Imitation Learning of Motion Coordination in Robots: a Dynamical System Approach

THÈSE N^{O} 5112 (2012)

PRÉSENTÉE LE 24 FÉVRIER 2012
À LA FACULTÉ SCIENCES ET TECHNIQUES DE L'INGÉNIEUR
LABORATOIRE D'ALGORITHMES ET SYSTÈMES D'APPRENTISSAGE
PROGRAMME DOCTORAL EN SYSTÈMES DE PRODUCTION ET ROBOTIQUE

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Elena GRIBOVSKAYA

acceptée sur proposition du jury:

Prof. H. Bleuler, président du jury
Prof. A. Billard, directrice de thèse
Prof. E. Burdet, rapporteur
Prof. M.-O. Hongler, rapporteur
Prof. S. Vijayakumar, rapporteur

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE
Suisse
2012

ABSTRACT

THE ease with which humans coordinate all their limbs is fascinating. Such a simplicity is the result of a complex process of motor coordination, i.e. the ability to resolve the biomechanical redundancy in an efficient and repeatable manner. Coordination enables a wide variety of everyday human activities from filling in a glass with water to pair figure skating. Therefore, it is highly desirable to endow robots with similar skills.

Despite the apparent diversity of coordinated motions, all of them share a crucial similarity: these motions are dictated by underlying constraints. The constraints shape the formation of the coordination patterns between the different degrees of freedom. Coordination constraints may take a spatio-temporal form; for instance, during bimanual object reaching or while catching a ball on the fly. They also may relate to the dynamics of the task; for instance, when one applies a specific force profile to carry a load.

In this thesis, we develop a framework for teaching coordination skills to robots. Coordination may take different forms, here, we focus on teaching a robot intra-limb and bimanual coordination, as well as coordination with a human during physical collaborative tasks. We use tools from well-established domains of Bayesian semi-parametric learning (Gaussian Mixture Models and Regression, Hidden Markov Models), nonlinear dynamics, and adaptive control. We take a biologically inspired approach to robot control. Specifically, we adopt an imitation learning perspective to skill transfer, that offers a seamless and intuitive way of capturing the constraints contained in natural human movements. As the robot is taught from motion data provided by a human teacher, we exploit evidence from human motor control of the temporal evolution of human motions that may be described by dynamical systems.

Throughout this thesis, we demonstrate that the dynamical system view on movement formation facilitates coordination control in robots. We explain how our framework for teaching coordination to a robot is built up, starting from intra-limb coordination and control, moving to bimanual coordination, and finally to physical interaction with a human.

The dissertation opens with the discussion of learning discrete task-level coordination patterns, such as spatio-temporal constraints emerging between the two arms in *bimanual manipulation tasks*. The encoding of bimanual constraints occurs at the task level and proceeds through a discretization of the task as sequences of bimanual constraints. Once the constraints are learned, the robot utilizes them to couple the two dynamical systems that generate kinematic trajectories for the hands. Explicit coupling of the dynamical systems ensures accurate reproduction of the learned constraints, and proves to be crucial for successful accomplishment of the task.

In the second part of this thesis, we consider *learning one-arm control policies*. We present an approach to extracting non-linear autonomous dynamical systems from kinematic data of arbitrary point-to-point motions. The proposed method aims to tackle the fundamental questions of learning robot coordination: (i) how to infer a motion representation that captures a multivariate coordination pattern between degrees of freedom and that generalizes this pattern to unseen contexts; (ii) whether the policy learned directly from demonstrations can provide robustness against spatial and temporal perturbations.

Finally, we demonstrate that the developed dynamical system approach to coordination may go beyond kinematic motion learning. We consider *physical interactions between a robot and a human* in situations where they jointly perform manipulation tasks; in particular, the problem of collaborative carrying and positioning of a load. We extend the approach proposed in the second part of this thesis to incorporate haptic information into the learning process. As a result, the robot adapts its kinematic motion plan according to human intentions expressed through the haptic signals. Even after the robot has learned the task model, the human still remains a complex contact environment. To ensure robustness of the robot behavior in the face of the variability inherent to human movements, we wrap the learned task model in an adaptive impedance controller with automatic gain tuning.

The techniques, developed in this thesis, have been applied to enable learning of unimanual and bimanual manipulation tasks on the robotics platforms HOAP-3, KATANA, and i-Cub, as well as to endow a pair of simulated robots with the ability to perform a manipulation task in the physical collaboration.

Keywords: Programming by Demonstration, Manipulation, Dynamical Systems, Coordination, physical Human-Robot Interaction

RÉSUMÉ

LA facilité avec laquelle les humains coordonnent les mouvements de tous les membres de leur corps est fascinante. Une telle aisance est le résultat d'un processus complexe de coordination motrice, à savoir, la capacité à résoudre la redondance biomécanique de manière efficace et reproductible. La coordination motrice permet l'exécution d'une large palette d'activités humaines: de remplir un simple verre d'eau au patinage artistique en couple. C'est pourquoi il est souhaitable de doter les robots d'une telle capacité.

Malgré la diversité de l'ensemble des mouvements coordonnés chez l'humain, ceux-ci partagent une similarité cruciale: ces mouvements sont régis par des contraintes sous-jacentes, qui, par essence, déterminent des motifs de coordination entre tous les degrés de liberté. Ces contraintes peuvent prendre une forme spatio-temporelle, comme par exemple lors d'une manipulation bimanuelle, ou lorsque l'on attrape une balle au vol. Elles peuvent aussi être apparentées à la dynamique d'une tâche, comme par exemple lorsque l'on applique un profil de force particulier pour déplacer une charge.

Dans cette thèse, nous développons une méthodologie permettant aux robots d'apprendre des modèles de coordination motrice. En particulier, nous nous concentrons sur l'apprentissage de la coordination de l'ensemble des joints d'un membre (p.ex., un bras ou une jambe), de la coordination bimanuelle, ainsi que de la coordination avec un humain lors de tâches collaboratives. Pour se faire, nous utilisons les outils des domaines bien établis que sont l'apprentissage semi-paramétrique Bayesien, la dynamique non-linéaire, et le contrôle adaptatif. De plus, nous suivons une approche bio-inspirée du contrôle robotique en adoptant une perspective qui considère l'apprentissage par imitation comme une méthode facilitant le transfert des capacités motrices. En effet, cette approche offre une méthode intuitive pour capturer les contraintes contenues dans les mouvements naturels, tels que ceux exécutés par les humains. Aussi, nous exploitons une évidence rapportée par des études en contrôle moteur, à savoir que l'évolution temporelle des mouvements humains peut être décrite par des systèmes dynamiques.

Tout au long de cette thèse, nous démontrons qu'une approche considérant la formation des mouvement par le biais de systèmes dynamiques facilite grandement le contrôle de la coordination chez les robots. Nous expliquons comment notre méthodologie d'apprentissage de la coordination motrice chez un robot se construit, à travers les problèmes liés à la coordination des degrés de libertés d'un membre, puis à la coordination bimanuelle, et enfin, à la coordination physique avec un humain.

Cette dissertation commence par la description d'une technique destinée à l'apprentissage de modèles de coordination discrets, tels que ceux qui sont déterminés par les contraintes spatio-temporelles émergeant entre deux bras lors de tâches bimanuelles. L'encodage de ces contraintes est effectué au niveau de la tâche, et s'accompli par une discrétisation de cette dernière en une séquence de contraintes bimanuelles. Une fois que ces contraintes sont apprises, le robot les utilise afin de coupler explicitement les deux systèmes dynamiques responsables de générer les trajectoires de chacune des deux mains. Le succès de notre méthode à apprendre et à reproduire les tâches démontrées prouve que ces contraintes sont un élément crucial à considérer pour une exécution réussie des tâches bimanuelles.

Dans la seconde partie de cette thèse, nous considérons l'apprentissage du contrôle coordonné d'un bras robotisé. Nous présentons une méthode servant à extraire un système dynamique autonome non-linéaire à partir de données cinématiques de mouvements point-à-point arbitraires. La méthode proposée vise à adresser les questions fondamentales de l'apprentissage de la coordination motrice en robotique que sont: (i) Comment inférer une représentation capable de capturer les motifs de coordination multivariés entre chacun des degrés de liberté d'un robot, et ensuite comment généraliser ces motifs à des contextes inconnus. (ii) A partir uniquement de démonstrations, comment un modèle peut-il garantir la robustesse du mouvement vis-à-vis de perturbations spatiales et temporelles.

Enfin, nous démontrons que notre approche dynamique de la coordination motrice peut aussi être appliquée à des problèmes qui ne sont pas purement cinématiques. Ici, nous considérons aussi l'interaction physique entre un robot et un humain dans les situations oÃź ceux-ci exécutent ensemble des tâches de manipulation. En particulier, nous considérons la tâche du transport collaboratif d'une charge. Nous étendons l'approche proposée dans la deuxième partie de cette thèse en incorporant l'information haptique dans le processus d'apprentissage. Grâce aux signaux haptiques, le robot devient capable d'adapter ses mouvements en fonction des intentions de l'humain. Cependant, même après l'apprentissage d'un modèle de la tâche par le robot, un humain reste un agent complexe à prédire. Afin de garantir une interaction robuste face à la variabilité intrinsèque des mouvements humains, nous incorporons le modèle de la tâche dans un système de contrôle en impédance contenant des gains capables de s'adapter automatiquement.

Les techniques développées dans cette thèse ont été appliquées afin de permettre à diverses plateformes robotiques (HOAP-3, KATANA et iCub) d'apprendre des tâches de manipulation unimanuelles et bimanuelles. De plus, la capacité d'exécuter une tâche en collaboration par le biais d'une interaction physique a été donnée à une paire de bras robotisés en simulation.

Mots clés: Apprentissage par Demonstration, Manipulation, Systèmes Dynamiques, Coordination Motrice, Interaction Physique entre Humain et Robot

To my mum and dad

Моим родителям посвящается

ACKNOWLEDGMENTS

I would like to express gratitude to my advisor, Aude Billard, for her expertise and enthusiasm for research. I was fortunate to receive her scientific guidance and professional advice. I greatly appreciated not only her vast knowledge of many areas of robotics and machine learning, but also her energy and involved work approach.

My special thanks also go to Abderrahmane Kheddar, who kindly welcomed me in his research group in AIST, Japan and spent a generous amount of time in scientific discussions. Collaboration with his group turned out to be an excellent learning experience. I am grateful to Prof. Kheddar for his steadfast enthusiasm and encouragement.

I gratefully acknowledge the experts, Etienne Burdet, Max-Olivier Hongler, and Sethu Vijayakumar for taking time to provide insightful comments. The president of the jury, Hannes Bleuler, helped me with the practical side of the thesis defense.

I am truly thankful to my coauthors Sylvain Calinon, Paul Evrard, and Mohammad Khansari. Our fruitful discussions were enriching for me, and innumerable laughs made the work process exciting.

I feel lucky to have worked with my colleagues in the LASA lab, Sylvain Calinon, Eric Sauser, Micha Hersch, Florent Guenter, Biljana Petreska, Basilio Noris, Brenna Argall, Mohammad Khansari, Jean-Baptiste Keller, Dan Grollman, and many others. I will be missing the energizing combination of scientific curiosity, humor, and creativity. Eric was always available to answer uncountable questions during my exploration of Linux, his passion for research and robots was truly motivating. I am indebted to Biljana, Basilio, and Jean-Baptiste for interesting (and distracting) discussions of both scientific and non-scientific character. I am thankful to Brenna for her friendship and great times we had in Lausanne. Brenna's feedback on my papers and advice on many academic issues are difficult to overestimate. I immensely value my friend Alexey for many reasons, for instance, for listening to my concerns and dispelling those that are particularly irrational.

I owe my family and friends in Russia infinite gratitude for letting me to pursue doctoral studies abroad and enduring my absence without a slightest sign of reproach. Their support helped me to pass through ups and downs of the doctoral studies. The unfailing optimism and strength of my mum and assurance of my dad have been the invaluable source of inspiration.

Finally, I thank Eric for being my best adventurous friend and for making everyday life joyful.

TABLE OF CONTENTS

List of Tables . . . . **xiii**

List of Figures . . . . **xv**

1 Introduction **1**
  1.1 Motivation 1
  1.2 Approach 3
    1.2.1 Imitation Learning 3
    1.2.2 A Dynamical System View on Motion Production 4
    1.2.3 Motion Control 4
  1.3 Major Contributions 5
  1.4 Contributions per chapter 7

2 Theoretical Background **9**
  2.1 The Analytical Robotics View on Manipulation Planning and Control 10
    2.1.1 Kinodynamic Planning 11
    2.1.2 From a Planning-Execution Scheme to a Unified Paradigm . . 13
    2.1.3 Feedback Planning 14
    2.1.4 Robot Control 16
      2.1.4.1 Impedance Control 16
      2.1.4.2 Impedance Control for Physical Human-Robot Interaction 18
    2.1.5 The Current Challenges 20
  2.2 Identification of Dynamical Systems and Stability Analysis 21
    2.2.1 Identification 22
      2.2.1.1 Parametric Methods 23
      2.2.1.2 Nonparametric Methods 26
    2.2.2 Stability Analysis 27
    2.2.3 The Current Challenges 31
  2.3 The Human Motor Control Insights into Motion Coordination 32
    2.3.1 Motion Production and Intra-limb Coordination 32
    2.3.2 A Dynamical System View on Bimanual Coordination 35
    2.3.3 Physical Interaction with Peers 36
    2.3.4 The Current Challenges 37
  2.4 Robot Learning 38
    2.4.1 Regression Techniques 39
      2.4.1.1 Global Regression Techniques: Gaussian Mixture Regression and Gaussian Process Regression 40
      2.4.1.2 Local Regression techniques: Locally Weighted Projection Regression 41
    2.4.2 Trajectory Modeling in Programming by Demonstration . . . 42

2.4.2.1 Learning Dynamical Systems for Motion Representations 43
2.4.2.2 Learning Constraints in Unimanual and Bimanual Tasks 45
2.4.3 Policy Derivation in Reinforcement Learning 47
2.4.3.1 Forward Reinforcement Learning 47
2.4.3.2 Inverse Reinforcement Learning 49
2.4.4 Learning for Physical Interaction 49
2.4.4.1 Learning Inverse Dynamics 50
2.4.4.2 Learning Variable Impedance 51
2.4.4.3 Learning for Physical Human-Robot Interaction . . 52
2.4.5 The Current Challenges 53
2.5 Conclusion 53

3 A Dynamical System Approach to Motion Representation and Coordination 57
3.1 Overview 57
3.2 Learning Online Motion Generation for Bimanual Tasks 59
3.2.1 Introduction 59
3.2.1.1 Set-up 61
3.2.2 Method Overview 61
3.2.2.1 The Learning System 61
3.2.2.2 The Motor System: Task Reproduction 66
3.2.3 Experimental Results 72
3.2.4 Discussion 76
3.2.5 Conclusions 78
3.3 Learning Nonlinear Dynamics of Motion in Robotic Manipulators . . 79
3.3.1 Introduction 79
3.3.2 Method 83
3.3.2.1 Problem Statement 83
3.3.2.2 Learning a Dynamical Motion Representation with Gaussian Mixture Regression 85
3.3.2.3 Stability Analysis 89
3.3.2.4 A Practical Approach to Ensure and Analyze Stability 92
3.3.3 Experimental Results 93
3.3.3.1 Learning Theoretical Dynamics 95
3.3.4 Application to Robot Control 108
3.3.4.1 Encoding Motion in the Operational Space 108
3.3.4.2 Set-up 110
3.3.4.3 Experiments with KATANA 110
3.3.4.4 Experiments with HOAP-3 111
3.3.4.5 Experiments with iCub 111
3.3.4.6 Results of Learning Dynamics from Motion Data . 111
3.3.4.7 Comparison with Dynamic Movement Primitives . 113
3.3.5 Conclusion 124
3.4 Learning Online Motion Generation for Intra-limb Coordination in Manipulation Tasks 125
3.4.1 Introduction 125
3.4.1.1 Learning Position and Orientation Control 127
3.4.1.2 Optimized Inverse Kinematics 128
3.4.2 Experimental Results and Discussion 129
3.4.2.1 Set-up 129
3.4.2.2 Results 130

3.4.2.3 Discussion and Conclusion 133
3.5 Discussion 133
3.5.1 Multi-Dimensional Systems, First Order Dynamics 133
3.5.2 Time Independency *vs* Time Dependency 134
3.5.3 Kinematic Controller 135
3.5.4 Choice of Statistical Framework 136
3.5.5 Real-time Adaptation to Perturbations 137
3.5.6 Single *vs* Several Attractors 137
3.5.7 Training Data 137
3.5.8 Kinesthetic Teaching 138
3.5.9 Practical Considerations 138
3.6 Conclusion 138

4 Learning Physical Human-Robot Coordination 141
4.1 Overview 141
4.2 Preliminary experiments with a HRP-2 robot 142
4.2.1 Hardware set-up 143
4.2.2 Algorithm 144
4.2.3 Reproduction Results 145
4.3 Model Introduction and a Simulation Set-up 146
4.4 Impedance Control 148
4.5 Two-stage Training Procedure 149
4.5.1 Passive Observation 150
4.5.2 Active Observation 151
4.6 Approach 151
4.6.1 Learning a Task Model 151
4.6.2 Learning Feedforward Control 153
4.6.3 Adaptive Impedance 154
4.7 Results 156
4.7.1 Learning a Task Model 156
4.7.2 Adaptation to Perturbations 156
4.7.3 Comparison with a Damping Controller 159
4.7.4 Impedance Adaptation 160
4.8 Discussion 161
4.9 Conclusion 163

5 Conclusion 165
5.1 Main Contributions and Limitations 165
5.2 Directions for Future Work 167

6 Appendices 169

References 175

Curriculum Vitae 192

LIST OF TABLES

3.1 Postures estimated by the *Learning system* 72
3.2 Glossary 84
3.3 Gaussian Mixture Regression 86
3.4 Model Training 94
3.5 Quantification of results 106
3.6 Quantification of results (Continuation) 107
3.7 On-line task reproduction 110
3.8 On-line Task Reproduction: Control over Position and Orientation . . 128

6.1 **Appendix I**: Derivation of constraint-consistent velocities for bimanual coordination 170
6.2 **Appendix I**: Derivation of the constraint-consistent velocities for bimanual coordination (continuation) 171
6.3 **Appendix II**: Comparison of the proposed method with Dynamical Movement Primitives, as defined in Hoffmann, Pastor, et al. (2009) . . 172
6.4 **Appendix III**: Bio-mimetic adaptive impedance (Ganesh, Albu-Schaffer, et al., 2010) 173

List of Figures

3.1 The robot is asked to sweeten tea. To do so, the robot moves its arms, so as to put a cube of sugar into a cup, and then brings the arms back on a table (*Tea task*). **Top**: An example of a sequence of three coordinated postures through which the robot transits when performing the *Tea task*. The three postures refer to three events: keeping arms in the initial position, putting the sugar in the cup, and keeping the arm at in the rest position. **Bottom:** The time series of the relative trajectory between the two hands (along the x axis). The superimposed arrows match stable postures (highlighted by ellipses) to the states illustrated on the top figure. 60

3.2 A hardware set-up. The humanoid robot HOAP-3 that we use in the experiments on bimanual coordination. The robot can freely manipulate objects in the space in front of its torso. The manipulated objects are marked with color patches and tracked by an external stereovision system. 61

3.3 A model overview. The arrows show an information flow across the system. A training set $\mathcal{D}$ is preprocessed by resampling and aligning the demonstrations. From a preprocessed dataset X, we then extract a set Π of stable postures. The set Π is further encoded in a HMM. After learning, a robot use a generalized set of spatio-temporal constraints P to reproduce a task. The robot's motion between and within stable postures is generated in real-time by a dynamical system controller. . 62

3.4 Dynamical Time Warping (DTW). Results of applying DTW to five trajectories of the shoulder's joint recorded during the demonstration of the Cube task. Note that, before DTW, local minima and maxima of the recorded signals are strongly misaligned. DTW helps to harmonize data in the time domain. 62

3.5 *Within-cluster* ΔC_1 and *between-cluster* $\delta(C_1, C_2)$ distances used to compute the Dunn index in Eq.3.5 66

3.6 An overview of the robot's *Motor System.* Each arm is controlled by a couple of dynamical controllers given by Eq.3.7-3.8. Within each arm, the controllers are coupled through robot's body constraints; Eq. 3.11. The coordination between the arms is ensured by spatial and temporal constraints; Eq.3.14, 3.17, 3.19. 67

3.7 The HMM encoding of *Tea task.* The demonstrated trajectories of the relative position $\boldsymbol{d}$; red dots are the starting points of stable postures extracted according to (their covariance matrices after training the HMM parameters are represented in bold ellipses). 71

3.8 Experimental set-ups. To validate our method, we investigate its performance in three manipulation tasks. The *tea task*: put a piece of sugar into a cup. The *cube task*: grasp a cube, lift it, and put it on top of the pedestal. The *tray task*: grasp a tray with both arms, lift it, and move the tray forward. 73

3.9 The *cube task.* (a) A robot tries to grasp a cube, but the cube is suddenly shifted from the position A to the position B, the direction of the perturbation is specified by a grey arrow. The robot adapts the trajectories of the both arms, so as to grasp the cube from the position B (the moment of grasping is highlighted with red cross). (b) Continuation of the task: while the robot is carrying the cube, the position of a pedestal is changed from C to D, and the robot brings the cube to the new location. Note that from grasping the cube (red cross) until releasing it (red circle), the robot preserves the learned relative position between the hands. (c) The positions A and B of the cube and C and D of the pedestal are superimposed with the workspace accessible to the robot in this task (light grey). Note that perturbations force the robot to operate almost on the boundary of its accessible workspace, however, the robot's *Motor System* successfully resolves the learned constraints and generates motion trajectories. (d) The photos of the robot at the different stages of the task completion. The yellow lines on photos are the trajectories of the robot. 74

3.10 The *tea task*. (a) A robot is bringing the two arms together so as to put a piece of sugar into a cup. During the motion, its right arm is pushed (as a result, the cup moved from the position A to B) so that, to accomplish the task, the robot has to quickly adapt both arms. The direction of the perturbation is specified by a grey arrow. (b) When the robot's is in the stable posture (the relative position between the arms is preserved) and is about to put a sugar in a cup, the right arm is pushes again (the cup moves from C to D). This time, however, the *Motor system* preserves the posture and adapts the left arm accordingly. (c) The positions A, B, C, and D of the cup are superimposed with the workspace accessible to the robot in this task (light grey). Note that perturbations force the robot to operate almost on the boundary of its accessible workspace, however, the robot's *Motor System* successfully resolves the learned constraints and generates motion trajectories. (d) The photos of the robot at the different stages of the task completion. The yellow lines on photos are the trajectories of the robot. 75

3.11 The *tray task.* (a) After grasping a tray (red cross), a robot is suddenly pushed (from the position A to B). The direction of the perturbation is specified by a grey arrow. This perturbation forces the robot to manipulate on the boundaries of its workspace. The robot successfully adapts its motion and ensures that the coordination constraints are satisfied. The robot carries the tray without dropping it and release it in the position C (red circle). (c) The positions A, B, and C of the tray are superimposed with the workspace accessible to the robot in this task (light grey). Note that perturbations force the robot to operate almost on the boundary of its accessible workspace, however, the robot's *Motor System* successfully resolves the learned constraints and generates motion trajectories. (d) The photos of the robot at the different stages of the task completion. The yellow lines on photos are the trajectories of the robot. 76

3.12 Task reproduction under temporal coordination constraints. The synchronization in the *Cube task*: at $t_1 = 150$, after the onset of the motion, the position of the cube is changed (the time of the perturbation is highlighted by a dashed red line). (a) Simultaneously, the robot's *Motor system* adapts the trajectories of the two arms to reach the cube in the new location. The velocity profiles under perturbations are smooth and close to bell-shaped, both in Cartesian (b) and joint space (c). The time instances $t_2..t_6$ refer to the boundaries of stable postures. (d) A schema of the HOAP-3 arm, the joints (SFE, SAA, SHR, EB) correspond to the velocities presented in figure (c). 77

3.13 The legend used in figures in Section 3.2. 82

3.14 The geometric interpretation of inference in GMR (see also Table 3.3). GMR approximates a dynamical system through a non-linear weighted sum of local linear models. Each regression matrix $A_k = \Sigma_k^{\mathcal{OI}}(\Sigma_k^{\mathcal{I}})^{-1}$ defines a local linear dynamics. Here, we illustrate inference with a single Gaussian and a pair of input ξ^I and output ξ^O. In the planar case, the regression defines a line with a slope given by the matrix A_k ($\xi^O = A_k\xi^I$). For each input ξ^I, GMR defines a conditional distribution $p(\xi^O|\xi^I)$, with the mean ξ^O and the conditional covariance $\hat{\Sigma}$. The conditional covariance $\hat{\Sigma}$ defines an error envelope around the regression output ξ^O (the expected error on the output predicted by the regression). 89

3.15 I. Illustration of a GMM/GMR encoding of an arbitrary dynamics. *Top left*: Two-dimensional projection of the data with superimposed the Gaussian Mixture envelope. Top right: All trajectories regenerated using Gaussian mixture regression when starting from 20 different locations in space converge correctly to the the origin, the attractor of the system. *Bottom left and right*: in blue (light grey in a black-and-white version), the region of applicability C that embeds all demonstrated trajectories. To empirically determine if C is a region of attraction, C is sampled equally and one measures if all trajectories originating from each of sampled point converges correctly to the target. 90

3.16 Accurate positioning of the Gaussian distribution at the attractor affects stability of the learned $\hat{f}(\boldsymbol{\xi})$. *Top*: the last Gaussian is positioned at the origin through the addition of synthetic datapoints. This modification guarantees asymptotic stability in the neighborhood of the attractor: the trajectories converge to the origin (the very right graph). *Bottom*: though the observed demonstrations converge to the origin (the very left graph), the EM training does not position the last Gaussian at the attractor automatically. Therefore, a motion generated by the learned dynamical representation $\hat{f}(\boldsymbol{\xi})$ converges to the spurious attractor (the very right graph). 92

3.17 Improvement in the stability of approximation with the increase in the number of Gaussian components 93

3.18 *System 1*. The proposed method encodes this system with 7 Gaussians; the learned system exhibits good precision in the area covered by demonstrations, outside this area the precision is also admissible except for a region in the direct proximity to y-axis, where actual trajectories represent an excess curvature as approaching to the equilibrium, e.g., a trajectory starting at the bound $x_2 = 2$. In this region, a flat part of trajectories is reproduced well, though the steep parts that were not demonstrated are attracted towards the region covered by the training set. 96

3.19 *System 2*. As the behavior of the system in the considered area is relatively simple, 2 Gaussians are sufficient to achieve the good performance, even in areas unseen during demonstration. Interestingly, the learned dynamics is extrapolated very well beyond the area covered by the training set. 97

3.20 *System 3*. Despite strong non-linearities in the observed trajectories, the dynamics is successfully approximated with 6 Gaussians. Note, even unseen, circular shape trajectories (starting around $x_2 \approx 0$) are reproduced correctly in both position and velocities spaces. 98

3.21 *System 4*. The system is strongly non-linear, 13 Gaussians are necessary to achieve a good precision in the considered region. Complex dynamics and increased number of Gaussians lead to less strong generalization abilities of the method. Indeed, trajectories started beyond the region covered by the training set tend to depart from the real trajectories generated by the dynamics, it is particularly noticeable in the velocity space, see section III-(g). However, even in this non-trivial case the system generates admissibly good results (the reproduced trajectories follow an observed motion pattern) from few demonstrations. 99

3.22 *System 5*. A strongly non-linear three-dimensional dynamical system. In this case, a slight increase in a number of demonstrations allows for the accurate approximation and generalization. 101

3.23 Learning motion with two attractors. 3-dimensional trajectories are generated by *System 5* that displays a periodic behavior. Trajectories were demonstrated in the neighborhood of two asymptotically stable attractors. During the reproduction, the system managed to accurately reproduce dynamics around both attractors. 102

3.24 A numerically estimated region of applicability of the System 4 (the red/black(in a black-and-white version) frame). An actual and spurious attractors are highlighted with circles. Note, the numerical method estimated a lower bound that goes along a trajectory with a good precision. Other bounds were left unchanged, i.e., in the other directions the considered region does not cross boundaries of the region of applicability. 103

3.25 Extrapolation properties of the GMMs encoding (better see in color). A color map reflects changes in values of the the likelihood (3.38) of datapoints, the dark-red (dark grey in the center) area represents an area of the most reliable inference regarding the velocity. For reconstructed trajectories starting outside this area, the deviation from the actual dynamics may be considerable. Interestingly, in the region of attraction of the origin, trajectories are strongly attracted towards a region covered by the training set. It is a useful property for practical applications as it enables the prediction of the system behavior outside the region covered during training, hence reducing the amount of training data required. 104

3.26 Robustness to perturbations. The target is shifted several times (to positions 2, 3, 4) after the onset of motion. 105

3.27 (a) Training trajectories of *System 4* are corrupted with a signal-dependent noise of variance $\sigma^2 = 20$. (b) The trajectories generated with the learned dynamics are superimposed with the training data. (c) The trajectories generated with the learned dynamics are superimposed with the trajectories of the actual dynamical system. Note, that the proposed method manages to accurately disentangle the motion dynamics from noise. 106

3.28 (a) If a trajectory in the operation space passes through non-reachable joint positions IK may return velocity in the operation space that sends a robot too far from original trajectory, so linear assumptions of approximation of kinematics does not satisfy and overall trajectory tracking will fail. (b) In the case of motion encoding with a dynamical system, after perturbation the robot will not try to return to the previous trajectory violating the linear approximation of kinematics, instead the dynamical system will generate other trajectory from the point where the robot occurs. 109

3.29 We encode tasks in a referential located at the target and moving with it$\{x^*y^*z^*\}$; this referential is expressed in the fixed global referential $\{xyz\}$(usually we choose one attached to static parts of a robot). Actually, the motion of the robot end-effector is expressed as moving a referential associated with the end-effector $\{x'y'z'\}$. 109

3.30 Set-up of the experiments. Left: KATANA puts a wooden brick into the container, to achieve the task the robot should lift the brick and move it following an elevating trajectory. Right: HOAP-3 grasps a box, to accomplish this task HOAP should approach the box with a specific orientation and than lower its arm, as the clench is small, see small figure in the corner. 112

3.31 The results of reproduction of dynamically generated trajectories on the robots. To check the generalization abilities of the learned dynamics the trajectories were reproduced from different initial positions. 112

3.32 KATANA experiment 1: Results of encoding and reproduction of the experiment where KATANA had to put a brick into a container. 114

3.33 KATANA experiment 2: *I. Real-time adaptation to perturbations*. The target is shifted several times from the position 1 to the position 4. First row: trajectory of the robot's end-effector; second row: velocity profile. *II. Generalization to the unseen context.* (a) The approximate workspace of KATANA is highlighted by the blue box (light grey in a black-and-white version), the reproduction is systematically tested starting the robot from positions on the starting plane (yellow/darker grey). (b) The robot is required to reproduce the motion from points monotonically covering the yellow/light-grey sub-part. For comparison, the part of space where the demonstrations are provided is in pink/dark-grey. Note, the demonstrations are sparse, but the system manages to generalize to other parts of the workspace. 115

3.34 HOAP-3 experiment: Results of encoding and reproduction of the experiment where HOAP-3 had to grasp a box. 116

3.35 (a) The humanoid robot iCub used in the experiments. We encode tasks in the frame of reference located at and moving with the target; this frame of reference is expressed in the fixed global frame of reference (we choose the one attached to static parts of a robot). (b) A human teacher demonstrates a ping-pong motion to the robot. (c) Three XSens motion capture sensors are attached to the hand, forearm, and upper arm of the demonstrator and allow the reconstruction of the motion of each joint. 117

3.36 Task generalization: our method vs. DMP (Hoffmann, Pastor, et al., 2009) in the ping-pong experiments. Here, due to the noise in the training data, our system tries to extract a generic pattern and, therefore, the reproduced trajectories do not follow the demonstrations exactly. However, in comparison with DMP generated trajectories, the trajectories produced by our method exhibit more similarity with demonstrations (in terms of the trajectory shape). The difference between the two method is particularly obvious when the robot starts its motion from locations unobserved during demonstration; DMP tends to generate unexpected swinging motions. 119

3.37 Robustness to spatial perturbations: our method vs. DMP (Hoffmann, Pastor, et al., 2009), learning a theoretical noise-free dynamics. Due to scaling that DMP performs for adapting a learned acceleration profile to the conditions after perturbation, a generated motion may have an unexpectedly excessive curvature (a) or can overshoot the target (b). . 120

3.38 Robustness to spatio-temporal perturbations: our method vs. DMP (Hoffmann, Pastor, et al., 2009) in the ping-pong experiment. The ball is moved up (from position (1) to position (2)) after the onset of the motion, DMP trajectories produce strong swings and tend to overshoot the target. 121

3.39 Robustness to temporal perturbations: our method vs. DMP (Hoffmann, Pastor, et al., 2009). The target has been shifted so that the duration of motion has been increased (a) or decreased (b). (a) The target is moved from position (1) to position (2), farther from the robot's end-effector, DMP takes the shortest path to the initial position of the target and stretches it to reach the shifted target position, this results in an almost straight line trajectory which potentially may violate external constraints implicitly encoded in the demonstrations. (b) The target shifted so as to decrease the duration of motion, in this case DMP scale the trajectory and produce the jerky motion right after the perturbation; in this case DMP require more time to reach the target than our system. 122

3.40 (a) Illustration of the high accuracy of DMP at reproducing trajectories that start in a small neighborhood of a demonstrated trajectory. When starting the motion at the same location as that demonstrated, reproduction fits the original signal very accurately. (b) When reproducing the noisy training data from the ping-pong task. DMP accurately fits each demonstration separately. This leads to over-fitting as each trajectory contain noise inherent to the physical world. 124

3.41 Geometrical illustration of stability and multi-dimensional correlation in the state-space. I. ***Stability problem***: stability of a dynamical system is defined by a maximum value of its *Lyapunov exponent* λ (in the linear case, it coincides with eigenvalues of a control matrix). (a) In systems with *negative* Lyapunov exponents volume between trajectories contracts; (b) In systems with *positive* Lyapunov exponents two arbitrary near trajectories diverge from each other exponentially fast. In the linear case, one may easily find Lyapunov exponents and estimate the global behavior of the overall system. In the non-linear case, the system may have different Lyapunov exponents in different parts of the state-space, moreover, non-linearities make analytical investigation of properties particularly tedious. IV. ***Multi-dimensional dynamics*** Analyzing dynamics of vector-valued timeseries requires their encoding in multi-dimensional state-spaces. Generally, one cannot unambiguously decouple dynamics of each dimension. Consider a simple 2D motion in Fig. II-(a), the phase-space of this motion in $\{\dot{x}_1, x_1\}$ is in Fig. II-(b): for each value x_1 there exist two different values of velocity, therefore, it is not possible to unambiguously encode dynamics of motion as two decoupled system $\dot{x}_1 = f_1(x_1)$, $\dot{x}_2 = f_2(x_2)$. However, if one look at the dependency $\dot{x}_1 = f(x_1, x_2)$ depicted at Fig. II-(c) this ambiguity can be easily eliminated. This problem is know in the literature on Dynamical Systems as a problem of searching for a *minimum embedding dimension*. In this particular example, the minimum embedding dimension is 4 (x_1, $\dot{x}_1$, x_2, $\dot{x}_2$). Alternatively, one may argue that in this case we may avoid an ambiguity and separate dimensions encoding $\ddot{x}_1 = f_1(x_1, \dot{x}_1)$, though it is possible in this particular case, it will lead to the necessity to analyze 5 state variables (x_1, $\dot{x}_1$, $\ddot{x}_1$, x_2, $\dot{x}_2$). Furthermore, to preserve a spatial correlation pattern between x_1 and x_2 the decoupled systems should be synchronized by an external mechanism. 126

3.42 Results of encoding the orientation phase of demonstrations in an experiment with HOAP-3. Note, the existence of non-linear correlation between an axis and an angle of rotation. 131

3.43 Experiments with the humanoid robot HOAP-3. Referentials display the change in the orientation of the robot's end-effector along the motion. Starting positions of the hand are highlighted by yellow circles. I. Generalization abilities of the method: the robot successfully grasped a box placed in different positions in the workspace. These configurations of the box have not been observed by the robot during demonstration. II. Real-time adaptation to perturbations: while the robot was moving towards the box its position was perturbed (a), both position and orientation were perturbed (b). Control of position and orientation through dynamical systems enables the smooth adaptation to both types of perturbations. 132

3.44 An example of a trajectory generated with DMP and starting far from an original demonstration. Note, that although the motion is globally asymptotically stable, the resulting trajectory makes little sense. . . . 135

4.1 We consider a task where a human and a robot lift a rigid beam in collaboration (Evrard et al., 2009). Training is accomplished through teleoperation of the HRP-2 robot through a haptic device. 143

4.2 [Taken from (Calinon, Evrard, et al., 2009)] The two sets of demonstrations are provided. In the first set, the teacher is blind-folded and the partner initiates and terminates the motion. In the second set, the roles are exchanged: the teacher leads the motion's onset and the offset. Note that the two set produce different force-velocity patterns. (a) Collected data are plotted in fine lines. An average demonstration in each set is plotted as a wide line with arrows. Right: when the human partner initiates the motion, the robot perceives positive interaction force, when the robot starts moving the force gradually decreases to zero by the end of the movement. Left: in contrast, when the robot (guided by the teacher) initiates the motion, the robot perceives negative force, when the human partner starts moving the force gradually increases to zero by the end of the movement. (b) The GMM encoding of the training sets. 144

