finding the optimal matrix Q_B, which minimizes the function $\tilde{L}(X_t, \lambda)$ in Eq. (7.16).

It remains to define the shape functions B_t^i. In our case, the shape functions should verify the following properties:

1. They are continuous.

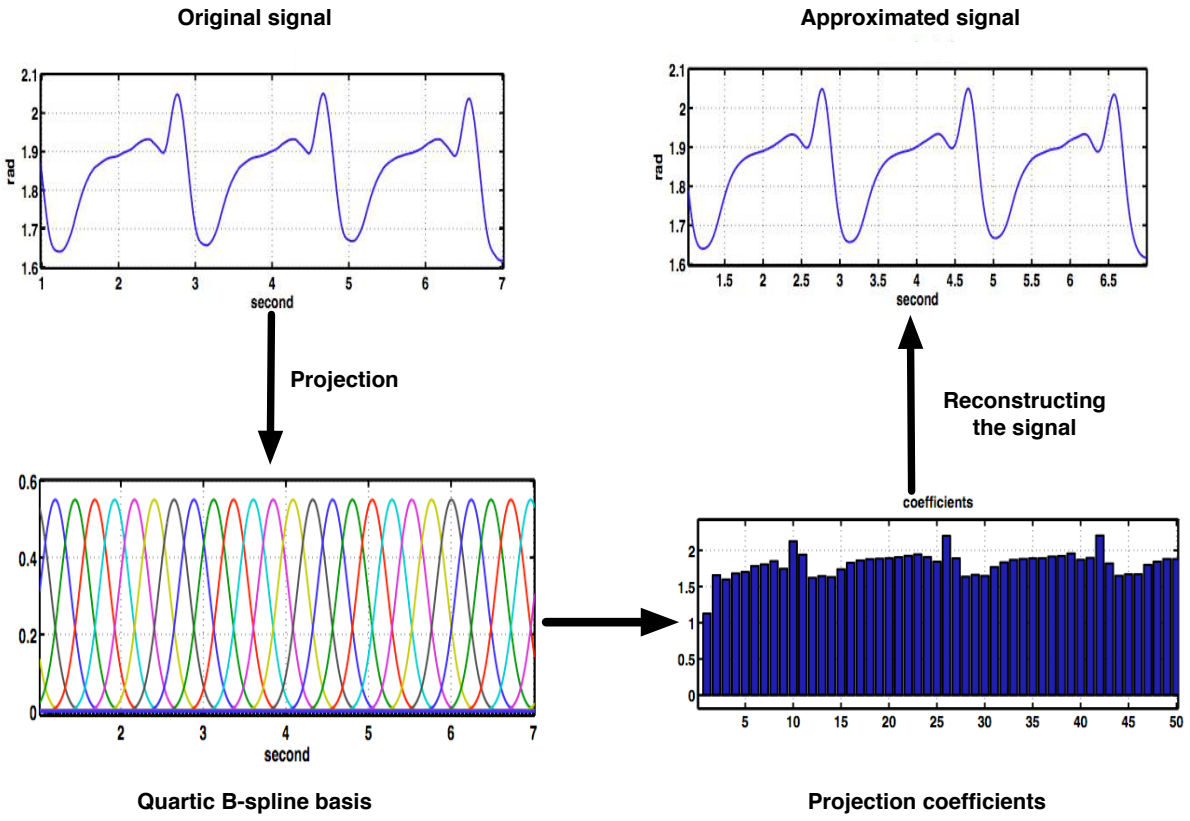


Figure 7.4: Example of using Quartic Bsplines as a basis of shape functions.

2. Their first and second derivatives are continuous.

That means $B_t^i \in \mathcal{C}^2$. Therefore, we suggest to use the *quartic B-spline functions*. Fig. 7.4 shows an example of using quartic Bsplines as a basis of shape functions.

7.5 Experimental Results

The experimental scenario that we have tested to validate the proposed method is the following

1. The robot carries a bar with its right hand.
2. The robot starts walking and depressing the vertical position of its pelvis. At the same time, it lifts up the carried bar.

The characteristics of the carried bar are: length= 2 m, weight= 0.7 Kg, cylindrical form with uniform density distribution. The robot grasps the bar at 0.35 m from its end.

Snapshots of the conducted motion are presented in Fig. 7.5. Note that the objective of using the carried bar is to generate an asymmetry in the kinematic structure, and also to perturb dynamically the motion by the movement of the bar.

To compare the obtained results with the results obtained by the method presented in [Kajita et al. 2003] (an overview of this method is given in Section 8.3.1), we use the real quantities measured by the sensors of the humanoid robot HRP-2.
Fig. 7.6 shows the x coordinate of the Zero Moment Point (ZMP) [Vukobratović and Borovac 2004]. The ground reaction force applied on the left foot are presented in Fig. 7.7. Fig. 7.8 shows the applied torque on the left knee.

From Figures 7.6, 7.7 and 7.8 we conclude that

1. The optimization method smoothes the shape of ZMP, and the oscillations have been avoided. Note that the ZMP trajectory of the optimized motion does not follow the designed ZMP trajectory. This is because the optimization method assures the stability of humanoid robot by guaranteeing that the spatial velocity and acceleration of the support foot are null, and it does not consider the trajectory of ZMP.
2. Using the optimization method avoids the surges in the ground reaction forces applied on the foot.
3. Using the optimization method not only minimizes the joint torques, but also it avoids the surges.

7.6 Conclusion

In this chapter, we have presented an optimization method for humanoid robot motions. The objective of this method is to smooth and improve the stability of the pre-calculated humanoid motions. To achieve this goal, the integral of Euclidean norm of the applied torques on the joints is minimized, and the dynamical stability conditions are transformed into guaranteeing that the spatial velocity and accelerations of the support foot are null.

The method is based on an efficient dynamics algorithm, which allows the calculation of the gradient function with respect to the control parameters analytically. The algorithm makes use of the theory of Lie groups and Lie algebra.

Experimental results using HRP-2 platform are provided to validate the proposed method. These results have pointed out that the proposed method not only smoothes the motion but also yields a dynamically stable motion.

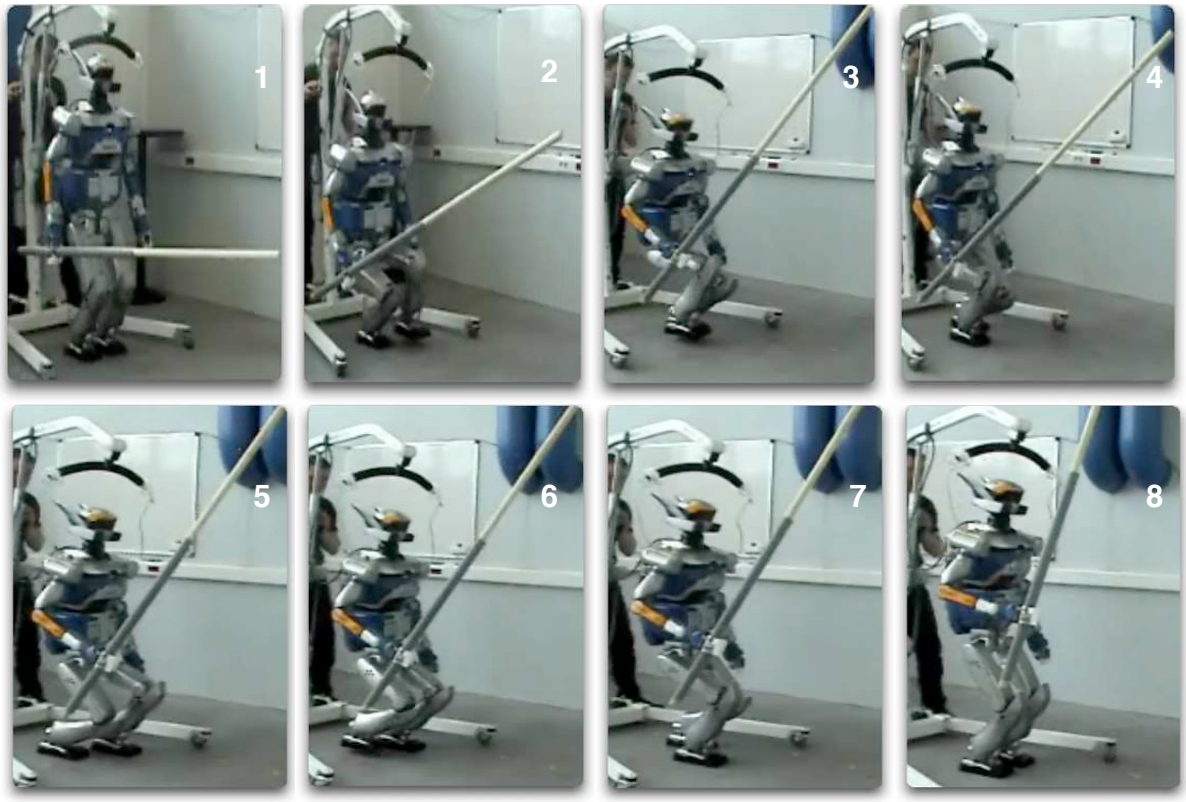
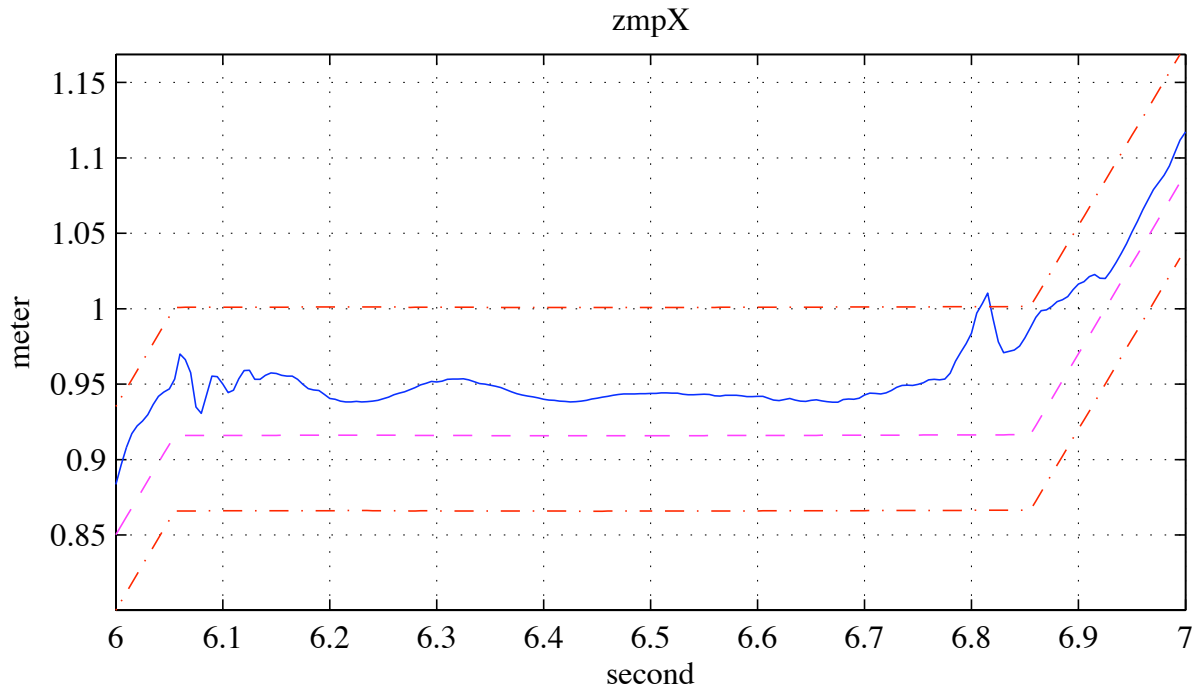
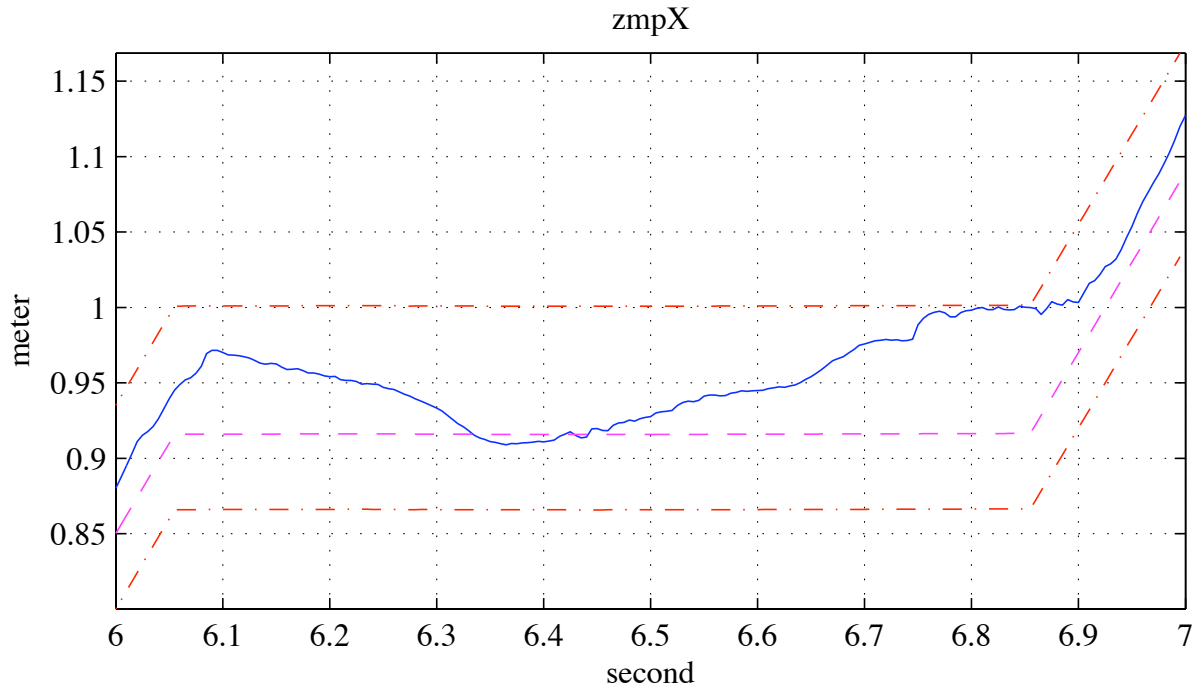


Figure 7.5: Snapshots of the conducted motion.



(a) Before optimization.



(b) After Optimization.

Figure 7.6: x coordinate of ZMP: the solid blue line is the measured ZMP_x, the red dash-dotted lines design the safe stability zone and the magenta dashed line denotes the designed reference. The oscillations around 6.1 and 6.8 seconds disappear after optimization as well as the surge at 6.81 second.

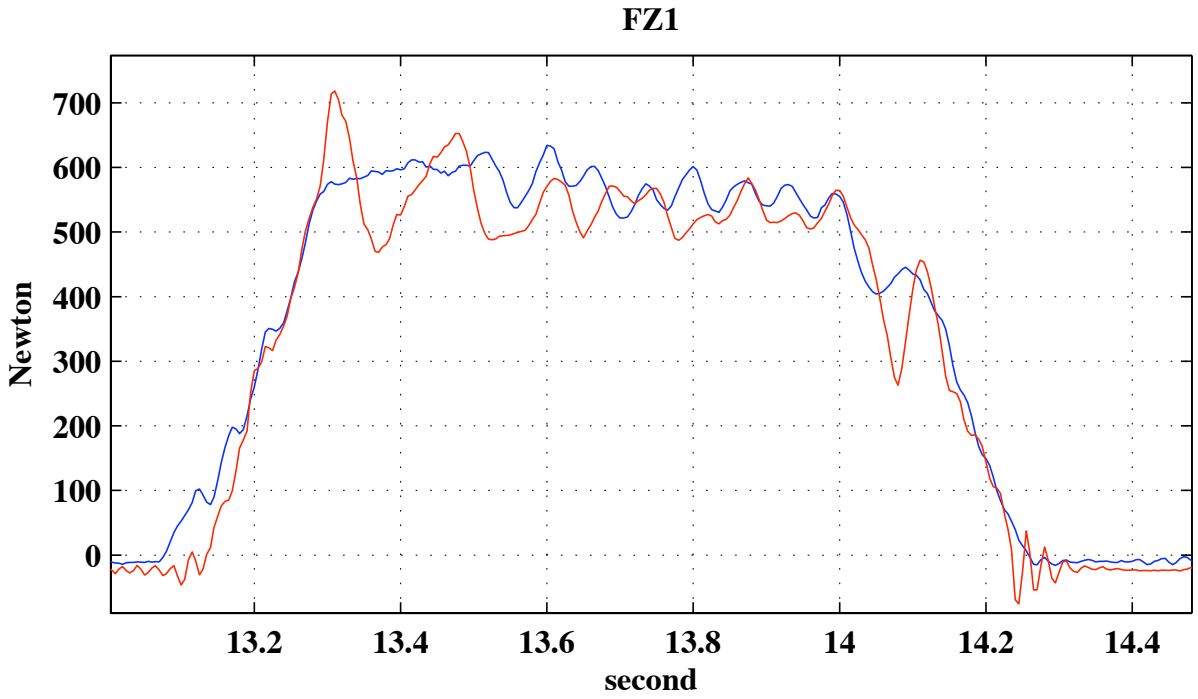


Figure 7.7: Applied ground reaction forces on the left foot: blue line for the optimization method and red line for the classical method. The surge at 13.3 second disappears after optimization.

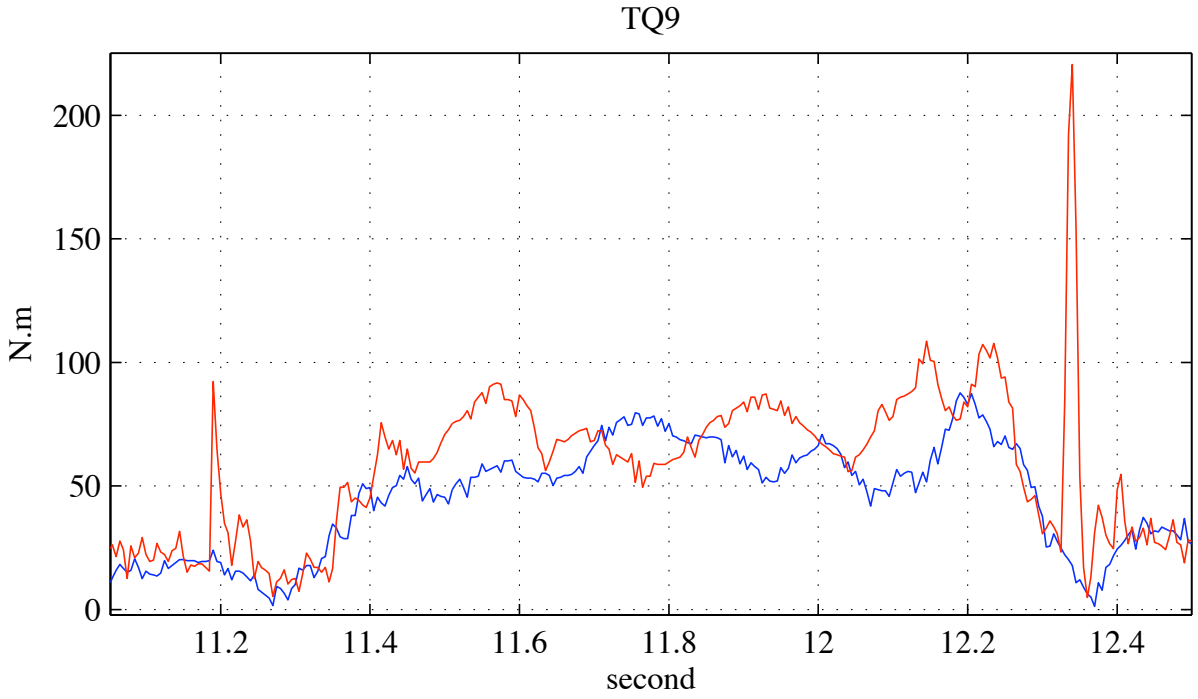


Figure 7.8: Applied torques on the left knee: blue line for the optimization method and red line for the classical method. The surges at 11.2 and 12.35 seconds disappear after optimization.