4.3 [Taken from (Evrard et al., 2009)] Dashed grey and solid blue lines show force-velocity patterns in failed reproduction trials where a human partner tries to stop the robot abruptly. The green and pink ovals represent Gaussian components of the learned task model and correspond to the these in Fig. 4.2-(b). 145

4.4 Two planar robots lift a beam in collaboration. For successful task completion the two robots have to coordinate and adapt their motions so as to avoid tilting the beam. The robot-leader substitutes the human. The desired kinematic plan $\boldsymbol{x}_{d,L}, \dot{\boldsymbol{x}}_{d,L}, \ddot{\boldsymbol{x}}_{d,L}$ of the robot-leader is predefined. The robot-follower anticipates the motion intentions of the robot-leader and adapts accordingly. During demonstration, the robot-follower learns to generate a desired kinematic command $\boldsymbol{x}_d, \dot{\boldsymbol{x}}_d, \ddot{\boldsymbol{x}}_d$ in response to the perceived force $\boldsymbol{f}$. The two robots are controlled by impedance control laws with desired stiffness, damping, and inertia. During task execution, the robot-follower adapts its desired stiffness $\tilde{\boldsymbol{K}}_d$ and inertia $\tilde{\boldsymbol{\Lambda}}_d$, so as to ensure accurate reproduction of a learned task model. 147

4.5 TWO-STAGE TRAINING PROCEDURE. To simulate real-world training, where the robot is teleoperated by a human, we adopt a two stage training procedure. Figs. (a), (e) present the robots' configurations during training. Desyncronization between the partners is greatly reduced during active observation, as expressed by the reduced tilting of the beam. *PASSIVE OBSERVATION*: The stiffness of the robot-follower is set to be low ($5N/m$) and the stiffness of the robot-leader is high ($50N/m$). This allows the robot-leader to impose its kinematic plan; see Fig. (b). The actual velocity of the robot-follower is higher than its reference signal and coincides with the actual and reference velocities of the robot-leader. Such a forced adaptation is achieved at the cost of considerable energy injection; see Figs. (c)-(d). The robot-follower perceives high positive external forces that are due to the effort of the robot-leader. After observing the task "passively", the robot-follower stores the kinematic information and discards the force signals. *ACTIVE OBSERVATION*: The stiffness of both partners is medium ($15N/m$). The robot-leader repeats the same reference kinematical profile as at the previous stage, while the robot-follower utilizes the kinematic profile acquired during passive observation. Improved synchronization decreases the magnitude of the forces perceived by both partners; see Figs. (c)-(d), solid line. The final training set is composed of the velocity signal recorded during passive observation, and the external forces/applied torques recorded during active observation. 150

4.6 After acquiring a *set of demonstrations* $\mathcal{D}$, the robot *learns* the task model $\dot{\boldsymbol{\xi}} = \hat{\mathbf{h}}(\boldsymbol{\xi})$ and a forward control signal $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{\xi})$ that maps the desired state $\boldsymbol{\xi}$ of the task model to actual motor commands.The dynamical system representation of the task model allows the robot to generate *reference signals* on-line adapting to the force applied by a human. The robot is controlled through an *impedance control law* so as to compensate for non-modeled aspects of the external dynamics. The *desired stiffness* $\tilde{\mathbf{K}}_d$ and *inertia* $\tilde{\boldsymbol{\Lambda}}_d$ are *adapted* during task execution. 152

4.7 The task model is represented by a dynamical system $\dot{\boldsymbol{\xi}} = \hat{\mathbf{h}}(\boldsymbol{\xi})$, $\boldsymbol{\xi} = [\dot{\boldsymbol{x}}_d;\boldsymbol{f}_d]$ and estimated from the training data. At each time step, the velocity $\dot{\boldsymbol{x}}_d$ and force $\boldsymbol{f}_d$ are inferred from these observed at the previous step. Their dynamical relationships follow vector fields displayed in blue. Dark gray lines show the demonstrations. One can observe an accurate fit between the inferred and demonstrated dynamics. Statistical inference extends prediction of the force-velocity pattern to ranges of these variables not observed during training. This offers a greater robustness during adaptation to a new human partner. 152

4.8 TASK LEARNING AND REPRODUCTION. In this experiment, the robot-follower learns the lifting task by observing demonstrations performed with different velocity profiles imposed by the leader. During reproduction, the robot-leader varies its kinematics plan from one attempt to another; it does so by changing motion duration and maximum velocity. The robot-follower, governed by the learned task and control model, adapts its motion and successfully accomplishes the task. (a) The state-space view of the data used for training a task model (gray line) and the reproduction attempts (green line). (b) The forward control signal generated by the follower during demonstration (gray line) and reproduction (green line). The two datasets correspond to the two joints of the planar robot-follower. (c)-(d) The time-series of the Cartesian velocity and trajectory of the robot-follower during reproduction. The follower (green line) adjusts its kinematics and synchronize with the leader (dashed blue line). 157
4.9 ADAPTATION TO PERTURBATIONS. Three cases are analyzed: the robot-leader changes the motion plan on-line and move the beam (1) higher than initially planned, (2) lower, but higher than the actual position of the beam at the moment the change decision is taken, and (3) stops at a position lower than the actual position at the decision-taking moment. In case (1), the robot-follower manages to re-accelerate (see the two peaks in the velocity profile). In cases (2) the robot-follower pro-actively decelerates. In case (3), the robot also manages to smoothly drop velocity below zero and lower the beam. 158
4.10 COMPARISON WITH A DAMPING CONTROLLER. We compare our system versus the damping controller. (a), (c) The planar robots performing lifting; the follower is controlled by our controller (a), and the damping controller (c). One may notice the improvements in coordination between the partners when the robot-follower adapts its kinematic profile (a): the beam is kept horizontal all along the motion. It is persistently tilted when the follower is controlled with the damping (c). (b) θ characterizes deviation of the beam from the horizontal orientation. (d) The leader has to apply considerably higher forces to make the system move. 159
4.11 AN IMPACT OF DESIRED INERTIA. We demonstrate three cases, where the follower executes the task with: 1) same impedance parameters as used during demonstration; 2) the desired inertia $\tilde{\mathbf{\Lambda}}_d$ is set higher than during demonstration; and 3) the desired inertia $\tilde{\mathbf{\Lambda}}_d$ is set lower than during demonstration. Note that changes in the inertia affects the perceived force and prediction of the desired velocity. 160
4.12 ADAPTATION OF UNKNOWN IMPEDANCE. In general, the impedance parameters that would be optimal for the task are unknown. Therefore, the follower should tune its stiffness and inertia during task execution. Arbitrary impedance parameters may cause undesirable effects, e.g. overestimated reference kinematics and contact instabilities (blue dashed line). Our algorithm provides an adaptation law, to tune the parameters and to ensure accurate reproduction (green lines in (a)-(b) show the results with impedance adaptation). 161

Chapter 1

INTRODUCTION

1.1 MOTIVATION

ALREADY in the eighties, we were amused by the fussy droid C-3PO from "Star Wars" helping its master, Anakin Skywalker, with various household chores. Beyond science fiction movies, existing robotic agents are still lacking the agility and motion skills that we as humans take for granted. One reason for this is that the principles underlying the production of coordinated body movements in humans are largely unknown. In this thesis, we take an engineering view of the problem, we do not aim to uncover the biological grounds of coordination in human motion. We investigate how, by merely observing the means and the effect of coordination in everyday human motions, we can extract control strategies that enable coordinated movements in robots.

Coordinated unimanual and *bimanual* movements involving object manipulation, either *autonomously* or *in collaboration* with peers, represent a key part of our motion repertoire. Teaching such movements to robots constitutes the research subject of this thesis[1]. The goal of a manipulation task is defined through its desired effect on the environment. In general, a given goal can be achieved by an infinite number of different movements, rather than by a single pre-specified trajectory. Stated in this way, the problem of motion learning appears to be ill-posed - it is not obvious how a subject decides on one particular motion signature (a distinctive movement pattern of an individual). However, it is intuitively clear that the human brain somehow resolves this redundancy in an efficient and repetitive manner. N. Bernstein, in his seminal work of 1967 (Bernstein, 1967), termed the remarkable ability to resolve motion redundancy *coordination*.

To make the above idea clearer, imagine that in one hand you hold a cup and in the other a piece of sugar. You want to sweeten your tea, therefore, the goal is to drop the sugar into the cup. At the trajectory level, this means that the two arms should move from a rest position to a target configuration where the sugar may be released right into the cup. Achieving this target configuration constitutes a "hard" task constraint that, generally speaking, can be satisfied by following trajectories quite dissimilar from each other. (You may even throw the sugar up in the air and try to catch it with the cup.) However, we humans exhibit a systematic motor behavior; that is, we follow regu-

---

[1] Throughout this thesis we mainly consider task space motions where generated Cartesian trajectories are converted into joint configurations through inverse kinematics.

lar motion profiles that may have both a practical (minimizing energy consumption or avoiding collision with the cup) and a communicative meaning (peers understand what we intend to do). Here, we adopt the view that such a *systematic motor behavior* distinguishes the *coordinated* motions from the *uncoordinated*. Therefore, a coordination pattern can be defined as a correlation that emerges between variables describing a motion and that is consistent across trials. Such patterns are usually task dependent and, therefore, it is difficult to conceive a unified analytical model for motion generation.

The existing analytical planning algorithms only resolve the "hard" task constraint, i.e. how to generate a collision-free path from a rest to target configuration, and ignore the second part of the problem where the motion itself may constitute the part of the task (Brock & Khatib, 2002). Furthermore, motion planning algorithms have increasingly large computation requirements and, therefore, are less reactive to dynamically changing environments. Human motor control concentrates on rather simplistic, from the robotics's point of view, pointing movements (Bullock & Grossberg, 1988; Todorov & Jordan, 2002) or basic types of rhythmic synchronization between the limbs (Haken et al., 1985), and does not provide a generic approach to tackle the problem of coordination in manipulation tasks. For these reasons, we exploit learning as a means for transferring human coordination skills to robots. Learning coordinated tasks requires solving two problems simultaneously: (1) learning to satisfy "hard" constraints that *any* successful task movement must satisfy (e.g., reaching the target configuration) and (2) learning to produce task movements that are *natural looking* and *easy to accomplish* (e.g., following a particular path). These two problems define the main theme of this thesis. The objective that we pursue is to develop learning algorithms that allow a robot to resolve hard task constraints and generate continuous coordinated motions.

1.2 APPROACH

As outlined in Section 1.1, our goal is to devise robot motion strategies from data provided by a human. We take the Programming by Demonstration (PbD) perspective to robot skill transfer. Originally, PbD in robotics has emerged to avoid tedious manual development of robot software. The two major factors contributed to the success of the PbD paradigm. At the interaction level, PbD is appealing as a human-friendly means to endow robots with various skills. At the computational level, PbD considerably speeds up the search for a task solution in the robot workspace as information contained in the demonstrations constrains the search area. The particular PbD perspective followed in this work argues that the robot's ability to *encode movements*, whether at the continuous trajectory level or at the discrete symbolic level, is the basis of skill transfer.

PbD is grounded on the concept of imitation learning that roboticists borrowed from the developmental psychology. Furthermore, we learn encodings for motion from data provided by a human demonstrator, i.e. we implicitly assume the existence of regularities in the motion data. Therefore, in our research, we are bound to adopt several biological hypotheses.

1.2.1 IMITATION LEARNING

From infancy and during the whole life, humans exhibit the ability to imitate their peers. The deceptively naive concept of imitation plays a fundamental role in the acquisition of a motion repertoire. The human Central Nervous System (CNS) will follow a sub-optimal solution[2] if it has been reinforced by positive results (Ganesh, Haruno, et al., 2010). From trial to trial, the sub-optimal solution is locally adapted to minimize error and effort; however, it still remains far from optimal. Consider, for instance, a teenager exercising basketball shooting. In the absence of proper guidance, his chances to acquire a stable shooting behavior are slim. A good shot is a unique combination of a balanced stance, a loose but accurate grip, and a powerful delivery motion; each of these components has a number of nuances that a novice can learn only from coach's guidance. This is explained by the fact that a teaching signal makes the CNS realize a globally optimal behavior. Research suggests that imitation of experienced individuals helps humans converge to more optimal motion strategies (Rizzolatti & Craighero, 2004).

In robotics, the "optimal" solution obtained through imitation learning may not necessarily be optimal in a strict mathematical sense, in contrast to solutions that would be produced by various planning algorithms (e.g., optimal in terms of the path length). Instead, optimality may be estimated as the extent to which a motion generated by the robot resembles one that the human would produce herself and, therefore, that she might expect the robot to execute in similar conditions. In this work, we adopt both the imitation learning strategy for transferring skills to robots and the above "soft" notion of movement optimality.

---

[2] For instance, the sub-optimality can be considered in terms of energy consumption. Even though a task might be successfully accomplished while spending less energy, an individual can adhere to a more energy-consuming and thus sub-optimal movement strategy if it still secures satisfactory results.

1.2.2 A DYNAMICAL SYSTEM VIEW ON MOTION PRODUCTION

At the level of motion planning, this thesis is driven by a dynamical system[3] view on motion production in humans (Bullock & Grossberg, 1988). The dynamical system hypothesis has emerged to oppose the more traditional planning-execution model (Schmidt, 1975). According to the later, the role of sensory feedback during execution is reduced to correcting deviations from the motion plan. As the execution system is disconnected from planning, it stiffly rejects all deviations (for example, this is how a PD controller tracks a reference trajectory). In contrast, the online corrections exhibited by biological systems are goal-directed: if, while you are stretching your arm to fetch a cup of coffee, your friend pushes the cup closer to you, you will neither stubbornly follow the preplanned motion, nor will you freeze to think about how to get the cup from its new location. You might keep talking to your friend, while the arm instantaneously adapts its movement. In this example, the arm behaves like a stable dynamical system (with the cup as the attractor).

Dynamical system motion representation is particularly well-suited for learning *coordinated motions*, as a dynamical system offers a *generative* mechanism to reproduce *systematically similar* motor behavior under varying environmental conditions (e.g., different initial conditions or a moving target). It does so by encoding motions through functions that capture the temporal evolution of a continuous family of task motions. In the previous example of grabbing a cup, the hand executes a multidimensional motion, where displacements and velocities along all dimensions are tightly correlated spatially and temporally, so that the hand follows a typical motion profile.

A multivariate dynamical system motion encoding, that we develop in this work, enables a robot to learn how the coordination between the variables describing the motion propagates in time. Another strength of the dynamical system view on motion formation is the fact that the planning and execution are no longer two separated mechanisms when driven by a dynamical system. The motor system can instantaneously react to unexpected sensory information and successfully reach the goal of the motion. The dynamical system approach is naturally robust against perturbations. That is, if we mapped the location of the cup into the attractor of the system, even under perturbations, the system would smoothly rearrange the motion so as to reach the cup. This thesis follows a dynamical system approach to learn and generate multi-dimensional motions, both in free-space and in collaborative tasks. As neither the exact biological principles, nor a concrete computational form for the dynamical systems, underlying arbitrary human motions currently exists, we suggest a machine learning approach to extract an estimate of a dynamical encoding directly from human demonstrations and to ensure its local stability.

1.2.3 MOTION CONTROL

At the level of motion control, we follow a hypothesis from human motion studies that the CNS combines feedforward (anticipatory) and feedback (compensatory) control of

---

[3] In this context, the term "dynamical" refers to the temporal evolution of a motion.

motion (Tee et al., 2010). These types of control provide the human with complementary capabilities: feedforward control helps in overcoming sensory delays and enables motion compliance, whereas feedback control can efficiently counteract instabilities. With respect to the current state-of-the-art in robotics, the necessity to combine the two control strategies becomes particularly apparent when we consider collaborative execution in tasks where a human and a robot should coordinate and interact physically.

PbD suggests an overall view on skill transfer, but does not impose any concrete methodology. The approach taken in this thesis is to bond machinery from different well-established mathematical domains - statistical machine learning, dynamical systems, and control - into a unified framework for learning coordination. In our work, we address an ongoing problem in robot learning; that is, learning from noisy time-series data, where some information may be missing. We propose a novel approach to learning motion dynamics from several demonstrations. The advantages of the proposed method include the ability to encode and generalize an extracted coordination pattern to an unobserved context, time-independency, and robustness against the perturbations. These advantages allow our generic method to be used as the basis for building more specialized approaches. In this thesis, we use the dynamical system encoding to learn bimanual coordinated tasks and to teach a robot to physically interact with a human in collaborative tasks.

1.3 MAJOR CONTRIBUTIONS

In this thesis, we have made progress towards a generic framework for learning and executing coordinated motions. This progress includes contributions to the following lines of research:

- Robot Learning

  Learning Motion Representations for Unimanual Coordinated Tasks

  How can one infer a compact motion representation that captures a multivariate correlation pattern which couples several degrees of freedom? Might this correlation be easily generalized to unseen contexts? How can one depart from encodings that assume explicit timing in favor of more convenient motion strategies that do not require non-intuitive heuristics for maintaining an internal clock? Can a task model learned directly from demonstrations provide robustness against spatial and temporal perturbations? We propose a novel algorithm that estimates an autonomous nonlinear dynamical function underlying an arbitrary goal-directed motion, and demonstrate how the proposed encoding addresses the research questions raised above. We utilize the strengths of statistical learning to extract the dynamical function through Gaussian Mixture Models (GMM)/ Gaussian Mixture Regression (GMR), and ensure the local stability of the estimate. To date, our work is one of few existing approaches to motion representation that provides actual robustness against temporal perturbations and enables multivariate encoding of a motion.

The dynamical system motion representation that we develop here is generic and can be used as a building block for more complex tasks than unimanual manipulation. In this thesis, we exploit the strengths of dynamical system encoding to couple the two arms in discrete bimanual tasks and to continuously interact with a human during collaborative manipulation.

Learning Bimanual Coordinated Tasks

What are the constraints coupling the two arms? How can these constraints be automatically extracted from noisy motion data? How can the controllers governing each of the two arms be coupled so as to ensure the reproduction of the learned constraints? In our work, we try to overcome limitations in existing engineering and robot learning approaches to bimanual manipulation. Namely, we investigate the explicit learning of bimanual constraints. The proposed approach differs from conventional motion planning algorithms where the constraints are imposed by hand. Our method also differs from most existing robot learning methods, that rely on implicitly capturing bimanual constraints from the training data. We take inspiration from research on coordination in human motion science: the process of coordination is driven by discrete transitions between the states of so-called coordination variables, i.e. parameters that couple several degrees of freedom. We introduce a set of hypotheses regarding the form of these variables, and demonstrate how this allows the robot to learn discrete bimanual coordination tasks. We exploit continuous Hidden Markov Models (HMM) for encoding the states of the coordination variables and for generating a most probable sequence of states for reproduction. The extracted coordination constraints are subsequently mapped to the two coupled dynamical systems that generate Cartesian space trajectories for the two robotic arms. The suggested model generates coordinated movements online, while handling perturbations and satisfying the learned coordination constraints.

Learning Task Models for Physical Human-Robot Interaction

How can the robot learn to share both the goals and means of a collaborative task? How can we use the haptic information for teaching the robot to anticipate human intentions? What type of controller can encapsulate the learned task model and compensate for unmodelled effects that inevitably emerge once the human is included in the robot control loop? We use our dynamical system approach to motion encoding to build out a novel method that combines Programming by Demonstration and adaptive control to teach a robot to physically interact with a human. Here, encoding task movements as dynamical systems enables the learning of action-perception coupling: learning a task model allows the robot to anticipate the partner's intentions and adapt its motion according to perceived forces. As the human represents a highly complex contact environment, a direct reproduction of the learned model may lead to sub-optimal results.

To compensate for unmodelled uncertainties, we enhance the learned task model with an adaptive control algorithm which tunes the impedance parameters, so as to ensure accurate reproduction.

- Physical Human-Robot Interaction

  The methods developed in this thesis provide a fundamental basis for addressing problems of continuous physical human-robot interaction. We argue that such interaction imposes important requirements on motion encoding in robots, that should be time-independent and that should naturally incorporate continuous action-perception coupling. The dynamical system encoding that we propose here satisfies these requirements.

- Robot Application

  This thesis contributes to the state-of-the-art in robotics by addressing several important theoretical questions related to motion planning and control. We also contribute to robotics implementation through various real-world applications of the proposed algorithms. We showcase that our methods can be successfully applied to robotics platforms that differ in the number of degrees of freedom and types of control.

1.4 CONTRIBUTIONS PER CHAPTER

The methods described in Chapters 3 and 4 have been published in peer-reviewed conference proceedings and scientific journals. References to the related publications are provided at the beginning of each of the subsection of these two chapters.

The topics addressed by each chapter and their contributions are briefly described below.

Chapter 2: THEORETICAL BACKGROUND Chapter 2 presents an overview of several research domains that constitute the theoretical grounding for this work. We first provide an account of how motion planning and control have been addressed in analytical robotics and human motion studies. We emphasize challenges existing in these fields and then move to surveying the robot learning domain, to which our work relates directly. We highlight major directions of research within the robot learning domain and explain how learning techniques suggested in this thesis approach the unresolved challenges of motion coordination.

Chapter 3: A DYNAMICAL SYSTEM APPROACH TO MOTION REPRESENTATION AND BIMANUAL COORDINATION In Chapter 3, we start by presenting an algorithm for learning bimanual tasks where coordination between the hands is important. Within this learning framework we assume that the individual basic movements are generated using the predefined VITE model of human reaching movements.

Being described by a linear dynamical system of the second order, the VITE model is limited in its ability to produce curved motions of an arbitrary shape. To address this shortcomings, we further introduce the problem of learning dynamics of arbitrary one-arm motions from multiple demonstrations. We propose our approach to learning locally stable dynamical systems from human demonstrations and provide an experimental illustration and validation of the method. We emphasize the novelty of our approach by formally comparing it with the other state-of-the-art approaches.

The chapter concludes with the extension of the dynamical system motion representation to learning coordination between the position and orientation components of a robot's motion in Cartesian space. Simultaneous learning and reproduction of both motion components in a coordinated manner offers a "pre-shape" motion strategy and endows the robot with the capability of smooth adaptation in the case of perturbations, which may affect the two motion components either separately or simultaneously.

Chapter 4: LEARNING PHYSICAL HUMAN-ROBOT COORDINATION In Chapter 4, we integrate learning motion dynamics and coordination, so as to endow a robot with the ability to physically interact with humans. We consider the problem of physical interaction between a robot and a human in situations where they jointly perform manipulation tasks, e.g. the collaborative carrying and positioning of a load. The novel framework introduces the augmented state, i.e. the state that encapsulates both the kinematic command and the perceived haptic input. We show how such a formulation allows the robot to learn and generate its velocity as a function of the incoming haptic information. The robot, therefore, is able to adapt its motion according to the perceived human intentions. We emphasize the novelty of our approach by comparing it with the damping controller traditionally used for controlling a robot during physical interaction.

Chapter 5: CONCLUSION This chapter revises the assumptions of the proposed algorithms and discuss the main limitations of our work. Finally we summarize and discuss the principal contributions of this thesis.

Chapter 2

THEORETICAL BACKGROUND

In this chapter, we strive to provide a multifaceted view of motion coordination and so we analyze how this problem has been addressed within different disciplines.

Motion coordination is a rather broad subject that covers topics such as hand coordination in manipulation tasks, locomotion, whole-body coordination, and other types of systematic motor behaviors. In this manuscript, we particularly concentrate on the coordination of robot motion in manipulation tasks in the context of unimanual and bimanual manipulation as well as in the context of physical interaction between a robot's and a human's hands. In such tasks, the problem of coordination essentially includes motion planning and control. Furthermore, we argue that, in coordinated tasks, planning and control are subject to spatio-temporal constraints in order to ensure the reproduction of coordination patterns.

The original concept by Bernstein of coordination patterns as synergies that simplify task execution is abstract and gives no indication of how to identify these patterns or how they are formed. For robotics applications we suggest that the *coordination patterns* are the *systematic correlations* between variables describing a movement. The correlations can be observed in the process of repeatedly executing a task under varying environmental conditions. The correlations can be encoded in different spaces, e.g., in the joint or Cartesian spaces or at the level of control signals. We limit our research to correlations and coordination patterns that are formed in the task space. Such a choice seems reasonable in the context of manipulation tasks, which we consider here, since the goals of coordination in these tasks are effectively formulated in the Cartesian task space.

We advocate the use of dynamical systems as a means to represent the coordination patterns of a motion. Specifically, we assume that the motion's trajectories can be generated by an autonomous dynamical system that defines a systematic coordination pattern:

$$\dot{\boldsymbol{\xi}} = f(\boldsymbol{\xi}) \tag{2.1}$$

where $\boldsymbol{\xi}$ is the state of the robot (for instance, the position of the robot's end-effector in Cartesian space), and $f(\boldsymbol{\xi})$ is the dynamic function that describes spatio-temporal evolution of $\boldsymbol{\xi}$.

In this section, we first provide a brief account of how planning and control of coordinated motion have been addressed in *analytical robotics*[1] (Section 2.1) and human motion studies (Section 2.3).

Sections 2.1 touches upon the use of dynamical systems for motion planning and control, and this discussion will be continued further when looking at coordination in humans 2.3. To provide necessary background on dynamical systems and their estimation, we include Section 2.2 that contains a review of existing methods for system identification and stability analysis.

While explaining which problems related to motion coordination and identification of nonlinear dynamical systems have been resolved, we will emphasize challenges that are difficult to address if we are bound by the problem definitions inherent to the analytical domains (Sections 2.1.5, 2.2.3, and 2.3.4). These speculations will lead us to Section 2.4, where we survey the robot learning domain to that our work relates directly. We will highlight major research directions within the robot learning domain and explain how the learning techniques suggested in this thesis address some of the unresolved challenges of motion coordination.

We do not attempt to provide a complete analysis of what has been done within the discussed domains; instead, we summarize relevant state of the art methods and, where it is possible, refer an interested reader to other sources.

2.1 THE ANALYTICAL ROBOTICS VIEW ON MANIPULATION PLANNING AND CONTROL

The production of coordinated movements relates to the processes of trajectory planning and execution (or a single intertwined process, as we develop in this thesis). These problems (planning and execution) have generated a long-standing interest in analytical robotics. Therefore, the research conducted on motion planning is abundant; an interested reader may refer to the seminal book by Latombe (1991) or to a more recent book by LaValle (2006).

In this review, we particularly concentrate on two directions of motion planning that might be directly associated with our work – kinodynamic (Section 2.1.1) and feedback planning(Sections 2.1.2 and 2.1.3). Similar to the methods developed in this manuscript, kinodynamic planning considers trajectory generation in the state-space rather than in the joint or task space, as classic planning methods do. In its turn, feedback planning raises a question that we also seek to answer: how to generate a trajectory online while adapting to a dynamically changing environment.

The latter group of methods is further categorized into: early methods that preplan “desired" trajectories and then update them in real-time only locally (Section 2.1.2), and more advanced approaches that do not depend on a single desired trajectory and allow a robot to choose a completely different path in real-time if it appears to be more

---

[1] Here and further, we will denote as *analytical* the approaches that rely on a thorough analysis and understanding of a problem at hand. We contrast analytical approaches with *data-driven* methods that do not aim at building an exact structured model or an algorithm, but rather aim at approximating an unknown system with an estimate extracted from observations.

optimal (Section 2.1.3).

Once a motion plan is generated, it needs to be converted to motor commands. This task is performed by control algorithms. When we consider the robot coordinating with a human, control algorithms have to accommodate important requirements of safety and adaptability. We review existing methods of interaction control in Section 2.1.4.

2.1.1 KINODYNAMIC PLANNING

In a classical formulation, motion planning is a purely geometric problem: given the geometry of a robot and static obstacles, compute a collision-free path between two given configurations. The vast majority of basic path planning algorithms consider only positional aspects of the path, while ignoring the temporal aspect and the dynamics of the robot itself. However, robot motions are often subject to kinematic and dynamic constraints (kinodynamic constraints) (LaValle, 2006). In the simplest form, kinodynamic constraints can take the form of bounds on velocity or acceleration. Moreover, the environment may contain moving obstacles that require a computed path to be parameterized by time so that the robot knows when it has to pass through a particular state.

Kinodynamic planning has emerged as an attempt to overcome these drawbacks of conventional planning methods (Canny et al., 1991; LaValle & Kuffner, 2001; Sahar & Hollerbach, 1986). Kinodynamic planning extends the path planning problem beyond the joint or task space into a state-space that includes both configuration parameters (a cartesian position of a robot's hand or a joint configuration) and the corresponding velocity parameters.

Most existing kinodynamic planners are based on random sampling planners, such as Probabilistic Road Maps (PRMs) (Kavraki et al., 1996) or Rapidly-exploring Random Trees (RRT) (Lavalle, 1998), since the integration of kinodynamic constraints into combinatorial planners is almost intractable computationally. Random sampling methods have been first introduced to solve geometric path planning problems for robots with many degrees of freedom (Kavraki et al., 1996). In contrast to global planners (Lozano-Perez, 1983), that explicitly build a representation of the environment, random planners replace this computationally expensive procedure with a probabilistic exploration of the environment, where collision tests are conducted at randomly picked configurations and on the paths connecting them. Random sampling allows a considerable reduction in the computation cost: the cost does not grow exponentially with the number of degrees of freedom. Reduction of the computational cost is particularly important for kinodynamic planning where the dimensionality of any problem is doubled so as to include velocities.

Motion planning under kinodynamic constraints answers a question – how to find a path through the state-space between a given initial and target state while satisfying the kinodynamic constraints and avoiding obstacles. The kinodynamic constraints define laws that should be satisfied during the robot's motion. Mathematically, the kinodynamic constraints are expressed in a form of difference equation (or in a differential form in case of a continuous constraint):

$$x_{t+1} = f(x_t, u_t) \tag{2.2}$$

where $x_t \in \mathbb{X} \subset \mathbb{R}^n$ is the state variable at time t, $\mathbb{X}$ is the state-space; $u_t \in \mathbb{U} \subset \mathbb{R}^m$ is the control signal, and $\mathbb{U}$ is the set of admissible control inputs.

The RRT-planner with imposed kinodynamic constraints proceeds as follows. Let us assume that x_0 is the robot's initial position and the root of a trajectory tree T. At each iteration a point x^r is picked randomly and the nearest vertex of the tree x_t^n is computed according to a proximity metric $\rho(x_t^n, x^r)$. A control input u_t is iteratively chosen according to $x_{t+1}^n = f(x_t^n, u_t)$ so that to build a branch connecting the existing vertex x_t^n with the random point x^r. By construction, the obtained local trajectory satisfies the kinodynamic constraints. If the newly computed vertex x_{t+1}^n passes the collision test, it is added to the tree. Such an iterative incremental procedure rapidly explores the state-space and produces the tree T rooted at the initial state and oriented along the time axis towards the target state.

Despite the successes of randomized planning algorithms, they have an important shortcoming – the sensitivity of the performance to a chosen proximity metric ρ. The dependence on the metric becomes especially critical in kinodynamic planning, as the Euclidian norm does not provide relevant insights regarding the actual distance between two points in the state-space.

The ideal metric is the optimal cost-to-go, i.e., the optimal cost for a robot to move from one state to another. The optimal cost has to consider both kinodynamic and global constraints. A kinodynamic constraint emerges, for instance, if the robot is moving forward and cannot turn backward immediately (i.e. making a turn requires additional effort). In this case a metric that equally measures the distance ahead and behind the robot would be misleading. A global constraint emerges, for instance, if the robot should pass through a labyrinth with non-convex obstacles. Two states representing different locations inside the labyrinth might be close in terms of the Euclidean metric, but in reality the distance between them might be considerable because of obstacles – a correct global metric should take into account actual geometrical constraints of the world.

Though many approaches use a simple weighted Euclidian proximity metric (LaValle & Kuffner, 2001), the use of more task-oriented solutions can improve the performance drastically (LaValle, 2006). Several approaches have been suggested that utilize a non-holonomic metric (Laumond et al., 1998) or the optimal *cost-to-go* (Glassman & Tedrake, 2010; Sundar & Shiller, 1997).

In practice, planning under kinodynamic constraints is highly a non-trivial problem. The complexity of the problem increases even more for manipulation tasks as it is complicated to derive kinodynamic constraints as given by Eq.2.2 for each particular task. Therefore, for now, the existing kinodynamic planners mainly address aerospace applications or motion planning for car-like robots (Cheng et al., 2001; Hartmann, 2005; Phillips et al., 2003).

The applications of kinodynamic planning most closely related to our work include the animation of virtual avatars that perform household tasks like sorting objects on a shelf or opening drawers (Y. Koga et al., 2004; Molina-Tanco & Hilton, 2000; Popovic et al., 2002; Yamane et al., 2004). To enhance the human-likeliness of the generated movements, the authors of these methods suggest ways to incorporate captured human motion information into the planning mechanism. Yet, kinodynamic planning in these applications is not resolved in the way described in this section, i.e., simultaneously with trajectory planning. Instead, kinodynamic contraints are imposed through a two-step procedure that consists of path planning and path post-processing. The planning phase generates a collision-free path; for this an expert should define "hard" constraints such as the object's location or grasping points on the object. In the post-processing phase, the generated path is smoothed and converted into a trajectory by fitting to a human velocity profile (recorded through a motion capture system) into the path.

2.1.2 FROM A PLANNING-EXECUTION SCHEME TO A UNIFIED PARADIGM

Motion planning and control are traditionally regarded as two distinct areas of research (Siciliano & Khatib, 2008). However, their integration can bring important advantages, especially if a robot is supposed to operate in a dynamically changing environment. A motion planner needs to make strong assumptions about the environment and requires the ability to accurately predict the evolution of the robot's state in the future given its current and target states. If these requirements are satisfied, the planner generates a globally optimal path that is guaranteed to converge to the target. However, global optimality is attained at a considerable computational cost. Moreover, by the time the generated trajectory is ready to be executed, the environment may have changed so that the planned motion is no more relevant. Unsatisfactory robot performance in varying conditions simulates the development of integrated approaches to motion generation (Baginski, 1998; Barraquand & Latombe, 1991; Faverjon & Tournassoud, 1987; McLean & Cameron, 1996; Quinlan, 1994).

Early integration attempts shared an important similarity: they all operated through local modification of a globally optimal trajectory. That is, the globally optimal trajectory was first generated by a classical planner and then, during execution, was locally modified through a mechanism that produced virtual repulsive forces. The authors of these methods were often motivated by a scenario where a robot navigated between moving obstacles. Within this group, the methods differed between one another in computational requirements: whether a particular method considered obstacles in the task space or in the joint space; and whether it generated repulsive forces, at the trajectory level (Quinlan, 1994) or at the level of control signals (Brock & Khatib, 2002).

The elastic strips framework (Brock & Khatib, 1997, 2002) has gained a particular popularity in the robotics community: it integrates a planning mechanism with the operation space control of Khatib (1987) and provides a unified framework for obstacle avoidance in the joint and task space. The elastic strips also decouple task and posture control; the decoupling enables a user to impose constraints on the robot's posture without affecting the task performance. The elastic strip is represented by a candidate path (generated by any available planner) and a corresponding elastic tunnel (formed by spheres centered on the path). The elastic tunnel bounds a free part of the workspace, where the candidate path can be safely modified to satisfy constraints or avoid obstacles. By extending the path representation with the notion of the tunnel, the authors eliminate expensive analysis of the validity of the modified trajectory: for a new trajectory to be valid, it simply should be contained within the tunnel.

Despite bringing apparent advantages in comparison with the more conventional decoupled planning-execution paradigm, the methods discussed in this section are still limited in their ability to tackle large environmental changes. As the trajectories are modified only locally, large-amplitude perturbations can make them irrelevant. In the next section, we review motion planning approaches grouped under the name of feedback planning. Feedback planning methods incorporate information about the current robot state into the planning process such that the robot can adapt to deviations from a plan.

2.1.3 FEEDBACK PLANNING

In the conventional motion planning formulation, feedback is not considered. If initial and target states are given, the solution produced by a planner is a geometric path. The path is then transformed into a time-parameterized trajectory. Although recent algorithms are able to produce feasible open-loop trajectories for high-dimensional and non-convex problems (Frazzoli et al., 2002; Kuffner et al., 2001), in many cases the feedback is fundamental and its absence might seriously deteriorate performance. For instance, at the onset of the motion, we do not know the target state exactly or the motion might not be predicted correctly due to disturbances or errors in the model of the environment.

Therefore, for implementation on real robots, preplanned open-loop trajectories are stabilized by a feedback controller (e.g., by a PD controller). Such a decoupled approach works well for free-space motions in an environment where perturbations are small. If, for instance, the robot has been moved far from the preplanned trajectory, the stabilization attempt may fail or, at least, result in a sub-optimal movement, the task can be accomplished by other, more desirable trajectories. Feedback planning algorithms, that explicitly consider the feedback stabilization during the planning process, can avoid this limitation.