Résumé du chapitre 8
Imitation des mouvements humains par un robot humanoïde

L'imitation des mouvements humains par un robot humanoïde est plus qu'un simple sujet de recherche. En réalité, il s'agit d'augmenter la capacité des robots humanoïdes pour qu'ils soient capables d'imiter les mouvements humains non seulement pour accroître leur autonomie, mais aussi pour augmenter leur réactivité lorsqu'ils interagissent avec l'homme.

De ce fait, l'imitation des mouvements humains a attiré l'attention des chercheurs. [Inamura et al. 2001] proposent d'abstraire les symboles principaux d'un mouvement humain en utilisant la théorie de mimésis.

[Pollard et al. 2002; Nakaoka et al. 2003] ont proposé une méthode pour transformer un mouvement de danse capturé à un mouvement que le robot peut exécuter. [Safonova et al. 2003] utilisent un mouvement capturé afin de générer un mouvement optimal pour la partie supérieur du robot humanoïde *Sarcos*. D'un manière similaire à la méthode précédente, [Ruchanurucks et al. 2006] proposent une méthode pour générer un mouvement de la partie supérieure d'un robot humanoïde afin d'imiter des mouvements humains préalablement capturés.

Dans le chapitre 8, nous proposons une méthode efficace permettant d'imiter des mouvements humains issus de la capture de mouvement. Les mouvements obtenus respectent les limites physiques du robot humanoïde telles que les limites angulaires, les limites des vitesses et les limites des couples appliqués sur les articulations du robot.

Nous nous focalisons sur l'imitation des mouvements de la partie supérieur du robot. À noter que les mouvements de la partie inférieure du robot pourraient être générés en faisant appel aux méthodes d'extraction de primitives. En ce qui concerne la stabilité dynamique, nous utiliserons une méthode basée sur le contrôle de la trajectoire de ZMP (Zero Moment Point) [Kajita et al. 2003].

La contribution majeure de ce chapitre 8 est de développer un algorithme d'optimisation destiné à engendrer les mouvements de la partie supérieure d'un robot humanoïde, ceci à partir de mouvements humains capturés. Les mouvements ainsi engendrés imitent les mouvements humains tout en respectant les limites physiques du robot humanoïde.

La formulation du problème de l'imitation comme un problème d'optimisation nous conduit au problème suivant:

$$\min_{\mathbf{q_t},\dot{\mathbf{q}}_\mathbf{t},\ddot{\mathbf{q}}_\mathbf{t}} \int_{t_0}^{t_f} \left\{ (\mathbf{q_t} - \mathbf{q_t^c})^T (\mathbf{q_t} - \mathbf{q_t^c}) + \sigma (P_t - P_t^c)^T (P_t - P_t^c) \right\} dt$$

sous les contraints

$$\begin{aligned}
\mathbf{M}(\mathbf{q_t})\ddot{\mathbf{q}}_\mathbf{t} + \mathbf{C}(\mathbf{q_t}, \dot{\mathbf{q}}_\mathbf{t}) &= \tau_t && \text{(Équation de la dynamique)} \\
\mathbf{q_{t_0}} = \mathbf{q_0}, \dot{\mathbf{q}}_{\mathbf{t_0}} &= \mathbf{0}, \ddot{\mathbf{q}}_{\mathbf{t_0}} = \mathbf{0} && \text{(Configuration initiale)} \\
\mathbf{q_{t_f}} = \mathbf{q_f}, \dot{\mathbf{q}}_{\mathbf{t_f}} &= \mathbf{0}, \ddot{\mathbf{q}}_{\mathbf{t_f}} = \mathbf{0} && \text{(Configuration finale)} \\
\tau^- \leq \tau_t &\leq \tau^+ && \text{(Limites des couples)} \\
\dot{\mathbf{q}}^- \leq \dot{\mathbf{q}}_\mathbf{t} &\leq \dot{\mathbf{q}}^+ && \text{(Limites des vitesses angulaires)} \\
\mathbf{q}^- \leq \mathbf{q_t} &\leq \mathbf{q}^+ && \text{(Limites angulaires)}
\end{aligned}$$

où τ_t désigne le vecteur des couples appliqués. $\mathbf{q_t}$, $\dot{\mathbf{q}}_\mathbf{t}$ et $\ddot{\mathbf{q}}_\mathbf{t}$ sont respectivement les vecteurs des angles, des vitesses et des accélérations des articulations du robot.

Les vecteur P_t et P_t^c désignent les positions cartésiennes des mains et de la tête du robot humanoïde et celles du mannequin virtuel dans le repère du bassin.
Le vecteur P_t est défini comme suit:

$$P_t = \begin{bmatrix} P_{\text{tête}} \\ P_{\text{main droite}} \\ P_{\text{main gauche}} \end{bmatrix}$$

Afin de résoudre ce problème d'optimisation nous proposons une méthode inspirée de la méthode développée dans le chapitre 7.

Par ailleurs, nous proposons une méthode de reparamétrage temporel des mouvements humains capturés. Le but de cette méthode est d'engendrer des mouvements qui soient dans les limites des vitesses angulaires du robot humanoïde, ceci à partir des mouvements humains capturés. Les mouvements obtenus seront utilisés par la suite comme solution initiale pour la méthode d'optimisation.

Nous montrons l'efficacité de notre méthode à travers des expérimentations sur la plate-forme HRP-2 № 14. Le mouvement que nous avons choisi est un mouvement de boxe. Ajoutons que l'imitation de ce mouvement par un robot humanoïde constitue un véritable défi.

L'expérimentation a montré que le mouvement imité conserve non seulement les caractéristiques remarquables et principales du mouvement humain capturé, mais respecte aussi les limites physiques du robot humanoïde.

"The important thing in science is not so much to obtain new facts as to discover new ways of thinking about them."

William Bragg

8

Human Motion Imitation By Humanoid Robot

Imagining a humanoid robot collaborates with humans to execute some daily tasks is now reality.
In order to increase the autonomous behavior of humanoid robots as well as improving their reactivity, the humanoid robot should be able to imitate human motions.

In recent years, the imitation of human motions by humanoid robots has been an active research field. In [Inamura et al. 2001] the human's behavior is abstracted into symbols by using mimesis theory. The observed symbols are analyzed into self motion elements which is regarded as a series of behavior. Then, a hidden Markov models for the description of the relation between the sequence of motion patterns and primitive symbols is adopted. After that a natural behavior can be generated and applied on a humanoid robot. However, even though some physical limits of the humanoid robot can be considered by this method, e.g angle limits, other limits, such as torque or joint velocity limits, are not integrated.

[Pollard et al. 2002] have proposed a method to transform a dance captured motion to a motion that the humanoid robot can execute. [Nakaoka et al. 2003] have realized a whole body control of humanoid robot to imitate Jongara-Bushi dance that is a traditional Japanese folk dance. To maintain the dynamical stability of humanoid robot, they control the trajectory of Zero Moment Point (ZMP) [Vukobratović and Borovac 2004] to be inside of the polygon of support. [Safonova et al. 2003] use also a pre-recorded human motion to generate optimal motion of the upper body of *Sarcos* humanoid robot. The function to be minimized is the difference between the recorded and executed motion by the robot. However, the previous methods do not

consider some physical limits of humanoid robot, e.g. torque limits.
[Ruchanurucks et al. 2006] have proposed a method to optimize upper body motion of humanoid robot in order to imitate a human captured motion. Their objective function preserves the main characteristics of the original motion, and at the same time it respects the physical constraints of the humanoid robot. However, the authors have mentioned that the resulting trajectories would meet the latter limits while the former limits are often violated. This is because their method considers the velocity and force constraints separately.

In this chapter, our objective is to generate a motion within the humanoid physical capabilities from a human captured motion. The physical limits are the angle, joint velocity and torque limits of the humanoid robot.

In this chapter, *we focus on the imitation of upper body motion*. On the other hand, the motion of lower body can be efficiently generated using leg motion primitives [Nakaoka et al. 2004; Nakaoka et al. 2005]. Concerning the dynamical stability of humanoid robot, it can be guaranteed by controlling the ZMP trajectory [Kajita et al. 2003].

The main contribution in this chapter is providing an optimization framework to generate the upper body motion of humanoid robot from human captured motions. The generated motions imitate the original human captured motion, and at the same time they respect the physical limits of humanoid robot.

The remainder of this chapter is organized as follows. In Section 8.1 the imitation problem is formulated and the problem solving procedure is explained. A pre-processing procedure for human captured motion is pointed out in Section 8.2. The implementation of the imitation procedure and an overview of the cart table model are given in Section 8.3. Some experimental results are given in Section 8.4 and Section 8.5 concludes the chapter.

8.1 Imitation Problem Formulation

The inputs of the imitation procedure are human captured motions. These motions are provided by a motion capture system as a skeleton of virtual actor and a sequence of the angular values of the virtual actor's joints.

Generally, the virtual actor has more degree of freedoms than the humanoid robot and as well its links lengths are different from those of the humanoid robot.

The imitation problem, from a kinematic point of view, is well known in computer graphics and it is called motion retargeting [Gleicher 1998]. The motion retargeting problem is formulated

as follows

$$\min_{\mathbf{q_t}} \int_{t_0}^{t_f} \left\{ (\mathbf{q_t} - \mathbf{q_t^c})^T (\mathbf{q_t} - \mathbf{q_t^c}) + \sigma (P_t - P_t^c)^T (P_t - P_t^c) \right\} dt$$

subject to

$$\begin{cases} \mathbf{q_{t_0}} = \mathbf{q_0} \\ \mathbf{q_{t_f}} = \mathbf{q_f} \\ \mathbf{q}^- \leq \mathbf{q_t} \leq \mathbf{q}^+ \end{cases} \tag{8.1}$$

where σ is a user defined constant. $\mathbf{q_t}$ and $\mathbf{q_t^c}$ are the joint positions of the upper body of the humanoid robot and the virtual actor respectively. P_t and P_t^c are the Cartesian positions of the hands, the head of the humanoid robot and the virtual actor in the pelvis frame. It is defined as follows

$$P_t = \begin{bmatrix} P_{\text{head}} \\ P_{\text{right hand}} \\ P_{\text{left hand}} \end{bmatrix} \tag{8.2}$$

Note that if the lengths of the virtual actor's links are largely different from those of humanoid robot, the vector P_t^c can be scaled to fit the humanoid robot size. $\mathbf{q}^-$ and $\mathbf{q}^+$ denote the minimal and maximal values of the vector $\mathbf{q}_t$ respectively.

The retargeting problem has been extremely studied in computer graphics during the last years, and we have actually many commercial graphic software that can solve it efficiently.

However, in the motion imitation by a humanoid robot additional difficulties arise such as the joints velocity and the torque limits.
By taking into account those additional constraints, the motion imitation problem becomes

$$\min_{\mathbf{q_t}, \dot{\mathbf{q}}_\mathbf{t}, \ddot{\mathbf{q}}_\mathbf{t}} \int_{t_0}^{t_f} \left\{ (\mathbf{q_t} - \mathbf{q_t^c})^T (\mathbf{q_t} - \mathbf{q_t^c}) + \sigma (P_t - P_t^c)^T (P_t - P_t^c) \right\} dt \tag{8.3}$$

subject to

$$\begin{aligned}
\mathbf{M}(\mathbf{q_t})\ddot{\mathbf{q}}_\mathbf{t} + \mathbf{C}(\mathbf{q_t}, \dot{\mathbf{q}}_\mathbf{t}) &= \tau_t && \text{(Dynamic equation)} \\
\mathbf{q_{t_0}} = \mathbf{q_0}, \dot{\mathbf{q}}_{\mathbf{t_0}} = \mathbf{0}, \ddot{\mathbf{q}}_{\mathbf{t_0}} &= \mathbf{0} && \text{(Initial configuration constraints)} \\
\mathbf{q_{t_f}} = \mathbf{q_f}, \dot{\mathbf{q}}_{\mathbf{t_f}} = \mathbf{0}, \ddot{\mathbf{q}}_{\mathbf{t_f}} &= \mathbf{0} && \text{(Final configuration constraints)} \\
\tau^- \leq \tau_t \leq \tau^+ & && \text{(Torque limits)} \\
\dot{\mathbf{q}}^- \leq \dot{\mathbf{q}}_\mathbf{t} \leq \dot{\mathbf{q}}^+ & && \text{(Velocity limits)} \\
\mathbf{q}^- \leq \mathbf{q_t} \leq \mathbf{q}^+ & && \text{(Joint limits)}
\end{aligned}$$

where τ_t is the vector of the applied torques on the humanoid robot's joints.

Using the recursive dynamic formulation introduced in Section 7.2.2 of Chapter 7, the

dynamic equation of motion can be written as follows

$$\tau_t = S^T F_t \tag{8.4}$$

where F_t is the applied forces on the humanoid robot joints.

In order to transform this optimization problem into a classical optimization problem, let us define

$$\begin{aligned} X_t &= \begin{bmatrix} \mathbf{q_t}^T & \dot{\mathbf{q}}_\mathbf{t}^T & \ddot{\mathbf{q}}_\mathbf{t}^T \end{bmatrix}^T \\ L(X_t) &= \int_{t_0}^{t_f} \left\{ (\mathbf{q_t} - \mathbf{q_t^c})^T (\mathbf{q_t} - \mathbf{q_t^c}) + \sigma \left(P_t - P_t^c\right)^T \left(P_t - P_t^c\right) \right\} dt \end{aligned}$$

$$G(X_t) = \begin{bmatrix} \tau_t - \tau^+ \\ -\tau_t + \tau^- \\ \dot{\mathbf{q}}_\mathbf{t} - \dot{\mathbf{q}}^+ \\ -\dot{\mathbf{q}}_\mathbf{t} + \dot{\mathbf{q}}^- \\ \mathbf{q_t} - \mathbf{q}^+ \\ -\mathbf{q_t} + \mathbf{q}^- \end{bmatrix}, \quad H(X_t) = \begin{bmatrix} \tau_t - S^T F_t \\ \mathbf{q_{t_0}} - \mathbf{q_0} \\ \dot{\mathbf{q}}_{\mathbf{t_0}} \\ \ddot{\mathbf{q}}_{\mathbf{t_0}} \\ \mathbf{q_{t_f}} - \mathbf{q_f} \\ \dot{\mathbf{q}}_{\mathbf{t_f}} \\ \ddot{\mathbf{q}}_{\mathbf{t_f}} \end{bmatrix} \tag{8.5}$$

Thus, the optimization problem (8.3) can be transformed into the following classical form

$$\begin{aligned} &\min_{X_t} \; L(X_t) \\ \text{subject to} \quad & \\ & H(X_t) = 0 \\ & G(X_t) \leq 0 \end{aligned} \tag{8.6}$$

The above optimization problem is similar to the optimization problem 7.15 and it can be solved by an analogues logic, i.e. using the augmented Lagrange multiplier method and the analytical calculation of the gradient function, and then discretize the configuration space in order to transform the optimization problem into a finite dimensional problem.

8.2 Pre-processing Human Captured Motion

The main challenging issue in the imitation of human captured motion is the fast dynamic. On account of the physical limits of the humanoid robot and the capacities of the motors in its joints, it is not able to follow a fast and highly dynamic motion.
Therefore, taking the human captured motion as an initial solution for the optimization problem (8.3) might yield a motion that is very different from the original one. In order to obtain a good initial guess for the optimization problem (8.3), one can slow down the captured motion.
Let us consider that we have a captured motion of N samples and the sampling frequency of this

motion is f (e.g. $f = 120$ Hz). Let us denote $q^c(n)$ the vector of joint values which corresponds to the sample number n.
A simple algorithm to transform the human captured motion into a motion within the joint velocity limits of the humanoid robot is given by the following pseudo code

Algorithm 8.1 Time re-parameterization of the human captured motion

Input: $n \leftarrow 1$
while $n \leq N-1$ **do**
  Calculate $\Delta\mathbf{q}^c \leftarrow |\mathbf{q}^c(n+1) - \mathbf{q}^c(n)|$;
  Input: $\Delta t \leftarrow \frac{1}{f}$
  while $\frac{\Delta\mathbf{q}^c}{\Delta t} > \dot{\mathbf{q}}^+$ **do**
    $\Delta t \leftarrow \Delta t + \frac{1}{f}$;
  end
  $n \leftarrow n+1$;
end

Recall that $\dot{\mathbf{q}}^+$ is the maximal value of humanoid robot's joint velocity vector. Fig. 8.7(b) shows the obtained motion after the application of time re-parameterization on the original human captured motion Fig. 8.7(a).

8.3 Implementation

It is clear that the self collision problem is not considered in the imitation problem formulation. Although, approximating the humanoid robot's links by cylinders and spheres and consider the distance between them as an additional constraint can solve the problem of self collision, this procedure might yield an imitated motion largely different from the original human captured motion on account of the approximation.

[Kanehiro et al. 2008] have proposed an efficient method to avoid the collision for a non-strictly convex objects. The method makes use of non-strictly convex polyhedra as geometric models of the robot and the environment without any approximation. Moreover, this method ensures that the velocities of the humanoid robot's joints are continuous. Applying this method as post-processing task can solve the problem of self collision and it yields a collision-free motion.

Furthermore, the dynamical stability of humanoid robot should be ensured. To achieve this goal, the trajectory of ZMP should be controlled to be inside of the polygon of support. The control of the trajectory of ZMP can be done by using the method of cart table model proposed in [Kajita et al. 2003].

The implementation of the imitation procedure can be summarized as follows

1. Pre-processing the upper body human captured motion to obtain a motion within the

velocity limits of the humanoid robot joints.

2. Solve the optimization (8.6) to obtain a motion within the humanoid robot capacities.
3. Apply the self collision avoidance method [Kanehiro et al. 2008] to transform the optimized motion to collision-free motion.
4. Calculate the ZMP trajectory of the obtained motion.
5. The calculated ZMP trajectory is then modified and restricted to be inside of the polygon of support foot (feet).
6. Once the modified ZMP trajectory is available, we use the cart table model [Kajita et al. 2003] to calculate the trajectory of the center of mass.
7. The horizontal trajectory of the free flyer (pelvis joint) is then calculated using the trajectory of the center of mass.

8.3.1 Cart Table Model: An Overview

The main idea of this method is to approximate the humanoid robot by a mass located at its Center of Mass (CoM) and it is equal to the total mass of the humanoid robot. Therefore, the complex problem of controlling the humanoid robot is transformed to control an inverted pendulum.

Let us define the Cartesian position of the center of mass (P_{CoM}) by

$$P_{CoM} = \begin{bmatrix} x \\ y \\ z_c \end{bmatrix} \tag{8.7}$$

We suppose that the vertical position of the mass z_c is constant.

The ZMP is a particular point of the horizontal plane at which the horizontal moments vanish. For the inverted pendulum, it is defined as follows

$$ZMP = \begin{bmatrix} p_x \\ p_y \end{bmatrix} = \begin{bmatrix} x - \frac{z_c}{g}\ddot{x} \\ y - \frac{z_c}{g}\ddot{y} \end{bmatrix} \tag{8.8}$$

where g is the absolute value of gravity acceleration.

It is clear from (8.8) that the two elements of ZMP are decoupled, and they are very similar. For that, in the sequel, we figure out how to control p_x only. Then the control for p_y can be obtained in similar way.

To control p_x, we consider the cart table model proposed in [Kajita et al. 2003], which is shown in Fig. 8.1. It depicts a running cart of mass m on a pedestal table whose mass is negligible. In our case, this mass is equal to the total mass of the humanoid robot and its center coincides with the CoM of humanoid robot.

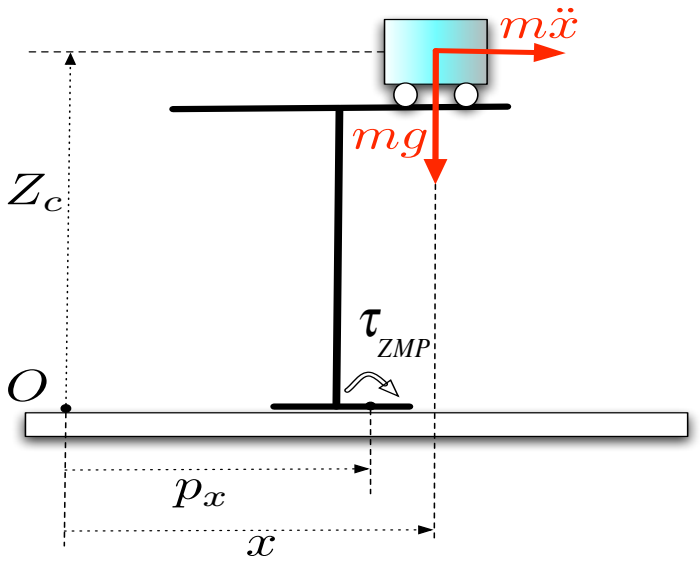


Figure 8.1: A cat table model

8.3.1.1 Controlling ZMP by Preview Control

Let us define a new variable u_x as the time derivative of $\ddot{x}$

$$u_x = \frac{d\ddot{x}}{dt} \tag{8.9}$$

The variable u_x is usually called the jerk.

Regarding u_x as the input of p_x, we can translate the equation of p_x into the following dynamical system

$$\begin{aligned} \frac{d}{dt}\begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix} &= \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u_x \\ p_x &= \begin{bmatrix} 1 & 0 & -\frac{z_c}{g} \end{bmatrix}\begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix} \end{aligned} \tag{8.10}$$

As the humanoid robot is controlled in discrete time, we discretize the system (8.10) with sampling time T_s as

$$\begin{aligned} z_{k+1} &= Az_k + Bu_k \\ p_k &= Cz_k \end{aligned} \tag{8.11}$$

where

$$
\begin{aligned}
z_k &\triangleq \begin{bmatrix} x(kT_s) & \dot{x}(kT_s) & \ddot{x}(kT_s) \end{bmatrix}^T \\
u_k &\triangleq u_x(kT_s) \\
p_k &\triangleq p_x(kT_s) \\
A &= \begin{bmatrix} 1 & T_s & T_s^2/2 \\ 0 & 1 & T_s \\ 0 & 0 & 1 \end{bmatrix} \\
B &= \begin{bmatrix} T_s^3/6 \\ T_s^2/2 \\ T_s \end{bmatrix} \\
C &= \begin{bmatrix} 1 & 0 & -\frac{z_c}{g} \end{bmatrix}
\end{aligned}
\tag{8.12}
$$

With a given reference for ZMP $\left(p_k^{ref}\right)$, the objective function to be minimized can be specified as

$$
J = \sum_{i=k}^{\infty} \left(Q e_i^2 + R \Delta u_i^2 \right) \tag{8.13}
$$

where $e_i \triangleq p_i - p_i^{ref}$ is the servo error, Q, $R > 0$. $\Delta u_i \triangleq u_i - u_{i-1}$.
Fig. 8.2 shows the ZMP tracking control problem.

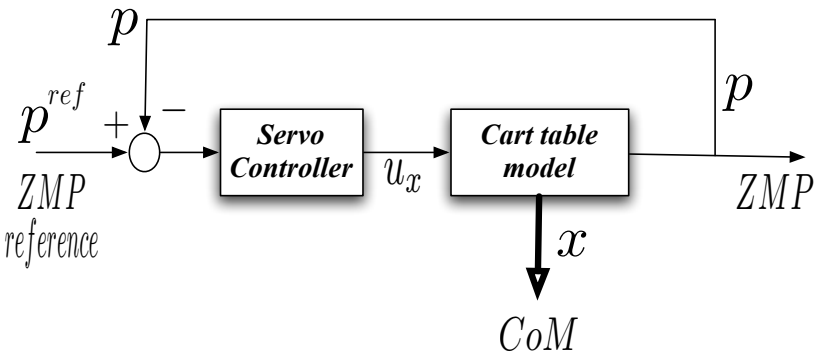


Figure 8.2: ZMP control problem.

We can then apply the classical Linear Quadratic Gaussian (LQG) technique. Thus, when the ZMP reference is previewed for N_L future steps at every sampling time, the optimal controller which minimizes the objective function (8.13) is given by

$$
u_k = -K z_k + \begin{bmatrix} f_1 & f_2 & \cdots & f_{N_L} \end{bmatrix} \begin{bmatrix} p_{k+1}^{ref} \\ p_{k+2}^{ref} \\ \vdots \\ p_{k+N_L}^{ref} \end{bmatrix} \tag{8.14}
$$

where K and f_i are calculated as

$$\begin{aligned} K &= \left(R + B^T \boldsymbol{P} B\right)^{-1} B^T \boldsymbol{P} A \\ f_i &= \left(R + B^T \boldsymbol{P} B\right)^{-1} B^T \left(A - BK\right)^{T*(i-1)} C^T Q \end{aligned} \tag{8.15}$$

$\mathbf{P}$ is the solution of the following Riccati equation

$$\boldsymbol{P} = A^T \boldsymbol{P} A + C^T Q C - A^T \boldsymbol{P} B \left(R + B^T \boldsymbol{P} B\right)^{-1} B^T \boldsymbol{P} A \tag{8.16}$$

The above optimal controller provides the trajectory of CoM of the robot. However, the trajectory of ZMP will not follow exactly the reference trajectory. This is because of the difference between the cart table model and detailed multibody model defined by the parameter of the real humanoid robot.

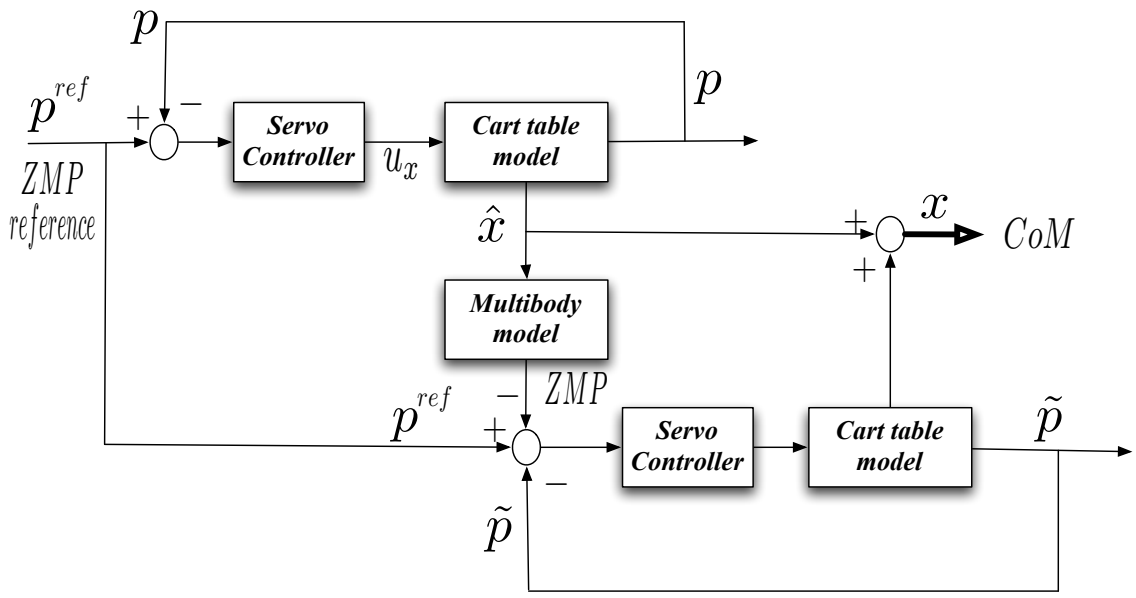


Figure 8.3: ZMP control problem with compensation

To fix the ZMP error due to this difference, again we can use the preview control. For that, we first calculate the CoM trajectory from the cart table model and obtain the ZMP error from the multibody model (see Section 9.1 for more details).

These information are stored into a memory buffer and loaded to be used after a delay time of $T * N_L$. By this way, the future ZMP error can be compensated by the preview control.

The servo controlling schema for the ZMP control becomes as shown in Fig. 8.3.

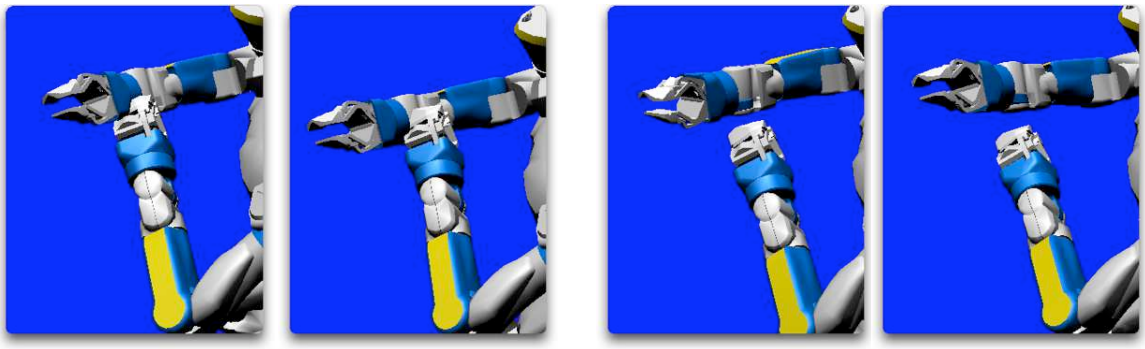
8.4 Experimental Results

We have chosen a boxing captured motion to validate our proposed method. As the movements of the upper body of the boxing motion is complex, the imitation by a humanoid robot is really a challenging issue. The vertical movement of the pelvis joint is also considered and the movements of the lower body is calculated using inverse kinematic.

As we have mentioned in Section 8.3, the self collision problem is addressed as a post processing task just after solving the optimization problem. In fact, after solving the optimization problem, we obtain a motion with collision between the hands of humanoid robot as pointed out in Fig. 8.4(a). Fig. 8.4(b) shows the self collision avoidance of the humanoid robot's hands by applying the method proposed in [Kanehiro et al. 2008].

Snapshots of the conducted motion using OpenHRP platform [Kanehiro et al. 2004] are presented in Fig. 8.5. Fig. 8.6 shows snapshots of the motion applied to the humanoid robot HRP2.

Fig. 8.7(a) shows the angular position trajectory of the virtual actor's right elbow, Fig. 8.7(b) shows the modified trajectory after the application of the time re-parameterization algorithm explained in Section 8.2. The optimized trajectory for the humanoid robot is given in Fig. 8.7(c). Note that the optimized trajectory respects the physical limits of HRP-2 humanoid robot that are not only the angle limits but also the joint velocity and torque limits. The angular velocity of humanoid robot's right elbow joint is shown in Fig. 8.8. Moreover, the applied torque on humanoid robot's right elbow joint before and after optimization is given in Fig. 8.9.



(a) Obtained motion after optimization.

(b) Applying self collision avoidance as post-processing task.

Figure 8.4: Self collisions avoidance.

8.5 Conclusion

In this chapter, the human motion imitation by a humanoid robot is considered. In order to generate an imitated motion within the humanoid robot capabilities, the imitation problem

is formulated as an optimization problem. The physical limits of the humanoid robot are transformed into constraints of the optimization problem and the objective function to be minimized is the difference between the angular values of the humanoid robot's joints and those of the virtual actor.

The experimental results have pointed out that the proposed method yields motions that preserve the salient characteristics of the original human captured motion and, at the same time, they respect the physical limits of the humanoid robot.

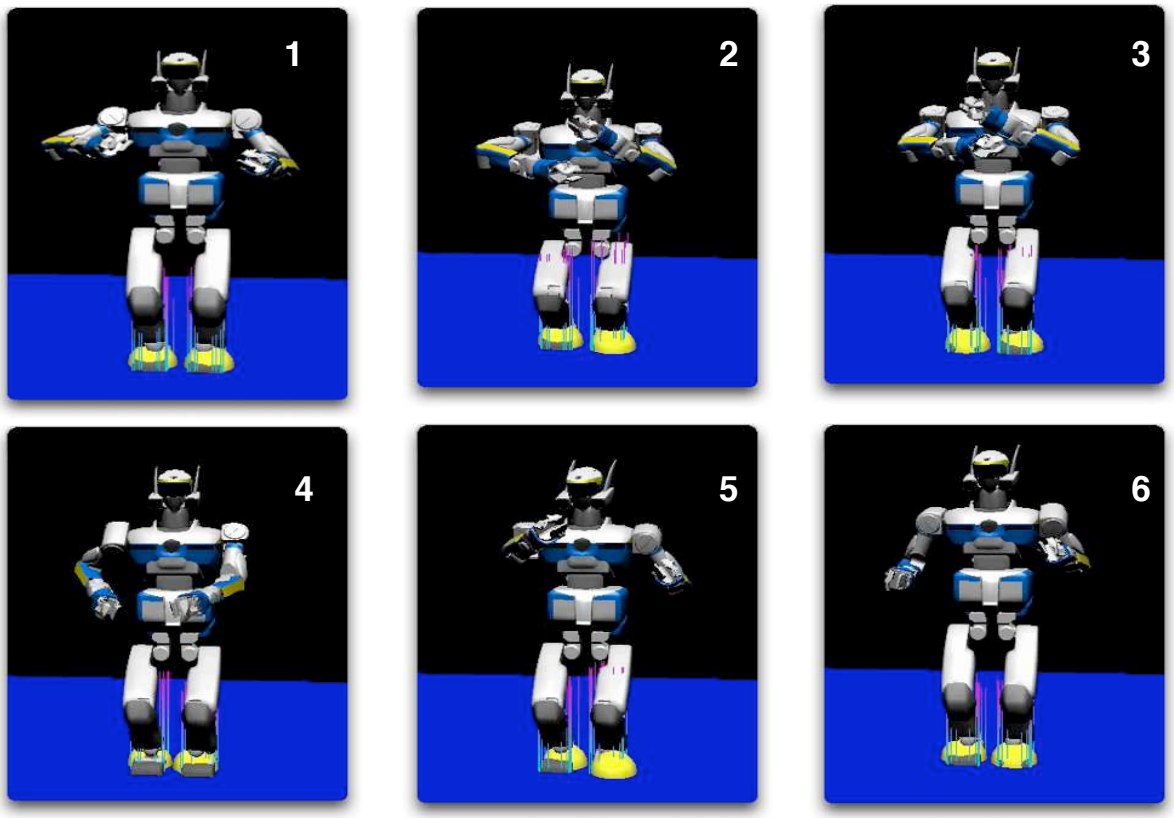


Figure 8.5: Snapshots of the simulated motion.

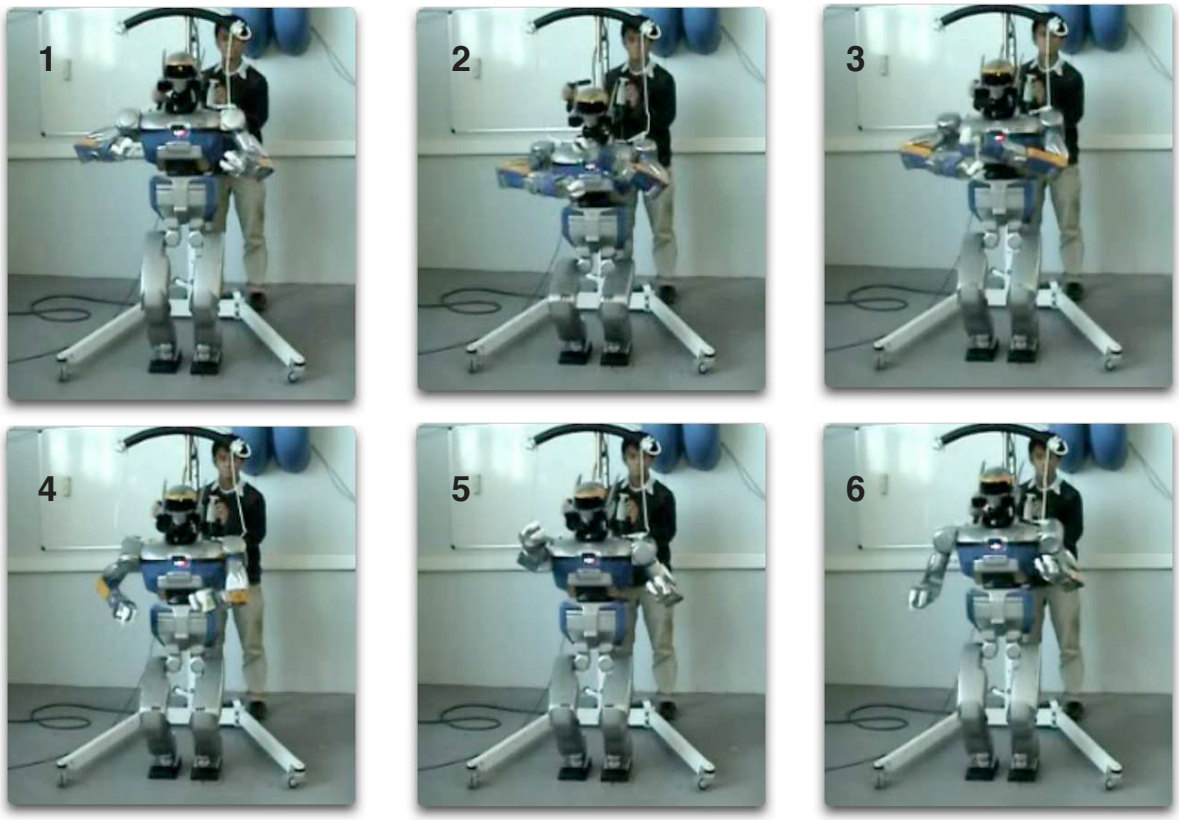
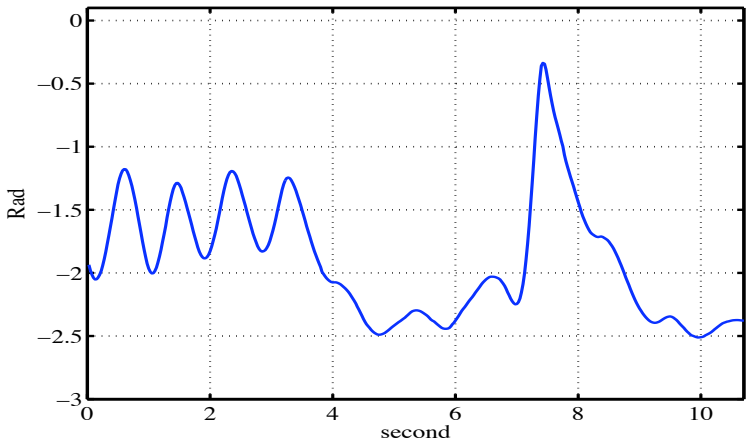
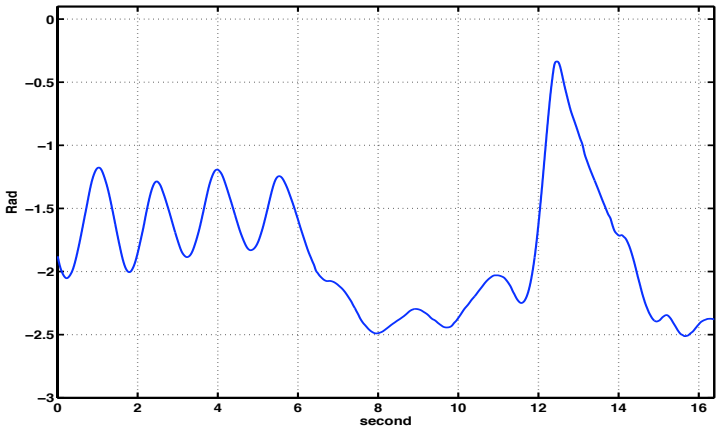


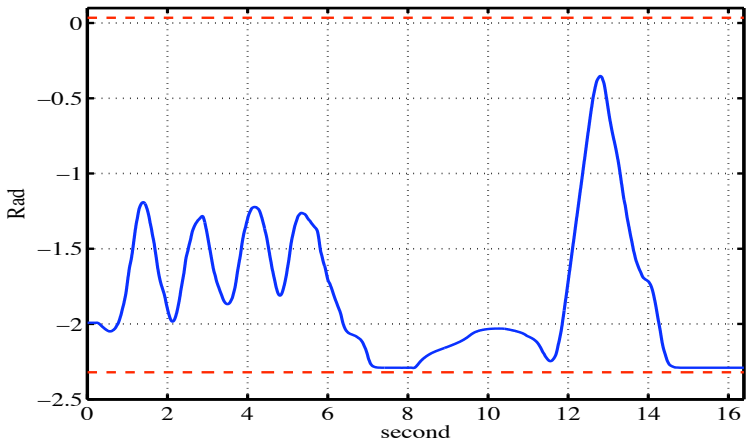
Figure 8.6: Snapshots of the conducted motion.