Potential fields (Khatib, 1986) is one of the early approaches to feedback planning. Under this approach a desired trajectory is generated though gradient descent along a potential function V that has a minimum at the target:

$$\dot{x} = -\nabla V(x) \tag{2.3}$$

where $V(x)$ is a potential function, e.g. $V = \frac{1}{2}\alpha(x_{\text{tar}} - x)^2$, where x_{tar} is the target state, $\alpha \in \mathbb{R}$ controls the speed of convergence. According to Eq.2.3, the robot's trajectory monotonously converges to the target x_{tar}. The potential fields offer several attractive features, for instance, they endow the robot with the ability to rapidly react to environmental changes. In Eq. 2.3, as we map the actual target position into the attractor x_{tar}, all perturbations of the target position will immediately modify the potential function $V(x)$ and consequently the trajectory. Moreover, the potential function can have a more complicated form than merely a quadratic function, e.g. it can incorporate not only attractors but also repellers so as to steer a robot away from obstacles. However, the motion of the robot guided by the potential field might be subject to a local minimum and, as a result, the robot may stop somewhere in the workspace before reaching its target. Some methods have been proposed to generate potential fields that do not suffer from the local minimum problem (Koditschek, 1987; Yun & Tan, 1997). Motion planning with potential fields is still active area of research; A. Masoud (2010); S. Masoud & Masoud (2002) extend the conventional potential fields framework so as to ensure resolution of kinodynamic constraints of the robot's body.

An interesting approach to simultaneously conduct feedback and kinodynamic planning is based on the concept of *funnels* (Burridge et al., 1999; Comer et al., 2003; Rizzi, 1998; Zefran & Burdick, 1998). A mathematical funnel brings a large set of initial conditions into a smaller set of final conditions (Mason, 1985). Each funnel represents a local area within the robot's workspace and enables linearization of a kinodynamic constraint in Eq. 2.2. Therefore, within a funnel, one can apply well-established tools of linear feedback control. Combining funnels allows one to obtain a global feedback planning policy that projects a broad set of initial robot's states into a target state.

One of the most recent and powerful implementations of this approach is proposed by Tedrake (2009) and Tedrake et al. (2010). Analogously to other randomized planning algorithms, the proposed method creates a tree of feasible trajectories by sampling randomly over a bounded region of the state-space. The novelty of the method is revealed once a new trajectory "branch" is added to the tree: the algorithm creates a local feedback controller and estimates its basin of attraction. Wherever the robot is located, if its position is within the basin of attraction, the feedback controller will generate correct commands. The algorithm terminates when the whole region of interest (the part of the robot's workspace) is contained within the basin of attraction of the tree. As a feedback controller, the authors choose the Linear Quadratic Regulator (LQR) controller, which, in addition to trajectory stabilization, builds a quadratic cost-to-go function. The benefit of building the cost-to-go function is two-fold: one simultaneously obtains a valid Lyapunov function (for estimation of the basin of attraction) and an optimal proximity metric. According to our discussion in Section 2.1.1 the design of a relevant proximity metric is a fundamental barrier to improving planners' performance and the method of Tedrake et al. (2010) suggests one way to overcome this metric problem.

Finally, a recent attempt to integrate visual feedback with a RRT planner in the context of manipulation tasks is described in (Diankov et al., 2009). The authors demonstrate how a simulated HRP-3 and a WAM robot manipulate objects in a complex environment with obstacles partly occluding the target. In this work, the robots do not plan the whole path before the movement's onset; instead, they sample trajectories as more visual information arrives. However, this approach suffers from two major drawbacks. First, a robot has to perform computationally expensive space exploration. Second, as a path is not defined beforehand, one is not able to fit a smooth velocity profile, therefore, the motion may appear jerky and non-intuitive.

2.1.4 ROBOT CONTROL

For a robot to be able to execute a trajectory generated by a planner, the latter should be converted into a sequence of motor commands. Algorithms that map a planned trajectory or, in a broader case, a desired behavior into executable commands are investigated in control theory (Astrom & Wittenmark, 2008; J.-J. Slotine & Li, 1991).

To enforce a robot to track a desired trajectory, one can use a proportional-derivative (PD) controller, as it does not require any knowledge of a robot's dynamics:

$$\boldsymbol{u} = \boldsymbol{K}(\boldsymbol{x} - \boldsymbol{x}_d) + \boldsymbol{D}(\dot{\boldsymbol{x}} - \dot{\boldsymbol{x}}_d) \tag{2.4}$$

where $\boldsymbol{u}$ is a control signal (e.g., joint torques or joint angles), $\boldsymbol{K}, \boldsymbol{D}$ are tunable gains, $\boldsymbol{x}_d, \dot{\boldsymbol{x}}_d$ are desired kinematic signals.

The algorithms that we propose in Section 3 rely on this controller to convert a learned kinematic behavior into commands. One drawback of the PD controller is the lack of compliance: a robot stiffly rejects all external disturbances. Such behavior is undesirable or even unsafe if the robot operates in a changing environment, where objects and other agents can move in an unpredictable way.

In Section 4, we go beyond the stiff tracking and investigate learning for physical coordination between a robot and a human. To provide a relevant background on interaction control, as it is addressed in analytical robotics, we further review work on impedance control (Section 2.1.4.1) and on application of impedance control to physical human-robot interaction (Section 2.1.4.2).

2.1.4.1 IMPEDANCE CONTROL

Manipulation tasks where a robot needs to physically interact with an environment have been a subject of active research; most existing approaches can be attributed to one of two fundamental control methodologies. The first approach, known as *hybrid position and force control*, is suggested by Raibert & Craig (1981). In hybrid position and force control, a task space is divided into position-controlled and force-controlled subspaces since position and force cannot be simultaneously controlled along the same direction. Therefore, hybrid control does not consider the dynamic coupling between a manipulator and an environment; however, ignoring this coupling might lead to instabilities (potential undesirable vibrations during contact with the environment) and to inaccuracies in position and force tracking.

The second approach aims to address these issues: Hogan (1985) proposes *impedance control*, where a mechanical impedance of a manipulator is regulated so as to match that of a target virtual model (typically chosen to be a spring-and-damper system). Impedance control establishes a dependency between the kinematical parameters of an end-effector and external force. Target system dynamics is described by the following equation:

$$\boldsymbol{\Lambda}_d(\ddot{\boldsymbol{x}} - \ddot{\boldsymbol{x}}_d) + \boldsymbol{D}_d(\dot{\boldsymbol{x}} - \dot{\boldsymbol{x}}_d) + \boldsymbol{K}_d(\boldsymbol{x} - \boldsymbol{x}_d) = \boldsymbol{f}, \tag{2.5}$$

where $\boldsymbol{\Lambda}_d, \boldsymbol{D}_d, \boldsymbol{K}_d$ are the matrices of desired inertia, damping, and stiffness, $\boldsymbol{f}$ is a vector of external forces perceived by the robot, and $\boldsymbol{x}_d, \dot{\boldsymbol{x}}_d, \ddot{\boldsymbol{x}}_d$ are the kinematic signals to be tracked. Note that here we define a desired impedance in a task space, as we consider it in our work (Section 4); a desired impedance can also be formulated in a joint space.

There are two ways to implement impedance control, depending on which control input a particular robotic platform admits. These two formulations of impedance control are referred to as *impedance* and *admittance* control.

Impedance control can be applied to force or torque controlled robots. Let us consider a rigid body dynamics model of a robotic arm:

$$\boldsymbol{\tau} = \boldsymbol{J}^T[\boldsymbol{\Lambda}(\boldsymbol{x})\ddot{\boldsymbol{q}} + \boldsymbol{\mu}(\boldsymbol{x}, \dot{\boldsymbol{x}})\dot{\boldsymbol{x}} + \mathbf{g}(\boldsymbol{x})] + \boldsymbol{J}^T\boldsymbol{f} \tag{2.6}$$

where $\boldsymbol{\tau} \in \mathbb{R}^{N_q}$ is a torque command, N_q is the number of controllable degrees of freedom; $\boldsymbol{J}$ is a Jacobian function of the arm, $\boldsymbol{\Lambda}$, $\boldsymbol{\mu}, \boldsymbol{g}$ are the inertia matrix, the coriolis term, and the gravitational term, respectively, all expressed in the Cartesian space. A torque control law that satisfies the rigid-body dynamics model Eq.2.6 and implements a desired impedance Eq. 4.5 can be written as:

$$\boldsymbol{\tau} = \boldsymbol{g} + \boldsymbol{J}^T[\boldsymbol{\Lambda}\ddot{\boldsymbol{x}}_d + \boldsymbol{\mu}\dot{\boldsymbol{x}}_d] + \boldsymbol{J}^T[\boldsymbol{\Lambda}\boldsymbol{\Lambda}_d^{-1}\boldsymbol{K}_d\boldsymbol{e}_x + (\boldsymbol{\Lambda}\boldsymbol{\Lambda}_d^{-1}\boldsymbol{D}_d + \boldsymbol{\mu})\dot{\boldsymbol{e}}_x + (\boldsymbol{\Lambda}\boldsymbol{\Lambda}_d^{-1} - \boldsymbol{I})\boldsymbol{f}]. \tag{2.7}$$

Essentially, the control signal $\boldsymbol{\tau}$ in Eq.2.7 consists of two parts: the feed-forward control $\boldsymbol{u} = \boldsymbol{g} + \boldsymbol{J}^T[\boldsymbol{\Lambda}\ddot{\boldsymbol{x}}_d + \boldsymbol{\mu}\dot{\boldsymbol{x}}_d]$, which attempts to track a desired trajectory with minimum forces, relying on the knowledge of the robot's dynamics, and the impedance control part, which aims to control interaction by adjusting to external forces and tracking errors. Robotic systems controlled with impedance control exhibit a stable dynamic interaction with a stiff environment but have a poor accuracy during free-space motions. Indeed, to ensure a stable interaction, a stiffness parameter $\boldsymbol{K}_d$ should be set to a rather low value. At the same time, a low stiffness combined with unmodelled friction might lead to the robot's inability to track a trajectory.

Many robotics platform accept only positional input. In this case, *admittance* control can be implemented if a robot is equipped with a force sensor. An admittance control law can be written as:

$$\boldsymbol{x} = \boldsymbol{x}_d - \boldsymbol{K}_d^{-1}(\boldsymbol{f} + \boldsymbol{D}_d\dot{\boldsymbol{x}}). \tag{2.8}$$

In essence, admittance control is similar to the stiff tracking of a desired trajectory $\boldsymbol{x}_d$, where interaction control is achieved through continuous adjustment of this trajectory $\boldsymbol{x}_d$ according to force measurements. In contrast to impedance, admittance control ensures motion accuracy in non-contact tasks but can result in instability during dynamic interaction with stiff environments.

This difference in the behavior of the two controllers has been studied by Valency & Zacksenhouse (2003). Ott et al. (2010) propose a unified framework, where a robot can switch between admittance and impedance control laws depending on task requirements.

Though impedance control has been under consideration for several decades, there are still a number of open questions related to its implementation on physical robots. One of the most critical issues is the choice of the impedance parameters: a widely accepted framework for tuning stiffness, damping, and inertia is missing (Buerger & Hogan, 2007). In most experimental studies, the parameters are hand-tuned using task knowledge, for instance, so as to make a robot stiffer along some, more constrained directions (Ott et al., 2005). The other important problem is that the rigid body model in Eq.2.6 represents only an approximation of the actual robot's dynamics, which in addition often contains nonlinear friction and other non-modelled effects. To compensate for inaccuracies of the model in Eq.2.6, one effectively would have to increase the stiffness – this makes an impedance controller less compliant than desirable. In Section 2.4.4, we review how research within robot learning attempts to overcome these limitations.

Impedance and admittance control have been applied to different manipulation tasks. For instance, in (Ott et al., 2005), the impedance of a humanoid robot Justin is controlled while the robot is opening a door. In (Ott et al., 2006), the same robotic platform is used to showcase a bimanual task of manipulating a ball: during the task execution, a human interferes by pushing and pulling the robot's arm; the robot complies to these perturbations, while managing to keep the ball as required by the task.

So far, we have discussed autonomous task execution. In the next section, we discuss how impedance algorithms are applied to control a robot during physical interaction with a human.

2.1.4.2 IMPEDANCE CONTROL FOR PHYSICAL HUMAN-ROBOT INTERACTION

Collaborative tasks can be deemed as an energy exchange between partners. Therefore, impedance control, which offers a means for regulating such an exchange, appears to be a suitable ground for algorithms that enable physical collaboration between a robot and a human.

Consider a task of bringing an object to a desired location. If a robot executes such a task autonomously, then position tracking with a PD controller given in Eq. 2.4 is a viable option. However, if several agents collaboratively manipulate a single object, the stiff position tracking would work only if all partners had the same desired trajectory and were perfectly synchronized. Obviously, such a situation is hardly possible, especially if one of the partners is a human. Conflicting kinematic agendas may result in significant interaction forces and eventually in a task failure. The use of impedance control, instead of a stiff PD controller, helps to smooth out discrepancies between motions of the partners.

In the last few decades, impedance control has been actively applied to control robots during physical interaction with other robots (M. Koga et al., 1992; Kosuge & Kazamura, 1996) and humans (Arai et al., 2000; Kosuge et al., 1993; Maeda et al., 2001; Rahman et al., 2002; Tsumugiwa et al., 2001).

M. Koga et al. (1992) consider the control of interaction between two robots. Their particular interest goes to a scenario where a manipulated object gets broken. In this case, the interaction force, perceived by the robots, changes abruptly. Therefore, the object's breakage can dangerously destabilize the two robots. The proposed algorithm ensures motion stability in this situation. Kosuge & Kazamura (1996) propose a decentralized control algorithm of multiple robots handling a single object in coordination. A motion command is given to one of the robots (referred to as a leader). The other robots estimate the motion of the leader through the motion of the object and handle the object based on the estimated reference. When considering physical interaction between the robots, the researchers can assign an equal impedance to all agents and, therefore, assume an equal load sharing, given that all robots track exactly the same trajectory.

Many works on physical human-robot interaction assume that only a human partner has knowledge about a task to be performed. Technically, this assumption means that the robot has no desired trajectory; that is, in Eq.4.5: $\boldsymbol{x}_d = \dot{\boldsymbol{x}}_d = \ddot{\boldsymbol{x}}_d = 0$. Therefore, the robot appears as a purely dissipative element: it does not inject any energy during the task execution. Consequently, the human partner has to apply more efforts than if he/she performs the task individually. This problem is discussed by Corteville et al. (2007). Yet, such control algorithms are useful in some applications, for instance, for manipulation of heavy objects: a robot can be programmed to keep an object at a specified altitude, while behaving as a passive element along a horizontal plane. The human still needs to guide the robot along the horizontal plane, but this requires less effort than an unaided manipulation.

To relieve a human partner from workload associated with guidance of a passive robot, researchers investigate two approaches. The first approach suggests implementing *varying impedance* parameters that should be fine-tuned so as to minimize the damping effect whenever it is possible (Duchaine & Gosselin, 2007; Rahman et al., 2002; Tsumugiwa et al., 2001). However, under this approach, the apparent dynamics of the robot perceived by the human is still passive. The second approach, *active following*, goes further, and allows the robot to input energy during interaction. This is accomplished by defining a desired motion of the robot (Corteville et al., 2007; Maeda et al., 2001).

In (Tsumugiwa et al., 2001), a collaborative task is divided into several phases based on a difficulty index (an onset, a moving phase, and an offset of a task). Different pre-defined impedance parameters are assigned to each of the phases. Willing to move beyond a heuristically pre-defined impedance, Duchaine & Gosselin (2007) argues that a robotic impedance should be tuned according to human's intentions. The researchers suggest that the derivative of the force perceived by the robot contains pertinent information regarding these intentions. They propose an algorithm that tunes a desired damping $\boldsymbol{D}_d$ of the admittance controller given by Eq. 2.8. The damping is increased if the robot detects a human intention to decelerate: the robot hence helps the partner to stop moving; if the robot perceives that the human intends to accelerate, $\boldsymbol{D}_d$ is decreased so as to minimize the damping effect.

Corteville et al. (2007) advocate the second approach and assert that, if desired kinematic signals are set to zero, then, even enhanced with a varying impedance, the robot still does not take a proactive stance in task execution and dissipates energy introduced by a human. They emphasize that active following necessitates task knowledge and the ability to predict the desired motion of the human. Maeda et al. (2001) and Corteville et al. (2007) independently propose model-based *active following* algorithms. The authors consider a one dimensional task of moving an object and assume that a human's trajectory follows the minimum jerk model. During task execution, the robot tries to identify the parameters of the minimum jerk model and then actively tracks the identified trajectory.

Most existing approaches to controlling physical human-robot interaction do not address explicitly the problem of stability of resulting controllers. One reason for this is the difficulty of formulating a stability criterion in case of interaction with a human. A general approach to ensure stability of an impedance controller during interaction with objects is by imposing a passivity condition (Colgate & Hogan, 1988; Hirata & Kosuge, 2000; Hirata et al., 2001). Buerger & Hogan (2007) emphasize limitations that stringent passivity constraints impose on a robot, particularly if the latter needs to interact with a human. Specifically, they argue that a robotic impedance might vary within larger bounds than it is allowed by the passivity constraints. The authors propose a less conservative stability criterion by making some hypotheses regarding impedance characteristics of a human arm.

2.1.5 THE CURRENT CHALLENGES

Summarizing our discussion on analytical motion planning and control, we may pinpoint the following challenges related to production of coordinated movements that have not been completely resolved as of yet and that we address in this manuscript.

- A particular path to follow is not considered to be a task by itself. In humans, task trajectories follow some coordination patterns and, therefore, represent an important part of a task. Even though a kinodynamic planner can generate a path satisfying differential constraints, the existing applications of kinodynamic planning mainly consider the robot's hardware constraints only. One reason for this is that, historically, matters of efficiency have preoccupied analytical robotics more than ergonomics or the comfort of a human user. Another reason is that within the scope of analytical robotics, a coordination constraint in the form given by

Eq. 2.2 has to be engineered by a researcher. This is the complicated and non-intuitive process if we think of the variety of that such constraints may take in manipulation tasks.

- State-space exploration is a computationally heavy process. Therefore, it may happen that real-time feedback planning of a multidimensional coordinated movement is practically infeasible. Additionally, the necessity of extensive state-space exploration makes planners sensitive to the proximity metric: a wrong metric may fail the search of a feasible path or lead to infinite computations.
- Despite recent advances in feedback planning, most planners still operate in open-loop and therefore require a heuristic timing mechanism for resampling a generated trajectory in the case of perturbations. Note that this problem also concerns the hybrid methods discussed in Section 2.1.2. The complexity of such a mechanism lies in the necessity to reestimate movement duration and reset a current time index if a perturbation occurs.
- Potential fields currently represent the most computationally efficient feedback planning method. Therefore, they are often applied to robotics applications. However, for each new task a potential function should be individually designed. As a number of task dimensions grows, rationalizing about a potential function quickly gets non-intuitive, therefore, many existing practical implementations consider only planar motions. Additionally, in manipulation tasks coordination requirements might be difficult to formalize analytically.
- Research on impedance control of physical human-robot interaction investigates two promising directions, variable impedance and active following. Existing methods demonstrate that for being an efficient partner, a robot needs task knowledge. Analytical ways of providing such knowledge, for instance by assuming a linear motion of a human (Corteville et al., 2007) or by devising different impedance parameters for different phases of a movement (Tsumugiwa et al., 2001), are of a limited use, given a variety of potential interactions.

In Sections 3 and 4, we approach some of these challenges. We suggest a motion planning method that operates in the state-space, incorporates the feedback loop, and allows for motion adaptation in real-time. This algorithm is further extended to enable active following during physical human-robot coordination.

2.2 IDENTIFICATION OF DYNAMICAL SYSTEMS AND STABILITY ANALYSIS

In the previous section we discussed the methods to analytical motion planning and control. In this section, we continue the overview of analytical approaches, but concentrate on providing the background on mathematical tools that relate to the adopted dynamical system approach.

Throughout our work, we promote the dynamical system approach as a means for efficient and compact representations of human movements. Due to a lack of existing computational models that explain arbitrary nonlinear motions, we suggest an approach to extract dynamical systems underlying observed trajectories directly from these data (Section 3.3 and 4). Our work therefore relates to methods of system identification. In classical control theory, one often faces a problem of regulating a system whose parameters are unknown. Before designing a control law, the unknown parameters need to be identified. As we explain in Section 2.2.1, despite the fact that the literature offers various approaches to estimate unknown dynamics, most progresses are achieved toward identification of *known dynamical systems* with *unknown parameters* (Section 2.2.1.1); the estimation of *completely unknown dynamics* is still under active research and no universal solution exists (Section 2.2.1.2).

Another important challenge associated with the use of dynamical systems for control or motion generation is stability analysis. We discuss major results of this field in Section 2.2.2.

2.2.1 IDENTIFICATION

Methods of *system identification* aim to estimate unknown parameters of a dynamical system. The problem can be formalized as follows. Let us consider a nonlinear dynamical system in a canonical form:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \boldsymbol{\alpha}) \tag{2.9}$$

where $\mathbf{x} \in \mathbb{R}^N$ is a state of the system, and $\boldsymbol{\alpha} \in \mathbb{R}^P$ is a vector of unknown parameters. Let us assume further, that a state $\mathbf{x}$ cannot be measured directly. Instead, one observes a value $\mathbf{y}(t) \in \mathbb{R}^M$:

$$\mathbf{y}(t) = \mathbf{g}(\mathbf{x}(t)) + \boldsymbol{\eta}, \tag{2.10}$$

where $\mathbf{g}$ is a measurement function and η is a measurement error taken as a Gaussian noise. It is further assumed that the system is sampled at discrete instants of time. The sampling is uniform; that is, measurements are performed at a fixed rate $t, t + \Delta t..t + (T - 1)\Delta t$.

Given a time series $\mathbf{y}_{t_i},\ i = 1..T$, the identification problem is to estimate the parameters $\boldsymbol{\alpha}$ and the states $\mathbf{x}(t)$

Our review concentrates on identification of autonomous dynamical systems that are deterministic, nonlinear, finite dimensional, and continuous in time. It is not assumed that a complete system state is available for measurements.

If an analytical form of a dynamical $\mathbf{f}$ and measurement $\mathbf{g}$ functions is known and only a set of parameters $\boldsymbol{\lambda}$ needs to be identified, then the problem is known as *parametric* identification. We review methods that fall into this group in Section 2.2.1.1. It also may happen that we have no knowledge of an underlying dynamical function $\mathbf{f}$. If this is the case, one needs to assume a functional basis that is suitable for approximating **f**; the unknown parameters are then the parameters of the functional basis. This problem is investigated by methods of *non-parametric* identification, which we review in Section 2.2.1.2.

2.2.1.1 PARAMETRIC METHODS

Parametric methods can be split into two broad categories. The methods of the first category rely on a global optimization of a cost function (e.g., least square optimization (Aguirre & Billings, 1995; Cremers & Huebler, 1987; Crutchfield & McNamara, 1987; Hegger, 1998) and shooting methods (Baake et al., 1992; Domselaar & Hemker, 1975; Schittkowski, 1994)). The methods of the second category perform local, *recursive* estimation of the unknown parameters $\boldsymbol{\alpha}$; that is, the parameters are updated at each point in time recursively, as the method proceeds through the time series (e.g., various implementations of the Kalman Filter fall into this group):

One of the most straightforward approaches for identifying unknown parameters $\boldsymbol{\alpha}$ is the *least square optimization* (Aguirre & Billings, 1995; Cremers & Huebler, 1987; Crutchfield & McNamara, 1987; Hegger, 1998). A least square cost function is defined as:

$$\begin{aligned} L &= \sum_{i=1}^{M} (\mathbf{g}^{-1}[\mathbf{y}_{t_i}] - \mathbf{f}(\mathbf{x}_{t_i}, \boldsymbol{\alpha}))^2 \\ \hat{\boldsymbol{\alpha}} &= \arg\min_{\boldsymbol{\alpha}} L \end{aligned} \tag{2.11}$$

Due to its apparent simplicity, the least square optimization has been applied to various problems including identification of chaotic systems (Crutchfield & McNamara, 1987). However, researchers pinpoint a number of issues that lead to an unsatisfactory performance in some conditions. If a dynamical system **f** is of an order higher than one (e.g., depends on a second derivative $\ddot{\mathbf{x}}$), the performance of the least square identification crucially dependant on the ability to accurately approximate higher derivatives of **x**, which might be not an easy task if measurements are noisy.

The other challenge is known as the *errors-in-variables* problem (Madansky, 1959) that stems from the fact that, in the system in Eqs. 2.9-2.10, not only the dependent variables **y** are corrupted by noise, but also the independent variables **x**. The error-in-variables problem is inevitable if we assume a measurement noise. Indeed, in our case, **x** are independent variables, but they are unobservable and need to be estimated; however, we cannot calculate their deterministic values due to the noise factor $\boldsymbol{\eta}$ in Eq.2.10. Noise therefore leads to a biased or even incorrect least-squared estimation.

The method of *total least squares* (TLS) (Boggs et al., 1987; Van Huffel & Vandewalle, 1991) attempts to address the problem of errors in variables. Taking into account the uncertainty of measurements, TLS suggests to use both measurements $\mathbf{y}_{t+\Delta t}$ and $\mathbf{y}_t$ for estimating a state $\mathbf{x}_t$; the idea behind is to minimize an effect of noise $\boldsymbol{\eta}$, which is assumed to affect $\mathbf{y}_{t+\Delta t}$ and $\mathbf{y}_t$ independently. However, it has been shown that TLS still produces suboptimal results (in the maximum likelihood sense) and significant estimation errors. Kostelich (2001); McSharry & Smith (1999) confirm a limiting applicability of TLS for identification of nonlinear systems.

Regarding the numerical feasibility of the optimization problem in Eq.2.11, some researchers have observed that both simple and total least squares, might result in a cost function that has an excessive number of local minimums. Therefore, the optimization gets too complex to be efficiently resolved with standard methods (Theiler, 1990; Theiler & Smith, 1995).

An alternative to the least square optimization is a maximum likelihood approach. The other group of global optimization methods is represented by shooting methods (Baake et al., 1992; Domselaar & Hemker, 1975; Schittkowski, 1994) that construct a likelihood function of an observed sample. Additionally, the methods of this group include an initial condition $\mathbf{x}_0$ into a set of parameters to be optimized. The initial value approach (Schittkowski, 1994) considers a single initial condition and optimizes the probability distribution $p(\mathbf{y}_0..\mathbf{y}_T|\mathbf{x}_0,\boldsymbol{\lambda})$ of observed data given states and parameters:

$$\{\hat{\mathbf{x}}_0,\hat{\boldsymbol{\lambda}}\} = \arg\max_{\mathbf{x}_0,\boldsymbol{\lambda}} p(\mathbf{y}_0..\mathbf{y}_T|\mathbf{x}_0,\boldsymbol{\lambda}). \tag{2.12}$$

It has been proved that the *initial value approach* theoretically renders optimal results. However, in practice, the quality of estimates depends on the accuracy of numerical optimization. Horbelt et al. (2000) demonstrate that estimated state trajectories quickly diverge. In the *multiple shooting* approach by Baake et al. (1992); Domselaar & Hemker (1975), initial conditions are estimated at several time steps $\tau_1..\tau_K$ along a sample time series. Through this amendment, an estimated trajectory is forced to stay closer to a true one.

$$\{\hat{\mathbf{x}}_{\tau_1},..,\hat{\mathbf{x}}_{\tau_K},\hat{\boldsymbol{\lambda}}\} = \arg\max_{\mathbf{x}_{\tau_1},..,\mathbf{x}_{\tau_K},\boldsymbol{\lambda}} p(\mathbf{y}_0..\mathbf{y}_T|\mathbf{x}_{\tau_1},..,\mathbf{x}_{\tau_K},\boldsymbol{\lambda}) \tag{2.13}$$

In contrast to methods discussed so far, which conduct a global optimization of parameters, the second group of identification methods reviewed in this section, *prediction-correction method*, suggests to estimate underlying trajectories by proceeding *recursively* through a data sample. A general idea behind such identification can be summarized as follows:

given a state $\mathbf{x}_t$, (2.14)
predict a state $\tilde{\mathbf{x}}_{t+\Delta t}$ and an observable $\tilde{\mathbf{y}}_{t+\Delta t}$,
correct the estimate $\tilde{\mathbf{x}}_{t+\Delta t}$ once an actual measurement $\mathbf{y}_{t+\Delta t}$ is available.

The Kalman filter (Kalman, 1960; Kalman & Bucy, 1960) and its different extensions (Extended and Unscented Kalman Filters (Anderson & Moore, 1979; Julier et al., 2000)) fall into this category of *recursive* approaches.

Let us assume that we collect observations $\mathbf{y}_1..\mathbf{y}_t$ and estimate states $\mathbf{x}_1..\mathbf{x}_t$ up to time t. From these data, the Kalman filter builds an *a priori* state estimate $\tilde{\mathbf{x}}_{t+\Delta t}$ and predicts a next observation $\tilde{\mathbf{y}}_{t+\Delta t}$. Once we obtain an actual observation $\mathbf{y}_{t+\Delta t}$, the Kalman filter re-estimates the *a priori* value $\mathbf{x}_{t+\Delta t}$ by calculating an *a posteriori* estimate as:

$$\mathbf{x}_{t+\Delta t} = \tilde{\mathbf{x}}_{t+\Delta t} + \mathbf{K}(\mathbf{y}_{t+\Delta t} - \tilde{\mathbf{y}}_{t+\Delta t}), \tag{2.15}$$

where $\mathbf{K}$ is a Kalman gain matrix computed from the covariance matrices of the state and observables:

$$\mathbf{K} = \mathbf{\Sigma}_{xy}\mathbf{\Sigma}_{yy}^{-1}, \quad \mathbf{\Sigma}_{xy} = \mathbb{E}[(\mathbf{x} - \tilde{\mathbf{x}})(\mathbf{y} - \tilde{\mathbf{y}})^T], \quad \mathbf{\Sigma}_{yy} = \mathbb{E}[(\mathbf{y} - \tilde{\mathbf{y}})(\mathbf{y} - \tilde{\mathbf{y}})^T] \tag{2.16}$$

The covariances $\mathbf{\Sigma}_{xy}$ and $\mathbf{\Sigma}_{yy}$ measure expected variance between the actual and *a priori* estimates. At each time step, the Kalman filter updates $\mathbf{K}, \mathbf{\Sigma}_{xy}, \mathbf{\Sigma}_{yy}$, and the estimates $\mathbf{x}_{t+\Delta t}, \mathbf{y}_{t+\Delta t}$.

In the linear case, one can easily compute covariance matrices $\mathbf{\Sigma}_{xy}, \mathbf{\Sigma}_{yy}$ and, consequently the gain matrix $\mathbf{K}$, analytically. For nonlinear systems, estimation of covariances gets more complicated. The Extended (EKF) and Unscented (UKF) Kalman Filters are designed to address nonlinear identification. EKF (Anderson & Moore, 1979) suggests linearizing a system's function $\mathbf{f}$ through a Taylor expansion. After the linearization, one can use analytical expressions of the linear Kalman filter. However, the linearization produces satisfactory results only if nonlinearities in $\mathbf{f}$ are weak (e.g., if $\mathbf{f}$ is a second-order polynomial). In contrast to EKF, UKF (Julier et al., 2000) enables a more accurate estimation: instead of using the linearization as a means to avoid estimation of covariances, UKF assumes that the density distribution of states $\mathbf{x}$ is Gaussian. The covariances are then computed as empirical covariances of samples from this distribution.

One should keep in mind that the approximations imposed by the UKF ignore nonlinearity in a state distribution. Therefore, if the state $\mathbf{x}$ follows a multimodal distribution, UKF might be inaccurate. The assumption of the unimodal normality is relaxed in methods based on Monte-Carlo sampling (e.g., importance sampling (Tanizaki, 1993), bootstrapping (Kitagawa, 1996), and particle filters (Arulampalam et al., 2002)). Monte Carlo-based methods approximate a state density numerically, by generating random samples and using them to estimate covariances in the Kalman Filter. The random sampling thus appears particularly useful if a probability density of a state $\mathbf{x}$ strongly deviates from a Gaussian distribution, such as, when this distribution is multimodal.

A specific objective of Kalman Filters pertains to estimation of a state trajectory. To simultaneously compute unknown parameters $\boldsymbol{\lambda}$, one needs to consider an augmented state: $[\mathbf{x}; \boldsymbol{\lambda}]^T$ (Bar-Shalom et al., 2001).

From an application point of view, recursive methods might appear to be a more attractive solution when online identification is necessary. However, global identification tends to produce more accurate results. Currently, there are no comparative studies that would systematically delineate computational advantages of one group over the other. To some extent, global and recursive identification might be considered complementary. The former focuses on the estimation of unknown parameters, and state trajectories are estimated as a byproduct. The latter particularly aims to uncover states trajectories, whereby the parameters can also be estimated.

2.2.1.2 NONPARAMETRIC METHODS

The previous section provides an overview of methods that allow for estimation of unknown parameters in case when a dynamical function itself has a known form. In many practical applications, one might have no structured knowledge about the dynamical function; that is, the only available information consists of a set of observed inputs and outputs. We refer to methods that offer a solution to such problems as methods for *nonparametric* identification (Peifer et al., 2002; Timmer et al., 2000; Voss & Kurths, 1997). Nonparametric, data-driven methods consider multivariate input-output data as instances of a dynamical system; these instances are employed to build an estimate of an underlying unknown dynamics.

Sometimes, it is known that **f** spans a set of continuous known basis functions, $\{\boldsymbol{\phi}_i(\mathbf{x})\}_{i=1}^{N}$ (J. Slotine & Coetsee, 1986):

$$\mathbf{f}(\mathbf{x}, \boldsymbol{\alpha}) = \sum_{i=1}^{P} \alpha_i \boldsymbol{\phi}_i(\mathbf{x}) \tag{2.17}$$

where $\boldsymbol{\alpha}$ are unknown parameters. Identification of the parameters $\boldsymbol{\lambda}_i$ can be performed using one of the methods discussed in the previous section.

If exact basis functions $\boldsymbol{\phi}_i(\mathbf{x})$ are unknown, it is still possible to estimate the function **f** using a set of some elementary functions. Essentially, under such a formulation, a subspace where the function **f** needs to be approximated is partitioned into a number of small cells (Sanner & Slotine, 1992) (this concept is analogous to the receptive fields or neurons); and each cell is associated with a basis function. The unknown parameters $\boldsymbol{\alpha}_i$ and those of basis functions are estimated through optimization. With respect to how the information is used to update parameters, one may distinguish two groups of methods: *local* methods, where only local data contribute to the calculation of the parameters of each cell and therefore updates can be done iteratively, and *global* methods, where all data are taken into account for estimating the parameters of each cell, and all parameters need to be updated simultaneously.

Local identification methods include nearest neighbor approaches (Farmer & Sidorovich, 1988; Lancaster & SalKauskas, 1986), look-up tables and cerebellar model articulation controllers (CMAC) (Miller et al., 1987), and memory based learning (Atkeson, 1992) (we examine this method in more details in Section 2.4.1, when we discuss regression techniques used in robot learning). The methods of this group differ in basis functions that they employ for approximation (e.g., binary functions (Albus, 1975), polynomials (Atkeson, 1992; Lane et al., 1992; E. Lee, 1986), or exponential functions (Buhmann, 2003; S. Lee & Kil, 1991)).

Global nonparametric methods for identification are largely represented by neural networks (Tomohisa et al., 2008; Travis et al., 2009; Wei & Amari, 2008) (see Hunt et al. (1992) for a discussion on the use of neural networks in control applications).

Specifically, the following recurrent neural networks have been applied to approximate dynamical systems: multilayered networks trained through back-propagation (Pineda, 1989; Werbos, 1980), radial basis networks (RBFs) (Sanner & Slotine, 1992; Tomohisa et al., 2008), Hopfield networks (Hopfield, 1984), and echo-state networks (Jaeger et al., 2007).

We do not go into the implementation details of particular algorithms as they are quite heterogeneous and some are distant from the methods used in our work. In Section 3.3, we review estimation of dynamical systems as it is addressed in robot learning. In a broad sense, these methods can also qualify as nonparametric identification approaches.

One should keep in mind that for both, global and local nonparametric identification algorithms, the number, location and size of cells greatly influence the accuracy of an approximation. This observation might appear contradictory to the fact that some networks possess the property of being *universal approximators* of an arbitrary nonlinear function on a compact set. For instance, J. Park & Sandberg (1991) prove this property for radial basis networks. However, while in theory a network may have the capacity to represent any nonlinear function, in practice, the quality of approximation depends on the choice of network's parameters. Consider, if in a part of a state space, a function $\mathbf{f}$ is changing rapidly, then the size of cells covering this part has to be smaller than this in a part where $\mathbf{f}$ varies slowly. The problem of a network architecture design is a complex one. Works that consider practical applications of neural networks often concentrate on a particular training procedure and rarely investigate theoretical properties of a given architecture; this observation is discussed in Hunt et al. (1992).

To compensate for a lack of principled theoretical grounds underlying a choice of a network's structure, some researchers attempt to quantify approximation properties by leveraging results from sampling theory (Petersen & Middleton, 1962) and Fourier analysis. Sanner & Slotine (1992) propose an algorithm for controlling a plant with unknown dynamics. They start by choosing radial basis functions as a basis set. Fourier analysis is then applied to compute a length scale of the function $\mathbf{f}$ from available input-output samples. The authors use the calculated length scale to quantify an effect of a number of RBFs and their variance on approximation error. Their quantification motivates a formulation of analytical conditions that these parameters should admit so as to achieve a given level of accuracy within a network. Gonzalez-Serranoa et al. (1998) provide a similar analysis for CMAC networks.