(a) Original human captured motion.



(b) Time re-parameterization of the human captured motion.



(c) Obtained motion after the optimization and considering the physical limits of the humanoid robot (HRP-2). Dash lines denote the angular limits of the elbow joint.

Figure 8.7: Angular position of the right elbow.

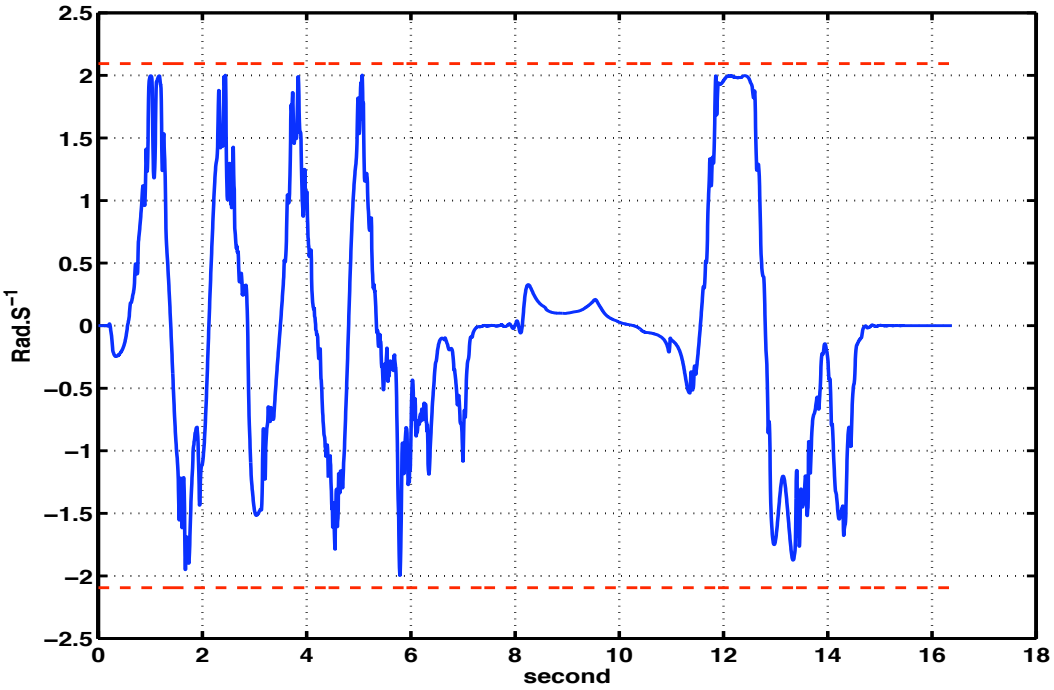


Figure 8.8: Angular velocity of humanoid robot's right elbow joint.

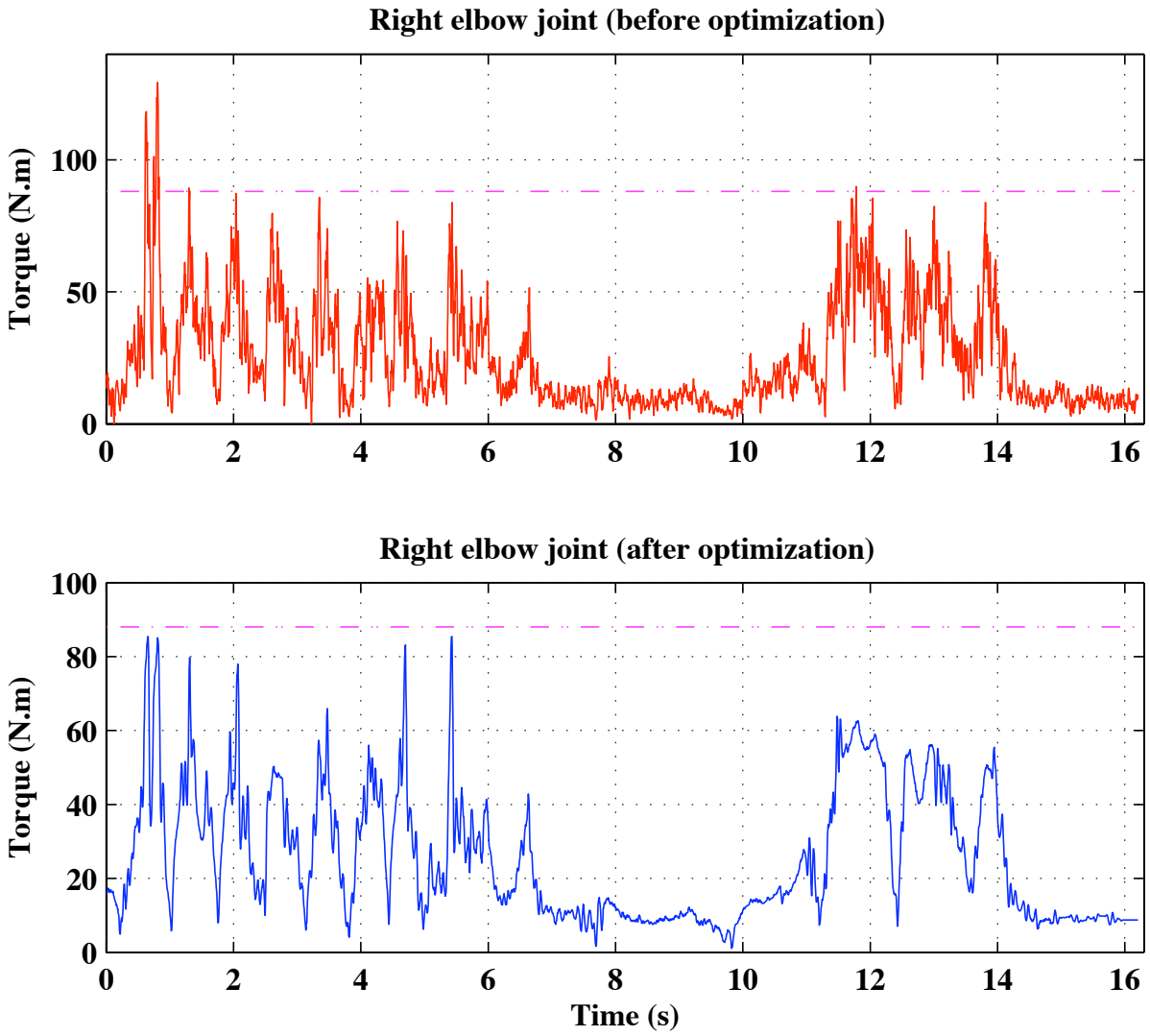


Figure 8.9: Applied torque on humanoid robot's right elbow joint.

Résumé du chapitre 9
Paramétrage temporel des chemins dans l'espace de configuration pour un robot humanoïde

Le problème de paramétrage des chemins est un problème ancien en robotique. Afin de bien distinguer la différence entre chemin et trajectoire, nous commençons par donner leurs définitions.

Un *chemin* désigne les positions des points dans l'espace de configuration, ou l'espace opérationnel, par lesquels le robot doit passer afin d'exécuter le mouvement désiré. Une *trajectoire* est un chemin où la notion du temps est ajoutée. Autrement dit pour chaque point du chemin on associe l'instant auquel le robot doit passer par ce point [Sciavicco and Siciliano 2000b].

En utilisant les définitions ci-dessus, le problème de paramétrage temporel est le problème de transformer un chemin en une trajectoire qui respecte les limites physiques du robot telles que les limites des vitesses angulaires, les limites des couples, ..., etc.

On retrouve ce problème de paramétrage temporel des chemins dans le contrôle des bras manipulateurs. Dans ce cas, l'objectif est de décroître le temps d'exécution des taches pour augmenter la productivité. La plupart des ces méthodes sont basées sur la théorie de commande optimale [Renaud and Fourquet 1992]. Dans la recherche sur les robots mobiles, le paramétrage temporel a pour but de transformer un chemin réalisable (chemin sans collisions) en trajectoire faisable par le robot [Lamiraux and Laumond 1998]. La trajectoire obtenue permet d'atteindre la position désirée le plus rapidement possible tout en respectant les contraintes.

Cependant, l'application de la théorie de la commande optimale dans le cas des robots humanoïdes est une tache très difficile. Non seulement l'équation dynamique du robot humanoïde est très complexe, mais l'application de la théorie de commande optimale exige en plus le calcul des dérivées du vecteur de configuration par rapport au chemin paramétrisé. Bien que ce calcul pourrait être fait en utilisant la géométrie différentielle, il se révèle très compliqué pour les systèmes avec grand nombre de degrés de liberté et à la structure cinématique en branches, ce qui est le cas pour les robots humanoïdes.

Pour cela, nous proposons de résoudre le problème de paramétrage temporel des chemins,

dans l'espace de configurations pour un robot humanoïde, numériquement par différences finis.

En général, le paramétrage temporel d'une fonction $f(x_t)$, où t désigne le temps, consiste à trouver une fonction réelle $\mathcal{S}_t$ de manière que $f(x_{\mathcal{S}_t})$ vérifie quelques contraintes temporelles, par exemple:

$$h(\mathcal{S}_t) \leq f(x_{\mathcal{S}_t}) \leq l(\mathcal{S}_t)$$

Dans notre cas, le problème de paramétrage temporel d'un chemin a pour but d'obtenir une trajectoire qui est non seulement dynamiquement stable, mais aussi dans les limites des vitesses angulaires du robot humanoïde.

Nous supposons qu'on dispose d'un chemin composé de L points. La période d'échantillonnage est ΔT. Nous définissons le temps total de la trajectoire initiale par $T = L\ \Delta t$.

Le problème de paramétrage temporel peut alors être formulé comme le problème d'optimisation suivant :

$$\min\ \mathcal{S}_T = \min_{s_t} \int_{t=0}^{T} s_t\ dt$$

sous les contraintes

$$s_t > 0 \tag{1}$$

$$\boldsymbol{p}_{s_t}^{-} \leq \boldsymbol{p}_{s_t} \leq \boldsymbol{p}_{s_t}^{+} \tag{2}$$

$$\boldsymbol{f}_{s_t}^{T}\ \boldsymbol{n} < 0 \tag{3}$$

$$\mu\ \boldsymbol{f}_{s_t}^{T}\ \boldsymbol{n} < \boldsymbol{\tau}_{s_t}^{T}\ \boldsymbol{n} < -\mu\ \boldsymbol{f}_{s_t}^{T}\ \boldsymbol{n} \tag{4}$$

$$\dot{\boldsymbol{q}}^{-} \leq \dot{\boldsymbol{q}}_{s_t} \leq \dot{\boldsymbol{q}}^{+} \tag{5}$$

où s_t est une fonction réelle, $\boldsymbol{p}_{s_t} \in \mathbb{R}^2$ désigne le vecteur ZMP, $\boldsymbol{p}_{s_t}^{-}$ et $\boldsymbol{p}_{s_t}^{+}$ désignent le polygone de support pour le robot humanoïde.

$\boldsymbol{f}_{s_t}$ et $\boldsymbol{\tau}_{s_t}$ désignent respectivement la force appliquée sur le(s) pied(s) du robot humanoïde et le moment de cette force sur le(s) pied(s).

μ est le coefficient de friction statique. $\dot{\boldsymbol{q}}_{s_t}$ désigne le vecteur des vitesses angulaires des articulations du robot humanoïde. $\dot{\boldsymbol{q}}^{+}$ et $\dot{\boldsymbol{q}}^{-}$ sont les limites supérieures et inférieures de ces vitesses.

Nous montrons dans le chapitre 9 que le problème ci-dessus peut être résolu en faisant appel à une discrétisation de l'espace des solutions à l'aide d'une base des fonctions B-splines.

Nous validons la méthode proposée à travers des expérimentations sur la plate-forme HRP-2 №14. Ces résultats expérimentaux ont révélé l'efficacité et la robustesse de notre méthode.

"Ce n'est pas l'homme qui arrête le temps, c'est le temps qui arrête l'homme."

François René Chateaubriand

9

Time Parameterization of Humanoid Robot Paths

Time parameterization of a path is an old problem in robotic research. First of all, let us start by defining a path and a trajectory.

A *path* denotes the locus of points in the joint space, or in the operational space, the robot has to follow in the execution of the desired motion, and a *trajectory* is a path on which a time law is specified [Sciavicco and Siciliano 2000b].

The time parameterization of a path is the problem of transforming this path to a trajectory which respects the physical limits of the robot, e.g. velocity limits, acceleration limits, $\cdots$ etc.

In the research works on manipulators, the time parameterization problem has the objective of reducing the execution time of the tasks, thereby increasing the productivity. Most of these approaches are based on time-optimal control theory (see [Renaud and Fourquet 1992] for an overview). In the framework of mobile robots, the time parameterization problem arises also to transform a feasible path to a feasible trajectory [Lamiraux and Laumond 1998]. The main objective, in this case, is to reach the goal position as fast as possible.

In our case, the application of time optimal control theory is however a difficult task. This is because not only the dynamic equation of the humanoid robot motion is very complex, but also applying time optimal control theory requires the calculation of the derivative of the configuration space vector of humanoid robot with respect to the parameterized path. Although such a calculation can be evaluated from differential geometry, it is a very difficult task in the case of systems with large number of degree of freedoms and branched kinematic chains, which is the case of humanoid robot. For that, we propose to solve the time parameterization problem

numerically using finite difference approach.

The remainder of this chapter is organized as follows. Section 9.1 gives an overview of the dynamic stability notion and the mathematical formulation of Zero Moment Point (ZMP). The time parameterization problem is formulated as an optimization problem under constraints in Section 9.2. Discretizing the solution space and solving the optimization problem is explained in Section 9.3. In Section 9.4 the implementation algorithm of the proposed method is summarized. An experimental example using the humanoid robot platform HRP2 is given in Section 9.5. Finally, Section 9.6 concludes this chapter.

9.1 Dynamic Stability and ZMP: An Overview

Definition 9.1 *Statically stable trajectory is a trajectory for which the trajectory of the projection of the Center of Mass (CoM) of the humanoid robot on the horizontal plane is always inside of the polygon of support (i.e. the convex hull of all points of contact between the support foot (feet) and the ground).*

Definition 9.2 *Dynamically stable trajectory is a trajectory for which the trajectory of Zero Moment Point (ZMP) is always inside of the polygon of support.*

Theoretically, any statically stable trajectory can be transformed into a dynamically stable one by slowing down the humanoid robot's motion.

However, our objective is to find the minimum time and dynamically stable trajectory from a statically stable one. In order to obtain a motion within the humanoid robot capacities, the joint velocity limits of the humanoid robot should be taken into account.

Let the ZMP on the horizontal ground be given by the following vector

$$\boldsymbol{p} = \begin{bmatrix} p_x & p_y \end{bmatrix}^T \tag{9.1}$$

To compute $\boldsymbol{p}$, one can use the following formula

$$\boldsymbol{p} = \boldsymbol{N} \frac{\boldsymbol{n} \times \boldsymbol{\tau}}{\boldsymbol{f}^T \boldsymbol{n}} \tag{9.2}$$

where the operator $\times$ refers to the cross product, and

- $\boldsymbol{N}$ is a constant matrix
$$\boldsymbol{N} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{9.3}$$
.
- the vector $\boldsymbol{n}$ is the normal vector on the horizontal ground $\left(\boldsymbol{n} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T\right)$.

- The vector $\boldsymbol{f}$ is the result of the gravity and inertia forces

$$\boldsymbol{f} = M\boldsymbol{g} - \sum_{i=1}^{n} m_i \ \ddot{\boldsymbol{c}}_i \tag{9.4}$$

where $\boldsymbol{g}$ denotes the acceleration of the gravity ($\boldsymbol{g} = -g\boldsymbol{n}$), and M is the total mass of the humanoid robot. The quantities m_i, $\ddot{\boldsymbol{c}}_i$ are the mass of the i^{th} link and the acceleration of its center of mass $\boldsymbol{c}_i$ respectively.

- $\boldsymbol{\tau}$ denotes the moment of the force $\boldsymbol{f}$ about the origin of the fixed world frame. The expression of $\boldsymbol{\tau}$ is the following

$$\boldsymbol{\tau} = \sum_{i=1}^{n} \left(m_i \ \boldsymbol{c}_i \times (\boldsymbol{g} - \ddot{\boldsymbol{c}}_i) - \dot{\mathcal{L}}_{c_i} \right) \tag{9.5}$$

where $\mathcal{L}_{c_i}$ is the angular momentum at the point $\boldsymbol{c}_i$

$$\dot{\mathcal{L}}_{c_i} = \boldsymbol{R}_i \left(\boldsymbol{I}_{c_i} \dot{\boldsymbol{\omega}}_i - (\boldsymbol{I}_{c_i} \boldsymbol{\omega}_i) \times \boldsymbol{\omega}_i \right) \tag{9.6}$$

$\boldsymbol{R}_i$ is the rotation matrix associated to the i^{th} link. $\boldsymbol{I}_{c_i}$, $\boldsymbol{\omega}_i$ and $\dot{\boldsymbol{\omega}}_i$ are its inertia matrix, angular velocity and angular acceleration respectively.

9.2 Time Parameterization Problem Formulation

Generally speaking, the time parameterization problem of a function $f(x_t)$, where t denotes time, consists in finding a real function $\mathcal{S}_t$ in such a way $f(x_{\mathcal{S}_t})$ verifies some temporal constraints, e.g,

$$h(\mathcal{S}_t) \leq f(x_{\mathcal{S}_t}) \leq l(\mathcal{S}_t) \tag{9.7}$$

Definition 9.3 *The spatial velocity vector $\boldsymbol{V}_t \in \mathbb{R}^6$ is defined as follows*

$$\boldsymbol{V}_t = \begin{bmatrix} \boldsymbol{v}_t \\ \\ \boldsymbol{\omega}_t \end{bmatrix} = \begin{bmatrix} \frac{d\boldsymbol{X}_t}{dt} \\ \\ \frac{d\boldsymbol{\theta}_t}{dt} \end{bmatrix} \tag{9.8}$$

where $\boldsymbol{X}_t \in \mathbb{R}^3$ designs the Cartesian position of the link, and $\boldsymbol{\theta}_t \in \mathbb{R}^3$ designs its orientation.