2.2.2 STABILITY ANALYSIS

Once a dynamical function is known, the next step toward its use for control is stability verification. In this section, we review three major approaches to stability analysis: Lyapunov stability theory (La Salle & Lefschetz, 1961; J.-J. Slotine & Li, 1991) and the passivity approach (Hill & Moylan, 1976), and contraction theory (Lohmiller & Slotine, 1998).

We start by defining the basic concept of stability in the Lyapunov sense. Consider a nonlinear autonomous system:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}), \tag{2.18}$$

where $\mathbf{x} \in \mathbb{R}^N$ is a state vector and $\mathbf{f} : \mathbb{R}^N \to \mathbb{R}^N$ is a continuously differentiable function.

A state $\bar{\mathbf{x}}$ is an *equilibrium* of Eq. 2.18, if $\mathbf{f}(\bar{\mathbf{x}}) = 0$ – once a state of the system 2.18 is equal to $\bar{\mathbf{x}}$ it remains equal to $\bar{\mathbf{x}}$ for all future times (J.-J. Slotine & Li, 1991).

For an equilibrium $\bar{\mathbf{x}}$ to be *stable*, it is necessary that, for any arbitrary $R > 0$, there exists $r > 0$ such that if a trajectory starts within a hypersphere of a radius r, $||\mathbf{x}(0) - \bar{\mathbf{x}}|| < r$, it will stay within a hypesphere of a radius R, $||\mathbf{x}(t) - \bar{\mathbf{x}}|| < R, t \to \infty$. If it is not possible to find such r, the equilibrium $\bar{\mathbf{x}}$ is *unstable*. For *asymptotic stability*, the theory additionally requires that $||\mathbf{x}(t) - \bar{\mathbf{x}}|| \to 0$. Typically, to simplify analysis, one transfers the equilibrium into the origin; we further adopt this simplification.

The seminal work by Lyapunov (1992) suggests two methods for stability analysis, the *linearization method* and the *direct method*. The former draws conclusions about a nonlinear system's local stability from the stability properties of its linear approximation. The latter (direct) method aims to overcome the restrictions of local stability and attempts to determine a scalar function (a *Lyapunov function*) and to assess stability by examining the time variation of this function.

By linearizing the system 2.18 around an equilibrium, we get:

$$\dot{\mathbf{x}} = \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)_{\mathbf{x}=\mathbf{0}} \mathbf{x} + O(\mathbf{x}), \tag{2.19}$$

where $O(\mathbf{x})$ are higher order terms of the expansion. According to the linearization method, a system 2.18 is locally asymptotically stable at the origin, if a Jacobian matrix $\mathbf{A} = \left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)_{\mathbf{x}=\mathbf{0}}$ is negative definite, i.e., real parts of all the eigenvalues of $\mathbf{A}$ are strictly negative.

For nonlinear systems, the linearization method is concerned with local stability which is ensured only as far as the linear approximation is valid. The direct Lyapunov method aims to relax the restricting requirements imposed by the linearization approach and to provide a more generic stability criteria for a nonlinear dynamics. The direct method is based on a physical observation: if the total energy of a system is continuously dissipated, then the system, whether linear or nonlinear, eventually settles down to an equilibrium point. Therefore, one may decide upon the stability of a system by examining the variation of a single scalar function.

A function $V(\mathbf{x})$ is said to be a *Lyapunov function* of the system 2.18, if $V(\mathbf{x})$ is positive definite, has continuous partial derivatives, and if its time derivatives along any state trajectory of Eq. 2.18 is negative semi-definite: $\dot{V}(\mathbf{x}) \leq 0$. A dynamical system for which it is possible to construct a Lyapunov function is stable. The link between an existence of Lyapunov function and stability is formalized in a number of theorems in the Lyapunov direct method. For instance, for a linear dynamical system one may consider a quadratic Lyapunov function $V(\mathbf{x}) = \mathbf{x}^T \mathbf{P} \mathbf{x}$, where $\mathbf{P}$ is a symmetric positive definite matrix. The stability condition then can be written as:

$$\mathbf{A}^T\mathbf{P} + \mathbf{P}\mathbf{A} < 0 \tag{2.20}$$

meaning that a matrix $\mathbf{A}^T\mathbf{P} + \mathbf{P}\mathbf{A}$ should be strictly negative definite.

Krasovskii theorem suggests an analogous formulation for nonlinear systems: if a matrix $\mathbf{A} + \mathbf{A}^T$ is negative definite in a neighborhood of an equilibrium, then the equilibrium point is asymptotically stable and a Lyapunov function of this system is $V(\mathbf{x}) = ([\mathbf{f}(\mathbf{x})]^T\mathbf{f}(\mathbf{x})$. An inequality in Eq. 2.20 is known as a *Linear Matrix Inequality* (LMI) (Boyd et al., 1994); methods for solving such inequalities generate a significant interest in optimization.

Passivity theory is often mentioned in the context of stabilization and system control. Passivity has been proven to be a useful concept for studying interconnected systems. Let us consider a dynamical system connected to a control input:

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \mathbf{f}(\mathbf{x}(t)) + \mathbf{u}(t) \\ \mathbf{y}(t) &= \mathbf{h}(\mathbf{x}(t)).\end{aligned} \tag{2.21}$$

A system is said to be *passive* if there exists a nonnegative storage function:

$$S(\mathbf{x}(t)) = -\sup \int_0^t \mathbf{y}^T(s)\mathbf{u}(s)ds \tag{2.22}$$

such that $\mathbf{S}(0) = 0$ and

$$S(\mathbf{x}) - S(\mathbf{x}_0) \leq \int_0^t \mathbf{y}^T(s)\mathbf{u}(s)ds. \tag{2.23}$$

Qualitatively, the condition given in Eq. 2.23 means that a system absorbs more energy from an external source than it outputs. The passivity is, therefore, characterized by the existence of a computable function $S(\mathbf{x})$, which can be interpreted as *stored energy* of the system. Under certain conditions, an energy function is also a Lyapunov function. It is also possible to show that, if a system has a Lyapunov function, then this system is passive and its Lyapunov function is also its storage function.

One of the advantages of passivity is that this formulation allows for a simplified treatment of interconnected systems, particularly, feedback systems. It has been shown that if several passive systems are interconnected through functions that satisfy some constraints, the resulting system remains passive. In this case, one can easily derive a storage function of the complete system as a sum of storage functions of the individual subsystems (Hill & Moylan, 1976; Willems, 1972).

Lyapunov stability theory and passivity analysis investigate a convergence to an equilibrium or to a single trajectory. In contrast, *contraction analysis* defines incremental stability between two arbitrary trajectories (Lohmiller & Slotine, 1998). Specifically, a nonlinear system is called *contracting* if initial conditions or temporary disturbances are forgotten exponentially fast so that all trajectories converge to a unique trajectory.

The infinitesimal squared distance between two trajectories of the system in Eq.2.18 is $\delta\mathbf{x}^T\delta\mathbf{x}$; and the rate of change of this distance is:

$$\frac{d}{dt}(\delta\mathbf{x}^T\delta\mathbf{x}) = 2\delta\mathbf{x}^T\delta\dot{\mathbf{x}} = 2\delta\mathbf{x}\frac{\partial\mathbf{f}}{\partial\mathbf{x}}\delta\mathbf{x} \tag{2.24}$$

If λ_{max} is the largest eigenvalue of the Jacobian $\frac{\partial\mathbf{f}}{\partial\mathbf{x}}$, then the following inequality holds ():

$$\frac{d}{dt}(\delta\mathbf{x}^T\delta\mathbf{x}) \leq 2\lambda_{\text{max}}\delta\mathbf{x}^T\delta\mathbf{x}. \tag{2.25}$$

From Eq. 2.25, one may see that, if λ_{max} is strictly negative, all trajectories of the system Eq. 2.18 converge to a single trajectory exponentially fast.

To render a notion of an infinitesimal distance $\delta\mathbf{x}^T\delta\mathbf{x}$ more generic, Lohmiller & Slotine (1998) introduce a *contraction metric* $\mathbf{M}(\mathbf{x},t)$, such that the distance is taken with respect to this metric:

$$\delta\mathbf{x}^T\mathbf{M}\delta\mathbf{x}, \tag{2.26}$$

where, $\mathbf{M}(\mathbf{x},t)$ is a symmetric, positive definite and continuously differentiable matrix. It can be shown that a system given by Eq.2.18 is contracting if a generalized Jacobian $(\dot{\mathbf{M}}+\mathbf{M}\frac{\partial\mathbf{f}}{\partial\mathbf{x}})\mathbf{M}^{-1}$ is negative definite. The existence of a contraction metric $\mathbf{M}$ that satisfies this condition ensures that an associated distance between trajectories is always decreasing.

One of advantages of a stability treatment offered by contraction theory relates to an analysis of nonlinear systems with uncertainty:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{s}(\mathbf{x}), \tag{2.27}$$

where $\mathbf{s}(\mathbf{x})$ is a function describing system perturbations. Assuming that one can establish the stability of a nominal system $\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x})$ and has very limited information regarding $\mathbf{s}(\mathbf{x})$, the problem consists in estimating how much of uncertainty the nominal system might absorb before getting unstable.

Lyapunov and passivity analysis has been successfully applied to many problems where parameters of a dynamical function $\mathbf{f}$ remain unchanged. Nonlinear systems with uncertainty are challenging for Lyapunov-based techniques: uncertainty can modify the location of equilibrium points and a Lyapunov function should account for this change. Some researchers attempt to use parameter-dependent Lyapunov functions to prove stability for a range of uncertain parameters (L. Andersson & Rantzer, 1999; Michel & Wang, 1993). However, in many cases, it may be difficult or even impossible to analytically obtain a parameterized expression for an equilibrium and consequently to design a Lyapunov function.

Contraction theory overcomes some restrictions and problems of the Lyapunov approach. Aylward et al. (2008) demonstrate that if a nominal system is contracting with respect to a contraction metric, then, even if a perturbation $\mathbf{s}(\mathbf{x})$ changes the position of an equilibrium (within some boundaries), a complete system Eq. 2.27 is contracting with respect to the same metric. Therefore, it becomes possible to verify whether the system is stable under a given amplitude of an uncertainty without explicitly tracking how it changes the location of the equilibrium.

A computational challenge crucial to both Lyapunov methods and contraction theory is designing a Lyapunov function or contraction metric for an arbitrary nonlinear system – a nontrivial process as often an analytical solution does not exist. Similar to the design of a Lyapunov function (see Eq. 2.20), the task of estimating a contraction metric might be formulated as a numerical optimization problem in terms of LMIs. LMI-based stability formulations have been first applied to linear systems and a special class of nonlinear systems (Boyd et al., 1994). Recently, Parillo (2003) extends the LMI framework to nonlinear systems for which a dynamical function $\mathbf{f}$ is a polynomial. The author leverages the ability of sum of squares (SoS) programming to efficiently solve LMIs systems. Recently, methods of SoS optimization have been adopted for stability analysis of unknown (Tedrake et al., 2010; Tobenkin et al., 2010) and uncertain (Aylward et al., 2008) systems.

2.2.3 THE CURRENT CHALLENGES

Summarizing our discussion on dynamical system identification and stability analysis, we may pinpoint the following challenges.

- Despite a large number of different methods for nonlinear identification, research in the field is still highly active. One reason for this is a lack of a generic framework: the algorithms are often developed for specific problems and classes of nonlinearities; therefore, their applicability to other types of nonlinearities is not apparent. Furthermore, training of some neural network-based methods is a computationally demanding process.

- A design of a Lyapunov function or contraction metric for stability analysis is a technically challenging process. Newly developed tools of SoS programming greatly simplify the design problem by transforming it into polynomial optimization. However, for these methods to be efficient, a dynamical function should be polynomial or allow for accurate approximation with a polynomial. Such a requirement is not always possible to fulfill, in this case, SoS optimization might render an over-conservative estimate of a Lyapunov function, which is of a limited practical use.

Taking these challenges into account, we suggest an algorithm for non-parametric identification of a motion dynamics. The method is particularly targeted for learning from multidimensional human motion data and able to approximate smooth non-linear low-frequency dynamical functions. The region of attraction of the learned dynamics is analyzed numerically. SoS-based techniques for the region of attraction analysis have been developed in the parallel to our work; furthermore, we also observe that due to a particular non-polynomial parametrization that we use, the SoS methods render conservative estimates.

2.3 THE HUMAN MOTOR CONTROL INSIGHTS INTO MOTION COORDINATION

In Section 2.1.5, we outline some challenges of the analytical robotics related to motion coordination in manipulation tasks. These challenges stimulated roboticists to seek inspiration in human motor control. Recently, mutual influence of robotics and human motor control has increased. There are two reasons for this. First, as robots' morphology tends to get more anthropomorphic, our expectations regarding human-likeliness of their movements are growing. Therefore, roboticists turn to the human motion studies, looking for important insights into computational models of human movements that can be transferable to robots. Second, open questions in robotics stimulated some directions in human motion studies that have not received sufficient attention before. For instance, for long time human motion studies have been mainly devoted to problems of motor control (that is, impedance characteristics of limbs, muscle activations, feedback mechanisms that ensure feedback tracking of preplanned trajectories), and motion planning has been of a secondary interest (Todorov, 1998). It is argued that the scientific community working on human motor control has started to fully appreciate the difficulty of motion planning once the lack of satisfactory robotics solutions for production of coordinated motion has become evident and critical.

In Section 2.3.1, we discuss two currently influential views on motion formation. We first outline the dynamical system view on motion production. Then the dynamical system approach is contrasted to the optimal control view on motion formation. From this discussion, we move to reviewing work on bimanual coordination (Section 3.2) and motion coordination during physical interaction with peers (Section 2.3.3).

2.3.1 MOTION PRODUCTION AND INTRA-LIMB COORDINATION

Similar to the analytical robotics view outlined in Section 2.1, the separation of planning and execution has been a prevalent approach to motion production in human motor control. Recently, open-loop trajectory generation in humans has been questioned. As a result, several generative close-loop models of motion generation have appeared.

It has been suggested that a broad class of arm movements including discrete, such as pointing or grasping, and rhythmic movements, such as swimming or swinging, can be thought of as being generated by an *attractor dynamics*. Specifically, this hypothesis assumes that a trajectory that the arm follows is a particular instantiation of a dynamical system:

$$\dot{x} = f(x) \tag{2.28}$$

where x is a state that describes the arm configuration in the joint or task space; f :

$\mathbb{R}^N \to \mathbb{R}^N$ is a continuously differentiable function. A state $\bar{x}$, such that whenever a trajectory that starts in a neighborhood of $\bar{x}$, converges to $\bar{x}$ and remains there infinitely, is said to be an *attractor* of the system given by Eq. 2.28. For essential references on nonlinear dynamical systems literature see Guckenheimer & Holmes (1993) and Strogatz (1994).

To depart from the open-loop paradigm, the dynamical system approaches emphasize the advantage of removing the explicit time dependency, so that a motion policy becomes *autonomous*, i.e. time-independent. From the human motor control point of view, explicit timing is cumbersome, as it assumes an additional level of complexity; that is, that the CNS maintains a clocking mechanism. It is disputed whether biological systems have access to such a clock at all (Ivry et al., 2002; Roberts & Bell, 2000). Importantly, in robotics applications, removing time-indexing of trajectories through the use of dynamical systems eliminates the need of heuristical mechanisms that scale trajectories if the timing of a motion has been perturbed (e.g., if a target has been moved closer) (Hoffmann, Pastor, et al., 2009; Pastor et al., 2009).

The power of modeling motor control with dynamical systems is further revealed by the ability of dynamical systems to integrate perceptual information into motion production (J. Kelso, 1995). This property is highly desirable in robotics, where it might resolve the challenge of integrating sensory information into state-space motion production.

Despite the obvious advantages that dynamical systems would bring into modeling of human motor control, there are few established computational approaches for generating curved motions (Petreska & Billard, 2009). Early attempts to find dynamical laws subserving human point-to-point movements develop computational models that accounts only for the "quasi-linear" trajectories, "bell-shaped" velocity profile (Bullock & Grossberg, 1988; Flash & Hogan, 1985) and $2/3$rd power law (Vivani & Terzuolo, 1982). These models, however, fall short at explaining reaching motions outside the planar space (Sternad & Schaal, 1999) and at accounting for the curvature of movements reaching for arbitrary points in space (Petreska & Billard, 2009). Recent approaches take a less categorical view and no longer search for a single invariant (Berret et al., 2008): it is ackowledged that the motion laws are task dependent (Admiraal & Kusters, 2004; Kang et al., 2005).

However, the exact form of dynamical control in humans is still undeciphered. Schaal et al. (2007) emphasize two reasons for that. First, modeling with nonlinear dynamical systems is challenging mathematically – optimization approaches, which we will discuss further, are more stereotypical and consequently more accessible to researchers. Second, with few exceptions, see (Bullock & Grossberg, 1988; Petreska & Billard, 2009; Schoner, 1990), dynamical systems approaches have concentrated on periodic behaviors or assumed that a discrete behavior is a part of an aborted limit cycle. However, in a broad perspective, optimization and dynamical system approaches to motion production provide complementary functions and can be combined in a unified framework: through optimization and adaptation of parameters, a single dynamical system can be applied to a whole class of tasks, rather than to one particular task only.

The appeal of optimality principles for human motor control consists in their ability to transform a performance criterion into predictions regarding the behavior of a given system. The adepts of the optimal control view on motion production argue that even though the nonlinear dynamical systems can reproduce relevant behaviors and offer other attractive properties, they do not possess the predictive power inherent to optimal control methods. Specifically, given a new task, one cannot predict a form of dynamical system that governs the task and, therefore, cannot predict task trajectories. However, if one knows an optimization criterion that corresponds to the analyzed task, then theoretically he/she may predict motion trajectories.

For decades researchers have exploited *optimality* concepts to investigate phenomena observed in human movements (Chow & Jacobson, 1971; Flash & Hogan, 1985; Nelson, 1983; J. Rasmussen et al., 2001; Todorov, 2004; Todorov & Jordan, 1998, 2002). The body of related research can be categorized according to the type of the considered control law: open-loop or closed-loop (feedback) control.

Open-loop approaches usually assume a deterministic dynamics of a motion and environment and optimize a trajectory without taking into account the role of online sensory feedback, and (Chow & Jacobson, 1971; Flash & Hogan, 1985; J. Rasmussen et al., 2001). Such methods typically yield an accurate prediction of an average behavior. Limitations of these methods are two-fold: they reduce the whole process of movement production to replication of a single movement and do not explain the variability typical for human movements. Still, there are motions that have been successfully explained from the stance of open-loop optimal control, for instance, movements that are mainly characterized by energy minimization or motion smoothness (jerk minimization). Research on open-loop optimal control of human motions has accumulated sufficient evidence that simple cost functions can illuminate the performance criteria behind different tasks. Still, for each new task one should manually pick a suitable cost function or a combination of these.

The other branch of optimal control methods concentrates on the *closed-loop* control. The methods of this group represent a more elaborated attempt to define computational grounds of human motions: they aim to directly model *sensorimotor integration* (Hoff & Arbib, 1993; Loeb & Levine, 1990; Shimansky et al., 2004).

A closed-loop optimal controller maps actual states of a body or an environment into control signals. This transformation is performed by a feedback mechanism that constitutes an essential part of the closed-loop optimization. In contrast to the open-loop optimization, which relies on a predefined trajectory, an optimal feedback controller lets the task constraints define a motion in real-time.

Many existing optimal feedback controllers aim to optimize a long-term performance as quantified by an optimal cost-to-go function. For every state and point in time, the cost-to-go function estimates how much cost will be accumulated from the current moment until the end of a movement.

As a control signal of an optimal feedback controller depends on an estimate of the current state, the resulting control sequence is optimal only when a state estimate is also optimal (e.g., in the maximum likelihood sense). Therefore, some algorithms build an internal estimate of the state. Specifically, when the sensors are noisy or delayed, a reconstruction of an internal estimate rather than the direct use of raw sensory measurements is necessary. Optimal feedback controllers with in-built optimal state estimators are able to anticipate state changes before corresponding sensory information arrives. For an optimal state estimation, one needs to know a model of body dynamics. Such a model is rarely known; hence the state estimation constitutes one of the challenges in optimal feedback control.

Todorov (2004); Todorov & Jordan (2002) develop a theory of motor coordination using tools of stochastic optimal feedback control. The authors share the same view on motion production as the proponents of the dynamical system view: the CNS does not track a single preplanned trajectory and does not reject all disturbances. In (Todorov & Jordan, 2003), they cast this view into a novel principle of *minimum intervention*; that is, the CNS corrects motion deviations only when they interfere with task performance.

Advancements of feedback optimal control in explaining human movements might be insightful to robotics. They offer an intuition regarding how to computationally model coordinated motions that constitute a part of a task and how to incorporate feedback into trajectory generation. Furthermore, investigation of optimality principles and cost functions can be considered as an attempt to analytically uncover coordination constraints underlying the motor behavior and typical motion signatures of humans. However, the process of choosing an optimality measure for each type of tasks is not yet automatized and requires design effort. This problem is similar to one in analytical motion planning: how to define a proximity metric so as to ensure the generation of a feasible and optimal path (see Section 2.1.1).

2.3.2 A DYNAMICAL SYSTEM VIEW ON BIMANUAL COORDINATION

Dynamical systems have been also used as a tool for modeling bimanual coordination. During last decades, human motor control has been investigating principles underlying the emergence and changes of coordination patterns. The synergetic approach (Haken, 1993) and a first dynamical system model of rhythmic bimanual coordination (Haken et al., 1985) have provided grounds to address these questions.

The dynamical system view of bimanual coordination emphasizes the self-organizing character of movement production. The dynamics of such coordination has been formalized in the Haken-Kelso-Bunz (HKB) model (Haken et al., 1985) – a system of two coupled nonlinear oscillators. This approach assumes that, under certain conditions, basic coordination patterns emerge as the result of coupling between the interacting limbs. The coupling between the limbs can be captured by a low-dimensional order parameter – *collective variable*. In (Haken et al., 1985), the collective variable chosen as the relative phase between limbs facilitates explanation of two preferred patterns of rhythmic coordination: the in-phase mode, i.e., the two limbs are moving symmetrically in opposite directions, and the anti-phase mode, i.e. parallel movements in the same direction.

Analogously to the HKB model of rhythmic bimanual coordination (Haken et al., 1985), Schoner (1990) proposes a dynamical model that captures the temporal synchronization tendencies observed in discrete bimanual coordination, see (J. Kelso et al., 1983). Synchronization consists in a mutual temporal dependency even if the spatial constraints imposed to each of the limbs are different. Schoner (1990) argues that discrete movements represent an aborted case of cyclical movements. He further constructs a model with two states, one for an initial and one for a target posture, that are represented by point attractors. Once an intention to move emerges, the motor system generates a limit cycle attractor that connects the initial and target state. The movement starts when the initial attractor is destabilized due to the intention to move; during the movement, the system follows the limit cycle attractor and eventually stabilizes at the target attractor. Intentions form a crucial part of the Schoner's model and should be modeled explicitly. Coordination is then expressed as a common timing of different limbs.

More recently, in (Jirsa & Kelso, 2005), the authors suggest a unified view on the HKB and Schoner's model. They propose an *excitator model* that accounts for both discrete and rhythmic coordinated movements. The coupled excitators are used to study bimanual movements and to explain their timing.

Besides the computational models that we have mentioned, there exists a body of experimental research that also argues that bimanual coordination in human manipulation relies on shared timing goals assigned to the two arms. That is, the motions of the two arms are generated independently and synchronized by a single timing mechanism that controls emergence of significant spatial events along the motion (e.g. the two arms simultaneously reach an object to be grasped). While assigning the principle role in coordination to the shared timing, these works do not exclude the possibility of coordination at the spatial, trajectory level. However, as of now, there is a lack of computational models that would incorporate spatial coupling of arbitrary discrete bimanual task.

2.3.3 PHYSICAL INTERACTION WITH PEERS

Physical human-human interaction may be observed when people are moving together bulky objects, dancing, or helping each to recover balance. Physical interaction typically entails energy exchange. As we discussed in the previous sections, during autonomous task execution (unless a manipulated object is very heavy), coordination can be characterized as a systematic kinematical behavior. That is, the kinematic variables of a coordinated motion exhibit important correlation patterns. During physical interaction, coordination necessarily extends to correlation between the interaction forces (difference between the forces applied by the peers) and the kinematic or dynamic behavior at each side.

In the literature, we can find multiple attempts to quantify unimanual and bimanual coordination even if the existing models are not necessarily generic. Research on coordination during physical human-human interaction is scarce. The existing studies are mainly experimental and do not suggest computational models. Here, we briefly review some experiments; our objective here is to highlight two aspects important to our work. First, similar to bimanual coordination, motion during physical interaction cannot be reduced to two independent unimanual movements. Second, the movements of the two partners are coupled through haptic information.

In a study designed to understand coordination between human peers, Sallnas & Zhai (2003) examine interaction of two persons manipulating objects in a virtual environment. The task consists in exchanging virtual objects of different sizes without dropping them. The conducted experiments are targeted to explore effects of different types of feedback. The two partners are requested to interact through haptic devices. During a part of the experiments, the force feedback is switched off, so that the partners can monitor each other's actions only visually. The experiments reveal that force feedback decreases the error rate significantly.

Gentry (2005) examines how two partners physically interact during dancing. The author describes dancing as a finite state machine – the dancers move through a sequence of poses and interact through force feedback. Dancers coordinate their actions (the motion pace and spatial details of dance postures) through external and internal cues. The rhythm of the music is an external cue that synchronizes the motion of each dancer's movement. The physical interaction is an internal cue, that allows one partner to send messages to another. Most of the communication is mediated through haptic cues even though the dancers keep the visual contact. Gentry (2005) has also experimented with professional couples dancing blindfolded and has found that they perform well communicating solely through haptics.

Reed & Peshkin (2008); Reed et al. (2007) have conducted studies where human partners have to accomplish a joint task (i.e., move a crank to a specified location). The authors demonstrate that dyads perform the task faster than individuals working alone, even though the partners consider each other to be an impediment.

2.3.4 THE CURRENT CHALLENGES

Summarizing our discussion on human motor control, we may pinpoint the following challenges that have not been completely resolved as of yet.

- Research in human motor control has mainly concentrated on motor control, motion planning still lacks a generic solution. Due to the diversity of manipulation tasks, systematical laboratorial modeling and manual analysis of such movements are problematic.

- Most of the existing approaches to explaining motion planning through dynamical systems aim at explaining linear reaching movements and are not applicable in a general case when a task requires curved trajectories.

- Except of very few work, there are no computational models for discrete bimanual coordination. Furthermore, the existing models explain only temporal synchronization. It is important to build generative algorithms that take into account spatial constraints and predict actual trajectories. Here, the same problem as mentioned in the previous point also arises: the diversity of discrete bimanual tasks makes it difficult to describe them with a single analytical model.

- Motion coordination during physical human-human interaction has not been modeled computationally. The complexity of such modeling follows from the fact that a model describing the motion of one partner requires a mechanism that predicts the motion of his/her peer along the task. However, the lack of well-developed approaches to motion planning logically implies the lack of satisfactory abilities to predict the task's movement.

2.4 ROBOT LEARNING

Motion planning in robots and human motor control discussed in the previous sections attempt to approach the generation of task movements and the resolution of related coordination constraints analytically. In analytical robotics such a perspective results in mathematically rigorous algorithms, that resolve many important problems, but that require a lot of computational resources and designing effort to define environmental and task models. In human motor control, for which the main goal is to understand motions rather than to devise efficient methods for executing them, the pure analytical approach does not go beyond constrained laboratory movements.

Robot learning relaxes the strict requirement of exact analytical solutions in favor of data-driven methods that devise motion policies π (mappings from a world state ξ to a robot action u) from available datasets:

$$u(\xi, t) = \pi(\xi, t) : \quad \pi : \mathbb{R}^{n+1} \rightarrow \mathbb{R}^{k+1} \tag{2.29}$$

where $\xi \in \mathbb{X} \subset \mathbb{R}^n$ and $u \in \mathbb{U} \subset \mathbb{R}^k$ are the chosen state and action. Note, that in this review we are mainly concerned with continuous state and action spaces. A training dataset is a sequence of state-action pairs $\{\xi_i, u_i\}_{i=1}^{i=M}$ acquired from either task demonstration conducted by an experienced teacher (as in methods of Programming by Demonstration), pure self-exploration or a combination of the two (as in methods of Reinforcement Learning). Policies derived under robot learning are most accurate within an observed range of states; outside of the observed values, the reliability of policies gradually decreases. Therefore, robot learning policies differ from analytical algorithms that do not have such a limitation on the area of applicability. To extend the area of applicability, robot learning emphasizes the importance of the *generalization ability* (the ability to extrapolate knowledge to unseen states) of a method. Robot learning also shares research questions with analytical robotics such as: developing *multi-dimensional* motion policies; and including *feedback* into learned models so as to facilitate adaptation to incoming sensory information.

We split this section into several parts as follows. In Section 2.4.1, we give an overview of several regression techniques that are often used in robot learning. In

Section 2.4.2, we describe the state of the art in Programming by Demonstration (PbD) with a focus on learning motion constraints in unimanual and bimanual tasks and learning motion dynamics. In Section 2.4.3, we outline current advances in Reinforcement Learning (RL). Section 2.4.4 explains how robot learning addresses robot control during physical interaction with the environment. Throughout our discussion, we will review how the challenges outlined in Sections 2.1.5 and 2.3.4 can be solved with robot learning tools.

We do not aim to provide a comprehensive overview of the robot learning field. Following the classification of robot learning techniques given in Argall et al. (2009), there are the two principal stages in the learning process: data acquisition and policy derivation. The scope of our discussion is deliberately limited to methods contributing to the second phase of learning – *policy derivation*. Within this scope we concentrate on some recent advances so as to delineate our contribution. We, therefore, exclude from our discussion problems and methods related to the first phase – data acquisition (e.g., we do not review demonstration techniques, correspondence problems (Nehaniv & Dautenhahn, 2002), or methods that aim at overcoming limitations in training sets). An interested reader may refer to more detailed reviews, e.g., by Argall et al. (2009); Billard et al. (2008); Calinon (2009).

2.4.1 REGRESSION TECHNIQUES

Statistical regression techniques are frequently used in robot learning to represent continuous motion policies. We further overview the basic theoretical formulations of several techniques. We will refer a reader interested in detailed information to relevant books. One can also refer to Calinon, D'halluin, et al. (2010) for an experimental comparison of several regression techniques in the context of learning robotic motions.

The generic regression problem, which we are interested in, can be summarized as follows:

$$\begin{aligned}&\text{given an input } \mathbf{x} \in \mathbb{R}^m \text{ and an output value } \mathbf{y} \in \mathbb{R}^n \\ &\text{estimate a regression function } \mathbf{f} : \mathbb{R}^m \rightarrow \mathbb{R}^n \text{ such that: } \mathbf{y} = \mathbf{f}(\mathbf{x}) + \eta\end{aligned} \tag{2.30}$$

where $\eta \sim \mathcal{N}(0, \sigma^2)$ is the Gaussian noise with zero mean and variance σ^2. We will distinguish two classes of regression techniques: (i) methods that learn non-linear functions globally, i.e. by covering the input space with basis functions of a predefined analytical form and/or of a predefined number (we will review Gaussian Mixture Regression (GMR) (McLachlan & Peel, 2000) and Gaussian Process Regression (GPR) (C. Rasmussen & Williams, 2006)), and (ii) methods that fit non-linear functions locally by using spatially localized models and that automatically adjust the number of local models to account for unknown nonlinearities in the target function (we will review Linear Weighted Regression (LWR) (Atkeson et al., 1997) and Locally Weighted Projection Regression (LWPR) (Schaal et al., 1998; Vijayakumar et al., 2005)).

2.4.1.1 GLOBAL REGRESSION TECHNIQUES: GAUSSIAN MIXTURE REGRESSION AND GAUSSIAN PROCESS REGRESSION

Gaussian Mixture Models (GMM) encode the joint distribution $p(\mathbf{x}, \mathbf{y})$ of input $\mathbf{x}$ and output $\mathbf{y}$ through a mixture of Gaussian distributions:

$$p(\mathbf{x}, \mathbf{y}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}, \mathbf{y} | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \tag{2.31}$$

where K is the number of Gaussian components in the mixture, π_k are the mixing coefficients, $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ are the mean and covariance of the kth component respectively. For approximation with GMMs, one needs to choose the number of components K and learn the parameters $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, k = 1..K$. Training is frequently accomplished through the Expectation Maximization procedure (Dempster et al., 1977). Once the GMM in Eq.2.31 is trained, GMR proceeds through estimating the regression function $f(\mathbf{x})$ as the expectation of the conditional distribution $p(\mathbf{y}|\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$:

$$\mathbf{y} = f(\mathbf{x}) = \mathbb{E}[p(\mathbf{y}|\mathbf{x}, \boldsymbol{\mu}, \boldsymbol{\Sigma})] \tag{2.32}$$

Where in Eq.3.22, $\boldsymbol{\mu}, \boldsymbol{\Sigma}$ denote to the whole set of parameters $\{\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}, k = 1..K$.

Gaussian Process Regression directly models the conditional distribution and aims to represent the observed outputs $\mathbf{y}$ with a Gaussian Process:

$$\mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}, K(X, X) + \sigma^2 I), \tag{2.33}$$

where $\boldsymbol{\mu}$ is the mean regression signal that is often chosen to be zero: $\boldsymbol{\mu} = \mathbf{0}$. X denotes the set of all observed inputs $\mathbf{x}$, I is the identity matrix, $K(X, X)$ is the covariance matrix computed using a given covariance function. The most popular general purpose covariance function is the Gaussian kernel:

$$k(\mathbf{x}_p, \mathbf{x}_q) = \sigma_s^2 \exp\left(-\frac{1}{2}(\mathbf{x}_p - \mathbf{x}_q)^T W (\mathbf{x}_p - \mathbf{x}_q)\right) \tag{2.34}$$

where σ_s^2 denotes the signal variance and W represents the width of the Gaussian kernel. For a more detailed analysis of GPR and other types of kernels see C. Rasmussen & Williams (2006). To estimate the regression function on unobserved values, GPR stores the whole training set. For each new input $\mathbf{x}^*$, the corresponding output $\mathbf{y}^*$ is computed as follows:

$$\mathbf{y}^* = f(\mathbf{x}^*) = k^T(\mathbf{x}^*, X)(K(X, X) + \sigma^2 I)^{-1}\mathbf{y} \tag{2.35}$$

where $k^T(\mathbf{x}^*, X)$ is the covariance between the new datapoint $\mathbf{x}^*$ and the training input X.

Note, global regression methods require a priori a determined modeling basis, which might pose problems if no information about the function f is available beforehand. In the case of GMR one has to choose the number of components and the proper initialization (the means and covariances of the Gaussian components), and for GPR one needs to select the right function space (the covariance functions). The important advantage of the global regression methods is that they are well-suited for the approximation of multi-dimensional functions and suffer less from the curse of dimensionality (the sparsity of training data in multi-dimensional spaces). In comparison to GPR, GMR is faster (the inversion of the covariance matrix in Eq. 2.35 is a computationally expensive process) and does not require storing of all the training data. In its turn, GPR has a different attractive property: the method does not depend on the choice of the number of mixture components. GPR also avoids over-generalization, which is possible in GMR. Indeed, for each input datapoint, GMR attempts to predict an output value, even if the input is far from the training dataset and no reliable inference can be made. On the contrary, for inputs far from the training set GPR does not attempt to generalize and returns output values close to the mean signal $\boldsymbol{\mu}$ (usually chosen as zero $\boldsymbol{\mu} = \mathbf{0}$; see Eq. 2.35).

2.4.1.2 LOCAL REGRESSION TECHNIQUES: LOCALLY WEIGHTED PROJECTION REGRESSION

Local methods are applicable when knowledge about the complexity of the underlying function f is limited, i.e. when it is impossible to decide beforehand how many basis functions will be necessary for approximation or which covariance function to choose. However, as these techniques allocate resources in a localized manner, with an increasing number of input dimensions, they encounter an exponential explosion in the number of local models required for accurate approximation. In contrast to global regression techniques, local regression is usually better adapted for incremental learning (i.e., if the range of observed training values is expanding during learning).

Both LWR and LWPR predict an output value by approximation with a combination of K locally weighted linear models. The predicted output $\mathbf{y}^*$ is given by:

$$\mathbf{y}^* = f(\mathbf{x}^*) = \frac{\sum_{k=1}^{M} w_k \bar{\mathbf{y}}_k(\mathbf{x}^*)}{\sum_{k=1}^{M} w_k} \tag{2.36}$$

with $\bar{\mathbf{y}}_k = \bar{\mathbf{x}}_k^T \boldsymbol{\theta}_k$, $\bar{\mathbf{x}}_k = [(\mathbf{x}^* - \boldsymbol{\mu}_k)^T, 1]^T$. Where w_k is the weight of a kth model, $\boldsymbol{\theta}_k$ is the vector of estimated parameters for the model and $\boldsymbol{\mu}_k$ is the center of the kth kernel. The weights w_k determine whether a data point $\mathbf{x}^*$ falls within the region of responsibility of the kth model (receptive field). The region of responsibility is characterized by the Gaussian kernel:

$$w_k = \exp\left(-\frac{1}{2}(\mathbf{x}^* - \boldsymbol{\mu}_k)^T W_k (\mathbf{x}^* - \boldsymbol{\mu}_k)\right) \tag{2.37}$$

where W_k is a distance matrix. During the learning process, both the distance matrices of the receptive fields W_k and the parameters $\boldsymbol{\theta}_k$ are adjusted to minimize error between the observed and predicted output.

LWPR has an advantage over LWR as it overcomes the curse of dimensionality by projecting high-dimensional input data into a subspace of lower dimensionality. One of the major challenges for LWPR is thus a choice of an efficient projection that would enable the best fitting with as few input dimensions as possible. A solution to this problem is suggest by Hoffmann, Schaal, & Vijayakumar (2009); Schaal et al. (1998).

2.4.2 TRAJECTORY MODELING IN PROGRAMMING BY DEMONSTRATION

As we emphasized in Section 2.4, from the whole body of work on Programming by Demonstration, our review concentrates on methods for motion policy derivation. In Programming by Demonstration, most approaches to trajectory modeling are built upon a *time-indexed* representation, either by exploiting the concept of spline decomposition (Aleotti & Caselli, 2006; Ude, 1993) or by explicitly encoding the time-indexed dependencies (Calinon et al., 2007).

Traditional means of encoding trajectories are based on spline decomposition after averaging across training trajectories (Aleotti et al., 2005; R. Andersson, 1989; Hwang et al., 2003; Yamane et al., 2004). Spline decomposition remains a powerful tool for quick trajectory formation. It is, however, heavily dependent on a heuristic for segmenting and aligning the trajectories. Furthermore, a spline representation, not being statistically-based, may have difficulties in coping with data noise inherent to a robotic application.

Non-linear regression techniques are proposed as a statistical alternative to spline-based representation (Calinon et al., 2007; Kulic et al., 2008; Schaal & Atkeson, 1994, 1998). These methods allow the systematic treatment of uncertainty by assuming the existence of noise in the data and, therefore, by estimating actual trajectories as a set of random variables with learned parameters.

These modeling methods encode an explicit time-precedence across the motion states. A motion policy takes the following form:

$$x = \pi(t), \quad \pi : \mathbb{R} \rightarrow \mathbb{R}^k \tag{2.38}$$

where t denotes a time index and x is a trajectory point either in the task or joint space. Following the generic notation of Eq.2.29, a time index t is a state of the policy $\xi = t$, a trajectory x a desired action, $u = x$.

However, similarly to spline-based approaches, most existing regression approaches to motion encoding consider a time-index as an input variable and virtually operate in "open-loop" (i.e. without a mechanism to adapt trajectories to perturbations or delays). The lack of positional feedback makes these methods sensitive to both temporal and spatial perturbations. To compensate for this, one needs to introduce an external mechanism for handling potential deviations from the desired trajectory during reproduction. Adaptation to deviations then relies on a heuristic to re-index the new trajectory in time or extrapolate it in space. Such re-indexing or extrapolation often comes at the cost of significant deviations from the desired velocity and acceleration profile, and, therefore, makes the motion look very different from the original demonstrations. Furthermore, finding a good heuristic is highly task-dependent and becomes particularly non-intuitive in multidimensional spaces (Schaal et al., 2003). Time-independent models, such as autonomous dynamical systems, have been recently advocated as an alternative to time-indexed approaches. Motion models based on dynamical systems are advantageous in that they do not depend on an explicit time-indexing and thus provide a closed-loop controller, while being able to model a broad class of non-linear behaviors.

2.4.2.1 LEARNING DYNAMICAL SYSTEMS FOR MOTION REPRESENTATIONS

Recently, the robot learning community has increased interest in exploiting dynamical systems for encoding observed trajectory data. (Dixon & Khosla, 2004; Hersch et al., 2008; Ijspeert, Nakanishi, & Schaal, 2001; Righetti et al., 2006). In Section 2.2, we reviewed identification methods applied to different engineering problems. Robot learning borrows some of these methods, but also strives to develop new approaches, that are more suitable for extracting nonlinear dynamics from collected motion data.

Earlier approaches to using dynamical systems for motion encoding, for instance (Dixon & Khosla, 2004), suggest a method that fits the parameters of a first-order linear dynamical system to the training data. As linear dynamics are limited in their capacity to produce curved trajectories, the authors model a curved motion as a set of linear dynamical systems and ensure continuity at transition points. Other approaches follow a different methodology to modeling curved motions: they modulate predefined linear dynamics with a non-linear estimate of a trajectory (Hersch et al., 2008) or a velocity profile (Ijspeert, Nakanishi, & Schaal, 2001; Righetti et al., 2006).

For instance, Ijspeert, Nakanishi, Shibata, & Schaal (2001) and Ijspeert et al. (2003) suggest one way to apply dynamical systems for motion production in robotics (similar to how dynamical systems have been applied in human motor control; see Section 2.3). The strength of their method, Dynamical Movements Primitives (DMP), is the ability to build a motion representation from a single demonstration and to ensure the global stability of the representation. However, even though the trajectories generated by DMP are guaranteed to converge to the target, their shape will be considerably deformed, as the method does not generalize accurately. This pitfall is caused by the fact that the algorithm does not eliminate the timing mechanism completely; instead, a phase variable s is introduced; s then controls the evolution of the movement. Their policy takes the following form (we provide a formal comparison between the DMP and our work in Appendix II):

$$\begin{aligned} \ddot{x} &= \pi(x, \dot{x}, s), \quad \dot{s} = -\alpha s \qquad (2.39) \\ \pi(x, \dot{x}, s) &= K_v(-\dot{x} + K_p(x_d - x)) + f(s)(x - x_0) \end{aligned}$$

where K_p and K_v are equivalent to the proportionate and derivative coefficients of the

PD controller, x_0, x_d are the initial and target positions respectively, and $f(s)$ is the modulation function that is learned from task demonstration through Locally Weighted Projection Regression (Vijayakumar et al., 2003) or Linear Weighted Regression (Atkeson et al., 1997). The modulation function $f(s)$ enhances the acceleration profile of a linear PD controller, so as to match the profile of the one contained in the observed data. Following the generic notation of Eq.2.29, a position and velocity of a robot define a state of the policy $\xi = [x, \dot{x}]$, an acceleration is a desired action, $u = \ddot{x}$; additionally, one can consider a phase variable s to be an analog of time, $t = s$.

The flexibility of the method is attained by controlling the phase variable through a linear dynamical system; this way, it becomes possible to control time as any other variable. The original approach of Ijspeert, Nakanishi, & Schaal (2001) contained several drawbacks: a sharp acceleration peak at the motion's onset and a difficulty with generalization if the onset is too spatially close to the target. These drawbacks have been addressed in the follow-ups by Pastor et al. (2009), Hoffmann, Pastor, et al. (2009), and D.-H. Park et al. (2008). Despite the improvements, the performance of DMP is still limited by the presence of the internal timer. That is, DMPs do not encode an actual closed-loop dependency between the motion variables, as it is done in autonomous dynamical systems. To summarize, the drawbacks of modulating the linear dynamics with a learned regression signal are: (1) The resultant encoding is uni-variate, and therefore it discards information about the correlation between degrees of freedom, that may be crucial for faithful reproduction (see Fig. 3.41 for an illustration of the uni-variate encoding problem). (2) Coupling the output of a predefined linear dynamical system with a regression estimate makes the overall system dependent on the temporal synchronization between the two signals, and thus is in effect time-dependent. To handle temporal perturbations, one would need a heuristic to maintain the synchronization. This would, however, no longer guarantee that the overall system is globally asymptotically stable. (3) By ensuring that the stable dynamical system takes precedence over the estimate when coming close to the attractor or after a given time period, one can show global stability of the complete estimate (Ijspeert et al., 2002a). In effect, the global dynamics of motion is increasingly dominated by the stable linear dynamical system, hence leading the motion to progressively depart from the learned dynamics. Alternative way to learn a PD type controller is introduced by Calinon, D'halluin, et al. (2009),Calinon, D'halluin, et al. (2010), and Calinon & Billard (2009). Their method eliminates the time-dependency of a derived policy, but essentially reduces robot's behavior to tracking a single learned trajectory, instead of generalizing a task and generating new trajectories if the context has changed.

The original DMP approach is validated by teaching the humanoid robot DB-2 to reach for a tennis ball (Ijspeert, Nakanishi, & Schaal, 2001). In this experiment, the robot learns a one-arm coordinated movement. Recent extensions of the algorithm address more complicated tasks, for instance in (Ude et al., 2010), the DB-2 learns to throw a ball. Pastor et al. (2009) have the Sarcos Slave Arm pouring water into a cup while adapting to different positions of the cup on a table.

The generic problem of learning auto-regressive dependencies from training data has been investigated by Ghahramani & Roweis (1999) and Roweis & Ghahramani (2001). The authors cast the problem as follows: a hidden state evolves according to an autonomous nonlinear dynamics corrupted by additive noise. The state is not directly observable, but it is known that the outputs nonlinearly relate to the states. The formulation can be formally summarized as follow:

$$\begin{aligned} x_{t+1} &= f(x_t) + w \\ y_t &= g(x_t) + v \end{aligned} \tag{2.40}$$

where $x \in \mathbb{R}^m$ is the state, $y \in \mathbb{R}^n$ is the observable output, w and v are zero-mean Gaussian noise, f and g are differentiable functions. Following the generic notation of Eq.2.29, a system state x maps into a policy state $\xi = x_t$, an observable y_t might be considered as a desired action $u_t = y_t$.

As the states x are not observable, the method first predicts the state x_t from the observation y_t and then learns the actual dynamics $f(x_t)$ and the mapping $g(x_t)$. Simultaneous estimation of the state and two functions, $f(x)$ and $g(x)$, is computationally expensive. Furthermore, the quality of learning crucially depends on the relevance of the initial guess for $g(x)$. In (Ghahramani & Roweis, 1999), the authors exploit the fact that under some assumptions the learning problem can be resolved analytically in closed form. The method is advantageous in applications where the dynamics is known to lie on a sub-manifold of a lower dimensionality.

The problem statement given in Eq.2.40 has been recently investigated by Wang. et al. (2008) andJ. Wang et al. (2006) with the objective of extracting whole-body coordination patterns of walking movements. Their method, Gaussian Process Dynamical Models (GPDM), is based on Gaussian Process Latent Variable Models (GP-LVM) (Lawrence, 2005) (a GP-based nonlinear dimensionality reduction technique). GPDM provides an extension to GP-LVM that enables the accurate representation of temporal dependencies when these are projected into a sub-space and reconstructed back into the original space. Learning whole-body motion dynamics with GPDM is showcased on walking movements that can be reduced to limit-cycle dynamics. The authors especially emphasize that their method can compensate for missing information (e.g., if, due to occlusion, a part of a trajectory where a human raises his leg is lost, the method is able to predict the behavior and, therefore, to reproduce the smooth pace).

2.4.2.2 LEARNING CONSTRAINTS IN UNIMANUAL AND BIMANUAL TASKS

A time-dependent formulation for learning unimanual tasks explored in Calinon et al. (2007), Calinon & Billard (2007a), and Hersch et al. (2008) addresses the problem of learning from multiple demonstrations and *combining constraints* in the task and joint space. The authors take the view that an actual task trajectory is not directly observable because of sensory noise. Furthermore, to extract task constraints, the robot should be provided with more than a single example: the variance across several demonstrations indicates the extent to which one or another part of a motion is constrained. They propose a generic framework that learns a task trajectory by weighting constraints in the task and joint spaces. The chosen statistical framework (Gaussian Mixture Models and Regression) allows the extraction of a compact task representation, convenient for fast reproduction. Calinon, D'halluin, et al. (2009); Calinon et al. (2007) demonstrate how their method learns the variance across several task demonstrations, so as to teach a humanoid robot HOAP-3 basic manipulation motions and task constraints (e.g., object-hand dependency).

The method of (Coates et al., 2008), which similar to the work of (Calinon, D'halluin, et al., 2009) exploits learning from multiple demonstrations, has been recently enhanced by Berg et al. (2010), so as to coordinate two surgery robots to perform rapid stitching.

The work of Calinon et al., as well as other approaches, assumes that the external constraints remain unchanged during demonstration and reproduction. For instance, the shape of a manipulated object imposes constraints on a particular movement's trajectory. If demonstrations are performed with the same object, then the constraints contained in the training set are identical. However, if we increase the complexity of the task and, from demonstration to demonstration, vary the objects' size, the constraints within the training set will vary. The problem of learning from data that contains varying constraints sets a new challenge in robot learning.

In Howard et al. (2010), the authors suggest one possible way to learn from *varying constraints*. They present a method that extracts an unconstrained policy from a set of demonstrations conducted under varying external constraints. The authors explain how learning can be implemented for potential-field based policies (Howard et al., 2008a,b) and for generic parametric policies (Howard et al., 2009a,b). Once learned, an unconstrained policy enables generalization: imposing suitable constraints allows the generation of satisfactory robot behaviors in novel settings. The authors consider hard constraints of the form:

$$A(x)u = 0 \tag{2.41}$$

where $A(x) \in \mathbb{R}^{n \times n}$ is a matrix describing a constraint. Given that $\pi(x)$ is the unconstrained policy, an actual action $u(x)$ that should be chosen by the robot under the constraint A is defined as:

$$u(x) = N(x)\pi(x), \text{ where } N(x) \text{ is the null space of } A \tag{2.42}$$

To clarify the idea behind Eq. 2.42, consider, for instance, the task of grasping a ball in a cluttered workspace (Howard et al., 2009a). The robot should avoid a barrier placed between its hand and the ball. The position of the barrier is varied from demonstration to demonstration, so that trajectories have different curvatures. In this case, averaging the demonstrations will not produce a valid policy. Furthermore, during reproduction, the barrier might be placed in an unobserved location. The authors suggest that, instead of trying to combine inconsistent constraints, the robot should rather uncover a policy $\pi(x)$ that is relevant in the absence of any obstacle. Such an unconstrained policy then can be modified (Eq.2.42) according to the actual barrier's position as defined by the constraints $A(x)$. All actions $\pi(x)$ perpendicular to the hyperplane defined by $N(x)$ will be canceled. The authors advocate that learning unconstrained policies extends the generalization abilities of the robot fundamentally. However, in the current formulation of their method, the constraints are manually developed by a human user.

Policy learning from varying constraints (Howard et al., 2009a) is illustrated in experiments with the humanoid robot ASIMO accomplishing two tasks: to manipulate a ball with the two arms and to clean a panel. The authors show that robot is able to grasp the ball despite the presence obstacles in its workspace, and to clean the panel successfully, whose orientation may vary between cleaning attempts. However, in each of these experiments, the authors engineer constraints by hand, which may be non-trivial for other tasks.

2.4.3 POLICY DERIVATION IN REINFORCEMENT LEARNING

In contrast to Programming by Demonstration that relies on external demonstrations, Reinforcement Learning emphasizes the importance of the robot's self-exploration. The exploration is governed by a reward function, so that during learning a robot aims to find a control sequence that maximizes the reward. We will distinguish two directions within this domain: *forward reinforcement* learning that addresses the problem of design and optimization of a *given* reward function, and *inverse reinforcement learning* that emphasizes the importance of extracting an *unknown* reward function from expert actions.

Even though the algorithms presented in this thesis relate to Programming by Demonstration, we include a brief review of reinforcement learning. We do so to emphasize that the research questions addressed in this thesis (learning coordination, online adaptation to perturbation, sensorimotor coupling) are also challenging when considered from the reinforcement learning perspective.

2.4.3.1 FORWARD REINFORCEMENT LEARNING

The goal of forward reinforcement learning is to maximize either an immediate reward or a reward over time (episodic reinforcement learning). Formally, at any state $x_t \in \mathbb{X}$ an algorithm chooses an action $u_t \in \mathbb{U}$ by drawing it from a stochastic parameterized policy $\pi(x_t, t|\theta)$ with parameters θ. As a result of the taken action, the system transits to a new state x_{t+1} drawn from a state transfer distribution $p(x_{t+1}|x_t, u_t)$. After taking the action, the system yields the reward $r(u_t, x_t) \in \mathbb{R}$. The objective of learning is to optimize the expected return:

$$J(\theta) = \int_{\mathbb{X}} \gamma(x) \int_{\mathbb{U}} \pi(x,t) r(u,x) dx du \tag{2.43}$$

where $\gamma(x)$ is a discount factor. We will consider two groups of forward reinforcement learning: *policy gradient* and *probabilistic policy search* algorithms. The policy gradient algorithms are local methods that optimize Eq.2.43 by gradient descent. Accumulated work on policy gradient learning creates a solid framework for estimation of the gradient from sampled data (Konda & Tsitsiklis, 2000; Sutton et al., 2000). However, even when applied to simple examples with rather few states, some policy gradient methods often turn out to be quite inefficient (Kakade, 2002). One of the recent advancements in policy gradient algorithms is the episodic natural actor critic (Peters & Schaal, 2008b). In essence, the method uses the Fisher Information metric (Amari, 1998) to project the gradient into a more efficient update direction. The episodic natural actor critic demonstrates a significant performance improvements in comparison with earlier gradient-based approaches.

Recently, probabilistic policy search has become an attractive alternative to gradient-based reinforcement learning (Bagnell et al., 2003). Policy search algorithms converge to a solution faster than policy gradient approaches. The performance improvements are achieved at the cost of introducing *a priori* knowledge concerning the form of the reward function. Precisely, to use policy search algorithms, one should make an assumption regarding a class of admissible policies. Learning then consists in discovering a good policy within the chosen class. In high-dimensional domains with continuous states and actions, such as in robotics, policy search is advantageous as it allows for the use of structured policies, integration of experts demonstrations, and fast online learning (Bagnell et al., 2003; Ng. & Jordan, 2000; Peters & Schaal, 2007; Toussaint & Goerick, 2007). At first, policy search algorithms are applied to optimization of immediate reward. The later work of Kober & Peters (2010); Koeber & Peters (2008), Policy Learning by Weighting Exploration with the Returns (PoWER), enables episodic reinforcement learning and, therefore, is well-suited for learning motion policies.

Due to its prohibitive computational cost, till recently, RL has been rarely used in robotics manipulation, where problems often are multidimensional. The development of statistical policy search algorithm such as PoWER (Koeber & Peters, 2008) and advancement in gradient policies has facilitated the use of RL in robotics applications. In (Kober & Peters, 2010; Koeber & Peters, 2008), training with the PoWER allows the WAM arm to learn complex tasks such as *ball in the cup* and *ping-pong*. These tasks require highly accurate motion coordination, therefore even a trained human teacher might not be able to provide satisfactory demonstrations; while self-exploration guided by the PoWER results in efficient motion policies. Among other applications of PoWER, we may emphasize experiments where the WAM robot flips pancakes (Kormushev, Calinon, & Caldwell, 2010) and the iCub robot learns two arm archery skills (Kormushev, Calinon, Saegusa, & Metta, 2010).

While the most existing approaches to probabilistic reinforcement learning are based on Expectation Maximization, Buchli et al. (2010); Theodorou et al. (2010) use a different mathematic machinery. Exploiting concepts from stochastic optimal control and path integrals, the authors derive a novel method, Policy Improvements with Path Integrals (PI^2).Currently, PI^2 is considered as one of the most efficient and easy ways to implement episodic reinforcement learning algorithms. PI^2 is particularly targeted for learning compliant robot locomotion (Theodorou et al., 2010), and can be also applied to multidimensional problems such as manipulation.

It is important to note that the existing Reinforcement Learning approaches to motion learning employ time-indexed representations. Therefore, the resulting policies do not necessarily suit for tackling real-time perturbations in the environment.

2.4.3.2 INVERSE REINFORCEMENT LEARNING

Forward reinforcement learning requires a user to define a reward function for each new task manually. We have already discussed the impediments of this requirement when reviewed work on optimal control for human motions (for instance, the necessity to manually pick up a cost function for each manipulation task). Assuming that during task execution a human produces a desired behavior, one may try to uncover the reward as a function that explains best the observed behavior. *Inverse Reinforcement Learning* (IRL) investigates learning of reward functions from human demonstrations.

A first attempt to extract a reward function has been undertaken in (Abbeel & Ng, 2004). This approach first defines features over the state-space and then assumes that a reward function is a linear combination of these features. Learning is then reduced to tuning of the mixing weights. The objective of optimization is to obtain a reward function that leads to the same behavior as the observed one. This method has resolved successfully several research problems, but it still suffers from some drawbacks. Specifically, Abbeel & Ng (2004) make strong assumptions about the process underlying the observed training data. For instance, in their method, the human is expected to act nearly optimal, so as to provide demonstrations that can be defined by a single statistical model. The IRL problem, the way it is casted in (Abbeel & Ng, 2004), is over-constrained and ill-posed: an infinite number of reward weights make demonstrated trajectories optimal. Ziebart et al. (2008) propose a novel principle built upon the maximum-entropy concept that resolves this ambiguity.

Ratliff et al. (2006) suggest another approach to uncovering a reward function, Maximum Margin Planning. Instead of relying on the stringent assumptions about the training data, the authors introduce a loss function that measures disagreement between a learned and desired policy. Using the loss function to learn a reward function makes the method of (Ratliff et al., 2006) agnostic to the assumption of the optimally of the human behavior. The later work of the same authors (Ratliff et al., 2009) provides important extensions of the original approach; specifically, they develop a non-linear version of the maximum margin planning algorithm that allows learning of a richer class of tasks.

2.4.4 LEARNING FOR PHYSICAL INTERACTION

In the previous sections, we discuss the major theoretical methodologies within robot learning. In this section, we explain how the methods of Programming by Demonstration and Reinforcement Learning are applied to investigate physical interaction between a robot and its environment and address challenges inherent in analytical robotics (see Section 2.1.5). Until recently, one could find only a few works on learning for physical interaction. Currently, this problem is gaining attention, which can be explained by advances in both learning techniques and hardware that now permits robots to operate alongside of humans.

In robot learning, physical interaction emerges in two, principally different, contexts: task demonstration (e.g., kinesthetic teaching) and task execution. Further, we will consider physical interaction during task execution. During task execution, a robot might enter into physical interaction with non-animated objects (e.g., object manipulation or locomotion) or with humans (e.g., collaborative tasks). In this section, we will discuss advances of learning for physical interaction along three directions: (i) learning inverse dynamics (Burdet & Codourey, 1998; Burdet et al., 1998; Nguyen-Tuong & Peters, 2010a; Vijayakumar et al., 2005), (ii) learning the variable impedance (Buchli et al., 2010; Calinon, Sardellitti, & Caldwell, 2010; Ganesh, Albu-Schaffer, et al., 2010; Mitrovic et al., 2008, 2010; B. Yang & Asada, 1996), and (iii) learning for Human-Robot Interaction (Ikemoto et al., 2009; D. Lee et al., 2010; Takeda et al., 2005; Z. Wang et al., 2009).

2.4.4.1 LEARNING INVERSE DYNAMICS

During free-space motions, a robot can be controlled by a high-gain PD controller. However, if a task implies physical interaction, the robot is required to exhibit compliance so as to guarantee safe interaction with the environment. Ideally, if one had an exact dynamic model of the robot, both accurate performance and compliance would be achieved through the inverse dynamics controller. In reality, one almost never has access to the exact dynamic model. Robot learning suggests data-driven ways to address this problem.

We distinguish purely data-driven approaches, which do not assume any structured knowledge about the robot rigid body model (Peters & Schaal, 2008a; Vijayakumar et al., 2005), and methods that employ the standard rigid body model as a baseline, which is further improved through learning (Nguyen-Tuong & Peters, 2010a,b). Vijayakumar et al. (2005) use the LWPR method to learn the inverse dynamics of a Sarcos arm. Learning proceeds in real-time: the robot is tasked to execute a motion, and the model update occurs while the robot is moving.

A more structured approach to estimating the rigid-body model has been considered in adaptive control and identification (An et al., 1988; Arimoto, 1993; Burdet & Codourey, 1998; Burdet et al., 1998; Ganesh, Albu-Schaffer, et al., 2010; W. Li, 1990). Specifically, the rigid-body model of a manipulator is known to be linear in the parameters $\boldsymbol{\theta}$ ($\boldsymbol{\theta}$ are the inertial, friction, and other unknown dynamic parameters):

$$\boldsymbol{\tau} = \boldsymbol{\Phi}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}})\boldsymbol{\theta} \tag{2.44}$$

where $\boldsymbol{\tau}$ denotes to joint torques, $\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}$ are joint angles, velocities, and accelerations of the robot, $\boldsymbol{\Phi}$ is a matrix containing nonlinear functions of joint angles, velocities, and accelerations. In adaptive control (Burdet & Codourey, 1998; Burdet et al., 1998), the dynamic parameters $\boldsymbol{\theta}$ are continuously tuned while the robot is tracking a reference trajectory.

Despite the convincing performance improvements associated with the use of parametric adaptive controllers, such estimation requires a lot of training data. Estimated parameters also need to be checked for physical plausibility and consistency. Furthermore, the linear approximation in Eq. 2.44 neglects more complex (e.g., nonlinear friction) and unmodelled effects in the robot's dynamics.

Depending on a robot and a task at hand, these characteristics can be impediment to accurate performance. Nguyen-Tuong & Peters (2010a,b) suggest to incorporate the analytical rigid-body model into learning inverse dynamics. The authors emphasize improvements in the robot's performance and generalization abilities associated with such *structured* learning. Their method is based on GPR and considers two ways to include the rigid-body model into learning: i) by using the rigid-body dynamics as the mean (μ of a GPR representation in Eq. 2.35), in which case, learning absorbs the errors between the rigid-body control and actual observed data; and ii) by using a special covariance function that incorporates the basis matrix $\mathbf{\Phi}$. The latter is shown to be a more generic and accurate approach.

2.4.4.2 LEARNING VARIABLE IMPEDANCE

Although an accurate inverse dynamics controller allows for both accuracy and compliance, it is often necessary to modulate the compliance of a robot. For instance, if a robot is tasked to serve water and carries a tray full of glass, stiffness is necessary to ensure that no perturbations affect the robot's arm. In such conditions, humans combine feedforward and impedance control (Franklin et al., 2008). Similarly, control over robot's impedance is necessary for stable interaction with external objects and people.

B. Yang & Asada (1996) outline a Reinforcement Learning approach to adjust the robot's impedance during high-speed insertion. The robot is required to track a reference trajectory so as to insert a ball into a hole. However, due to uncertainty in the model of the environment, the hole is not aligned precisely with the trajectory and the ball often collides with a chamfer surface. Full impedance control (including stiffness, damping, and inertia) is therefore necessary to cope with these uncertainties and to prevent the robot from bouncing on the surface during high-speed execution. The authors propose a reward function that simultaneously minimizes interaction forces and deviations from the reference kinematics. To ensure the consistency of the learning process, they suggest a *progressive learning* scheme: Initially, the robot starts with a low-speed execution, where positional discrepancies are dominating. Hence, the robot updates its estimate of the necessary stiffness. Once an optimal stiffness value is found, the robot gradually increases its velocity and updates the damping and inertia parameters. Buchli et al. (2010) explain how the PI^2 algorithm can simultaneously optimize a robot't trajectory and adapt the impedance. The authors use a reward function that represents the trade-off between stiffness and tracking precision.

Calinon, Sardellitti, & Caldwell (2010) adopt a more intuitive view on the problem of variable impedance. They encode demonstrations with GMM and use the learned variance to specify the stiffness of a WAM robotic arm. Specifically, the authors make an assumption that in parts of the task demonstrations that are constrained (low variance), the robot has to follow the learned trajectory while strictly rejecting disturbances (high stiffness). On the contrary, while tracking other, less constrained parts (high variance), the robot is allowed to be compliant (low stiffness). In these, less constraint parts, people can interfere in the task execution without danger of physical damage either for them or for the robot. To validate their approach the authors apply their method to an ironing task where the physical contact with a surface is important.

2.4.4.3 LEARNING FOR PHYSICAL HUMAN-ROBOT INTERACTION

Learning physical Human-Robot Interaction is largely unexplored topic in robot learning. The few existing approaches proceed through learning trajectories from demonstrations and then replaying them on a robot either through a stiff PD controller (Ikemoto et al., 2009; Takeda et al., 2005) or through incorporating them into a hardcoded impedance controller (D. Lee et al., 2010; Z. Wang et al., 2009). Another way to categorize the existing methods is according to whether they use haptic information to decide on a motion trajectory (Ikemoto et al., 2009; Takeda et al., 2005; Z. Wang et al., 2009) or if the choice is solely governed by visual information (e.g., coming from a vision motion capture system) (D. Lee et al., 2010).

Ikemoto et al. (2009) propose an algorithm for teaching a robot to stand up with the help of a human. During interaction, force measurements from the skin sensors are used to adjust the timing of the motion so as to synchronize with the human partner. Takeda et al. (2005) present a robotic dance partner. The robot learns dancing movements by observation and encodes these with an Hidden Markov Models (HMMs). During task execution, force measurements are used to recognize motions of the human partner and choose the appropriate movement sequence for the robot.

The learning framework of D. Lee et al. (2009, 2010) is built upon the *mimesis imitation model* (Inamura et al., 2004; Nakamura et al., 2005). The mimetic communication model for physical interaction is a multilayered framework: at the low level it stores observed motion primitives (of both partners) encoded with the continuous HMMs, at the upper level the framework stores discrete features, so-called interaction patterns. During the task execution, the robot observes actions of the partner and visually recognizes to which motion primitive and interaction pattern his/her actions relate. After the recognition of the interaction primitive, the robot decides on which particular motion primitive to execute. The motion primitive essentially defines a robot's reference trajectory. D. Lee et al. (2010) extends the mimesis communication model to include a hardcoded impedance controller that allows the robot to handle the transition between the non-contact and contact parts of the task. The essential characteristic of this method is that a motion primitive is chosen based on visual information about a partner's motion. Therefore, the robot does not adapt its reference trajectory to incoming haptic signals.

Another HMM-based approach to learning for physical HRI is proposed by Z. Wang et al. (2009). The authors consider a hand-shaking scenario. In this method, the authors encode motion trajectories into an HMM where the hidden variables represent the human impedance. Such an encoding requires the robot to measure human impedance and further recognize which motion model to choose. The motion model is chosen at the onset of the task and governs the robot through the rest of the task without adaptation. In collaborative tasks, the lack of online adaptation can deteriorate performance and prevent the human partner from relying on a robotic assistant confidently.

2.4.5 THE CURRENT CHALLENGES

Summarizing our discussion of robot learning, we may pinpoint the following challenges related to the production of coordinated movements, which have not been completely resolved as of yet and that we aim to address in this manuscript.

- Despite the acknowledged advantages of learning motions as dynamical systems, this problem has not been fully addressed. The existing methods either lack learning of an autoregressive dependency (and, therefore, cannot generalize motions to unseen contexts efficiently) or are computationally complex (e.g., some methods additionally require learning of a low-dimensional sub-manifold that embeds a dynamics) and do not ensure the stability of a learned estimate.
- The current approaches to learning bimanual tasks do not address the problem of automatically extracting the constraints coupling the two arms . Instead, the methods rely on an implicit encoding of the constraints (through learning exact motion trajectories) or design the constraints manually.
- Learning control for continuous physical human-robot interaction is largely an unexplored topic. One of the reasons is the lack of flexible algorithms to learn and predict motion conditioned on the environment and perceived haptic information.

2.5 CONCLUSION

In this chapter, we have provided an overview of how motion coordination has been addressed in analytical robotics, human motor control, and, finally, in robot learning, the domain of our particular interest.

We have outlined two challenging research directions in analytical robotics: how to incorporate kinodynamic constraints into the path planning process, and how to make the robot adapt to uncertainties through feedback planning. Resolution of these two problems would advance robot motion coordination significantly. Precisely, we provided evidence that a planner that could solve kinodynamic constraints would account not only for basic characteristics such as a path length, but for actual constraints related either to a robot's body or to task requirements. We discussed that feedback planning, in its turn, explored how to generate and coordinate movement if the environmental information was only partial or dynamically changing. It has been emphasized that the existing planning methods relied on human analysis and extensive path search, which were the impediments when a robot should accomplish a variety of tasks in real-time. In their pursuit of alternative solutions, roboticists have turned to human motor control from which they have borrowed some models and assumptions. Our work also followed such a *bio-inspired* perspective.

To elucidate the origins of some hypotheses that we exploited in our work, we reported on how motion coordination has been tackled in human motor control and reviewed three research direction: intra-limb motion production, bimanual coordination, and coordination during physical interaction with peers. Our review discussed that the existing literature contained ad-hoc computational models of coordination in motions that were intentionally constrained within laboratory experiments. Still, the proposed models provided important insights regarding possible ways for motion production. We used these insights (e.g., the dynamical nature of human arm movements, sensorimotor integration) to support the dynamical system approach adopted in this manuscript. Furthermore, the lack of a common approach to modeling coordination motivated our search for a generic framework applicable to a broad class of motions (see Chapter 3). The dynamical view on motion production provided a fruitful ground for modeling coordination: it was advocated that dynamical systems could grasp complex forms of coordination, such as bimanual coordination.

We suggested a robotics formulation for bimanual coordination in Section 3.3. In Chapter 4, we extended the dynamical system view of motion coordination so as to explore physical interaction between a robot and a human. Here, we exploited evidence from human motor studies that force feedback governed motion coordination during physical interaction. We integrated haptic information into a dynamical system that generated motion for a robot. Taking into account the considerations of analytical robotics and human motor control, we suggested an alternative, data-driven approach to the encoding and generation of coordinated movements.

In robotics, data-driven methods have been developed in the context of robot learning. We reviewed the state of the art in robot learning, emphasized some problem statements typical to this field, and explained why a robot learning treatment of motion coordination could avoid some of the pitfalls of purely analytical methods. Our work described in Chapters 3 and 4 contributes to this direction of research.

Robot learning solutions often exhibited limited generalization abilities (i.e., the reviewed methods were *not applicable* to the *whole workspace*). However, they were more generic than those analytical (i.e., applicable to a *broad class* of tasks). We discussed the methods for improving generalization that have been suggested under PbD and Reinforcement Learning.

It has been discussed that another challenge of learning coordinated movements related to constraint extraction. Many proposed learning techniques relied on an implicit encoding of constraints, which could lead to poor generalization. To overcome this limitation, in Section 3.2, we suggest a method to explicitly extract bimanual coordination constraints from training data. Finally, in comparison to other domains of robot learning, very little has been done to learn within a field of physical Human-Robot Interaction. In Chapter 4 of this manuscript, we extend the dynamical system learning from Chapter 3.3 and propose a novel approach to teaching a robot to coordinate its movements while physically interacting with a human.

Chapter 3

A DYNAMICAL SYSTEM APPROACH TO MOTION REPRESENTATION AND COORDINATION

3.1 OVERVIEW

LEARNING motion coordination requires a mechanism for encoding and reproduction of coordination patterns and temporal constraints.

In this chapter, we first present our work on learning bimanual coordination. We outline a generic framework that combines Dynamical Systems movement control and Programming by Demonstration (PbD) to teach a robot bimanual coordination tasks. We consider learning of spatio-temporal constraints that couple the two hands to act synchronously. The proposed algorithm consists of two parts: a learning system that processes demonstrated data and extracts spatio-temporal coordination constraints, and a motor system that reproduces the movements in real-time while satisfying the learned constraints. In this algorithm, the motor system exploits strength of the VITE model of human reaching movements to generate trajectories. The proposed model accounts for learning of a sufficiently broad class of manipulation tasks as demonstrated through several robotics experiments.

However, the method for learning bimanual coordination is built upon a simplifying assumption: robotic motions are generated by a pre-defined VITE model. We then explain that though the use of a hard-coded computational model of human motion (VITE) has its advantages (e.g. simplicity (only two free parameters) and applicability in the whole workspace), the hard-coded representation confines the robot to follow linear trajectories. From human motor control (see Section 2.2), we know that many coordinated motions cannot be reduced to solely linear trajectories.

To address this limitation, we further define the problem of learning dynamics of nonlinear motions in the robotic context. We propose an approach for learning locally stable dynamical systems from multiple human demonstrations. The method allows the representation of coordination patterns in a compact analytical form. We provide experimental illustration and validation of the method. The chapter concludes with the extension of the dynamical system motion representation to learning coordination between the position and orientation components of a robot's motion in Cartesian space. Simultaneous learning and reproduction of both motion components in a coordinated manner offers a "pre-shape" motion strategy. Furthermore, it endows the robot with the ability to adapt motion smoothly in the case of perturbations that affect the two motion components either separately or simultaneously.

This chapter is organized as follows:

Section 3.2 presents a method for learning coordination in the case of bimanual tasks. A robot estimates parameters of bimanual constraints using Hidden Markov Models. In this section, we start our exploration of a dynamical system approach to motion coordination: we apply a pre-defined linear dynamical model to generate trajectories for both arms of a robot.

Section 3.3 extends the dynamical system approach to motion coordination followed in Section 3.2. While Section 3.2. assumes that a motion is driven by a known linear dynamical system, in Section 3.3. we present a method for statistical learning of nonlinear dynamical systems and discuss its advantages for both learning and reproduction of nonlinear coordinated motions. We also examine the ability of the proposed algorithm to quickly adapt a motion under external perturbations.

Section 3.4 applies the dynamical system approach presented in Section 3.3 to intra-limb coordination. We explain how dynamical systems can be used for controlling the position and orientation of a robot's hand during manipulation tasks.