Definition 9.4 *The spatial acceleration vector $\dot{\boldsymbol{V}}_t \in \mathbb{R}^6$ is defined as follows*

$$\dot{\boldsymbol{V}}_t = \begin{bmatrix} \boldsymbol{a}_t \\ \\ \dot{\boldsymbol{\omega}}_t \end{bmatrix} = \begin{bmatrix} \frac{d\boldsymbol{v}_t}{dt} \\ \\ \frac{d\boldsymbol{\omega}_t}{dt} \end{bmatrix} \tag{9.9}$$

In order to obtain a causal motion, the function $\mathcal{S}_t$ should be a strictly increasing function, that means $\frac{d\mathcal{S}_t}{dt} > 0$.
Therefore we will express $\mathcal{S}_t$ as the integral of a strictly positive function $s_t > 0$, as follows

$$\mathcal{S}_t = \int_{\tau=0}^{t} s_\tau \; d\tau \tag{9.10}$$

Let us write the finite difference approximation of the quantities in Eqs (9.8) and (9.9) with respect to the mapping function $\mathcal{S}_t$

$$\boldsymbol{V}_{\mathcal{S}_t} = \begin{bmatrix} \frac{d\boldsymbol{X}_t}{d\mathcal{S}_t} \\ \\ \frac{d\boldsymbol{\theta}_t}{d\mathcal{S}_t} \end{bmatrix} \approx \begin{bmatrix} \frac{\Delta \boldsymbol{X}_t}{s_t \; \Delta t} \\ \\ \frac{\Delta \boldsymbol{\theta}_t}{s_t \; \Delta t} \end{bmatrix} \tag{9.11}$$

$$\begin{aligned} \dot{\boldsymbol{V}}_{\mathcal{S}_t} = \frac{d \; \boldsymbol{V}_{\mathcal{S}_t}}{d \; \mathcal{S}_t} &\approx \frac{1}{\Delta \mathcal{S}_t} \left(\boldsymbol{V}_{\mathcal{S}_t} - \boldsymbol{V}_{\mathcal{S}_{t-1}} \right) \\ &\approx \begin{bmatrix} \frac{\frac{\Delta \boldsymbol{X}_t}{\Delta t} s_{t-1} - \frac{\Delta \boldsymbol{X}_{t-1}}{\Delta t} s_t}{s_t^2 s_{t-1} \Delta t} \\ \\ \frac{\frac{\Delta \boldsymbol{\theta}_t}{\Delta t} s_{t-1} - \frac{\Delta \boldsymbol{\theta}_{t-1}}{\Delta t} s_t}{s_t^2 s_{t-1} \Delta t} \end{bmatrix} \end{aligned} \tag{9.12}$$

9.2.1 Minimum Time and Dynamically Stable Trajectory

Let us suppose that we have a path which consists of L points. At first, we transform this path to a trajectory by considering a uniform time distribution function. In other words, we suppose that $s_t = 1 : \; \forall t$ in (9.10). We denote the time horizon $T = L \; \Delta t$.

In our case, we would obtain a minimum time trajectory which is not only dynamically stable but also within the joint velocity limits of the humanoid robot. Therefore, the optimization problem can be formulated as follow

$$\min \; \mathcal{S}_T = \min_{s_t} \int_{t=0}^{T} s_t \; dt \tag{9.13}$$

subject to

$$s_t > 0 \tag{9.14}$$

$$\boldsymbol{p}_{s_t}^{-} \leq \boldsymbol{p}_{s_t} \leq \boldsymbol{p}_{s_t}^{+} \tag{9.15}$$

$$\boldsymbol{f}_{s_t}^{T} \; \boldsymbol{n} < 0 \tag{9.16}$$

$$\mu \; \boldsymbol{f}_{s_t}^{T} \; \boldsymbol{n} < \boldsymbol{\tau}_{s_t}^{T} \; \boldsymbol{n} < -\mu \; \boldsymbol{f}_{s_t}^{T} \; \boldsymbol{n} \tag{9.17}$$

$$\dot{\boldsymbol{q}}^{-} \leq \dot{\boldsymbol{q}}_{s_t} \leq \dot{\boldsymbol{q}}^{+} \tag{9.18}$$

where $\boldsymbol{p}_{s_t}$ is the ZMP vector, $\boldsymbol{p}_{s_t}^{-}$ and $\boldsymbol{p}_{s_t}^{+}$ design the polygon of support for the humanoid

robot. $\boldsymbol{f}_{s_t}$ and $\boldsymbol{\tau}_{s_t}$ denote the applied force on the foot (feet) and the moment of this force about the origin of the fixed world frame respectively. μ is the coefficient of the static friction. The vector $\dot{\boldsymbol{q}}_{s_t}$ denotes the joint velocity of the humanoid robot. $\dot{\boldsymbol{q}}^+$ and $\dot{\boldsymbol{q}}^-$ design its upper and lower limits.

The constraints of the above optimization problem can be analyzed as follows

- Constraint (9.15) guarantees that the ZMP is inside of the polygon of support.
- Constraint (9.16) ensures that the foot is in contact with the ground (the humanoid robot will not jump).
- Constraint (9.17) prevents the foot from sliding around the Z-axis.
- Constraint (9.18) guarantees that the obtained trajectory respects the joint velocity limits of the humanoid robot.

Let us write $\boldsymbol{p}_{s_t}$, $\boldsymbol{f}_{s_t}$, and $\boldsymbol{\tau}_{s_t}$as functions of s_t

$$\boldsymbol{p}_{s_t} = \boldsymbol{N}\ \frac{\boldsymbol{n}\times\boldsymbol{\tau}_{s_t}}{\boldsymbol{f}_{s_t}^T\,\boldsymbol{n}} \tag{9.19}$$

where

$$\begin{aligned}\boldsymbol{\tau}_{s_t} &= \sum_{i=1}^{n}\left(m_i\ \boldsymbol{X}_t^{c_i}\times\left(\boldsymbol{g}-\tilde{\tilde{\boldsymbol{c}}}_i\right)-\tilde{\tilde{\mathcal{L}}}_{c_i}\right)\\ \boldsymbol{f}_{s_t} &= M\boldsymbol{g}-\sum_{i=1}^{n}m_i\ \tilde{\tilde{\boldsymbol{c}}}_i\end{aligned} \tag{9.20}$$

in which

$$\begin{aligned}\tilde{\tilde{\boldsymbol{c}}}_i &= \frac{\frac{\Delta\boldsymbol{X}_t^{c_i}}{\Delta t}s_{t-1}-\frac{\Delta\boldsymbol{X}_{t-1}^{c_i}}{\Delta t}s_t}{{s_t}^2 s_{t-1}\Delta t}\\ \tilde{\tilde{\mathcal{L}}}_{c_i} &= \boldsymbol{R}_i\left(\boldsymbol{I}_{c_i}\tilde{\tilde{\boldsymbol{\omega}}}_i-(\boldsymbol{I}_{c_i}\tilde{\boldsymbol{\omega}}_i)\times\tilde{\boldsymbol{\omega}}_i\right)\\ \tilde{\tilde{\boldsymbol{\omega}}}_i &= \frac{\frac{\Delta\boldsymbol{\theta}_t^{c_i}}{\Delta t}s_{t-1}-\frac{\Delta\boldsymbol{\theta}_{t-1}^{c_i}}{\Delta t}s_t}{{s_t}^2 s_{t-1}\Delta t}\\ \tilde{\boldsymbol{\omega}}_i &= \frac{\Delta\boldsymbol{\theta}_t^{c_i}}{s_t\Delta t}\end{aligned} \tag{9.21}$$

In similar way we obtain

$$\dot{\boldsymbol{q}}_{s_t} = \frac{\Delta\boldsymbol{q}_t}{s_t\Delta t} \tag{9.22}$$

It is clear that the optimization problem (9.13) is polynomial in s_t, so the gradient and Jacobian functions can be calculated easily.

9.3 Discretization of Solution Space

In real fact, the space of the admissible solutions of the minimization problem (9.13) is very large. In order to transform this space to a smaller dimensional space, we can use a basis of

shape functions (e,g cubic B-spline functions).

Let us consider a basis of shape functions $\boldsymbol{B}_t$ that is defined as follows

$$\boldsymbol{B}_t = \begin{bmatrix} B_t^1 & B_t^2 & \cdots & B_t^l \end{bmatrix}^T \tag{9.23}$$

where B_t^i denotes the value of shape function number i at the instant t, the dimension of $\boldsymbol{B}_t$ is l defines the dimension of the basis of shape functions.
The projection of s_t into the basis of shape functions B_t can be given by the following formula

$$s_t = \sum_{i=1}^{l} s_B^i \; B_t^i = \boldsymbol{s}_B^T \; \boldsymbol{B}_t \tag{9.24}$$

Thus, the optimization problem (9.13) can be rewritten as follows

$$\begin{aligned} &\min_{\boldsymbol{s}_B} \sum_{k=1}^{l} s_B^k \int_{t=0}^{T} B_t^k \; dt \\ &\text{subject to} \\ &\boldsymbol{s}_B^T \; \boldsymbol{B}_t > 0 \\ &\boldsymbol{p}_{\boldsymbol{s}_B}^- \leq \boldsymbol{p}_{\boldsymbol{s}_B} \leq \boldsymbol{p}_{\boldsymbol{s}_B}^+ \\ &\boldsymbol{f}_{\boldsymbol{s}_B}^T \; \boldsymbol{n} < 0 \\ &\mu \; \boldsymbol{f}_{\boldsymbol{s}_B}^T \; \boldsymbol{n} < \boldsymbol{\tau}_{\boldsymbol{s}_B}^T \; \boldsymbol{n} < -\mu \; \boldsymbol{f}_{\boldsymbol{s}_B}^T \; \boldsymbol{n} \\ &\dot{\boldsymbol{q}}^- \leq \dot{\boldsymbol{q}}_{\boldsymbol{s}_B} \leq \dot{\boldsymbol{q}}^+ \end{aligned} \tag{9.25}$$

The optimization problem has been transformed into finding the vector $\boldsymbol{s}_B \in \mathbb{R}^l$.

Note that the support polygon is a function of $\boldsymbol{s}_B$, and it depends on the horizontal position of CoM. However, as the given path is a statically stable one, it can be split into various sections. Each section is a statically stable path which has a fixed support polygon that is independent of $\boldsymbol{s}_B$.

9.4 Implementation Algorithm

The algorithm of the implementation can be summarized as follows

1. Given a path which is supposed to be statically stable.
2. Split the path into various sections depending on the place and shape of support polygon during the motion. The support polygon for each section is fixed.
3. Choose Δt, e.g. $\Delta t = 5.10^{-3} s$.
4. Transform each section of the initial path to a trajectory by considering a uniform time distribution function ($s_t = 1, \forall t$).

5. Calculate $\Delta \boldsymbol{q}_t$, $\boldsymbol{X}_t^{c_i}$, $\Delta \boldsymbol{X}_t^{c_i}$ and $\Delta \boldsymbol{\theta}_t^{c_i}$ for each section of the path.
6. Calculate the cubic B-spline functions.
7. Solve the optimization problem (9.25) for each section with the initial solution obtained from the above steps.

9.5 Experimental Results

The considered example is a collision-free reaching motion in a cluttered environment. In this example, the task for the humanoid robot is to move its left hand to the specified position, and at same time to keep the projection of its CoM inside of the polygon of support (statically stable motion). This collision-free motion was created using the method proposed in [Kanehiro et al. 2008]. Fig. 9.1 shows snapshots of the simulated motion. In the initial configuration, the robot is standing on its right foot and surrounded by a torus, a cylinder and a box.

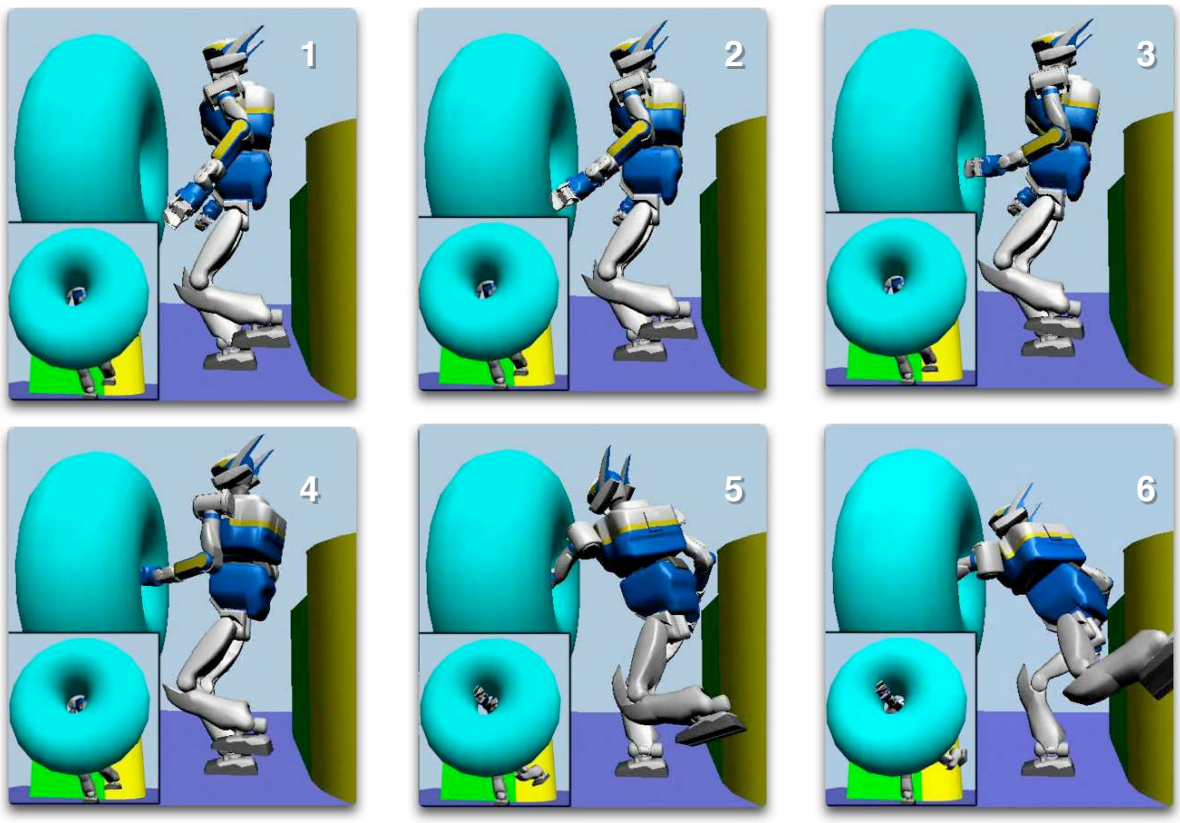


Figure 9.1: Whole body reaching with collision avoidance

Once the collision-free path is available, this path is converted to a trajectory using a uniform time distribution function. The obtained trajectory is then used to initialize Problem (9.25). Solving optimization problem (9.25) yields a minimum time and dynamically stable trajectory.

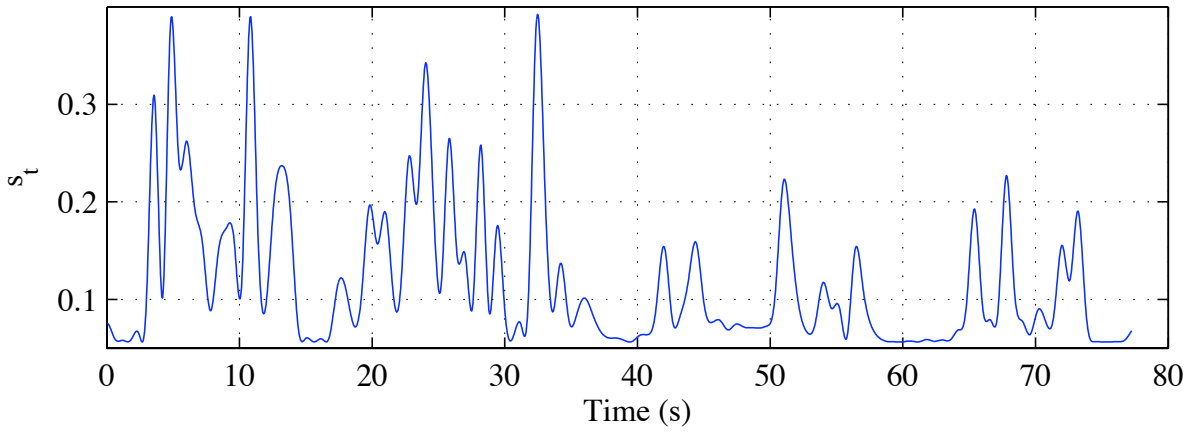


Figure 9.2: Time parameterization function (s_t)

The computation time obviously depends on the length of the path and on the dimension of the basis of cubic B-spline functions. For this scenario, we have chosen: $\Delta t = 5.10^{-3}s$, and a basis of 120 cubic B-spline functions. The function s_t is given in Figure 9.2. The duration of minimum time dynamically stable trajectory is around ($9s$) instead of ($77s$) for the original uniform time distribution trajectory. That means the optimized trajectory is around 9 times faster than the original one. However it's not always the case. For example, if the original trajectory is fast and dynamically unstable, then the optimized trajectory will be slower. Though the optimized trajectory is the minimum time trajectory among the dynamically stable trajectories. The trajectory of ZMP is given in Figure 9.3.

Fig. 9.4 shows snapshots of the conducted motion applied to the humanoid robot HRP2.

9.6 Conclusion

In this chapter, we proposed a numerical method to solve the time parameterization problem of humanoid robot paths. The main contribution of this method is transforming a statically stable path to a minimum time and dynamically stable trajectory which respects the velocity limits of the humanoid robot's joints.

The initial statically stable path can be calculated using inverse kinematic methods or the motion planning methods. This path, by definition, is a pure geometric description of the motion.

The effectiveness of the proposed method has been validated using the humanoid robot HRP2.