Each section contains experimental validation and a discussion of the contributions and the limitations of each method.

3.2 LEARNING ONLINE MOTION GENERATION FOR BIMANUAL TASKS

The method presented in this section has been previously published in:

Gribovskaya E. and Billard, A. G. Combining Dynamical Systems Control and Programming by Demonstration for Teaching Discrete Bimanual Coordination Tasks to a Humanoid Robot. *Proceedings of IEEE/ACM International Conference on Human-Robot Interaction*. 2008.

3.2.1 INTRODUCTION

For complex manipulation tasks, we, as humans, often use both arms to accomplish a task quickly and skillfully. In robotics, investigation of methods for planning and control of bimanual coordination is an important step towards greater autonomy and better performance.

We have emphasized in Chapter 2.3 that, while motion learning in general has been a subject of active research in Programming by Demonstration (PbD), bimanual coordination has received little attention so far. That is, bimanual coordination has not been treated as a constraint *per se*: two-arm manipulation is accomplished through the precise imitation of either a demonstrated trajectories (Zollner et al., 2004) or given object/hand dependencies (Calinon & Billard, 2007b). Such methods demonstrate good performance in a static environment, but fail if we allow dynamical changes in the environment (for instance, if we allow a human user to move objects that the robot is trying to manipulate).

We aim to address some of the limitations of the existing analytical and PbD approaches to bimanual manipulation and propose a method for learning coordination constraints. We also suggest an algorithm for the task reproduction that enforces a robot to adhere to the estimated coordination constraints and that yields robustness towards perturbations.

We take inspiration from bimanual coordination as addressed in human motor control: as discussed in Chapter 2.2, researchers in this field have adopted the coordination dynamics view on the problem. Here, we briefly reiterate some concepts of coordination dynamics that we exploit in the current section.

Coordination dynamics successfully explains and predicts the emergence of coordination patterns in rhythmical movements. According to coordination dynamics, a discrete coordination pattern corresponds to an *attractor* in the state-space of a *collective variable* (a parameter that governs the evolution of a coordinated motion). The motion then consists in a transition from a starting position to this attractor. Therefore, to predict the trajectories of a coordinated task, one needs to choose a suitable collective variable and decide on a dynamical system that governs it.

For clarification, let us consider the Haken-Kelso-Bunz model (HKB) (Haken et al., 1985). In this model, the authors address oscillatory motions of two fingers. The HKB model suggests that the relative angular phase between the fingers is a suitable collective variable for this movements. A dynamical system for the relative phase is then derived from the equations of two coupled oscillators (each oscillator represents one finger). From the previous experimental findings of the same authors it is known that rythmical finger motions exhibit several typical features (e.g., a tendency of the fingers to synchronize, a dynamical switch of a phase between the fingers). The proposed HKB model accords with these observations and successfully predicts trajectories of the fingers.

Taking the stance of coordination dynamics, we demonstrate that, in the case of discrete bimanual motions, the relative position between two hands is a plausible candidate for a collective variable. Stable positions (attractors) of the relative position represent stable *coordinated postures* (coordination patterns), which a robot should attain sequentially to accomplish a task; this concept is illustrated in Fig. 3.1. Specifically, we suggest that a discrete coordinated movement is described by a set of coordination patterns changing each other dynamically as the motion evolves in time. The patterns are presumably task-dependent and, hence, we propose a learning algorithm that enables a robot to automatically extract them from task demonstrations.

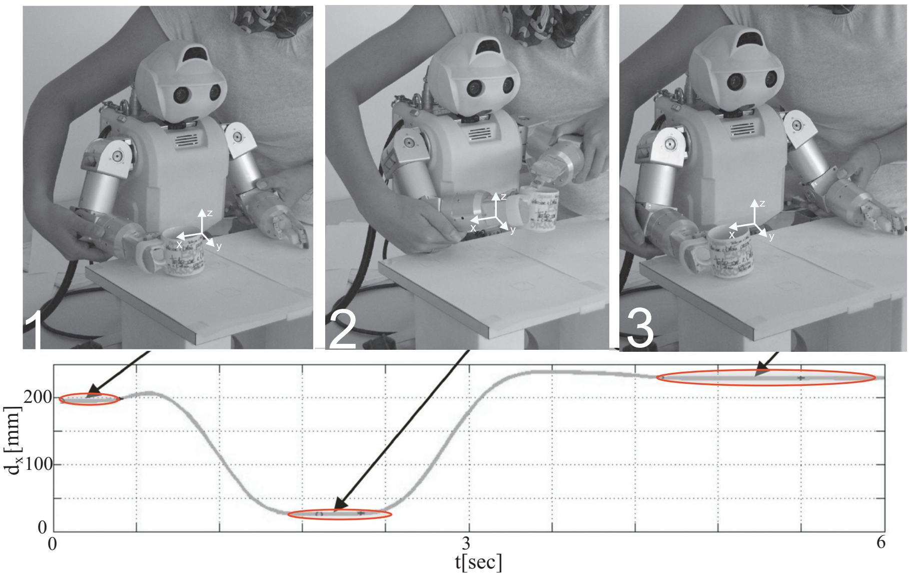


Figure 3.1: The robot is asked to sweeten tea. To do so, the robot moves its arms, so as to put a cube of sugar into a cup, and then brings the arms back on a table (*Tea task*). **Top**: An example of a sequence of three coordinated postures through which the robot transits when performing the *Tea task*. The three postures refer to three events: keeping arms in the initial position, putting the sugar in the cup, and keeping the arm at in the rest position. **Bottom:** The time series of the relative trajectory between the two hands (along the x axis). The superimposed arrows match stable postures (highlighted by ellipses) to the states illustrated on the top figure.

3.2.1.1 SET-UP

The experiments reported in this section are conducted using a humanoid robot HOAP-3 from Fujitsu with 28 degrees of freedom (DOFs), four DOFs per each arm; see Fig. 3.2. Here, we are interested in arms' movements (8 DOFs total); all other DOFs are set so as to keep the robot in the seated position; Fig. 3.2.

The manipulated objects are marked with color patches. During learning and reproduction, these patches are tracked by an external stereovision system.

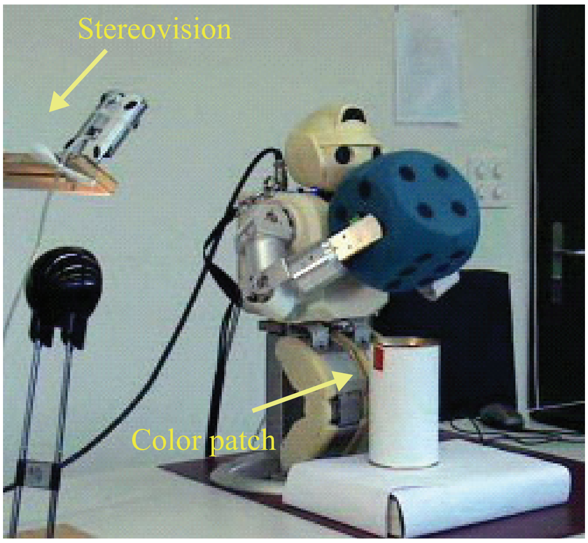


Figure 3.2: A hardware set-up. The humanoid robot HOAP-3 that we use in the experiments on bimanual coordination. The robot can freely manipulate objects in the space in front of its torso. The manipulated objects are marked with color patches and tracked by an external stereovision system.

3.2.2 METHOD OVERVIEW

The suggested model consists of two systems; see Fig.3.3: (1) a *learning system* that extracts and learns task constraints and (2) a *motor system* that dynamically generates movements from the learned task model.

In our experiments, demonstrations are provided through kinesthetic teaching: the robot motors are set into a passive mode and a human teacher guides the robot's arms through a task. Kinesthetic teaching exempt us from a problem of finding a correspondence between motions of a teacher and a robot. Furthermore, a robot cannot move its limbs into all the configuration available to humans. When the teacher directly moves the robot's arms, he/she perceives the robot's limitations, and properly adjusts the motion according to the robot's geometry.

3.2.2.1 THE LEARNING SYSTEM

During learning a robot builds a task model by observing several demonstrations.

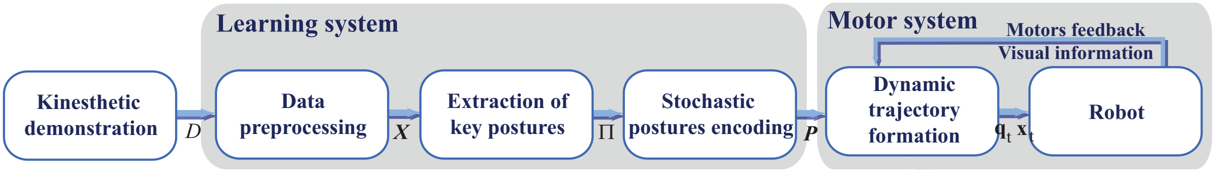

Learning system
Motor system
Motors feedback
Visual information
Kinesthetic demonstration
D
Data preprocessing
X
Extraction of key postures
Π
Stochastic postures encoding
P
Dynamic trajectory formation
q_t x_t
Robot

Figure 3.3: A model overview. The arrows show an information flow across the system. A training set $\mathcal{D}$ is preprocessed by resampling and aligning the demonstrations. From a preprocessed dataset X, we then extract a set Π of stable postures. The set Π is further encoded in a HMM. After learning, a robot use a generalized set of spatio-temporal constraints P to reproduce a task. The robot's motion between and within stable postures is generated in real-time by a dynamical system controller.

1. *Training data*

   A training set $\mathcal{D}$ of each task consists of M demonstrated trajectories of a length N^k, $k = 1..M$. Each trajectory is a sequence of joint angles of the right and left arms $\boldsymbol{q}_t^{R,k}$, $\boldsymbol{q}_t^{L,k}$, $t = 1..N^k$, where $\boldsymbol{q}_t^{R,k}, \boldsymbol{q}_t^{L,k} \in \mathbb{R}^{N_q}$ (N_q is a dimensionality of a joint space; $N_q = 4$ in our case. Here and further the upper indices "^{R}" and "^{L}" refer to the right and the left arm respectively).

   The recorded training set $\mathcal{D}$ is smoothed with the one-dimensional Gaussian filter and resampled to a fixed length N_u.

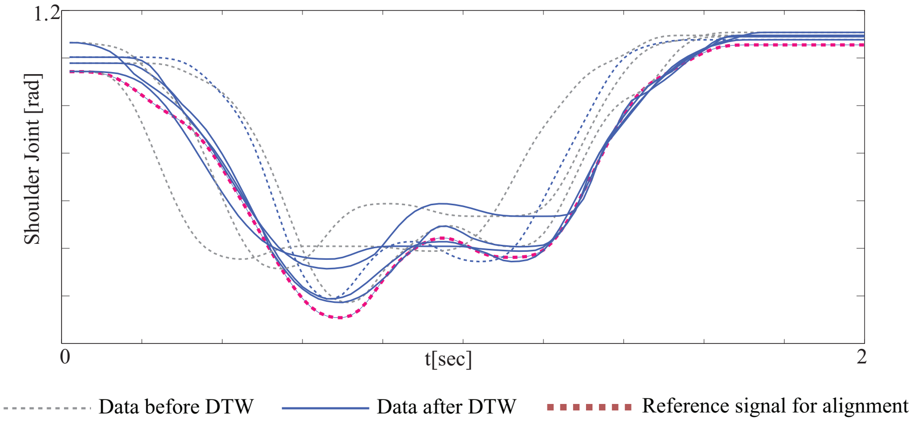


Figure 3.4: Dynamical Time Warping (DTW). Results of applying DTW to five trajectories of the shoulder's joint recorded during the demonstration of the Cube task. Note that, before DTW, local minima and maxima of the recorded signals are strongly misaligned. DTW helps to harmonize data in the time domain.

During experimentation, we observe that the demonstrations are often temporally misaligned. This is due to variability of human movements: a human teacher cannot ensure a stable pace across trials. Therefore, extracting constraints from a set of such suboptimal trajectories requires a more sophisticated approach than merely averaging constraints extracted from each raw trajectory. If not reduced prior training, the misalignments blur temporal and spatial constraints of a task. Here, we align raw data with Dynamic Time Warping (DTW) (Sakoe & Chiba, 1978); see Fig. 3.4. DTW performs nonlinear transformation of the time axis of a signal so as to align it with the time axis of a reference signal. Let us consider two trajectories $\boldsymbol{x}^0$ and $\boldsymbol{x}$ of the same length T, assuming further that $\boldsymbol{x}^0$ is the reference trajectory.

Next, we construct a warping path F, which matches pairs of points of the trajectories $\boldsymbol{x}^0$ and $\boldsymbol{x}$: $F = \{c_k\},\ k = 1..T$ where $c_k = (i_k; j_k)$, i_k and j_k are time indices of the points to be matched $\boldsymbol{x}_i^0$ and $\boldsymbol{x}_j$. The optimal path F is estimated through dynamic programming by minimizing the distance between the trajectories $I(\boldsymbol{x}^0, \boldsymbol{x}, F)$ defined as follows (Sakoe & Chiba, 1978):

$$\begin{gathered} I(\boldsymbol{x}^0, \boldsymbol{x}, F) \sim \sum_{k=1}^{T} \gamma(c_k) \qquad (3.1) \\ F = \arg\min_F I(\boldsymbol{x}^0, \boldsymbol{x}, F) \end{gathered}$$

where $\gamma(c_k) = \|\boldsymbol{x}_i^0 - \mathbf{x}_j\|$ is the distance between matching points. The optimization in Eq. 3.1 is performed under several constraints that control the computational cost and the flexibility of the warping path F. For our data, the slope constraint p, controlling the flexibility of the warping path, is an important parameter: the smaller p the less flexible warping path. We set the slope constraint experimentally as $p = 2$, which in average correspond to the decrease in the optimized function Eq. 3.1 by approximately 40 percent of its initial value. Larger values of the slope constraint cause significant distortion of the data; see the original reference by Sakoe & Chiba (1978) for detailed information on the parameter setting.

After pre-processing of the training set $\mathcal{D}$, the end-effectors trajectories $\boldsymbol{x}_t^R, \boldsymbol{x}_t^L \in \mathbb{R}^3$ and the relative position between the two arms $\boldsymbol{d}_t = \boldsymbol{x}_t^R - \boldsymbol{x}_t^L$ are calculated through the direct kinematics.

2. *Key postures extraction*

We consider a coordinated bimanual motion as a dynamical transition across a set of stable postures. To automatically extract these postures from the data set $X = \{\boldsymbol{x}_t^{R,k}, \boldsymbol{d}_t^k\}_{t=1..N_u}^{k=1..M}$[1] ($N_u$ is a unified length of the demonstrated trajectories after resampling), we apply the Mean Square Velocity (MSV) analysis to each observed trajectory of the relative position $\{\mathbf{d}_t^k\}_{t=1..N_u}$, $k = 1..M$. Next, we outline the details of our MSV analysis.

Lieberman & Breazeal (2004) suggest a method to automatically segment trajectories of complex movements into episodes. The authors start from a segmentation method developed by Mataric (2000), which is based on the following observation: when humans execute a complex action, they typically change the direction and speed of a motion between each segment of the action. To extract motion episodes, Mataric (2000) is looking for changes in the velocity profile

---

[1] To unambiguously generate trajectories for the two arms, we need to consider, in addition to the relative position $\boldsymbol{d}$, a position of one of the arms. Here, we choose to use the motion data of the right arm.

of a motion. However, in her algorithm, at least two parameters (lower and upper bounds for the mean square velocity variation) should be chosen manually for each motion to be segmented. To overcome this drawback, Lieberman & Breazeal (2004) propose a way to compute these parameters directly from motion data.

Here, we extend their algorithm so as to extract stable states from the relative position between the hands $\boldsymbol{d}$. (Following our hypothesis, each stable state of the relative position $\boldsymbol{d}$ defines a coordination pattern between the two arms.) The motion data are noisy, and therefore, the stability is considered in a loose sense and corresponds to a state where $\boldsymbol{d}$ remains approximately constant during a certain time interval and its velocity $\dot{\boldsymbol{d}}$ significantly decreases or drops to zero [2]; see Fig.3.1.

We define the Mean Square Velocity (MSV) function V for each demonstration as follows:

$$V_t = \dot{\boldsymbol{d}}^2_{x,t} + \dot{\boldsymbol{d}}^2_{y,t} + \dot{\boldsymbol{d}}^2_{z,t}, t = 1..N_u, \tag{3.2}$$

And a threshold for segmentation:

$$V^* = \langle V \rangle - 0.5\sigma_V; \tag{3.3}$$

where $\langle V \rangle$ and σ_V are respectively the mean and the standard deviation of V.

To find episodes' boundaries, the algorithm proceeds as follows. For each time t, if $V^k_{t-1} < V^*$ and $V^k_t \geq V^*$, search through the remaining times until $V^k_{t^*} < V^*$ is found. Then the time stamp t is assigned to be the beginning of the i episode, $t_{i,\text{start}}$, and t^* is the end of this episode,$t_{i,\text{end}}$. We consider that the mean value of $\boldsymbol{d}$ on the interval $[t_{i,\text{start}}, t_{i,\text{end}}]$ describes the key posture $\boldsymbol{d}_i$.

To specify the key postures of the right arm $\hat{\boldsymbol{x}}^R_i$, we set up a correspondence between them and the key postures of the relative position $\boldsymbol{d}_i$. As a result the key postures of the right arm take the following values: $\boldsymbol{x}^R_{i,\text{start}} = \boldsymbol{x}^R_{t_{i,\text{start}}}, \boldsymbol{x}^R_{i,\text{end}} = \boldsymbol{x}^R_{t_{i,\text{end}}}, i = 1..N_\Pi$ (N_Π is a number of extracted postures).

The dataset used for stochastic posture encoding is as follows:

$$\Pi = \{\boldsymbol{d}_i, \boldsymbol{x}^R_{i,\text{start}}, t_{i,\text{start}}, T_i, \;\; i = 1..N_\Pi\}. \tag{3.4}$$

Here $T_i = t_{i,\text{end}} - t_{i,\text{start}}$.

3. *Stochastic postures encoding*

   We have several motivations to encode the extracted postures Π with HMMs. These are: (1) to get rid of the spurious postures (the postures that are not relevant for reproduction and are caused by accidental deceleration of the teacher); see Fig.3.7 [3]; (2) to merge postures that are close spatially and temporally; (3) to extract spatial and temporal characteristics of the postures.

---

[2] In practice, we usually observe the velocity $\dot{\boldsymbol{x}}$ decreasing below a certain threshold.

[3] Spurious postures are the postures that do not contribute into task execution and appear in a training

We encode the training set Π with a continuous HMMs. The most generic approach is to use a *fully-connected* HMM model. However, learning of this model requires a large training set, which is almost never available in the context of PbD. Hence, we incorporate prior knowledge about the data structure through biased transitional probabilities. We use a *left-right* model, where no self-transitions are not-allowed. Indeed, the states in our model represent temporally ordered trajectory events, thus the assumption of using the left-right model does not bring any limitations. Each emission probability is approximated with a single multivariate Gaussian distribution and represents a key posture $\boldsymbol{d}_i$, with its time properties $t_{i,\text{start}}, T_i$, and a corresponding position of the right arm $\boldsymbol{x}^R_{i,\text{start}}$.

We apply the Baum-Welsh algorithm (Rabiner, 1989), which estimates the HMMs parameters through the *local* expectation maximization. Because of the local optimization, a proper initialization of the parameters of HMMs plays a crucial role in convergence of training. If we initialize the parameters randomly, the algorithm most likely will converge to a suboptimal solution. The K-means method (MacQueen, 1967) is regularly used for HMMs initialization. However, this algorithm assumes that a number of clusters (hidden states in HMM) is known in advance. It appears logical to choose the number of clusters to be equal to a number of extracted stable postures N_Π. However, due to noise in training data, the number of extracted stable postures varies from one demonstration to another.

Further, we explain a criterion that estimates a number of cluster N_P. In literature, one can find different methods to validate clusters for K-means (Hubert & Arabie, 1985). Usually, such methods are formulated as a criterion (or an index) that characterizes how well a discovered partition explains data and how reliable it is (e.g., whether it is sensitive to outliers). Here, we extend the Dunn index (Bezdek & Pal, 1998) and use the extended criterion to validate the number of clusters within training data. Our criterion aims to maximize *between-cluster* distances while minimizing *within-cluster* distances and a number of clusters c; see Fig. 3.5. The proposed criterion reads as follows:

$$\Gamma(U) = \frac{\max_{1\leq i,j\leq c} \delta(C_i, C_j)}{c \max_{1\leq l\leq c} \Delta(C_l)}; \tag{3.5}$$

where U is a current partition, consisting of a set of clusters $\{C_i, i = 1..c\}$, $\delta(C_i, C_j) = \min_{s\in C_i, g\in C_j}(\|s - g\|)$ is the *between-cluster* distance, $\Delta(C_i) = \max_{s,p\in C_i}(\|s - p\|)$ is the *within-cluster* distance. We analyze possible partitions by iteratively increasing the number of clusters c starting from $c = 2$. The maximum value of the criterion in Eq.3.5 points out to the optimal partition. After encoding of an extracted set of postures Π with HMM and generating a se-

---

set either due to the noise in robot sensing or due to flaws in human demonstrations. Such postures emerge sporadically and are not represented in all demonstrations. Therefore, the probability to transit through them is comparatively small. To prevent a robot from reproducing these postures, we fix the threshold for the HMM transitional probabilities to be 0.2; that is, the postures with lower transition probabilities are

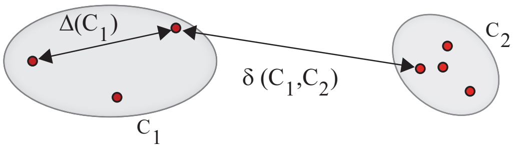


Figure 3.5: *Within-cluster* ΔC_1 and *between-cluster* $\delta(C_1, C_2)$ distances used to compute the Dunn index in Eq.3.5

quence of the most probable postures, we obtain a learned set of spatio-temporal constraints P:

$$P = \{\hat{\boldsymbol{d}}_i, \hat{\boldsymbol{x}}^R_{i,\text{start}}, \hat{\boldsymbol{x}}^R_{i,\text{end}}, \hat{t}_{i,\text{start}}, \hat{T}_i; i = 1..N_P\} \tag{3.6}$$

where $\hat{\boldsymbol{d}}_i, \hat{\boldsymbol{x}}^R_{i,\text{start}}, \hat{\boldsymbol{x}}^R_{i,\text{end}}$ are learned values of the relative position and positions of the right arm in the beginning and in the end of a ith stable posture; $\hat{t}_{i,\text{start}}, \hat{T}_i$ are the time of emergence and the duration of a ith posture.

3.2.2.2 THE MOTOR SYSTEM: TASK REPRODUCTION

Next, we explain how a generalized set of postures Π learned with the HMMs is used for the task reproduction; see Fig.3.3.

1. *Hybrid controller*

   In the previous work of ours, Hersch & Billard (2008) propose a hybrid controller of reaching movements in humanoid robots. The controller is based on the Grossberg's model of human reaching movements – *Vector Integration to Endpoint* (VITE) (Bullock & Grossberg, 1988), and follows a current trend in human motor control. That is, movements are not planned in a single frame of reference, but rather several frames are involved in planning. For instance, one can think of a motion being planned in an internal referential (in a joint space) or in an external referential, e.g., linked to a manipulated object (in the Cartesian space). The planning under multiple referentials introduces redundancy and raises a question of how to combine motions generated in different referentials. Hersch & Billard (2008) show that the redundancy can be used to efficiently avoid joint limits. We use their controller as a basis for the proposed bimanual motor system; see Fig. 3.6.

   At each time step, the desired trajectories $\boldsymbol{q}^R_d, \boldsymbol{q}^L_d$ and $\boldsymbol{x}^R_d, \boldsymbol{x}^L_d$ are generated by VITE controllers (we present only the equations for the right arm, the equations for the left arm are identical):

$$\ddot{\boldsymbol{q}}^R_d = \dot{\boldsymbol{q}}^R + \boldsymbol{\alpha}^R_{\boldsymbol{q}}(-\dot{\boldsymbol{q}}^R + \boldsymbol{\beta}^R_{\boldsymbol{q}}(\boldsymbol{q}^R_{\text{tar},i} - \boldsymbol{q}^R)); \tag{3.7}$$

$$\ddot{\boldsymbol{x}}^R_d = \dot{\boldsymbol{x}}^R + \boldsymbol{\alpha}^R_{\boldsymbol{x}}(-\dot{\boldsymbol{x}}^R + \boldsymbol{\beta}^R_{\boldsymbol{x}}(\boldsymbol{x}^R_{\text{tar},i} - \boldsymbol{x}^R)); \tag{3.8}$$

---

discarded.

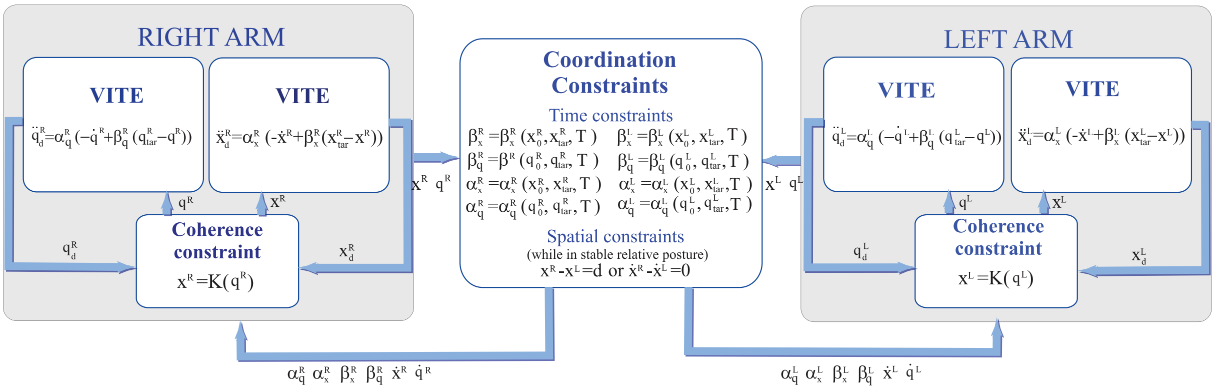

RIGHT ARM
VITE
$\ddot{q}^R_d=\alpha^R_q(-\dot{q}^R+\beta^R_q(q^R_{tar}-q^R))$
VITE
$\ddot{x}^R_d=\alpha^R_x(-\dot{x}^R+\beta^R_x(x^R_{tar}-x^R))$
q^R
x^R
q^R_d
Coherence constraint
$x^R=K(q^R)$
x^R_d
x^R q^R
Coordination Constraints
Time constraints
$\beta^R_x=\beta^R_x(x^R_0,x^R_{tar},T)$
$\beta^L_x=\beta^L_x(x^L_0,x^L_{tar},T)$
$\beta^R_q=\beta^R(q^R_0,q^R_{tar},T)$
$\beta^L_q=\beta^L_q(q^L_0,q^L_{tar},T)$
$\alpha^R_x=\alpha^R_x(x^R_0,x^R_{tar},T)$
$\alpha^L_x=\alpha^L_x(x^L_0,x^L_{tar},T)$
$\alpha^R_q=\alpha^R_q(q^R_0,q^R_{tar},T)$
$\alpha^L_q=\alpha^L_q(q^L_0,q^L_{tar},T)$
Spatial constraints
(while in stable relative posture)
$x^R-x^L=d$ or $\dot{x}^R-\dot{x}^L=0$
x^L q^L
LEFT ARM
VITE
$\ddot{q}^L_d=\alpha^L_q(-\dot{q}^L+\beta^L_q(q^L_{tar}-q^L))$
VITE
$\ddot{x}^L_d=\alpha^L_x(-\dot{x}^L+\beta^L_x(x^L_{tar}-x^L))$
q^L
x^L
q^L_d
Coherence constraint
$x^L=K(q^L)$
x^L_d
α^R_q α^R_x β^R_x β^R_q $\dot{x}^R$ $\dot{q}^R$
α^L_q α^L_x β^L_x β^L_q $\dot{x}^L$ $\dot{q}^L$

Figure 3.6: An overview of the robot's *Motor System.* Each arm is controlled by a couple of dynamical controllers given by Eq.3.7-3.8. Within each arm, the controllers are coupled through robot's body constraints; Eq. 3.11. The coordination between the arms is ensured by spatial and temporal constraints; Eq.3.14, 3.17, 3.19.

where $\boldsymbol{\alpha}^R_{\mathbf{x}}, \boldsymbol{\alpha}^R_{\mathbf{q}}, \boldsymbol{\beta}^R_{\mathbf{x}}, \boldsymbol{\beta}^R_{\mathbf{q}}$ are empirical constants; we will discuss a way to compute them in a section on temporal constraints.

In the beginning of each motion segment i, a target Cartesian position of the right $\boldsymbol{x}^R_{\text{tar},i}$ and left $\boldsymbol{x}^L_{\text{tar},i}$ arms are set as follows: $\boldsymbol{x}^R_{\text{tar},i} = \hat{\boldsymbol{x}}^R_{\text{end},i}$, $\boldsymbol{x}^L_{i,\text{tar}} = \hat{\boldsymbol{x}}^R_{i,\text{end}} - \hat{\boldsymbol{d}}_i$. In the end of the ith segment the target positions are set so as to lead a robot to a next stable posture: $\boldsymbol{x}^R_{\text{tar},i} = \hat{\boldsymbol{x}}^R_{\text{start},i+1}$, $\boldsymbol{x}^L_{i,\text{tar}} = \hat{\boldsymbol{x}}^R_{i+1,\text{start}} - \hat{\boldsymbol{d}}_{i+1}$. The target joint angles $\boldsymbol{q}^R_{\text{tar},i}, \boldsymbol{q}^L_{\text{tar},i}$ are computed from the target Cartesian positions.

Generally, a desired arm configuration $\boldsymbol{q}_d$ might be incompatible with a desired end-effector position $\boldsymbol{x}$ (violation of robot's body constraints) or desired positions of the two arms do not satisfy a spatial coordination constraint. We assume that one can find positions consistent with the constraints in a neighborhood of the desired positions.

We consider estimation of a consistent pair $\{\boldsymbol{x},\ \boldsymbol{q}\}$ as a constrained optimization problem with a functional to be minimized:

$$\mathbf{H}(\boldsymbol{q}^R,\boldsymbol{q}^L,\boldsymbol{x}^R,\boldsymbol{x}^L) = (\boldsymbol{q}^R - \boldsymbol{q}^R_d)^T \boldsymbol{W}^R_q (\boldsymbol{q}^R - \boldsymbol{q}^R_d) + (\boldsymbol{x}^R - \boldsymbol{x}^R_d)^T \boldsymbol{W}^R_x (\boldsymbol{x}^R - \boldsymbol{x}^R_d) +$$
$$(\boldsymbol{q}^L - \boldsymbol{q}^L_d)^T \boldsymbol{W}^L_q (\boldsymbol{q}^L - \boldsymbol{q}^L_d) + (\boldsymbol{x}^L - \boldsymbol{x}^L_d)^T \boldsymbol{W}^L_x (\boldsymbol{x}^L - \boldsymbol{x}^L_d); \tag{3.9}$$

where $\boldsymbol{W}^R_{\mathbf{q}}, \boldsymbol{W}^L_{\mathbf{q}}, \boldsymbol{W}^R_{\mathbf{x}}, \boldsymbol{W}^L_{\mathbf{x}}$ are the positive diagonal matrices, that control the influence of each of the controllers in Eq. 3.7-3.8 [4]. The optimization problem takes then the following form:

$$\min_{\boldsymbol{q}^R,\boldsymbol{q}^L,\boldsymbol{x}^R,\boldsymbol{x}^L} \mathbf{H}(\boldsymbol{q}^R,\boldsymbol{q}^L,\boldsymbol{x}^R,\boldsymbol{x}^L) \tag{3.10}$$

2. *Enforcing robot's body constraints*

In this section, we explain how one can resolve the robot's body constraints:

---

[4] "T" refers to the transpose operator.

$\boldsymbol{x}_d = K(\boldsymbol{q}_d)$, where $K(\boldsymbol{q}_d)$ is the kinematic function of an arm.

To ensure the consistency between the Cartesian and joint trajectories, we solve the optimization problem in Eq. 3.10 under the robot body constraints (note, we consider the robot's body constraints in the differential form):

$$\dot{\boldsymbol{x}}^R = \boldsymbol{J}^R \dot{\boldsymbol{q}}^R; \;\; \dot{\boldsymbol{x}}^L = \boldsymbol{J}^L \dot{\boldsymbol{q}}^L; \tag{3.11}$$

where $\boldsymbol{J}^R, \boldsymbol{J}^L$ are the Jacobians of the right and the left arm. The problem in Eq.3.10-3.11 is resolved at each time step. An analytical solution can be found using the Lagrange multipliers (see Appendix I for derivation).

After optimization, we obtain values $\Delta\boldsymbol{q}^R$, $\Delta\boldsymbol{q}^L$ that update the joint angles commands to be sent to the robot.

$$\begin{aligned}\Delta\boldsymbol{q}^R = (&\boldsymbol{W}_{\boldsymbol{q}}^R + (\boldsymbol{J}^R)^T \boldsymbol{W}_{\boldsymbol{x}}^R \boldsymbol{J}^R)^{-1}((\boldsymbol{J}^R)^T \boldsymbol{J}_{\boldsymbol{x}}^R(\boldsymbol{x}_d^R - \boldsymbol{x}^R) \\ &+ \boldsymbol{W}_{\boldsymbol{q}}^R(\boldsymbol{q}_d^R - \boldsymbol{q}^R)).\end{aligned} \tag{3.12}$$

The updated joint commands $\boldsymbol{q}^R + \Delta\boldsymbol{q}^R$ and $\boldsymbol{q}^L + \Delta\boldsymbol{q}^L$ are guaranteed to be coherent with the robot's geometry; these commands also bring robot's hands in the Cartesian position that are close to the desired values.

Note that the reproduction of joint trajectories calculated through Eq.3.12 does not necessarily lead to the desired Cartesian trajectories. The optimization in Eq.3.10 searches for a trade-off between Cartesian and joint space control. Therefore, the resulting motion depends on the weight matrices $\boldsymbol{W}_{\boldsymbol{q}}$ and $\boldsymbol{W}_{\boldsymbol{x}}$. For instance, if the Cartesian weights $\boldsymbol{W}_{\boldsymbol{x}}$ take precedence over the joint weights $\boldsymbol{W}_{\boldsymbol{q}}$, the robot tracks the generated Cartesian trajectory. The choice of weights and associated redundancy are caused by motion planning in multiple referentials. This redundancy can be exploited to avoid joint limits (Hersch & Billard, 2008). If the joint angle space is convex, the joint angle controller given by Eq. 3.7 will never bring a robot to a joint boundary (unless a target lies on the boundary). Therefore, to avoid joint limits, the robot might gradually move from Cartesian control to joint space control when approaching the workspace boundaries. Such an adaptation is done by varying the weight matrices $\boldsymbol{W}_{\boldsymbol{q}}^R, \boldsymbol{W}_{\boldsymbol{q}}^L, \boldsymbol{W}_{\boldsymbol{x}}^R, \boldsymbol{W}_{\boldsymbol{x}}^L$; see (Hersch & Billard, 2008). For instance, by setting $\boldsymbol{W}_{\boldsymbol{x}}^R$ to zero, one obtains a pure joint angle controller for the right arm, while, by setting $\boldsymbol{W}_{\boldsymbol{q}}^R$ to zero, the result is a pure end-effector position controller.

As the right arm gets closer to one of the joint limits, the corresponding element $w_{q_i}^R$ of the matrix $\boldsymbol{W}_{\boldsymbol{q}}^R$ gets bigger. Eventually, a ratio $\frac{w_x^R}{w_{q_i}^R}$ tends to zero, which leads to a pure joint angle controller and therefore allows avoiding the joint limit. To achieve this, Hersch & Billard (2008) suggest to define the ratio

$\frac{w_x^R}{w_{q_i}^R}$ as follows:

$$\frac{w_x^R}{w_{q_i}^R} = 0.5\gamma(1 - \cos(2\pi \frac{q_i^R - q_{i,min}^R}{q_{i,max}^R - q_{i,min}^R})); \tag{3.13}$$

where $q_{i,\text{min}}^R$ and $q_{i,\text{max}}^R$ are the joint angle boundaries, q_i^R is a joint angle position at time t, and γ is a constant that defines the maximum value of $\frac{w_x^R}{w_{q_i}^R}$. Finally, by setting the cartesian weights $\boldsymbol{W}_{\boldsymbol{x}}^R$, $\boldsymbol{W}_{\boldsymbol{x}}^L$ to be identity matrices, one might compute the diagonal elements of matrices $\boldsymbol{W}_{\boldsymbol{q}}^R$, $\boldsymbol{W}_{\boldsymbol{q}}^L$ through Eq.3.13. Inside the robot's workspace, the Cartesian weights $\boldsymbol{W}_{\boldsymbol{x}}$ take much larger values than the joint weights $\boldsymbol{W}_{\boldsymbol{q}}$ and, therefore, the robot follows the desired Cartesian path; see (Hersch & Billard, 2008) for a comparison of joint space and Cartesian control.

3. *Enforcing spatial coordination constraints* (while in a stable posture)

   While transiting through a stable posture, a robot has to adhere to learned spatial constraints. To enforce their resolution, we extend the optimization problem in Eq.3.10-3.11 with spatial coordination constraints as follows:

$$\dot{\boldsymbol{x}}^R - \dot{\boldsymbol{x}}^L = 0 \tag{3.14}$$

   After solving the joint constrained optimization problem (Eq. 3.9, 3.11 and 3.14), we obtain:

$$\begin{aligned} \Delta\boldsymbol{q}^R &= (\boldsymbol{M}_1)^{-1}\boldsymbol{M}_2; \\ \Delta\boldsymbol{q}^L &= [(\boldsymbol{J}^L)^{-1}\boldsymbol{J}^R]\boldsymbol{q}^R \end{aligned} \tag{3.15}$$

   where

$$\begin{aligned} \boldsymbol{M}_1 =& \boldsymbol{W}_{\boldsymbol{x}}^R\boldsymbol{J}^R + (\boldsymbol{J}^R)^{-T}\boldsymbol{W}_{\boldsymbol{q}}^R + \boldsymbol{W}_{\boldsymbol{x}}^L\boldsymbol{J}^R + (\boldsymbol{J}^L)^{-T}\boldsymbol{W}_{\boldsymbol{q}}^L(\boldsymbol{J}^L)^{-1}\boldsymbol{J}^R; \\ \boldsymbol{M}_2 =& \boldsymbol{W}_{\boldsymbol{x}}^R(\boldsymbol{x}_d^R - \boldsymbol{x}^R) + (\boldsymbol{J}^R)^{-T}\boldsymbol{W}_{\mathbf{q}}^R(\mathbf{q}_d^R - \mathbf{q}^R) + \\ & \boldsymbol{W}_{\boldsymbol{x}}^L(\boldsymbol{q}_d^L - \boldsymbol{q}^L) + (\boldsymbol{J}^L)^{-T}\boldsymbol{W}_{\mathbf{q}}^L(\boldsymbol{q}_d^L - \boldsymbol{q}^L). \end{aligned} \tag{3.16}$$

4. *Enforcing temporal coordination constraints*

   For each posture, a learned task model Π defines its temporal constraints: occurrence time $\hat{t}_{i,\text{start}}$ and duration $\hat{T}_i$. To ensure the synchronization between the arms as well as the timing of a movement, we require a robot to reproduce the learned temporal constraints. Note that, by varying the parameters α and β[5] of the VITE controller in Eq. (3.7)-(3.8), we can adjust motion duration and the velocity of the robot along a movement. Next, we show how to compute the parameters α and β so as to guarantee that the learned temporal constraints are fulfilled. Being a linear dynamical system, the VITE model has a single attractor

---

[5] To simplify notation, we further omit indices ^{R},^{L} and $_{x,q}$ and refer simply to α and β while assuming that the results are applicable to both arms, in Cartesian and joint spaces. The derivations are made for a one dimensional case.

that can be stable or unstable depending on the parameters α and β. Therefore, while adjusting α and β so as to satisfy the learned temporal constraints, we need to keep in mind the requirement of stability. For the VITE system to be stable, its eigenvalues λ_1 and λ_2 should be either real negative or complex conjugate with negative real parts. We choose the latter as the strictly negative real eigenvalues might produce a sharply peaked velocity profile, which is undesirable for implementation on a robot: $\lambda_1 = -m + ni, \lambda_2 = -m - ni$, where $m, n \in \mathbb{R}, m > 0$ are the parameters to be computed, i is the imaginary unit. The parameters α, β can be expressed through m and n as:

$$\alpha = 2m; \beta = \frac{4n^2 + \alpha^2}{4\alpha}. \tag{3.17}$$

To obtain expressions for the parameters m and n, we consider the VITE controller from Eq.3.7 as a differential equation and resolve it analytically under boundary conditions:

$$x(0) = x_0;\ \dot{x}(0) = \dot{x}_0;\ x(T) = x_{\text{end}};\ \dot{x}(T) = \dot{x}_{\text{end}}; \tag{3.18}$$

where T is the learned motion duration, $x_0, \dot{x}_0, x_{\text{end}}, \dot{x}_{\text{end}}$ are respectively the current robot's position and velocity and these in the end of the movement. The boundary conditions have to satisfy the following requirement: $||\frac{x_{\text{end}} - x_{\text{tar}}}{x_0 - x_{\text{tar}}}|| \approxeq 0$, where x_{tar} is a target position. Additionally, according to the definition of an attractor, the velocity in the end of the movement must satisfy: $||\dot{x}_{\text{end}}|| \approxeq 0$.

Taking into account the boundary conditions in Eq. 3.18, we obtain the following formulas for m, n:

$$n = \frac{\pi}{T}; \quad m = \frac{1}{T} \log \left(\frac{x_0 - x_{\text{tar}}}{x_{\text{end}} - x_{\text{tar}}} \right). \tag{3.19}$$

If, during a motion, the robot encounters an external perturbation (e.g. a manipulated object is shifted or a robot's arm is being pushed), the *motor system* has to adapt the velocity of both arms accordingly: for this, the motor system recomputes the parameters α and β taking into account new environmental information and a current configuration of a robot.

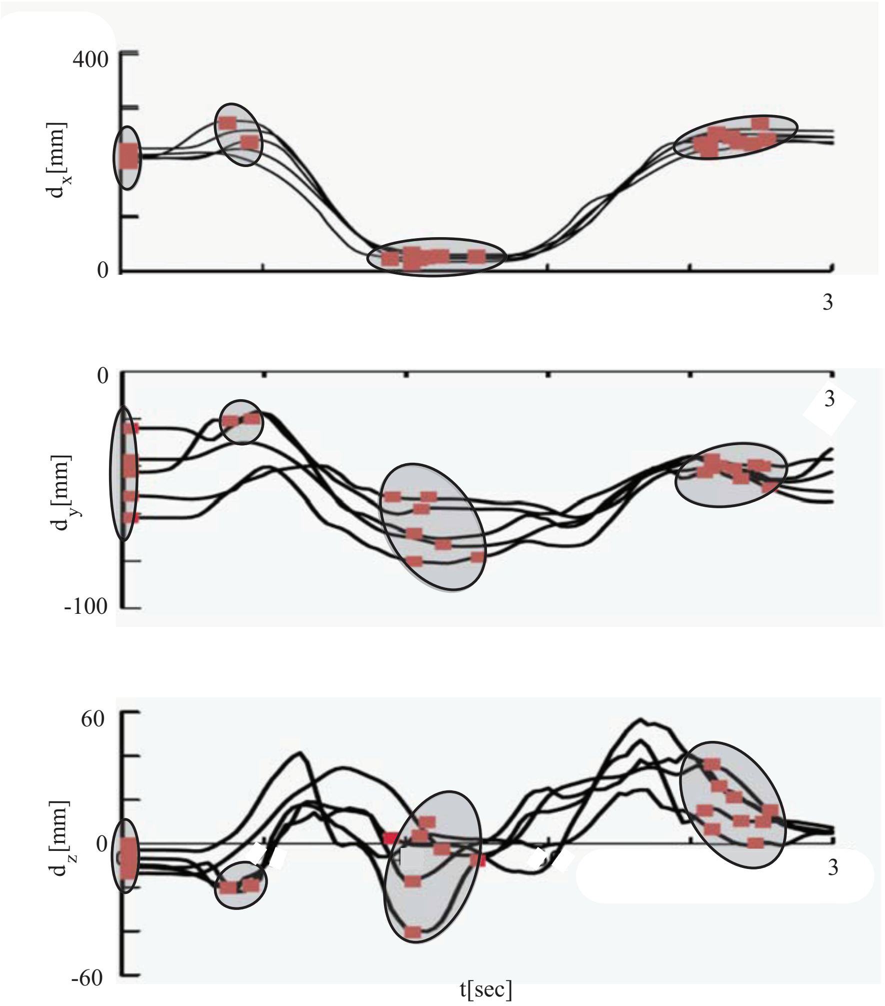


Figure 3.7: The HMM encoding of *Tea task*. The demonstrated trajectories of the relative position ***d***; red dots are the starting points of stable postures extracted according to (their covariance matrices after training the HMM parameters are represented in bold ellipses).

Table 3.1: Postures estimated by the *Learning system*

Task	**A number of postures extracted from the training data**	**A number of postures after encoding with HMM**
Tea task	6	4
Cube task	5	3
Tray task	7	4

3.2.3 EXPERIMENTAL RESULTS

To validate our method, we conduct three experiments in which a humanoid robot is taught to bimanual coordinated movements: manipulating a bulky cube (*Cube task*), putting a sugar into a cup (*Tea task*), and moving a tray (*Tray task*); see Fig. 3.8. Our objective in these experiments is twofold: (i) to demonstrate that the robot can extract coordination constraints and (ii) to illustrate that the robot's *motor system* adapts under perturbations, so as to fulfill learned task constraints.

During task reproduction, the positions of manipulated objects are tracked with an external stereovision system that has a wider angle of view in comparison with cameras in-built in the robot. All trajectories of the robot's arms are calculated with respect to a frame of reference located at the center of the robot's waist.

- *Task Learning*

  Fig. 3.7 shows an example of posture encoding in *Tea task*. The *learning system* determines four stable postures along the time series of the relative position $\boldsymbol{d}$. Only three of these postures (first, third, and forth) are statistically relevant. The second posture is spurious: it does not correspond to any specific movement pattern and appears only in two demonstrations out of five. The learning algorithm discovers that the probability to transit through this posture is lower than the chosen threshold, therefore, the robot is not required to reproduce it.

  Table 3.1 provides a quantitative summary of results on task learning: a number of initially extracted stable postures and a final number after the statistical encoding with HMMs . Note that the segmentation procedure tends to extract more postures than it is statistically relevant: Encoding with HMMs allows for casting off some unnecessary postures.

- *Enforcing Spatial Constraints*

  Fig. 3.9 - 3.11 share a same legend: each figure contains a graph with robot's hands trajectories (projected into the axial plane), the workspace accessible to the robot in each task (the accessible workspace is estimated as a space where the robot is able to maintain learned coordination constraints), and a series of snapshots with superimposed trajectories.

  Fig.3.9 presents results of the reproduction of *Cube Task*. To test the ability of the *motor system* to adapt to external perturbations, we first change the position of the cube while the robot is trying to grasp it, and then we change the position

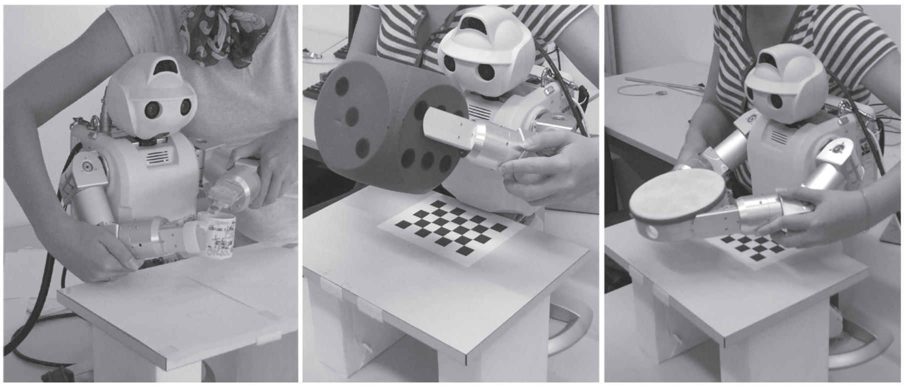


Figure 3.8: Experimental set-ups. To validate our method, we investigate its performance in three manipulation tasks. The *tea task*: put a piece of sugar into a cup. The *cube task*: grasp a cube, lift it, and put it on top of the pedestal. The *tray task*: grasp a tray with both arms, lift it, and move the tray forward.

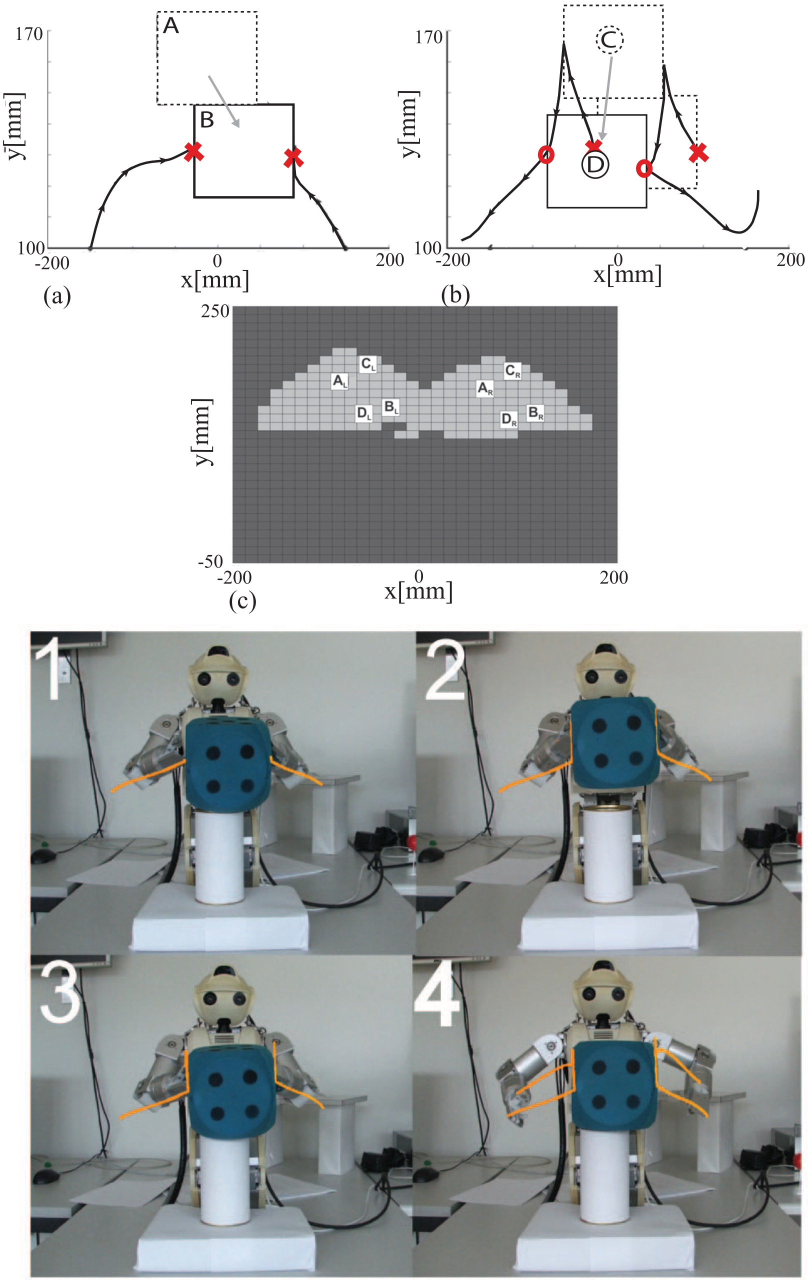
of the pedestal on top of which the robot has to put the cube. The second perturbation is applied while the robot is in a stable posture (carrying the cube) and hence it has to maintain the relative position between its arms so as not to drop the cube. The perturbations are applied so that all objects positions are reachable to the robot.

Fig. 3.10 presents results of the reproduction of *Tea task*. In this experiment we apply perturbations also twice, as in *Cube task*; in both cases we simulate a situation where the right arm of the robot is suddenly pushed. This was achieved by sending a perturbed command to the arm's joints. In the first case, a jerk is initiated while the robot is moving the arms towards each other. In the second case, we send a jerk command, when the robot is opening its gripper to put a piece of sugar into a cup. The sudden changes in the robot's configuration are detected as a discrepancy between a planned robot's joint configuration and the feedback from the motors. In each case, the robot readapts the position of both arms to make sure that the sugar will not fall outside the cup.

Fig. 3.11 presents results of the reproduction of *Tray task*. We change the position of the tray, while the robot is trying to grasp it. This perturbation forces the robot to manipulate on the boundaries of its workspace. The robot successfully adapts its motion and ensures that the coordination constraints are satisfied. The robot carries the tray without dropping it.

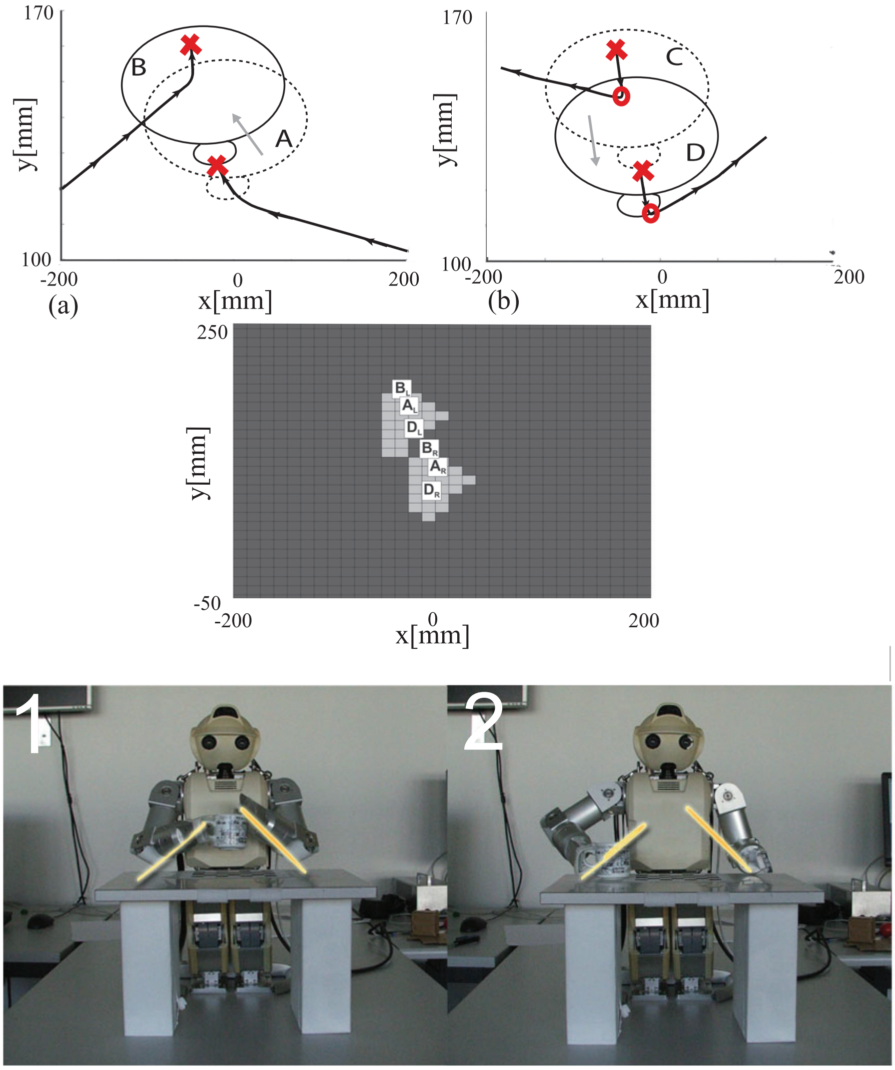
- *Enforcing Temporal Constraints*

  Fig. 3.12 illustrates the preservation of the synchronization feature of the movement of the two arms of the robot during the *Cube task*. At time t_1 while the robot was moving the arms towards the cube, we changed the position of the cube. Both arms adapted their trajectory simultaneously to handle this perturbation and reached the target simultaneously. In the same figure we show the velocity profiles of both arms in both Cartesian and joint-angle spaces. We see



(a) (b) (c) (d)

Figure 3.9: The *cube task.* (a) A robot tries to grasp a cube, but the cube is suddenly shifted from the position A to the position B, the direction of the perturbation is specified by a grey arrow. The robot adapts the trajectories of the both arms, so as to grasp the cube from the position B (the moment of grasping is highlighted with red cross). (b) Continuation of the task: while the robot is carrying the cube, the position of a pedestal is changed from C to D, and the robot brings the cube to the new location. Note that from grasping the cube (red cross) until releasing it (red circle), the robot preserves the learned relative position between the hands. (c) The positions A and B of the cube and C and D of the pedestal are superimposed with the workspace accessible to the robot in this task (light grey). Note that perturbations force the robot to operate almost on the boundary of its accessible workspace, however, the robot's *Motor System* successfully resolves the learned constraints and generates motion trajectories. (d) The photos of the robot at the different stages of the task completion. The yellow lines on photos are the trajectories of the robot.

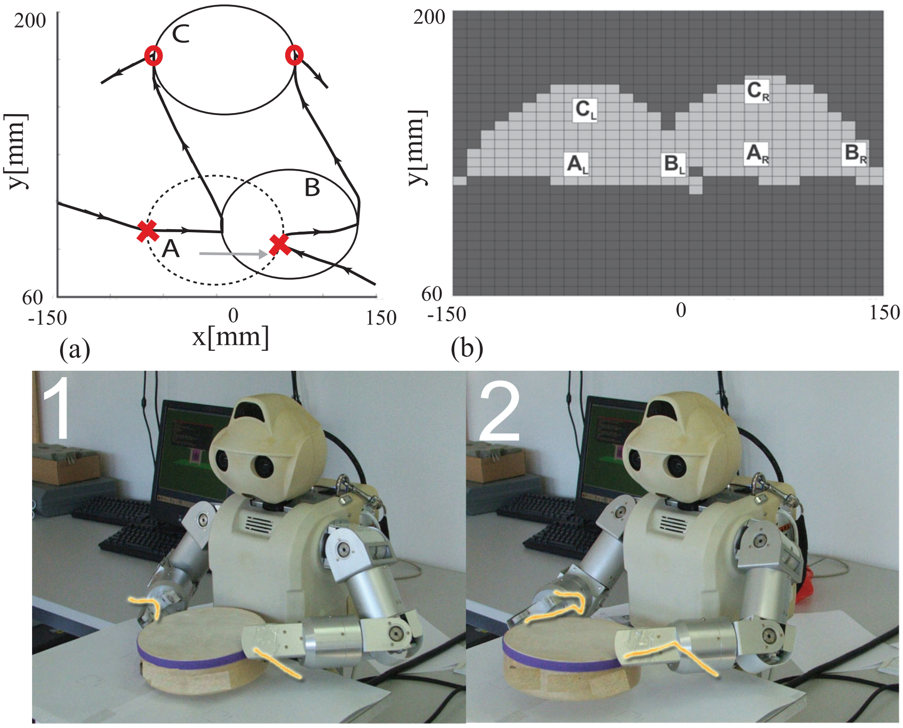


(a) (b)

(d)

Figure 3.10: The *tea task*. (a) A robot is bringing the two arms together so as to put a piece of sugar into a cup. During the motion, its right arm is pushed (as a result, the cup moved from the position A to B) so that, to accomplish the task, the robot has to quickly adapt both arms. The direction of the perturbation is specified by a grey arrow. (b) When the robot's is in the stable posture (the relative position between the arms is preserved) and is about to put a sugar in a cup, the right arm is pushes again (the cup moves from C to D). This time, however, the *Motor system* preserves the posture and adapts the left arm accordingly. (c) The positions A, B, C, and D of the cup are superimposed with the workspace accessible to the robot in this task (light grey). Note that perturbations force the robot to operate almost on the boundary of its accessible workspace, however, the robot's *Motor System* successfully resolves the learned constraints and generates motion trajectories. (d) The photos of the robot at the different stages of the task completion. The yellow lines on photos are the trajectories of the robot.

that the *motor system* produce smooth and bell-shaped velocity profiles similar to these of humans.



(c)

Figure 3.11: The *tray task*. (a) After grasping a tray (red cross), a robot is suddenly pushed (from the position A to B). The direction of the perturbation is specified by a grey arrow. This perturbation forces the robot to manipulate on the boundaries of its workspace. The robot successfully adapts its motion and ensures that the coordination constraints are satisfied. The robot carries the tray without dropping it and release it in the position C (red circle). (c) The positions A, B, and C of the tray are superimposed with the workspace accessible to the robot in this task (light grey). Note that perturbations force the robot to operate almost on the boundary of its accessible workspace, however, the robot's *Motor System* successfully resolves the learned constraints and generates motion trajectories. (d) The photos of the robot at the different stages of the task completion. The yellow lines on photos are the trajectories of the robot.

3.2.4 DISCUSSION

The method presented in this section aims to address some features of bimanual coordination so that to guarantee robot's performance in manipulation tasks. We investigate two types of constraints: spatial constraints (e.g., the two arms have to maintain a specific spatial relation between each other) and temporal constraints (the two arms have to synchronize). The satisfactory performance is deemed achieved if the robot manages to transit through a set of stable postures (bimanual constraints), while manipulating objects. Note that, while forced to adhere to coordination constraints, the robot is free to depart from the trajectories shown during demonstration.

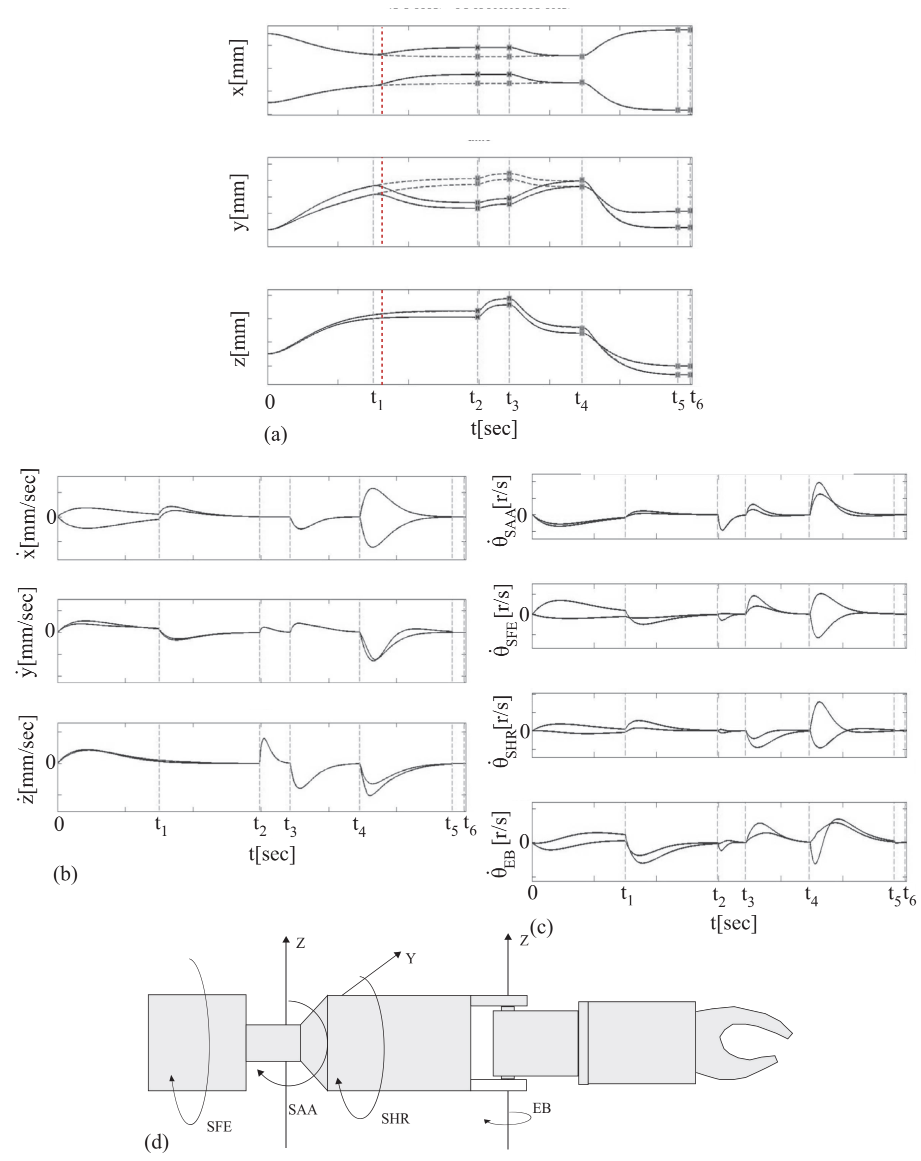


Figure 3.12: Task reproduction under temporal coordination constraints. The synchronization in the *Cube task*: at $t_1 = 150$, after the onset of the motion, the position of the cube is changed (the time of the perturbation is highlighted by a dashed red line). (a) Simultaneously, the robot's *Motor system* adapts the trajectories of the two arms to reach the cube in the new location. The velocity profiles under perturbations are smooth and close to bell-shaped, both in Cartesian (b) and joint space (c). The time instances $t_2..t_6$ refer to the boundaries of stable postures. (d) A schema of the HOAP-3 arm, the joints (SFE, SAA, SHR, EB) correspond to the velocities presented in figure (c).

During task reproduction, we generate several perturbations so as to test the robustness of the proposed method. We, indeed, manage to show that within the accessible parts of the workspace, the robot handles the perturbations well. The method is also capable of reproducing task motions in an unobserved context. In the *Cube* experiment, we perturb the cube's position while the robot is reaching for it; in the second part of the motion, the pedestal is also shifted. The experiments confirm that the robot can cope with this novel situation. However, we should emphasize that, in the current framework, the robot does not learn to associate positions of manipulated objects with extracted postures; this mapping needs to be done manually.

Yet, we should also mention limitations that might affect the robot's performance. The trajectories generated by the *motor system* built upon the VITE model are quasi-straight. Therefore, if a task requires a robot to reproduce a significantly curved motion (e.g., to put an object into a box without bumping into it), the performance might be unsatisfactory. We will address this limitation in the next section.

The magnitude of a perturbation should be sufficiently small and the perturbation should be fast so that the robot can still reach the target during the allocated motion time. Failing this, the system can potentially react with jerky movements. Therefore, it would be desirable to develop a mechanism that (r)estimates a physically plausible motion duration so as to ensure that both arms can satisfy imposed time constraints.

The presented algorithm concentrates on kinematic aspects of bimanual manipulation tasks; the manipulated objects are designated as light and their dynamical properties are not considered. If a robot is supposed to manipulate heavy objects, then their dynamical properties must be taken into consideration. In this case, the robot would need to learn constraints on the interaction force between the two end-effectors, in addition to spatial and temporal kinematic constraints.

3.2.5 CONCLUSIONS

In this section, we presented a novel approach to learning discrete bimanual coordination skills in a humanoid robot. We explained that the approach consisted of two components: (1) a *learning system* accountable for automatic extraction of spatio-temporal coordination constraints across the two end-effectors; and (2) a *motor system* built of coupled dynamical systems and capable of handling online perturbations occurred during coordinated motions.

The system was validated in three experiments where a humanoid robot has been taught discrete bimanual coordinated tasks. We demonstrated that the system successfully reproduced the tasks under various external perturbations.

3.3 LEARNING NONLINEAR DYNAMICS OF MOTION IN ROBOTIC MANIPULATORS

The method presented in this section has been published in:

Gribovskaya E., Khansari M., and Billard, A. G.. Learning Nonlinear Multivariate Dynamics of Motion in Robotic Manipulators. *International Journal of Robotics Research, Vol. 30(1), pp. 80-117.* 2011.

3.3.1 INTRODUCTION

This work was done in collaboration with Mohammad Khansari, who was also a PhD student at the LASA laboratory when this thesis was conducted. Mr. Khansari's input pertains to the following a) derivation of the local stability at the origin (eq. 3.28-3.29); b) experiments with the Katana robot.

IN the previous section, we demonstrated how a robot could learn tasks that required coordination between the two arms. The considered coordination constraints were discrete, and motion trajectories were generated by a pre-defined linear dynamical system. In this section, we suggest a more generic approach to motion representation that does not reduce the robot's motion repertoire to straight-line movements.

A core issue within robot control is to ensure that, if perturbed, the robot's motion can be rapidly recomputed, so that the robot ultimately accomplishes a task at hand. Perturbations may force the robot either to depart from its original trajectory or to be delayed. In the rest of this section, we will refer to the former type of perturbations as *spatial* perturbations and to the latter as *temporal* perturbations.

We focus on a low-level continuous representation of coordinated motions (Schaal et al., 2003; Ude et al., 2004). We investigate the problem of building encodings that can be easily modulated to enable re-use of a skill in novel contexts. An overview of requirements for an effective movement encoding are summarized in Ijspeert, Nakanishi, & Schaal (2001). Most relevant to the presented method are the notions of *reusability* of the representation (the encoding should allow a robot to reproduce a task in parts of the workspace where no demonstrations are provided), and the notion of *robustness* to perturbations (an ability of an encoding to ensure that a motion may be quickly adapted to perturbation and changes in a dynamic environment).

The idea of having task representations, which differ from traditional time-dependent trajectories has been challenging researchers for decades. For instance, in (P. Li & Horowitz, 2001a,b), the authors propose the concept of passive velocity field control (PVFC). One of the features of PVFC is that a desired behavior of a system under control is expressed as a dynamical system (according to the definition provided in (P. Li & Horowitz, 2001a), a velocity field is a function that maps system configurations into desired velocities). The motivation of the PVFC's authors is similar to ours: they explicitly emphasize that for some tasks, motion coordination and robustness should take precedence over tracking a timed trajectory. However, the major objectives of their work are distinct from ours: they aim to develop a control framework that would allow a robot to be passive (i.e., not injecting energy into an environment) while following a desired velocity field. When discussing the design of such fields, the authors state that this process is nontrivial and highly task dependent; they illustrate their algorithm with experiments where desired velocity fields are designed analytically. The method, which we propose in this section, aims to deduce such velocity fields directly from motion data. However, due to hardware constraints (the considered robotic platforms accept positional input) and our particular research question, we rely on a simpler method, a PD controller, to reproduce learned dynamical systems.

Recent works on *feedback planning* (Brock et al., 2008; Tedrake et al., 2010) also emphasize the need to develop an encoding of robot motion that embeds the ability to adapt or even re-generate trajectories on the fly. Though the planners are able to generate trajectories taking into account different external and internal constraints, the planning might require significant computation time. This is an impediment for robotic applications that need an immediate response in the case of perturbations.

The approach of learning motions as dynamical systems that we follow here, was suggested as an alternative to classical planning algorithms (Schaal et al., 2007). Autonomous dynamical systems encode trajectories through a time-independent function that defines the temporal evolution of a motion. The advantages of the dynamical system motion representation as opposed to providing a robot with a single pre-planned trajectory are three-fold. (1) The use of this representation exempts one from re-indexing trajectories while recovering from perturbation or during adaptation to new initial conditions (robustness to temporal perturbations). (2) Motion planning with dynamical systems allows for on-line adaptation to spatial perturbations, and therefore does not require additional algorithms to replan a complete trajectory or to re-scale an existing one. (3) The dynamical system motion representation offers a means to generalize motions in areas of the workspace not covered during demonstration.

One of the main limitations on using dynamical systems for motion encoding is possible instability of a learned dynamics. The primary concern is, therefore, to ensure stability of the estimate. Once the stability issue is resolved, dynamical systems are able to handle more complex constraints, like the presence of obstacles or robot's physical limitations (e.g. joint limits (Hersch & Billard, 2008)). Existing literature that derives a stable dynamical system does so by imposing an external stabilizer (e.g., a linear stabilizer in Dynamical Movements Primitives (DMP) (Ijspeert, Nakanishi, & Schaal, 2001; Pastor et al., 2009)). A disadvantage of this approach is that the external stabilizer distorts a temporal pattern of a dynamics (see experiments in Section 3.3.4.7). Our work concentrates on building a stable dynamical motion representation that does not rely on an external stabilizer and, therefore, preserves a spatio-temporal pattern of a demonstrated motion. However, as we learn a non-linear dynamical function, we can guarantee only local stability, while DMP guarantees a global stability. Yet, we argue that non-linear motions are usually driven by local coordination constraints, e.g., related to a shape of a manipulated object. Therefore, reproducing a nonlinear motion far from an originally observed context is not necessarily helpful. Furthermore, statistical learning is local by nature; therefore, one cannot ensure that inference far from the demonstrations is relevant in the statistical sense.

We ground our work on an assumption that human motions contain regularities that can be represented by a dynamical system. As there is no unified approach to representing arbitrary via-point motions, we develop a method to learn them from motion data. We do so by discovering non-linear dynamical laws that govern kinematic invariants contained in the data.

To model the natural variability of human motions, dynamical models often include a signal-dependent noise that is represented by a multiplicative Gaussian noise (Harris & Wolpert, 1998). The signal-dependent noise, partly caused by muscle fatigue and imprecision in sensor feedback, is considered as an inherent limitation of human motor control (Shadmehr et al., 2010). Therefore, in our work, we assume that learning of the deterministic part that accounts for motion dynamics should be sufficient for the design of robot control.

In this section, we consider the problem of estimating a *time-independent* model of motion through a set of first order non-linear multivariate dynamical systems. We exploit the strength of *parametric statistical techniques* to learn correlations across the variables of the system and show that the proposed method allows one to discover a coarse representation (dependent on a limited number of free parameters) of the dynamics. We demonstrate advantages of our approach as an alternative to the *time-dependent* methods, by ensuring robustness to external *spatio-temporal perturbations* through on-line adaptation of the motion. Here, under robustness to perturbations we particularly refer to the ability of the system to react to changes in the environment that are encapsulated by motion parameters, such as a desired target position and motion duration. Therefore, the system is able to cope with uncertainties in the position of a manipulated object, duration of motion, and perturbations associated with robot's body limitation (e.g., joint velocity and torque limits).