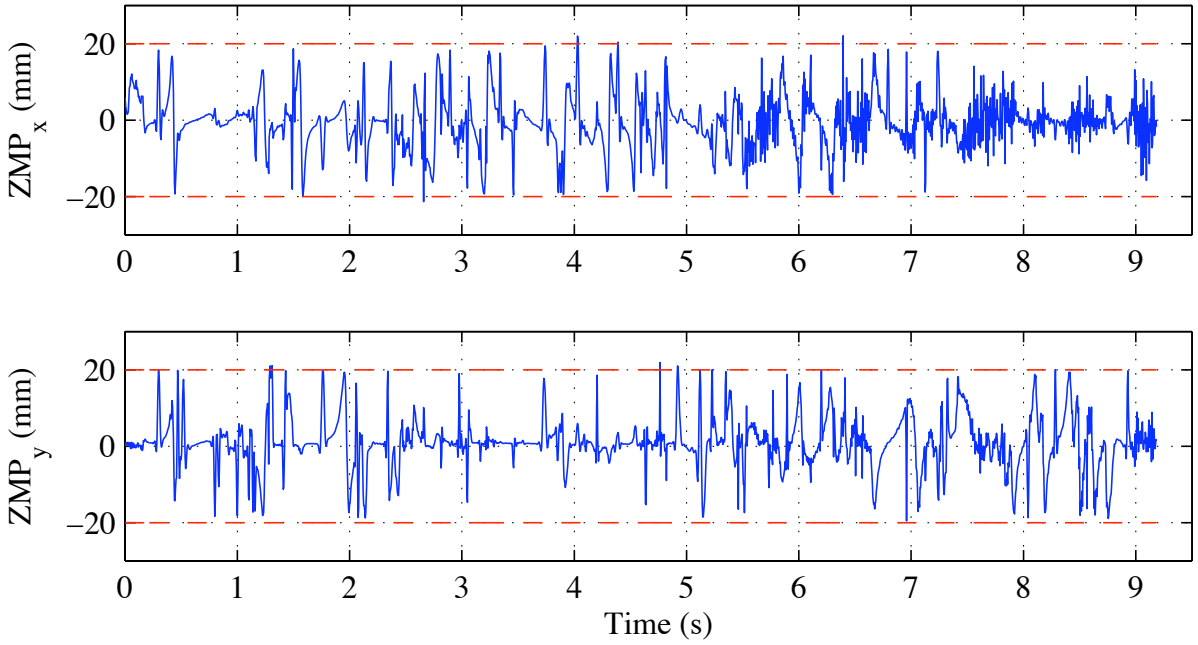


Figure 9.3: ZMP_x and ZMP_y trajectories in solid lines, and the safety zones for dynamical stability are designed by the dashed lines

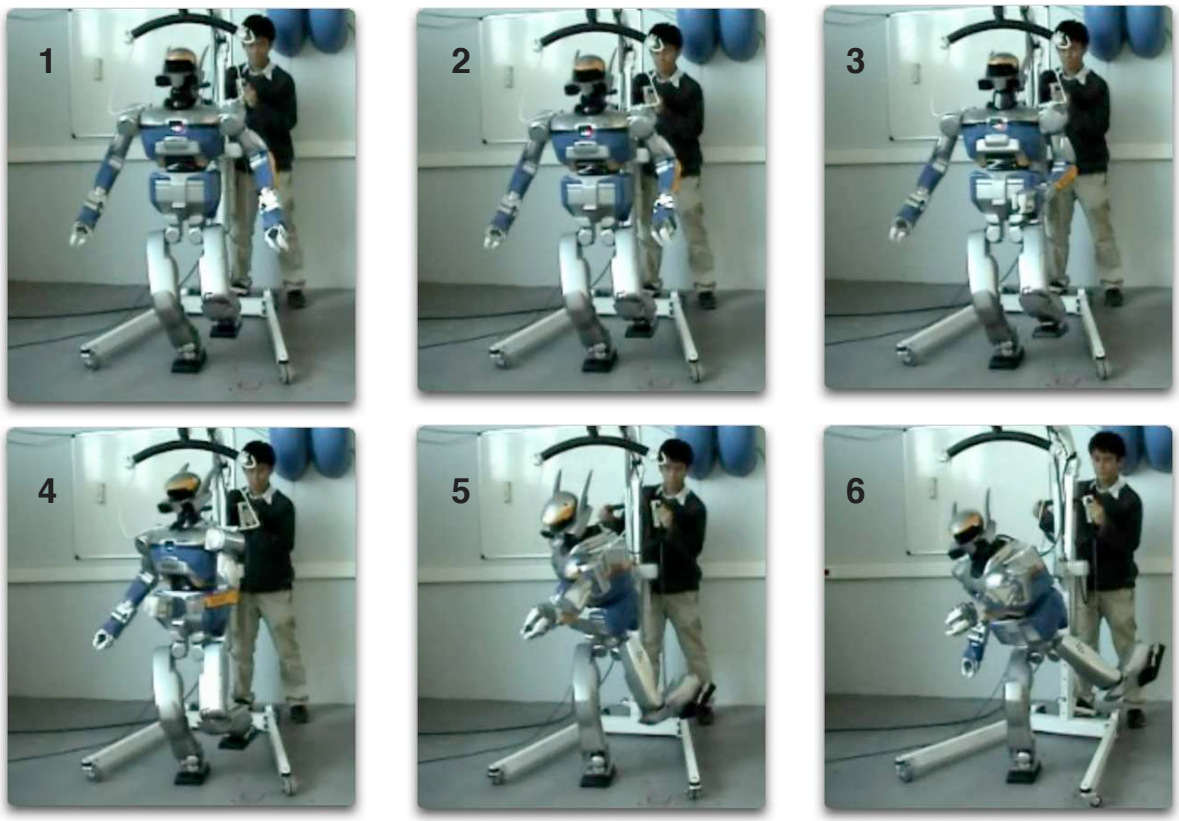


Figure 9.4: Snapshots of the whole body reaching motion

Conclusion générale

Le travail présenté dans de cette thèse se situe au carrefour de plusieurs disciplines dont les principales sont l'identification des systèmes dynamiques et l'optimisation des mouvements. Il ouvre également la voie à divers plusieurs sujets de recherche qui pourraient être dérivés de notre travail.

Nous avons notamment considéré dans un premier temps l'identification des systèmes linéaires et de quelques classes des systèmes non-linéaires puis dans un second temps l'optimisation des mouvements des robots humanoïdes.

Bien que l'identification des systèmes linéaires ait été beaucoup étudiée durant ces dernières années, le cas des expérimentations multiples n'a pas fréquemment abordé. Quoique la plupart des méthodes conventionnelle d'identification des systèmes linéaires échouent à identifier le système linéaire dans ce cas, nous montrons que le traitement simultané des données fournit un modèle précis et fiable du système linéaire étudié.

Cependant, il est bien connu que le modèle linéaire a des limites et que la plupart des systèmes réels manifestent des comportements non-linéaires. Pour cette raison, nous nous sommes intéressés à l'identification des systèmes non-linéaires. Comme la définition d'un modèle universel pour la classe des systèmes non-linéaires est une tâche inaccessible, nous nous sommes focalisés sur deux classes spécifiques de systèmes non-linéaires. Ces classes sont celle des séries de Volterra d'ordre fini et à horizon infini et celle des systèmes quadratiques en l'état.

Pour les séries de Volterra à horizon infini, nous avons montré que ces séries peuvent être approchées par une réalisation de dimension finie dans l'espace d'état et avons proposé une méthode d'identification de cette réalisation. La méthode est basée sur une méthode d'optimisation de type gradient combinée à une méthode de paramétrisation locale. L'utilisation de la méthode de paramétrisation locale non seulement réduit le nombre de calculs de gradient au minimum, mais aussi surmonte le problème de non-unicité de la solution optimale. De plus, nous avons proposé une méthode de projection séquentielle afin d'estimer une solution initiale des paramètres de la réalisation dans l'espace d'état. Cette estimation a servi comme valeur initiale pour la méthode d'optimisation de type gradient. Une comparaison avec des méthodes conventionnelles pour l'identification des systèmes non-linéaires a révélé l'efficacité de notre méthode et les précisions des résultats obtenus ont largement dépassé ceux des méthodes classiques.

Par ailleurs, nous avons montré que les systèmes quadratiques en l'état jouent un rôle important dans plusieurs domaines de recherche. De ce fait, nous avons développé une méthode pour identifier cette classe de systèmes. Cette méthode est basée sur la minimisation de l'erreur de sortie et une méthode de paramétrisation locale du modèle d'état des systèmes quadratiques.

Nous avons alors considéré l'application de ces méthodes d'identification à la modélisation et à la synthèse de la locomotion humaine. Étant donné que ce problème est très complexe non seulement du fait des non-linéarités mais aussi à cause de sa nature multivariable, la contribution majeure de la méthode proposée est de montrer que la locomotion humaine pouvait être modélisée par des systèmes linéaires, ceci grâce à une décomposition de la structure du corps humain en cinq chaînes cinématiques simples et à l'utilisation de la paramétrisation "exponential-map".

D'autre part, nous avons traité le problème de l'optimisation des mouvements. Bien que nous ayons appuyé notre étude sur l'application aux robots humanoïdes, les méthodes développées peuvent être étendues à des systèmes anthropomorphiques plus généraux.

En premier lieu, nous avons développé une méthode qui a pour but d'optimiser les mouvements des robots humanoïdes. Les entrées de cette méthode sont des mouvements calculés par des méthodes de planification de mouvement et les mouvements obtenus sont des mouvements optimisés et dynamiquement stable (le robot conserve son équilibre et ne chute pas).

En second lieu, nous avons considéré l'imitation des mouvements humains par un robot humanoïde; à cet effet nous avons formalisé ce problème comme un problème d'optimisation et avons proposé une méthode efficace de résolution. Les mouvements obtenus non seulement imitent les mouvements humains capturés, mais respectent aussi les limites physiques du robot humanoïde.

En dernier lieu, nous avons abordé le problème du paramétrage temporel des chemins dans l'espace de configuration pour un robot humanoïde. Bien que ce problème de paramétrage temporel soit un problème ancien en robotique, il n'était pas abordé dans le cas des robots humanoïdes. Afin de résoudre ce problème, nous avons proposé une méthode numérique destinée à transformer un chemin statiquement stable à une trajectoire qui est non seulement dynamiquement stable mais aussi dans les limites des vitesses angulaires du robot humanoïde.

Les méthodes proposées ci-dessus, qu'il s'agisse de l'optimisation des mouvements, de l'imitation des mouvements humains ou du paramétrage temporel, ont été validées avec succès sur la plate-forme HRP-2 №14. Les résultats expérimentaux ont également mis en exergue l'efficacité et la robustesse de ces méthodes.

Perspectives

Étant donné que les nombreuses contributions de ce travail de thèse sont pluridisciplinaires,

nous pouvons imaginer plusieurs directions de recherche en perspectives.

- *Identifications des systèmes dynamiques*
  - Un axe de recherche prometteur est de combiner une méthode d'optimisation globale avec une méthode de paramétrisation locale afin d'identifier les séries de Volterra et les systèmes quadratiques en l'état. La méthode de paramétrisation locale rejette les directions dans lesquelles la fonction de l'erreur de sortie ne change pas, par conséquent le problème de non-unicité du minimum global est surmonté.
  - L'utilisation conjointe de plusieurs expérimentations dans l'identification de systèmes dynamiques a été restreinte ici essentiellement aux systèmes linéaires. Une extension à une classe de systèmes plus riche pourrait être aisément envisagée. Rappelons que cette extension au cas des systèmes quadratiques en l'état a été développée dans le chapitre 5.
  - L'application de la théorie de l'identification des systèmes dynamiques à la modélisation et à la synthèse de la locomotion humaine constitue une première étape vers la synthèse d'autres types des mouvements humains comme la course, le saut et la voltige.
- *Optimisation des mouvements*
  - Afin de faire la connexion avec la théorie de l'identification des systèmes dynamiques, nous pouvons penser à enrichir le modèle dynamique du robot humanoïde en identifiant le modèle des articulations flexibles situées aux chevilles. Nous pouvons penser que les résultats obtenus en utilisant ce modèle enrichi gagneront en précision.
  - Une idée prometteuse est d'intégrer les contraintes de ZMP (Zero Moment Point) et de l'évitement des obstacles comme des contraintes supplémentaires dans la formalisation du problème d'optimisation des mouvements du robot et de l'imitation des mouvements humains par un robot humanoïde.
  - Plusieurs applications intéressantes de la méthode de paramétrage temporel que nous avons proposée dans le chapitre 9 peuvent être envisagées. Par exemple, transformer un chemin en une trajectoire lisse en minimisant la fonction de jerk (variation de l'accélération).

Enfin, l'adaptation des méthodes développées pour l'optimisation des mouvements des robots humanoïdes, l'imitation des mouvements humains et le paramétrage temporel des chemins aux applications temps réel devrait être considérée comme une perpective réalisable. À cet effet, nous pouvons envisager quelques modifications de la fonction objectif, relaxer les critères de convergence ou considérer un délai fixe entre le mouvement réel et celui reproduit par le robot humanoïde, ce délai constituant le temps nécessaire pour optimiser un segment fixe du mouvement réel observé. Cette axe de recherche fait l'objet d'un travail de recherche déjà engagé.

"Science may set limits to knowledge, but should not set limits to imagination."

Bertrand Russell

10

Conclusion and Prospective Research

In this thesis, we gave new insights into nonlinear system identification and motion optimization. There are several research directions that stem from our work. Some of them are the subject of actual and future research works in our research groups.

The research work in this thesis mainly deals with the identification of linear systems and some classical nonlinear models, and the optimization of humanoid robot motions.

Although linear system identification has been subject of extensive research investigation during the last years, the case of multiple short data sets is infrequently considered. Though most of classical linear system identification fail in identifying the linear system in this case, we have pointed out that treating the data sets simultaneously provides an accurate and reliable model of the linear system.

However, the linear models have their limitations. It is well known that most real life systems show nonlinear dynamic behavior; a linear model can only describe such a system for a small range of input and output values. Therefore, we were interested in nonlinear system identification. As building a universal nonlinear system model is a very difficult task, we tackled in particular two models of nonlinear systems. These models are finite degree Volterra series with infinite horizon and quadratic in-the-state systems.

For finite degree Volterra series with infinite horizon, a state space realization has been pointed out and we have proposed a new method to identify it. This method is based on a local gradient search in a local parameterization of the state space realization. Using the local parameterization not only reduces the amount of the gradient calculations to the minimal value,

but also overcomes the nonuniqueness problem of the optimal solution. Moreover, we proposed a sequential projection method to provide an initial estimation of the parameters of state space realization. This estimation is used to initialize the gradient search method. A comparison with the conventional methods for nonlinear system identification has borne out the outperformance of the proposed method.

Furthermore, we have pointed out that the quadratic in-the-state systems enjoy a useful model in many research fields. Thus we have described a method to identify the quadratic in-the-state systems. This method is based on output error identification and local parameterization of the state space model of quadratic systems.

The application of system identification methods to model and to synthesize human locomotion has been described as well. As such the general identification problem is very challenging because of its nonlinearity and multi-dimensional nature, the core contribution of the proposed method is to show that human locomotion can be modeled by linear systems thanks to a decomposition of the human body structure into simple kinematic chains and using exponential-map parameterization.

Besides system identification, *motion optimization* fills an important part of this research work. Although we emphasized on the application to humanoid robots, the methods can be extended to a general anthropomorphic system.

We developed an optimization framework for humanoid robot motions. This framework takes as input a pre-calculated motion which is provided by motion planning techniques and the output is an optimized and stable motion.

Moreover, the imitation of human captured motions by a humanoid robot is considered as well. Unlike the conventional methods for human motion imitation, the proposed method provides a unified optimization framework for human motion imitation by a humanoid robot. The generated motions not only imitate the original human captured motion, but also respect the physical limits of humanoid robot.

Finally, the time parameterization of humanoid robot paths is discussed. The time parameterization problem is an old problem in robotic. Nevertheless, this problem has not been investigated, up to our best knowledge, in the case of humanoid robots. To solve this problem, we proposed a numerical method that has the objective of transforming a statically stable path to a minimum time and dynamically stable trajectory that respects the velocity limits of humanoid robot's joints.

The above methods for motion optimization, human motion imitation and time parameterization have been successfully validated on the *humanoid robot HRP2*.

Prospective Further Work

Since the original contributions in this thesis are applied to several research fields, many

research directions can be considered in further research work.

- *System identification and its applications*
  - An interesting further research is combining a global optimization method and a local parameterization method in order to identify finite degree Volterra series and quadratic in-the-state system. The local parameterization method projects out the directions that do not change the value of output error function, and therefore overcomes the problem of nonuniqueness global optimal solution, if it exists.
  - The key point in the algorithm of linear system identification using multiple data sets is that the model is differentiable with respect to its vector of parameters. Therefore, the extension of the proposed method to models more complex than the linear model can be developed in an analogues way. Recall that the extension for quadratic in-the-state system is described in Chapter 5.
  - The application of system identification theory to model and synthesize human locomotion can be further extended to other kinds of human motions such as running, leaping and vaulting.

- *Motion optimization*
  - To make the connection with system identification theory and its application, we can think of enhancing the dynamical model of the humanoid robot by identifying the model of flexible joints situated in the ankle joints. We expect that the obtained results using this enhanced dynamical model will be more accurate.
  - A promising idea is integrating the Zero Moment Point (ZMP) constraints and obstacle avoidance as additional constraints into the formulation of humanoid motion optimization and human motion imitation.
  - Several interesting applications stem from the time parameterization method described in Chapter 9. For instance, transforming a path into a smooth trajectory by minimizing the jerk function.

Finally, adapting the developed methods for humanoid motion optimization, human motion imitation and time parameterization to deal with realtime applications can be considered as well. To achieve this goal, we may make some simplifications of the objective functions, consider some relaxation of the convergence criteria or consider a fixed delay between the real motion and the reproduced one, this delay constitutes the required computation time to optimize a fixed segment from the observed real motion. This line of research is the subject of an ongoing research.

A

Proofs

In this appendix, we give the proofs of the theorems and the lemmas reported in the thesis.

A.1 Proofs of Chapter 2

A.1.1 Proof 2.1

The proof of both equations (2.54) and (2.55) is very similar. Using Eq. (2.52) and by multiplying on the right by $\frac{1}{\sqrt{N}}Q_2^T$ *we obtain*

$$\frac{1}{\sqrt{N}}Y_{1+\alpha,\alpha,N}Q_2^T = \underbrace{\frac{1}{\sqrt{N}}\Gamma_\alpha X_{1+\alpha,N}Q_2^T}_{i} + \underbrace{\frac{1}{\sqrt{N}}\Phi_\alpha U_{1+\alpha,\alpha,N}Q_2^T}_{ii} + \underbrace{\frac{1}{\sqrt{N}}V_{1+\alpha,\alpha,N}Q_2^T}_{iii} \tag{A.1}$$

Let us consider the term (ii)*, using Eq. (2.53) we obtain*

$$U_{1+\alpha,\alpha,N} = R_{11}Q_1 \tag{A.2}$$

As the matrix Q *is an orthonormal matrix, we have*

$$Q_1Q_2^T = \mathbf{0} \tag{A.3}$$

As a consequence the term (ii) is equal to zero. Using the independency of u_t and v_τ, $\forall t, \tau$, we can write

$$\frac{1}{\sqrt{N}} U_{1+\alpha,\alpha,N} V_{1+\alpha,\alpha,N}^T = O_N(\epsilon) \tag{A.4}$$

where $O_N(\epsilon)$ is a matrix of appropriate dimension and ϵ-norm for finite N and vanishing when $N \to \infty$.
Using Eq. (A.2), we find that

$$\frac{1}{\sqrt{N}} R_{11} Q_1 V_{1+\alpha,\alpha,N}^T = O_N(\epsilon) \tag{A.5}$$

Caused by the persistency of excitation of the input signal, the matrix R_{11} is invertible. Therefore

$$\frac{1}{\sqrt{N}} Q_1 V_{1+\alpha,\alpha,N}^T = O_N(\epsilon) \tag{A.6}$$

we can similarly derive that

$$\begin{aligned} \frac{1}{\sqrt{N}} U_{1,\alpha,N-\alpha} V_{1+\alpha,\alpha,N}^T &= O_N(\epsilon) \\ \frac{1}{\sqrt{N}} \left(R_{21} Q_1 + R_{22} Q_2\right) V_{1+\alpha,\alpha,N}^T &= O_N(\epsilon) \end{aligned} \tag{A.7}$$

Using (A.6), we find that

$$\frac{1}{\sqrt{N}} R_{22} Q_2 V_{1+\alpha,\alpha,N}^T = O_N(\epsilon) \tag{A.8}$$

As the matrix R_{22} is invertible, we can write

$$\frac{1}{\sqrt{N}} Q_2 V_{1+\alpha,\alpha,N}^T = O_N(\epsilon) \tag{A.9}$$

As a result, we obtain Eq. (2.54). The proofs of Eqs (2.55 - 2.58) can be easily derived similarly to the above proof. For more details see [Verhaegen 1994]. ■

A.2 Proofs of Chapter 3

A.2.1 Proof 3.1

The linear system (3.17) is stable if and only if $\rho(A) < 1$ and $\|u_t^i\|_2 < \infty$. As a consequence of the stability of the linear system, we obtain that $\|\hat{x}_t^i\|_2 < \infty$.
Let us define

$$\tilde{u}_t^i = \begin{bmatrix} \hat{x}_{t-1}^i \\ u_t^i \end{bmatrix} \tag{A.10}$$

Using (A.10), the system (3.23) can be reformulated as follows

$$\begin{aligned}\zeta_t^{i,j} =& A\zeta_{t-1}^{i,j} + \begin{bmatrix}\dfrac{\partial A}{\partial \theta_j} & \dfrac{\partial B}{\partial \theta_j}\end{bmatrix} \tilde{u}_t^i \\ \frac{\partial \hat{y}_t^i}{\partial \theta_j} =& C\zeta_t^{i,j} + \begin{bmatrix}\dfrac{\partial C}{\partial \theta_j} & \mathbf{0}\end{bmatrix} \tilde{u}_t^i\end{aligned} \tag{A.11}$$

Thus, the above linear system is stable because $\rho(A) < 1$ *and* $\|\tilde{u}_t^i\|_2 < \infty$. ■

A.3 Proofs of Chapter 4

A.3.1 Proof 4.1

Consider that the total number of parameters of a truncated Volterra series of degree l *and horizon length* T *is* $C(l,T)$.
Let us define $C_n(T)$ *as the total number of parameters of the Volterra kernels of degree* n *(the number of elements of the matrices* $w_n(\tau_1,\tau_2,\ldots,\tau_n),\ 0 \le \tau_n \le \tau_{n-1} \le \cdots \le \tau_1 \le T$*).*

It is obvious that

$$C(l,T) = \sum_{n=1}^{l} C_n(T) \tag{A.12}$$

Suppose S *is any finite set of elements and denote the number of these elements by* $\sharp S$*, then*

$$\begin{aligned}C_n(T) &= pm^n \,\sharp\{0 \le \tau_n \le \tau_{n-1} \le \cdots \le \tau_1 \le T\} \\ &= pm^n \sum_{k=0}^{T} \sharp\{0 \le \tau_n \le \tau_{n-1} \le \cdots \le \tau_1 \le k\} \\ &= pm^n \sum_{k=0}^{T} \binom{k+n-1}{n-1}\end{aligned} \tag{A.13}$$

where $\binom{\cdot}{\cdot}$ *denotes the binomial coefficient defined as follows*

$$\binom{N}{K} = \frac{N!}{K!(N-K)!} \tag{A.14}$$

Using the following properties

$$\begin{aligned} \binom{K}{N} &= \binom{N}{N-K} \\ \sum_{k=0}^{T} \binom{k+q}{k} &= \binom{T+q+1}{T} \end{aligned} \tag{A.15}$$

One could rewrite $C_n(T)$ as follows

$$C_n(T) = \binom{T+n}{T} p m^n = \binom{T+n}{n} p m^n \tag{A.16}$$

Finally, by using (A.12) and (A.16) we obtain

$$\begin{aligned} C(l,T) &= p \left\{ \sum_{n=1}^{l} \binom{T+n}{n} m^n \right\} \\ &= p \left\{ \sum_{n=1}^{l} \frac{(T+n)!}{n!T!} m^n \right\} \end{aligned} \tag{A.17}$$

■

A.3.2 Proof 4.2

Each subsystem may be viewed as a linear system with input $(u_t \otimes Z_t^{i-1})$. So, Z_t^i will be asymptotically stable if and only if the linear system is stable $(\rho(A^i) < 1)$ and $u_t \otimes Z_t^{i-1}$ is bounded. Assuming that Z_t^1 is stable $(\rho(A^1) < 1)$, so $u_t \otimes Z_t^1$ is bounded and Z_t^2 is stable if and only if $(\rho(A^2) < 1)$. And so on. ■

A.3.3 Proof 4.3

Equation (4.19) shows that the tangent space of the manifold of equivalent realizations at $(A^i, B^i, C^i : i = 1, 2, \cdots, l)$ is equal to the column space of the matrix M_θ (4.20). Since the left null space of the matrix M_θ is orthogonal complement to the column space, the directions in which the value of the cost function $J_N(\theta)$ changes are those related to left null space of M_θ. ■

A.3.4 Modified search algorithm with dimension reduction

A first order expansion of prediction error $E_N(\theta)$ (4.12) around θ

$$E_N(\theta+\delta\theta) \approx E_N(\theta)+\psi_N(\theta)\delta\theta \tag{A.18}$$

where

$$\psi_N(\theta) \triangleq \frac{\partial E_N(\theta)}{\partial \theta^T} \tag{A.19}$$

is the jacobian of the error vector $E_N(\theta)$.

By restricting the parameter update to directions orthogonal to the equivalence class we can write the perturbation as a function of a vector $\tilde{\delta\theta}$

$$\delta\theta(\tilde{\delta\theta}) = Q_2\tilde{\delta\theta} \tag{A.20}$$

where Q_2 is given in (4.21). With this restriction the linearization becomes

$$E_N(\theta+\delta\theta(\tilde{\delta\theta})) \approx E_N(\theta)+\psi_N(\theta)Q_2\tilde{\delta\theta} = E_N(\theta)+\tilde{\psi}_N(\theta)\tilde{\delta\theta} \tag{A.21}$$

By using the first order expansion of the error vector the criterion (4.11) becomes

$$J_N(\theta+\delta\theta(\tilde{\delta\theta})) = \frac{1}{N}\|E_N(\theta)+\tilde{\psi}_N(\theta)\tilde{\delta\theta}\|^2 \tag{A.22}$$

which is a quadratic function of $\tilde{\delta\theta}$. The Gauss-Newton algorithm is based on a successive updating rule of the system parameters θ as follows

$$\begin{aligned} \hat{\theta}^{i+1} &= \hat{\theta}^i - \delta\theta \\ &= \hat{\theta}^i - Q_2\tilde{\delta\theta} \end{aligned} \tag{A.23}$$

where $\tilde{\delta\theta}$ minimizes (A.22) and it can be obtained by the following formula

$$\tilde{\delta\theta} = (\tilde{\psi}_N^T(\hat{\theta}^i)\tilde{\psi}_N(\hat{\theta}^i)+\lambda^i I)^{-1}\tilde{\psi}_N^T(\hat{\theta}^i)E_N(\hat{\theta}^i) \tag{A.24}$$

Thus, the update rule becomes

$$\hat{\theta}^{i+1} = \hat{\theta}^i - Q_2\left(Q_2^T\psi_N^T\psi_N Q_2+\lambda^i I\right)^{-1}Q_2^T\psi_N^T E_N \tag{A.25}$$

A.3.5 Semidefinite problem formulation

The objective is transforming Eq. (4.41) to a semidefinite optimization problem.

Using the property

$$\Gamma_\alpha^1(1:(\alpha-1)p,:)A^1 = \Gamma_\alpha^1(p+1:\alpha p,:) \tag{A.26}$$

and replacing the stability condition $\rho(A^1) < 1$ by equivalent Lyapunov inequalities

$$\rho(A^1) < 1 \iff \exists P \geq \delta I_{n_1} : \; P - A^1 P A^{1T} \geq \delta I_{n_1}$$

where $\delta > 0$, we can transform the minimization problem (4.41) into the following one

$$\min_{\hat{A}^1} \; J\left(\hat{A}^1\right) \triangleq \| \, L_{\hat{A}^1} \left(\mathcal{Q}_s^\dagger - \mathcal{Q}_s^\ddagger \hat{A}^1\right) R_{\hat{A}^1} \|_F^2 \tag{A.27}$$

subject to

$$\begin{bmatrix} P - \delta I_{n_1} & X^T \\ X & P \end{bmatrix} \geq 0_{2n_1} \tag{A.28}$$

where

$$\begin{aligned} \mathcal{Q}_s^\dagger &\triangleq \mathcal{Q}_s(p+1:\alpha p,:) \\ \mathcal{Q}_s^\ddagger &\triangleq \mathcal{Q}_s(1:(\alpha-1)p,:) \\ X &\triangleq \hat{A}^1 P \end{aligned} \tag{A.29}$$

where $R_{\hat{A}^1}$ and $L_{\hat{A}^1}$ are positive-definite matrices.

If we let $R_{\hat{A}^1} \triangleq P$ and $L_{\hat{A}^1}$ be equal to the identity matrix $\left(L_{\hat{A}^1} \triangleq I_{\alpha(p-1)}\right)$, the problem is converted to an optimization problem that involves minimizing a linear function over symmetric cones [Lacy and Bernstein 2003] as follows

$$\min_x c_x^T x \tag{A.30}$$

subject to

$$\begin{bmatrix} -\mathcal{Q}_s^\ddagger & \mathcal{Q}_s^\dagger \end{bmatrix} X_3 \begin{bmatrix} 0_{n_1 \times n_1} \\ I_{n_1} \end{bmatrix} = X_2 \tag{A.31}$$

$$\begin{bmatrix} 0_{n_1} & I_{n_1} \end{bmatrix} X_3 \begin{bmatrix} 0_{n_1} \\ I_{n_1} \end{bmatrix} - \delta I_{n_1} = \begin{bmatrix} I_{n_1} & 0_{n_1} \end{bmatrix} X_3 \begin{bmatrix} I_{n_1} \\ 0_{n_1} \end{bmatrix} \tag{A.32}$$

$$X_1 \geq \|X_3\|_F \tag{A.33}$$

$$X_3 \geq 0 \tag{A.34}$$

where $x_i = vec(X_i)$, $c_x = \begin{bmatrix} 1 & 0_{1 \times \{4n_1^2 + n\alpha(p-1)\}} \end{bmatrix}^T$, $x \triangleq \begin{bmatrix} x_1 & x_2^T & x_3^T \end{bmatrix}^T \in \mathbb{R}^{4n_1^2 + n_1\alpha(p-1)+1}$, and $X_1 = x_1$ represents the value of the cost function $J\left(\hat{A}^1\right)$, $X_2 \in \mathbb{R}^{\alpha(p-1) \times n_1}$ represents the matrix $L_{\hat{A}^1}\left(\mathcal{Q}_s^\dagger - \mathcal{Q}_s^\ddagger \hat{A}^1\right) R_{\hat{A}^1}$, and $X_3 \in \mathbb{R}^{2n_1 \times 2n_1}$ represents the matrix in (A.28). The equality constraint (A.31) ensures that the blocks of X_3 corresponding to P and Q. The equality

constraint (A.32) ensures that the matrix P is used to construct the blocks [1,1] and [2,2] of X_3. The quadratic constraint (A.33) enforces the stability constraint (A.28).

To transform the minimization problem into a form suitable for use with convex optimization methods, we rewrite the equality constraints as a function of x_i and obtain the following optimization problem

$$\min_x c_x^T x \tag{A.35}$$

subject to

$$A_x x = b_x \tag{A.36}$$

$$x_1 \geq \|x_2\|_F \tag{A.37}$$

$$X_3 \geq 0_{2n_1} \tag{A.38}$$

where $A_x \in \mathbb{R}^{n^2+4n_1^2\alpha(p-1)+n_1\alpha(p-1)+1}$ is given by

$$A_x = \begin{bmatrix} 0_{n_1\alpha(p-1)\times 1} & -I_{n_1\alpha(p-1)} & \begin{bmatrix} 0_{n_1} & P \end{bmatrix} \otimes \begin{bmatrix} -\mathcal{Q}_s^{\ddagger} & \mathcal{Q}_s^{\dagger} \end{bmatrix} \\ 0_{n_1^2\times 1} & 0_{n_1^2\times n_1\alpha(p-1)} & \begin{bmatrix} 0_{n_1} & I_{n_1} \end{bmatrix} \otimes \begin{bmatrix} 0_{n_1} & I_{n_1} \end{bmatrix} - \begin{bmatrix} I_{n_1} & 0_{n_1} \end{bmatrix} \otimes \begin{bmatrix} I_{n_1} & 0_{n_1} \end{bmatrix} \end{bmatrix} \tag{A.39}$$

and $b_x \in \mathbb{R}^{n_1^2+n_1\alpha(p-1)}$ is given by

$$b_x \triangleq \delta \begin{bmatrix} 0_{n_1\alpha(p-1)\times 1} \\ vec(I_{n_1}) \end{bmatrix} \tag{A.40}$$

B

Lie Groups and Algebras

To understand the definition of Lie groups and algebras, we start by giving some basic definitions from topology.

A **manifold** is a mathematical space in which every point has a neighborhood which resembles Euclidean space, and the dimension of this Euclidean space determines the dimension of the manifold. Note that the global structure may be more complicated than a simple Euclidean space.

Examples of one dimensional manifold includes a line, a circle and two separate circles. For these examples every point has a neighborhood that looks like a segment of a line. For two dimensional manifold, every point has a neighborhood that looks like a disk. As examples, we find a plane, the surface of a sphere and the surface of a torus.

The manifolds are important objects not only in mathematics but also in physics. The introduction of manifolds has offered a great tool to express and understand complicated structures in terms of the well-understood properties of simpler spaces (Euclidean spaces).

One of the important application fields of manifolds is the analytical mechanics, which has been developed by *Simeon Poisson, Charles Jacob Jacobi* and *William Rowan Hamilton*. For mechanical systems, all possible states can be viewed as points of an abstract space, which is the phase space in Lagrangian and Hamiltonian formalisms of classical mechanics. This space is, indeed, a high dimensional manifold, whose dimension corresponds to the degrees of freedom of the system and where the points are specified by their generalized coordinates. Another geometrical and topological aspects of classical mechanics were investigated by *Henri Poincaré*,

one of the founders of modern topology and the special theory of relativity.

A special kind of manifolds is differentiable manifolds, which are locally similar enough to Euclidean space to allow to do calculus (limits, derivatives, integrals and infinite series). Therefore, one can define directions, tangent spaces, and differentiable functions on that manifold. Each point of a n-dimensional differentiable manifold has a tangent space. This is a n-dimensional Euclidean space consisting of the tangent vectors of the curves through the point.

For a more formal mathematical definition of manifolds and the history of manifolds and varieties refer to [Bourbaki 2005].

Using the above definitions, a *Lie group* can be defined as a differentiable manifold that carries also the structure of a group and its operations. An example of Lie group is the orthonormal matrix Θ in $R^{3\times3}$, which is called $SO(3)$. Note that this group consists of the rotation matrices in Euclidean space. Another example of Lie group is the group of homogeneous transformation which is the special Euclidean group or $SE(3)$. Given a rotation $\Theta \in SO(3)$ and translation $b \in R^3$, the homogeneous matrix $\in SE(3)$ is defined as follows

$$G = \begin{bmatrix} \Theta & b \\ 0 & 1 \end{bmatrix} \tag{B.1}$$

An important concept associated with each Lie group is the notation of *Lie algebra*. The Lie algebra associated to a Lie group is the tangent space at the identity element of the Lie group, which completely captures the local structure of the group. We denote, the Lie algebra of $SO(3)$ and $SE(3)$ by $so(3)$ and $se(3)$ respectively.
For more details on Lie groups and algebras refer to [Bourbaki 2005; Rosenfeld and Wiebe 1997; Erdmann and Wildon 2006].

In this thesis, the main used notations and operations on Lie groups and Lie algebra are:

1. Skew operator:

$$\begin{aligned} &[.] : \omega \in R^3 \rightarrow so(3) \\ &[\omega] = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \end{aligned} \tag{B.2}$$

2. $(.,.)$ operator:

$$\begin{aligned} &(.,.) : \{\omega, v\} \in R^3 \rightarrow se(3) \\ &(\omega, v) = \begin{bmatrix} [\omega] & v \\ 0 & 0 \end{bmatrix} \end{aligned} \tag{B.3}$$

3. Matrix exponential:

$$e^{(\omega,v)} = \exp \begin{bmatrix} [\omega] & v \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \exp([\omega]) & Av \\ 0 & 1 \end{bmatrix} \tag{B.4}$$

where

$$\begin{aligned} \exp([\omega]) &= I + \frac{\sin\phi}{\phi}[\omega] + \frac{1-\cos\phi}{\phi^2}[\omega]^2, \phi = \|\omega\| \\ A &= I + \frac{1-\cos\phi}{\phi^2}[\omega] + \frac{\phi-\sin\phi}{\phi^3}[\omega]^2 \end{aligned} \tag{B.5}$$

4. Adjoint map on $SE(3)$:

$$\begin{aligned} &Ad_G(h) : se(3) \to se(3) \\ &Ad_G(h) = \begin{bmatrix} \Theta & 0 \\ [b]\Theta & \Theta \end{bmatrix} \begin{bmatrix} h_\omega \\ h_v \end{bmatrix} \end{aligned} \tag{B.6}$$

where $G \in SE(3)$ is defined as in (B.1), and $h = (h_\omega, h_v) \in se(3)$.

5. Dual adjoint operator:

$$\begin{aligned} &Ad^*_G(h^*) : se(3)^* \to se(3)^* \\ &Ad^*_G(h^*) = \begin{bmatrix} \Theta^T & \Theta^T[b]^T \\ 0 & \Theta^T \end{bmatrix} \begin{bmatrix} M \\ F \end{bmatrix} \end{aligned} \tag{B.7}$$

where $G \in SE(3)$, and $h^* = (M, F) \in se(3)^*$.

6. Lie bracket operator:

$$ad_g(h) = [g, h] = \begin{bmatrix} [g_\omega] & 0 \\ [g_v] & [g_\omega] \end{bmatrix} \begin{bmatrix} h_\omega \\ h_v \end{bmatrix} \tag{B.8}$$

where $g, h \in se(3)$. $h = (h_\omega, h_v)$ and $g = (g_\omega, g_v)$.

7. Dual Lie bracket operator:

$$ad^*_g(h^*) = [g, h^*] = \begin{bmatrix} [g_\omega]^T & [g_v]^T \\ 0 & [g_\omega]^T \end{bmatrix} \begin{bmatrix} M \\ F \end{bmatrix} \tag{B.9}$$

where $g = (g_\omega, g_v) \in se(3)$ and $h^* = (M, F) \in se(3)^*$.

References

A. De Carli, M. M. and Ruberti, A. 1966. Speed Control of Induction Motors by Frequency Variations. In *Proc. 3rd IFAC Congrees*. London.

Allgöwer, F. and Zheng, A. 2000. *Nonlinear Model Predictive Control*. Birkhäuser.

Aoki, M. 1990. *State Space Modeling of Time Series*, 2nd ed. Springer-Verlag.

Åström, K. and Bohlin, T. 1965. Numerical Identification of Linear Dynamic Systems from Normal Operating Records. In *IFAC Symposium on Self-Adaptive Systems*. Teddington, UK.

Bastogne, T., Richard, A., and Sibille, P. 1998. Identification des Systèmes Multivariables : Méthodes des Sous-espaces. *Journal Européen des Systèmes Automatisés 32,* 2 (April), 235–265.

Beckett, R. and Chang, K. 1968. An Evaluation of the Kinematics of Gait by Minimum Energy. *Journal Biomechanics 1,* 147 – 159.

Bernstein, N. 1967. *The coordination and regulation of movements*. Oxford: Pergamon Oxford: Pergamon Press.

Bertsekas, D. P. 1995. *Nonlinear Programming*. Athena Scientific.

Billings, S. 1980. Identification of Nonlinear Systems: a Survey. *IEE Proceedings 127*, 272–285.

Billings, S., Jamaluddin, H., and Chen, S. 1992. Properties of Neural Networks With Applications to Modelling Non-linear Dynamical Systems. *Int. J. Control 55,* 1, 193–224.

Bourbaki, N. 2005. *Elements of Mathematics: Lie Groups and Lie Algebras*. Springer, Chapter 7–9.

Brockett, R. 1972. System theory on group manifolds and coset spaces. *SIAM J.Control 10*, 265–284.

Brockett, R. 1976. Volterra series and geometric control theory. *Automatica 12*, 167–176.

Chen, H. and Maciejowski, J. 2000a. An Improved Subspace Identification Method for Bilinear Systems. In *39th IEEE Conf. Decision Control*. Sydney, Australia, 1573–1578.

Chen, H. and Maciejowski, J. 2000b. *CUEDSID Toolbox User's Guide*. Cambridge University System Identification Toolbox, Cambridge University Engineering Dept.

Chen, H. and Maciejowski, J. 2000c. Subspace identification method for deterministic bilinear systems. *IFAC Conf. System Identification*.

DeMoor, B., Vandewalle, J., Vandenberghe, L., and Mieghem, P. V. 1988. A Geometrical Strategy for the Identication of State Space Models of Linear Multivariable Systems with Singular Value Decomposition. In *Proc. IFAC 88*. Beijing, China, 700–704.

Doyle, F. J., Pearson, R. K., and Ogunnaike, B. A. 2001. *Identification and Control Using Volterra Models*. Springer.

Erdmann, D. K. and Wildon, M. J. 2006. *Introduction to Lie Algebras*. Birkhäuser.

Fourier, J. 1798. Mémoire sur la statique. *Journal de l'École Polytechnique*.

Gauss, C. 1829. Uber ein neues allgemeins grundgesetz der mechanik. *J. Reine Angew. Math. 4*, 232–235.

Gleicher, M. 1998. Retargetting Motion to New Characters. In *ACM SIGGRAPH*. 33–42.

Gleicher, M. 2001. Motion Path Editing. *ACM Symposium on Interactive 3D Graphics*.

Gopinath, B. 1969. On the Identification of Linear Time-invariant Systems from Input-ouput Data. *Bell Syst. Tech. J. 48*, 1101–1113.

Grassia, F. S. 1998. Practical Parameterization of Rotations Using the Exponential Map. *Journal of Graphics Tools 33*.

Guilamo, L., Kuffner, J., Nishiwaki, K., and Kagami, S. 2006. Manipulability Optimization for Trajectory Generation. In *Proc. IEEE International Conference on Robotics and Automation*. 2017–2022.

Haber, R. and Unbeheauen, H. 1990. Structure Identification of Nonlinear Dynamic Systems - a Survey on Input/Output Approaches. *Automatica 26*, 651–677.

Ho, B. L. and Kalman, R. E. 1965. Effective Construction of Linear State Variable Models From Input Output Data. *Regelungstechnik 12,* 545—548.

Hsu, E., Pulli, K., and Popovic, J. 2005. Style translation for human motion. In *Proceedings of ACM SIGGRAPH*. Los Angeles , USA, 1082 – 1089.

Inamura, T., Nakamura, Y., Ezaki, H., and Toshima, I. 2001. Imitation and primitive Symbol Acquisition of Humanoids by the Integrated Mimesis Loop. In *IEEE International Conference on Robotics and Automation*. 4208–4213.

Isidori, A. 1995. *Nonlinear Control Systems*. Springer-Verlag.

Juang, J. and Pappa, R. 1985. An Eigensystem Realization Algorithm for Modal Parameter Identification and Model Reduction. *J. Guidance, Control and Dyn. 8*, 620–627.

Juditsky, A. et al. 1995. Nonlinear Black-box Modeling in System Identification: Mathematical Foundations. *Automatica 31,* 12, 1725 – 1750.

Kajita, S., Kanehiro, F., Kaneko, K., Fujiwara, K., Harada, K., Yokoi, K., and Hirukawa, H. 2003. Biped Walking Pattern Generation by using Preview Control of Zero-Moment Point. In *Proc. IEEE International Conference on Robotics and Automation*. 1620–1626.

Kanehiro, F., Hirukawa, H., and Kajita, S. 2004. OpenHRP: Open architecture Humanoid Robotics Platform. *International Journal of Robotics Research 23,* 2, 155–165.

Kanehiro, F., Lamiraux, F., Kanoun, O., Yoshida, E., and Laumond, J.-P. 2008. A Local Collision Avoidance Method for Non-strictly Convex Objects. In *2008 Robotics: Science and Systems Conference*. Zurich, Switzerland.

Kaneko, K., Kanehiro, F., Kajita, S., Hirukawa, H., Kawasaki, T., Hirata, M., Akachi, K., and Isozumi, T. 2004. Humanoid Robot HRP-2. In *Proc. IEEE International Conference on Robotics and Automation*. 1083–1090.

Khatib, O. 1987. A Unified Approach for Motion and Force Control of Robot Manipulators: The Operational Space Formulation. *IEEE Journal of Robotics and Automation 3,* 1, 43–53.

Kibangou, A. Y., Favier, G., and Hassani, M. M. 2005. Selection of Generalized Orthonormal Bases for Second-order Volterra Filters. *Signal Processing 85,* 12 (December), 2371–2385.

Kovar, L. and Gleicher, M. 2002. Footskate Cleanup for Motion Capture Editing. In *ACM SIGGRAPH Symposium on Computer Animation*. 97 – 104.

Kung, S. 1978. A New Identification and Model Reduction Algorithm via Singular Value Decomposition. In *Proc. 29th Conf. on Circuits, Systems and Computers*. 705–714.

Kwon, T. and Shin, S. Y. 2005. Motion Modeling for On-Line Locomotion Synthesis. In *Eurographics/ACM SIGGRAPH Symposium on computer Animation*.

Lacy, S. L. and Bernstein, D. S. 2003. Subspace identification with guaranteed stability using constrained optimization. *IEEE Trans. Automatic Control 48,* 7 (JULY), 1259–1263.

Lamiraux, F. and Laumond, J.-P. 1998. From Paths to Trajectories for Multi-body Mobile Robots. In *The Fifth International Symposium on Experimental Robotics V*. Springer-Verlag, London, UK, 301–309.

Laumond, J.-P., Arechavaleta-Servin, G., Truong, T., Hicheur, H., Pham, Q., and Berthoz, A. 2007. The Words of the Human Locomotion. In *3th International Symposium of Robotics Research (ISRR 2007)*. Hiroshima (Japan).

Lee, J. 1998. Modeling and Identification for Nonlinear Predictive Control: Requirements, Current Status and Future Research Needs. In *International Symposium on Nonlinear Model Predictive Control: Assessment and Future Directions*. Ascona, Switzerland, 91–107.

Lee, S., , Kim, J., Park, F., Kim, M., and Bobrow, J. 2005. Newton-type algorithms for dynamics-based robot movement optimization. *IEEE Trans. Robotics 21,* 4, 657– 667.

Liu, K. and Skelton, R. E. 1992. Identification of Linear Systems from their Pulse Responses. In *Proc. American Control Conference*. 1243–1247.

Ljung, L. 1995. *system Identification Toolbox User's Guide*. MathWorks.

Ljung, L. 1999. *System Identification: Theory for the User*, 2nd ed. Prentice Hall Informations and Systems Sciences.

Lo, J. and Metaxas, D. 1999. Recursive Dynamics and Optimal Control Techniques for Human Motion Planning. *Computer Animation*, 220–234.

Mathews, V. and Sicuranza, G. 2000. *Polynomial Signal Processing*. Wiley.

McKelvey, T. and Helmersson, A. 1997. System identification using an over-parametrized model class- improving the optimization algorithm. In *31th IEEE Conf. Decision Control*. San Diego, California USA.

McMahon, T. A. 1984. Mechanics of Locomotion. *International Journal of Robotics Research*.

Mohler, R. 1991. *Nonlinear System, II: Applications to Bilinear Control*. Prentice Hall.

Monin, A. and Salut, G. 1996. I.I.R. Volterra Filtering with Application to Bilinear Systems. *IEEE Trans. Signal Processing 44,* 9 (September), 2209–2221.

Moonen, M., Moor, B., Vandenberghe, L., and Vandewalle, J. 1989. On- and Off-line Identification of Linear State-space Models. *Int. J. Control 49,* 1 (July), 219–232.

Moor, B. D. 1997. *DaISy: Database for the Identification of Systems*. Department of Electrical Engineering, ESAT/SISTA, K.U.Leuven, Belgium, *http://homes.esat.kuleuven.be/˜smc/daisy/.*

Multon, F., France, L., Cani, M.-P., and Debunne, G. *1999. Computer Animation of Human Computer Animation of Human Walking: a Survey.* Journal of Visualization and Computer Animation.

Nakamura, Y. *1991.* Advanced Robotics: Redundancy and Optimization. *Addison-Wesley.*

Nakaoka, S., Nakazawa, A., Kanehiro, F., Kaneko, K., Morisawa, M., and Ikeuchi, K. *2005. Task Model of Lower Body Motion for a Biped Humanoid Robot to Imitate Human Dances.* IEEE/RSJ International Conference on Intelligent Robots and Systems, *3157–3162.*

Nakaoka, S., Nakazawa, A., Yokoi, K., Hirukawa, H., and Ikeuch, K. *2003. Generating Whole Body Motions for a Biped Humanoid Robot from Captured Human Dances. In* IEEE International Conference on Robotics and Automation. *3905–3910.*

Nakaoka, S., Nakazawa, A., Yokoi, K., and Ikeuchi, K. *2004. Leg Motion Primitives for a Dancing Humanoid Robot. In* IEEE International Conference on Robotics and Automation. *Vol. 1. 610–615.*

Nijmeijer, H. and van der Schaft, A. *1996.* Nonlinear Dynamical Control Systems. *Springer.*

Nørgaard, M. *2000.* Neural Network Based System Identification Toolbox. *Department of Automation, Department of Mathematical Modeling, Technical University of Denmark.*

Nørgaard, M., Ravn, O., Poulsen, N. K., and Hansen, L. K. *2000.* Neural Networks for Modelling and Control of Dynamic Systems. *Springer-Verlag, London, UK.*

Nowak, R. and Veen, B. V. *1994. Efficient Methods for Identification of Volterra Filter Models.* Signal Processing 38, *3, 417–428.*

Overschee, P. V. and Moor, B. D. *1994. N4SID: Subspace Algorithms for the Identification of Combined Deterministic - Stochastic Systems.* Automatica 30, *01 (Jan), 75–93.*

Park, F., Bobrow, J., and Ploen, S. *1995. A Lie Group Formulation of Robot Dynamics.* International Journal of Robotics Research 14, *6, 1130–1135.*

Pearson, R. *1995. Nonlinear Input/Output Modeling.* J. Process Control 5, *197–211.*

Pearson, R. *2000.* Discrete-Time Dynamic Models. *Oxford.*

Pettre, J. and Laumond, J. *2006. A Motion Capture Based Control-Space Approach for Walking Mannequins.* Journal of Computer Animation and Virtual World 17, *109—126.*

Pham, Q., Hicheur, H., Arechavaleta-Servin, G., Laumond, J., and Berthoz, A. *2007. The formation of trajectories during goal-oriented locomotion in human. ii. a maximum smoothness model.* European Journal of Neuroscience 26, *8 (Novembre), 2391–2403.*

Pollard, N., Hodgins, J., Riley, M., and Atkeson, C. *2002. Adapting Human Motion for the Control of a Humanoid Robot. In* IEEE International Conference on Robotics and Automation.

Renaud, M. and Fourquet, J. Y. *1992. Time-optimal motions of robot manipulators including dynamics.* The robotics review 2, MIT Press, *225–259.*

Ribarits, T., Deistler, M., and McKelvey, T. *2004. An Analysis of the Parametrization by Data Driven Local Coordinates for Multivariable Linear Systems.* Automatica 40, *789–803.*

Richard, M., Zexiang, L., and Shankar, S. *1994.* A Mathematical Introduction to Robotic Manipulation. *CRC Press.*

ROCKAFELLAR, R. *1973. Penalty Methods and Augmented Lagrangians in Nonlinear Programming. In* Proc. 5th IFIP Conference on Optimization techniques.

ROCKAFELLAR, R. *1974. Augmented Lagrange Multiplier Functions and Duality in Nonconvex Programming.* SIAM J.Control 12, *2 (May).*

ROSENFELD, D. B. AND WIEBE, B. *1997.* Geometry of Lie groups. *Springer.*

RUCHANURUCKS, M., NAKAOKA, S., KUDOH, S., AND IKEUCHI, K. *2006. Humanoid Robot Motion Generation with Sequential Physical Constraints. In* Proc. IEEE International Conference on Robotics and Automation. *2649–2654.*

RUGH, W. J. *1981.* Nonlinear System Theory: The Volterra/Wiener Approach. *The Johns Hopkins.*

SAFONOVA, A., POLLARD, N., AND HODGINS, J. *2003. Optimizing Human Motion for the Control of a Humanoid Robot. In* Proc. Applied Mathematics and Applications of Mathematics.

SCHETZEN. *1989.* The Volterra and Wiener theories of nonlinear systems. *New York: Wiley-Interscience.*

SCIAVICCO, L. AND SICILIANO, B. *2000a.* Modelling and Control of Robot Manipulators. *Springer-Verlag.*

SCIAVICCO, L. AND SICILIANO, B. *2000b.* Modelling and Control of Robot Manipulators. *Springer-Verlag, Chapter Trajectory Planning, 185–212.*

SENTIS, L. AND KHATIB, O. *2006. A Whole-Body Control Framework for Humanoids Operating in Human Environments. In* Proceedings of the IEEE Internactional Conference in Robotics and Automation.

SHOEMAKE, K. *1985. Animating Rotation with Quaternion Curves. In* ACM SIGGRAPH'85. *245 – 254.*

SJOBERG, J. ET AL. *1995. Nonlinear Black-box Modeling in System Identification: a Unified Overview.* Automatica 31, *12, 1691 – 1724.*

SJOBERG, J., HJALMARSSON, H., AND LJUNG, L. *1994. Neural Networks in System Identification. In* 10th IFAC symposium on system identification. *Copenhagen, Denmark.*

SKRINAR, A., BURDETT, R., AND SIMON, S. *1983. Comparison of Mechanical and Metabolic Energy Consumption During Normal Gait.* Journal of orthopedic research 1, *1, 63–72.*

SÖDERSTRÖM, T. AND STOICA, P. *1989.* System identification. *Prentice Hall, New York.*

SOHL, G. AND BOBROW, J. *2000. A Recursive Multibody Dynamics and Sensitivity Algorithm for Branched Kinematics Chains. Tech. rep., Department of Mechanical Engineering, University of California.*

STEINBACH, M. C. *1997. Optimal motion design using inverse dynamics. Tech. rep., Konrad-Zuse-Zentrum fur Informationstechnik Berlin, Deutschland.*

STURM, J. *1999.* Using SeDuMi 1.0x, a MATLAB Toolbox for Optimization Over Symmetric Cones. *Dept. Quantiative Economics, Maastricht Univ, Netherlands.*

SUN, H. C. AND METAXAS, D. N. *2001. Automating Gait Generation. In* ACM SIGGRAPH. *Los Angeles , USA.*

TAKEUCHI, Y. *1996.* Global Dynamical Properties of Lotka-Volterra Systems. *World Scientific.*

Thau, F. E. *1972. On the Feedback Control of Nonlinear Population Dynamics.* IEEE Trans. Syst. Man, Cybern. (Corresp) SMC-2*, 430–433.*

Thieme, H. R. *2003.* Mathematics in Population Biology*. Princeton University Press.*

VanOverschee, P.*,* DeMoor, B.*,* and Suykens, J. *1991. subspace algorithm for system identification and stochastic realization. In* Proc. MTNS*. Japan, 589–595.*

Verhaegen, M. *1991. A Novel Non-iterative MIMO State Space Model Identification Technique. In* Proc. 9th IFAC/IFORS Symp. on Identification and System Parameter Estimation*. 1453–1458.*

Verhaegen, M. *1994. Identification of the Deterministic Part of MIMO State Space Models Given in Innovations Form from Input-Output Data.* Automatica 30*, 1, 61–74.*

Verhaegen, M.*,* Varga, A.*,* and Grubel, G. *1994. Some Experience with the MOESP Class of Subspace Model Identification Methods in Identifying the BO105 Helicopter. Tech. rep., German Aerospace Research Establishment.*

Viberg, M. *1995. Subspace-based methods for the identification of linear time-invariant systems.* Automatica 31*, 12, 1835–1851.*

Volterra, V. *1930.* Theory of Functionals and of Integral and Integro-Differential Equations*. Dover Publications.*

Volterra, V. *1931.* Leçons sur la Théorie Mathématique de la lute pour la Vie*. Paris: Gauthier-Villars.*

Vukobratović, M. and Borovac, B. *2004. Zero-Moment Point—Thirty Five Years of its Life.* International Journal of Humanoid Robotics 1*, 1, 157–173.*

Wiley, D. and Hahn, J. *1997. Interpolation Synthesis for Articulated Figure Motion. In* Proceedings of IEEE Virtual Reality Annual International Symposium (VRAIS'97)*.*

Winter, D. *1990.* Biomechanics and Motor Control of Human Movement*, 2nd ed. John Wiley.*

Wit, C.*,* Siciliano, B.*,* and Bastin, G. *1996.* Theory of Robot Control*. Springer-Verlag.*

Yamaguchi, G. *1990.* Performing Whole-body simulations of Gait with 3D*. Springer-Verlag, Chapter Multiple Muscle Systems: Biomechanics and Movement Organization.*

Yoshida, E.*,* Belousov, I.*,* Esteves, C.*,* and Laumond, J.-P. *2005. Humanoid Motion Planning for Dynamic Tasks. In* Proc. IEEE-RAS International Conference on Humanoid Robots*.*

Yoshida, E.*,* Esteves, C.*,* Sakaguchi, T.*,* Laumond, J.-P.*,* and Yokoi, K. *2006. Smooth Collision Avoidance: Practical Issues in Dynamic Humanoid Motion. In* Proc. IEEE International Conference on Intelligent Robotics and Systems*.*

Yoshida, E.*,* Kanoun, O.*,* Esteves, C.*,* and Laumond, J.-P. *2006. Task-driven support polygon humanoids. In* Proc. IEEE-RAS International Conference on Humanoid Robots*.*

Auteur : Wael SULEIMAN

Titre : Identification et contrôle des systèmes non linéaires : application aux robots humanoïdes.

Résumé de la thèse:

Le travail de recherche dans ce mémoire aborde les problèmes de l'identification des systèmes non linéaires et également de l'application de la théorie d'optimisation. Dans une première partie, nous proposons des méthodes efficaces et nouvelles afin d'identifier les systèmes linéaires dans le cas d'expérimentations multiples, les séries de Volterra à horizon infini et les systèmes quadratiques en l'état. Dans une seconde partie, nous appliquons la théorie d'identification à la modélisation de la locomotion humaine.

Nous abordons ensuite l'optimisation des mouvements des robots humanoïdes, l'imitation des mouvements humains par un robot humanoïde et enfin le paramétrage temporel des chemins dans l'espace des configurations pour un robot humanoïde. Les résultats expérimentaux de nos méthodes sur la plate-forme HRP-2 ont révélé non seulement leur efficacité, mais aussi leurs bonnes performances qui dépassent largement celles des méthodes conventionnelles.

Mots clés : Identification des systèmes; Systèmes linéaires; Systèmes non linéaires; Séries de Volterra; Systèmes quadratiques en l'état; Locomotion humaine; Robot humanoïde; Imitation des mouvements; Paramétrage temporel; Plate-forme HRP-2; Expérimentations.

Author : Wael SULEIMAN

Title : Identification and Control of Nonlinear Systems : Application to Humanoid Robots.

Summary of the thesis:

The research work in this thesis deals with both nonlinear system identification and anthropomorphic motion optimization. Firstly, we consider the identification of three systems that are: linear systems in multiple experiments case, finite degree Volterra series with infinite horizon and quadratic in-the-state systems. Thus we propose novel and efficient identification methods. Secondly, we apply the system identification techniques to synthesize and to model human locomotion.

Besides, we address the optimization of humanoid robot motions, the imitation of human captured motions by a humanoid robot and finally the time parameterization of humanoid robot paths in the configuration space. The experimental validation of our proposed methods on the humanoid platform HRP-2 has pointed out that they not only outperform the conventional methods, but also provide a unified framework to control humanoid robots.

Keywords: System identification; Linear systems; Nonlinear systems; Volterra series; Quadratic in-the-state systems; Human locomotion; Humanoid robot; Motion imitation; Time parameterization; HRP-2 platform; Experiments.