The term *perturbation* has been treated rather broadly in the current robotics research; however, to the best of our knowledge, no established classification of perturbations can be found in the literature. We therefore suggest to classify perturbations according to the following criteria. (1) *Spatial vs. temporal*; perturbations that either affect the position of the robot in space or modify the planned motion duration. These perturbations are often coupled, e.g. as the robot is pushed farther from the target, both spatial and temporal perturbations occur. (2) *External vs. internal* (or self-generated); whether a perturbation has been applied externally (e.g., a robot has been pushed away while tracking a trajectory) or generated internally (e.g., if a motion planner autonomously generates a spatial perturbation to avoid an obstacle). (3) *Instantaneous vs. continuous*; whether the perturbation has an impulse character (e.g. in the case of a sudden push or a jerk) or the perturbation is applied continuously and thus systematically modifies the robot's motion (e.g. if a human applies a continuous force to slow down the robot's motion). The suggested classification is not exhaustive; however, it may prove useful for a qualitative comparison of the existing motion planning methods.

According to this classification, our method handles spatial and temporal perturbations which are externally-generated[6]; and applied instantaneously or continuously.

This section is divided as follows. Section 3.3.2.1 starts with a formalization of the problem at hand. This is followed by a technical description of the modeling approach: Section 3.3.2.2 introduces our learning approach for estimating a motion dynamics; and Section 3.3.2.3 presents an iterative algorithm to improve stability of the learned dynamics. Finally, in Section 3.3.3, we validate our method by estimating the motion dynamics from trajectories generated with given dynamical laws; in this way we may systematically verify approximation qualities of the method. We, further, show how the same framework can be used to learn the motion dynamics of manipulation tasks with different robotic platforms. To emphasize advantages of our approach as compared to the state-of-the-art methods in the field, we provide an experimental comparison with Dynamic Movements Primitives (Ijspeert, Nakanishi, & Schaal, 2001; Pastor et al., 2009). The legend used in graphs throughout the paper is summarized in Fig. 3.13. The glossary is in Table 3.2.

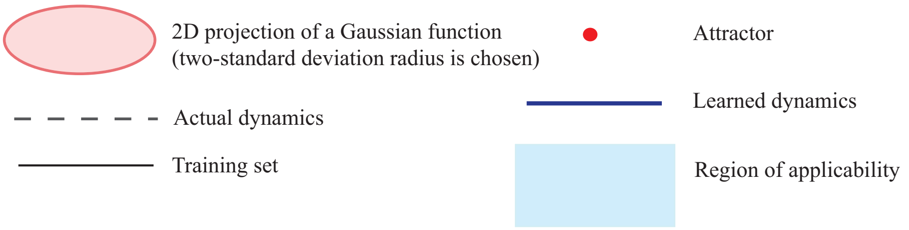

2D projection of a Gaussian function
(two-standard deviation radius is chosen)
Attractor
Learned dynamics
Actual dynamics
Training set
Region of applicability

Figure 3.13: The legend used in figures in Section 3.2.

---

[6] Limited adaptation to internally-generated temporal perturbations is addressed through adaption to joint torque limits

3.3.2 METHOD

In Sections 2.2.1 and 3.3, we review methods for extracting a dynamical system from observed data that are proposed within the fields of system identification and robot learning. Many of these methods concentrate on the direct modeling of the dynamical function. We take the other approach and suggest estimating a joint distribution of all observed data. The dynamical function is then computed through from this distribution.

In this section, we formalize the task of learning dynamical motion representations (Section 3.3.2.1) and explain the proposed statistical approach (Section 3.3.2.2). We also discuss the problem of ensuring the stability of a learned system (Section 2.2.2).

3.3.2.1 PROBLEM STATEMENT

Let us assume that a state[7] of a robot during a motion can be unambiguously described by a variable $\boldsymbol{\xi}$ and that the workspace of a robot forms a sub-space X in $\mathbb{R}^N$.

Consider further that the state $\boldsymbol{\xi}$ is governed by an *Autonomous Dynamical System* $\langle X, f, T \rangle$ (as per Definition 1-2, Table 3.2). Then, for all starting locations $\boldsymbol{\xi}_0 \in X$, the temporal evolution of a robot's motion is *uniquely determined* by a *state transition map* (Definition 2, Table 3.2) $f(t, t_0, \boldsymbol{\xi}_0) = \boldsymbol{\xi}(t), \forall \boldsymbol{\xi}_0, \boldsymbol{\xi} \in X$.

Let us further assume that the state transition map f is a non-linear continuous and continuously differentiable function and that the system is driven by a first order differential equation[8] with a single equilibrium point (attractor) $\bar{\xi}$, such that:

$$\begin{aligned} &\forall t \in T = [t_0; \infty]; \boldsymbol{\xi}, \dot{\boldsymbol{\xi}} \in \mathbb{R}^N \\ &\dot{\boldsymbol{\xi}}(t) = f(\boldsymbol{\xi}(t)) \\ &\dot{\bar{\boldsymbol{\xi}}} = f(\bar{\boldsymbol{\xi}}) = 0. \end{aligned} \tag{3.20}$$

Let a set $\mathcal{D}$ of M N-dimensional demonstrated datapoints $\{\boldsymbol{\xi}^i, \dot{\boldsymbol{\xi}}^i\}_{i=1}^M$ be instances of the above motion model corrupted by a multiplicative zero-mean Gaussian noise. The problem then is to reconstruct a noise-free estimate $\hat{f}$ of f from the set of demonstrations $\mathcal{D}$. To this end, we will approximate the function in a subregion[9] $C \subset \Delta \subset X$, so that:

$$\begin{aligned} &\hat{f} : C \to C \\ &\hat{f}(\boldsymbol{\xi}(t)) \approxeq f(\boldsymbol{\xi}(t)), \forall \boldsymbol{\xi} \in C. \end{aligned} \tag{3.21}$$

---

[7] The state of a dynamical system represents the *minimum amount of information* required to describe the effect of past history on the future development of this system (Hinrichsen & Pritchard, 2000).

[8] Considering solely *first order* dynamical systems is not restrictive to learning only first order relationships between trajectory and velocity, as one can always convert dynamics of an arbitrary order into a canonical system of first order ODEs.

[9] Estimating the dynamics in the whole state-space X would be practically infeasible due to the excessive number of demonstrations that such estimation would require.

Table 3.2: Glossary

Definition 1: The *state-space* $X \subset \mathbb{R}^N$ includes all possible instantiations of $\boldsymbol{\xi}$, such that $\boldsymbol{\xi}(t) \in X$ at each time step $t \in T = \mathbb{R}^+ = [0; \infty]$.

Definition 2: A *dynamical system* is the tuple $\langle X, f, T \rangle$, with $f : t \rightarrow f^t$ a continuous map of X onto itself.

Definition 3: A dynamical system is *differentiable* if $\exists f : T \times X \rightarrow X$ such that for all $t_0 \in T, \boldsymbol{\xi}_0 \in X$ the problem:

$$\dot{\boldsymbol{\xi}} = f(t, \boldsymbol{\xi}(t)), \;\; t \geq t_0, t \in T$$

$$\boldsymbol{\xi}(t_0) = \boldsymbol{\xi}_0$$

has a unique solution.

A dynamical system governed by a time-independent transition map with $f(t, \boldsymbol{\xi}(t)) \triangleq f(\boldsymbol{\xi}(t))$ is an *Autonomous Dynamical System.*

Definition 4. An *equilibrium state* $\bar{\xi} \in X$ of a dynamical system is such that

$$f(t, t_0, \bar{\boldsymbol{\xi}}) = 0.$$

Definition 5. An equilibrium state $\bar{\boldsymbol{\xi}} \in X$ is *stable* if $\exists \epsilon > 0$ and $\delta = \delta(\epsilon)$ such that

$$\forall \boldsymbol{\xi}_0 \in B(\bar{\boldsymbol{\xi}}, \delta) \Rightarrow f(\boldsymbol{\xi}_0) \in B(\bar{\boldsymbol{\xi}}, \epsilon),$$

$B(\bar{\boldsymbol{\xi}}, \delta) \subset X$ is a hypersphere centered at $\bar{\boldsymbol{\xi}}$ with radius δ. $\bar{\boldsymbol{\xi}}$ is an *attractor* of f.

Definition 5. An *attractive state* is an equilibrium state $\bar{\boldsymbol{\xi}}$ of a local flow, if there exists $\rho > 0$ such that:

$$\forall \boldsymbol{\xi}_0 \in B(\bar{\boldsymbol{\xi}}, \rho) \Rightarrow \lim_{t \to \infty} f(\boldsymbol{\xi}_0) = 0.$$

$B(\bar{\boldsymbol{\xi}}, \delta) \subset X$ is a hypersphere centered at $\bar{\boldsymbol{\xi}}$ with radius δ. $\bar{\boldsymbol{\xi}}$ is an *attractor* of f.

Definition 6. An equilibrium point $\bar{\boldsymbol{\xi}}$ is *asymptotically stable* if it is both stable and attractive.

Definition 7. A set $\Delta \subset X$ is a *Region of Attraction (or Basin of Attraction)* of an equilibrium $\bar{\boldsymbol{\xi}}$ if:

$$\Delta(\bar{\boldsymbol{\xi}}) = \{\boldsymbol{\xi}_0 \in X; \lim_{t \to \infty} f(\boldsymbol{\xi}_0) = \bar{\boldsymbol{\xi}}\}$$

See Fig. 3.41-II for illustration.

Definition 8. A dynamical system is *globally asymptotically stable* at the equilibrium $\bar{\boldsymbol{\xi}}$ if $\bar{\boldsymbol{\xi}}$ is an asymptotically stable attractor and $\Delta \equiv \mathbb{R}^N$.

C is referred further to as the *region of applicability* of a learned dynamics.

Without loss of generality, we can transfer the attractor to the origin[10], so that $\bar{\boldsymbol{\xi}} = 0 \in C \subset X$ is now the equilibrium point of f and by extension of its estimate $\hat{f}$, i.e. $\hat{f}(0) = f(0) = 0$. If C is contained within the *region of attraction* Δ of $\bar{\boldsymbol{\xi}}$ (see Definition 7, Table 3.2), then the estimate $\hat{f}$ is asymptotically stable at $\bar{\boldsymbol{\xi}}$ in C and any motion initiated from $\boldsymbol{\xi}(t_0) \in C$ will asymptotically converge to the target $\bar{\boldsymbol{\xi}}$.

3.3.2.2 LEARNING A DYNAMICAL MOTION REPRESENTATION WITH GAUSSIAN MIXTURE REGRESSION

We learn a dynamical motion representation in two step. Firstly, we estimate a joint distribution a training set $\mathcal{D}$. The dynamical function is then constructed from this distribution at the second step.

One way to obtain the joint distribution $p(\boldsymbol{\xi}, \dot{\boldsymbol{\xi}})$ is to encode the training data $\mathcal{D}$ with *Gaussian Mixture Models* (GMMs). GMMs define a joint distribution function $p(\boldsymbol{\xi}^i, \dot{\boldsymbol{\xi}}^i)$ as a mixture of a finite set of K Gaussian distributions $G^1..G^K$ (with $\boldsymbol{\mu}^k$, $\boldsymbol{\Sigma}^k$, and π_k being respectively the mean value, the covariance matrix, and the prior probability of a kth Gaussian G^k):

$$p(\boldsymbol{\xi}^i, \dot{\boldsymbol{\xi}}^i) = \frac{1}{K} \sum_{k=1}^{K} \pi_k G^k(\boldsymbol{\xi}^i, \dot{\boldsymbol{\xi}}^i; \boldsymbol{\mu}^k, \boldsymbol{\Sigma}^k) \tag{3.24}$$

and

$$\boldsymbol{\mu}^k = [\boldsymbol{\mu}^k_{\boldsymbol{\xi}} \; ; \; \boldsymbol{\mu}^k_{\dot{\boldsymbol{\xi}}}] \text{ and } \boldsymbol{\Sigma}^k = \begin{pmatrix} \boldsymbol{\Sigma}^k_{\boldsymbol{\xi}} & \boldsymbol{\Sigma}^k_{\boldsymbol{\xi}\dot{\boldsymbol{\xi}}} \\ \boldsymbol{\Sigma}^k_{\dot{\boldsymbol{\xi}}\boldsymbol{\xi}} & \boldsymbol{\Sigma}^k_{\dot{\boldsymbol{\xi}}} \end{pmatrix} \tag{3.25}$$

Where each Gaussian probability distribution G^k is given by:

$$\begin{aligned} &G^k(\boldsymbol{\xi}^i_t, \dot{\boldsymbol{\xi}}^i_t; \boldsymbol{\mu}^k, \boldsymbol{\Sigma}^k) = \\ &\qquad \frac{1}{\sqrt{(2\pi)^{2N} |\boldsymbol{\Sigma}^k|}} e^{-\frac{1}{2}(([\boldsymbol{\xi}^i_t;\dot{\boldsymbol{\xi}}^i_t] - \boldsymbol{\mu}^k)^T (\boldsymbol{\Sigma}^k)^{-1} ([\boldsymbol{\xi}^i_t;\dot{\boldsymbol{\xi}}^i_t] - \boldsymbol{\mu}^k))}. \end{aligned} \tag{3.26}$$

The parameters $\boldsymbol{\mu}^k$ and $\boldsymbol{\Sigma}^k$, $k = 1..K$ are initialized using the *K-means* clustering algorithm starting from a *uniform mesh* and trained iteratively through Expectation-Maximization (EM) (Dempster et al., 1977); we will discuss training in more details in Section 3.3.2.4.

Once we obtain the joint distribution $p(\boldsymbol{\xi}, \dot{\boldsymbol{\xi}})$, we can use it to infer the derivative $\dot{\boldsymbol{\xi}}$ for each observed state $\boldsymbol{\xi}$. According to the Bayes theorem, this inference is done through taking the expectation of the conditional distribution $p(\dot{\boldsymbol{\xi}}|\boldsymbol{\xi})$:

$$p(\dot{\boldsymbol{\xi}}|\boldsymbol{\xi}) \sim \frac{p(\dot{\boldsymbol{\xi}}, \boldsymbol{\xi})}{p(\boldsymbol{\xi})}, \quad \dot{\boldsymbol{\xi}} = \mathbb{E}[p(\dot{\boldsymbol{\xi}}|\boldsymbol{\xi})] \tag{3.27}$$

---

[10] To simplify the notation, we keep the same notation for the domains C and X after translation at the origin.

Table 3.3: Gaussian Mixture Regression

Let us assume that, we have pairs of matched input $\xi^{\mathcal{I}}$ and output $\xi^{\mathcal{O}}$ datapoints. The joint probability of these data can be modeled using Gaussian Mixtures Models. A probability that a datapoint $\eta = [\xi^{\mathcal{O}}; \xi^{\mathcal{I}}]$ belongs to a particular GMM is defined by

$$p(\eta) = \sum_{k=1}^{K} \pi_k \, \mathcal{N}(\eta; \mu_k, \Sigma_k) =$$

$$= \sum_{k=1}^{K} \pi_k \, \frac{1}{\sqrt{(2\pi)^D |\Sigma_k|}} \, e^{-\frac{1}{2}\left((\eta-\mu_k)^T \Sigma_k^{-1} (\eta-\mu_k)\right)}$$

where π_k are prior probabilities and $\mathcal{N}(\mu_k, \Sigma_k)$ are Gaussian distributions defined by means μ_k and covariance matrices Σ_k:

$$\mu_k = \begin{bmatrix} \mu_k^{\mathcal{I}} \\ \mu_k^{\mathcal{O}} \end{bmatrix}, \quad \Sigma_k = \begin{bmatrix} \Sigma_k^{\mathcal{I}} & \Sigma_k^{\mathcal{I}\mathcal{O}} \\ \Sigma_k^{\mathcal{O}\mathcal{I}} & \Sigma_k^{\mathcal{O}} \end{bmatrix}.$$

For a given input $\xi^{\mathcal{I}}$ and a given mixture component k. Gaussian Mixture Regression computes the distribution of $\xi^{\mathcal{O}}$ as:

$$p(\xi^{\mathcal{O}} | \xi^{\mathcal{I}}, k) = \sum_{k=1}^{K} h_k \mathcal{N}(\hat{\eta}_k, \hat{\Sigma}_k), \text{where} \quad \begin{aligned} \hat{\eta}_k &= \mu_k^{\mathcal{O}} + \Sigma_k^{\mathcal{O}\mathcal{I}} (\Sigma_k^{\mathcal{I}})^{-1} (\xi^{\mathcal{I}} - \mu_k^{\mathcal{I}}), \\ \hat{\Sigma}_k &= \Sigma_k^{\mathcal{O}} - \Sigma_k^{\mathcal{O}\mathcal{I}} (\Sigma_k^{\mathcal{I}})^{-1} \Sigma_k^{\mathcal{I}\mathcal{O}}. \end{aligned} \tag{3.22}$$

where $h_k = p(k|\xi^{\mathcal{I}})$ is the probability of the component k to be responsible for the observed input $\xi^{\mathcal{I}}$

$$h_k = \frac{\pi_k p(\xi^{\mathcal{I}}|k)}{\sum_{i=1}^{K} \pi_i p(\xi^{\mathcal{I}}|i)} = \frac{\pi_k \, \mathcal{N}(\xi^{\mathcal{I}}; \mu_k^{\mathcal{I}}, \Sigma_k^{\mathcal{I}})}{\sum_{i=1}^{K} \pi_i \, \mathcal{N}(\xi^{\mathcal{I}}; \mu_i^{\mathcal{I}}, \Sigma_i^{\mathcal{I}})}. \tag{3.23}$$

Alternatively, by using the linear transformation property of Gaussian distributions, the conditional expectation of $\xi^{\mathcal{O}}$ given $\xi^{\mathcal{I}}$ can be approximately defined by a single normal distribution with the parameters:

$$p(\xi^{\mathcal{O}} | \xi^{\mathcal{I}}, k) \sim \mathcal{N}(\hat{\eta}, \hat{\Sigma}), \text{where} \quad \begin{aligned} \hat{\mu} &= \textstyle\sum_{k=1}^{K} h_k \hat{\mu}_k \\ \hat{\Sigma} &= \textstyle\sum_{k=1}^{K} h_k^2 \, \hat{\Sigma}_k. \end{aligned}$$

Comparing Eq.3.27 and Eq.3.20, we observe that the following holds:

$$\dot{\boldsymbol{\xi}} = \hat{f}(\boldsymbol{\xi}) = \mathbb{E}[p(\dot{\boldsymbol{\xi}}|\boldsymbol{\xi})]. \tag{3.28}$$

The process of estimating the expectation of the conditional distribution in Eq.3.28 is called Gaussian Mixture Regression (GMR); see Table 3.3 for details. Taking the expectation of the conditional distribution in Eq. 3.22, we can write down the estimate $\hat{f}$ as follows:

$$\hat{\dot{\boldsymbol{\xi}}} = \hat{f}(\boldsymbol{\xi}) = \sum_{k=1}^{K} h_k(\boldsymbol{\xi})(\boldsymbol{A}_k\boldsymbol{\xi} + \boldsymbol{B}_k), \tag{3.29}$$

where $\boldsymbol{A}_k = \boldsymbol{\Sigma}^k_{\dot{\boldsymbol{\xi}}\boldsymbol{\xi}}(\boldsymbol{\Sigma}^k_{\boldsymbol{\xi}})^{-1}$, $\boldsymbol{B}_k = \boldsymbol{\mu}^k_{\dot{\boldsymbol{\xi}}} - \boldsymbol{A}_k\boldsymbol{\mu}^k_{\boldsymbol{\xi}}$, $h_k(\boldsymbol{\xi}) = \frac{p(\boldsymbol{\xi};\boldsymbol{\mu}^k_{\boldsymbol{\xi}},\boldsymbol{\Sigma}^k_{\boldsymbol{\xi}})}{\sum_{k=1}^{K} p(\boldsymbol{\xi};\boldsymbol{\mu}^k_{\boldsymbol{\xi}},\boldsymbol{\Sigma}^k_{\boldsymbol{\xi}})}$, $h_k(\boldsymbol{\xi}) > 0$, and $\sum_{k=1}^{K} h_k(\boldsymbol{\xi}) = 1$.

The representation in Eq.3.29 defines the approximation $\hat{f}$ is a mixture on linear dynamics weighted with coefficients $h_k(\boldsymbol{\xi})$. Note that these coefficients are nonlinear functions of the state $\boldsymbol{\xi}$, therefore, the function $\hat{f}$ is also nonlinear. Its stability is examined in Section 2.2.2.

A geometric illustration of the GMR inference in the case of single Gaussian is presented in Fig. 3.14 and the GMR procedure is summarized in Table 3.3. Fig. 3.15 further illustrates the encoding process from GMM to GMR for a non-linear dynamical system with a single attractor.

We emphasize that, in our framework, the goal of a movement is mapped into the attractor of a dynamical system. So far we have assumed that the attractor $\bar{\boldsymbol{\xi}}$ of a system in Eq. 3.29 coincides with the origin. To explain the behavior of the system under perturbations, we need to take into account an offset $\boldsymbol{\xi}_{\text{offset}}$. The offest defines the location of an object with respect to a frame of reference associated with its initial location (i.e., without perturbations $\boldsymbol{\xi}_{\text{offset}} = 0$). Therefore, in a general case, the system in Eq. 3.29 takes the form:

$$\hat{\dot{\boldsymbol{\xi}}} = \hat{f}(\boldsymbol{\xi} - \boldsymbol{\xi}_{\text{offset}}). \tag{3.30}$$

At the onset of a motion $\boldsymbol{\xi}_{\text{offset}}$ is set to zero. Each time a manipulated object is perturbed, the value $\boldsymbol{\xi}_{\text{offset}}$ is calculated as the difference between the initial and the new location of the goal. Essentially, a perturbation affects the calculation of a velocity signal $\hat{\dot{\boldsymbol{\xi}}}$ (and hence might lead to a nonsmooth velocity profile); the smoothness of the motion trajectory $\hat{\boldsymbol{\xi}}$ however is preserved. A recent work of ours also explores temporal scaling of a learned dynamics (Kim et al., 2010), so as to provide a robot with the ability to generate faster or slower movements depending on task requirements.

Several methods for identification of unknown dynamical functions, which we discuss in Sections 2.2.1 and 3.3, directly estimate a dependency between the state and its derivatives, for instance, through the least-square optimization. From this point of view, learning the joint distribution $p(\dot{\boldsymbol{\xi}}, \boldsymbol{\xi})$ might appear as a too complicated solution.

One however should note that such a formulation assumes uncertainty in the observed states $\boldsymbol{\xi}$ and, therefore, helps to mitigate the noise in the training values $\boldsymbol{\xi}$ (which effect is similar to this of error-in-variables; see Section 2.2.1). Specifically, the estimate in Eq.3.29 takes into account the covariance (noise) of the state $\boldsymbol{\Sigma}^k_{\boldsymbol{\xi}}$.

We further discuss the approximation properties of the estimator in Eq. 3.29. One can interpret Eq. 3.29 either as a nonlinear mixture of linear dynamical systems or as a linear mixture of the scalar basis functions $h_k(\boldsymbol{\xi})$ weighted with linear coefficients $\boldsymbol{A}_k\boldsymbol{\xi} + \boldsymbol{B}_k$. Taking the latter view, it is possible to discover some similarities with the nonparametric identification methods discussed in Section 2.2.1.2, particularly, with RBFs networks. Indeed, $h_k(\boldsymbol{\xi})$ are normalized Gaussian functions. However, there are a number of important differences between our method and RBFs networks. One of them is that the coefficients $\boldsymbol{A}_k\boldsymbol{\xi} + \boldsymbol{B}_k$ accept the multivariate input $\boldsymbol{\xi}$. Therefore, the approximation accounts for correlation between the dimensions of the state $\boldsymbol{\xi}$. Additionally, our method estimates the parameters in Eq. 3.29 by optimizing the expected log-likelihood. This should lead, at least theoretically, to an optimal solution (in the maximum likelihood sense).

However, the quality of approximation depends on whether the number of components K in the mixture is sufficient to represent an actual underlying function f. We would prefer to have as few components as possible so as to tune less parameters. Hence, our algorithm starts with a small number of Gaussians and gradually increases them until a stable and accurate estimate is found. The details are further discussed in Section 3.3.2.4.

The scope of our work does not encompass a formal investigation of what frequency of f the method can accommodate. Nevertheless, after examining the collected motion data, we can conclude that they do not display high-frequency changes. Therefore, we assume that the frequency of data sampling and filtering is much higher than the frequency of the underlying dynamical function. Therefore, the demonstrated data are representative of the dynamics. Given the class of considered applications, we perceive this assumption as reasonable. Yet, we do not claim that our method is uniformly applicable for learning any type of dynamics.

Note that here we learn multivariate functions, and hence the complexity of learning expands with the number of considered task dimensions. As this number grows, one needs a large amount of training data to estimate a motion representation accurately. This problem affects many statistical learning approaches, we discuss it in more details in Section 3.5.1.

EM estimation of GMMs requires inversion of the covariance matrices $\boldsymbol{\Sigma}^k$, which is not possible when these matrices are singular. These singularities might be caused, for instance, by severe data over-fitting, e.g. when one of the Gaussian components collapses into a single datapoint and the log-likelihood function of EM goes to infinity. Whether or not the singularities occur during training depends on the quality of training data and on the number of mixture components. In the experiments reported in the paper, this problem does not arise due to 1) the nature of trajectory data, which are sampled at a high frequency (therefore, EM has a sufficient amount of data to estimate the parameters); 2) a coarse encoding with a low number of mixture components. Alternatively, one may choose a variational treatment of GMMs (Attias, 1999) that assumes prior distributions over unknown parameters. Instead of estimating crisp values of covariances, as this is implemented in the standard formulation of GMMs, the variational approach learns a statistical distribution that simultaneously defines a family of covariances. After training a single covariance can be chosen as the expectation of the learned distribution. The variation learning therefore does not run into numerical instabilities.

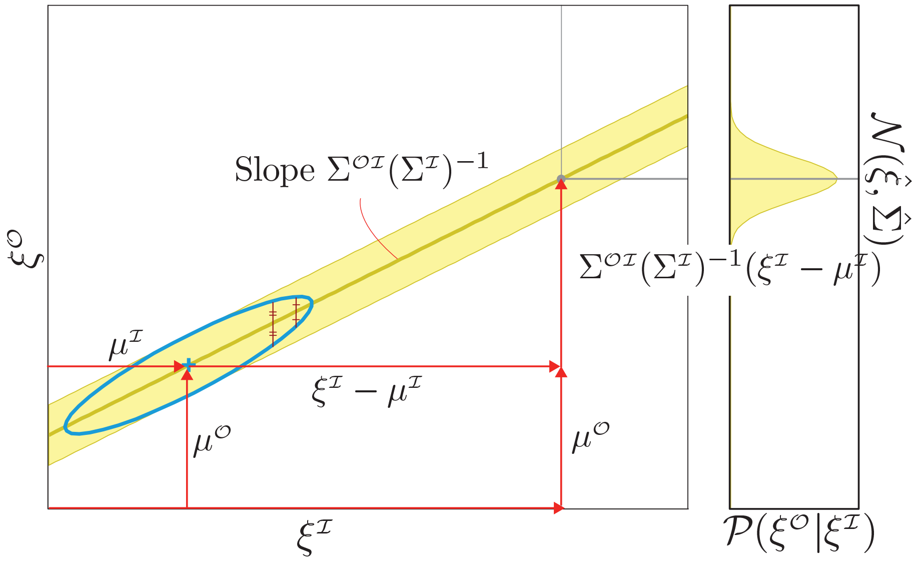

Slope $\Sigma^{\mathcal{O}\mathcal{I}}(\Sigma^{\mathcal{I}})^{-1}$
$\xi^{\mathcal{O}}$
$\mathcal{N}(\hat{\xi},\hat{\Sigma})$
$\Sigma^{\mathcal{O}\mathcal{I}}(\Sigma^{\mathcal{I}})^{-1}(\xi^{\mathcal{I}}-\mu^{\mathcal{I}})$
$\mu^{\mathcal{I}}$
$\xi^{\mathcal{I}}-\mu^{\mathcal{I}}$
$\mu^{\mathcal{O}}$
$\mu^{\mathcal{O}}$
$\xi^{\mathcal{I}}$
$\mathcal{P}(\xi^{\mathcal{O}}|\xi^{\mathcal{I}})$

Figure 3.14: The geometric interpretation of inference in GMR (see also Table 3.3). GMR approximates a dynamical system through a non-linear weighted sum of local linear models. Each regression matrix $A_k = \Sigma_k^{\mathcal{O}\mathcal{I}}(\Sigma_k^{\mathcal{I}})^{-1}$ defines a local linear dynamics. Here, we illustrate inference with a single Gaussian and a pair of input ξ^I and output ξ^O. In the planar case, the regression defines a line with a slope given by the matrix A_k ($\xi^O = A_k \xi^I$). For each input ξ^I, GMR defines a conditional distribution $p(\xi^O|\xi^I)$, with the mean ξ^O and the conditional covariance $\hat{\Sigma}$. The conditional covariance $\hat{\Sigma}$ defines an error envelope around the regression output ξ^O (the expected error on the output predicted by the regression).

3.3.2.3 STABILITY ANALYSIS

Reviewing methods for stability analysis in Section 2.2, we observe that though several theories have been developed (e.g., Lyapunov, passivity, and contraction theories), existing practical solutions are often suitable for a particular class of nonlinearities. In particular, for examining the stability in the Lyapunov sense, one needs to design a Lyapunov function, and for examining the stability as defined in contraction theory, one needs a contraction function. Manual design of these function is tedious and often unfeasible. Efficient numerical solutions, such as Sum of Square Programming (SoS) (see Section 2.2), are applicable to polynomial dynamical systems.

In our case, learned dynamical functions are not polynomial. Therefore, we do not use SoS methodsand, instead, conduct stability analysis in two steps as follows: (1) a

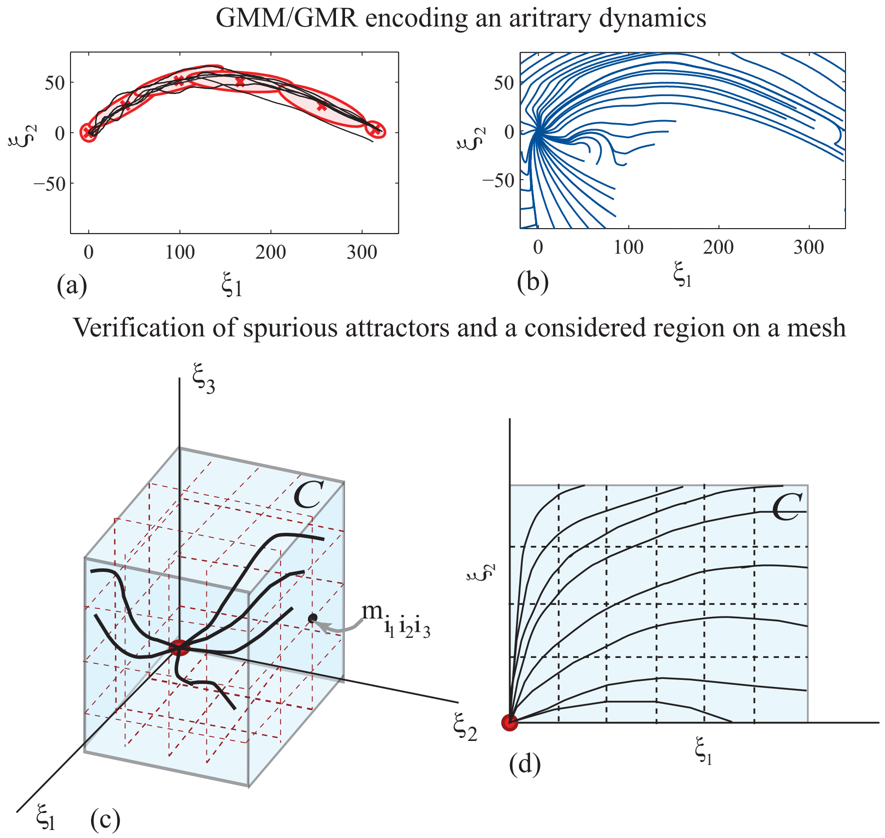


Figure 3.15: I. Illustration of a GMM/GMR encoding of an arbitrary dynamics. *Top left*: Two-dimensional projection of the data with superimposed the Gaussian Mixture envelope. Top right: All trajectories regenerated using Gaussian mixture regression when starting from 20 different locations in space converge correctly to the the origin, the attractor of the system. *Bottom left and right*: in blue (light grey in a black-and-white version), the region of applicability C that embeds all demonstrated trajectories. To empirically determine if C is a region of attraction, C is sampled equally and one measures if all trajectories originating from each of sampled point converges correctly to the target.

system is linearized in a neighborhood of an attractor and the asymptotic stability of the attractor; (2) the region of attraction of a particular attractor is estimated.

In the general case of multivariate non-linear systems, theoretical estimation of the region of attraction is still an open problem. In practice, to evaluate whether a region of interest is contained within a region of attraction, one relies on numerical procedures. We suggest an algorithm for estimating a region of attraction in Section 3.3.2.4.

To verify that a learned dynamical function $\hat{f}(\boldsymbol{\xi})$ is stable around an attractor, we start from the observation that GMR approximates $\hat{f}(\boldsymbol{\xi})$ as a non-linear weighted sum of linear dynamical systems; see Eq. 3.29. Stability of the system $\hat{f}(\boldsymbol{\xi})$ depends on the learned parameters (the matrices $\boldsymbol{A}_k$, $\boldsymbol{B}_k$ and mixing coefficients h_k). We demonstrate how to modify the GMMs procedure so as to ensure that $\hat{f}(\boldsymbol{\xi})$ is locally stable around the attractor (and consequently around the target).

Local stability at the attractor

Let us assume that, in the neighborhood of the attractor, the system $\hat{f}(\boldsymbol{\xi})$ is governed solely by the last Kth Gaussian [11]. In other words, let us assume that there exists a neighborhood of the attractor, where for all points $\boldsymbol{\xi}$ the mixing coefficients $h_k(\boldsymbol{\xi})$ except for the Kth one are zeros: $\exists B(\epsilon)$such that $\forall \boldsymbol{\xi} \in B(\epsilon) \;\; h_k(\boldsymbol{\xi}) \simeq 0, \;\; k = 1..K-1$, where $B(\epsilon)$ is a hypersphere of radius ϵ. In this region, the system governed by Eq.3.29 reduces to:

$$\dot{\boldsymbol{\xi}} = \boldsymbol{A}\boldsymbol{\xi} + \boldsymbol{B} \tag{3.31}$$

with $\boldsymbol{A} = \boldsymbol{\Sigma}_{K,\dot{\boldsymbol{\xi}}\boldsymbol{\xi}} \boldsymbol{\Sigma}^{-1}_{K,\boldsymbol{\xi}}$ and $\boldsymbol{B} = \boldsymbol{\mu}_{K,\dot{\boldsymbol{\xi}}} - \boldsymbol{A}\boldsymbol{\mu}_{K,\boldsymbol{\xi}}$.

The system driven by Eq. 3.31 is asymptotically stable if the eigenvalues of the symmetric matrix $\tilde{\boldsymbol{A}} = (\boldsymbol{A} + \boldsymbol{A}^T)/2$ are all strictly negative. For a $m \times m$-dimensional matrix to be negative definite, all its i-th order leading principal minors should be negative if i is odd and positive if i is even. Stability, therefore, is guaranteed when the following set of constraints is satisfied:

$$\begin{aligned} &\|\tilde{\boldsymbol{A}}_{[1:i,1:i]}\|(-1)^i < 0 \quad \forall i = 1,...,m \text{ that is satisfied if} \\ &(1)\; \tilde{a}_{ii} < 0 \text{ and } (2)\; \tilde{a}_{ij} \ll \tilde{a}_{ii} \;\; \forall \;\; i,j = 1,...,m \text{ and } \; i \neq j, \end{aligned} \tag{3.32}$$

where $\tilde{\boldsymbol{A}} = \{\tilde{a}_{ij}\}_{i,j=1}^{N}$.

Fig. 3.16 provides a geometrical illustration of how the stability conditions in Eq.3.32 affect the shape of the Kth Gaussian distribution in the mixture. When projected on the $\{\boldsymbol{\xi}, \dot{\boldsymbol{\xi}}\}$ plane, a Gaussian distribution corresponds to an ellipse, whose principal axis forms a negative slope. This results in a flow of motion toward the attractor along all dimensions. For the EM training to result in such an elongated Gaussian, training data must homogeneously cover the space around the target. This means that one should demonstrate the robot how to approach the target by starting from locations all around the target. In practice, as the training set is finite and gives only a partial

---

[11] In practice, as we seek to avoid the over-fitting, the Gaussians are set apart sufficiently, therefore at the origin the influence of all other Gaussians except for the last one becomes zero (up to numerical precision).

coverage of the state space, an estimate $\hat{f}(\boldsymbol{\xi})$ will be imprecise: it will exhibit both an undesired rotation of the principal axis and a shift of the attractor's location, see Fig. 3.16. Additional measures, thus, should be taken to guarantee the asymptotic convergence to the target. In the next section, we describe the practical implementation of such measures.

3.3.2.4 A Practical Approach to Ensure and Analyze Stability

1. Ensure local stability empirically.

   To compensate for the lack of data around the origin, we suggest to generate an additional *synthetic* data set by rotating a subset of training data selected within a small neighborhood of the attractor [12]. We also set the center of the last Gaussian of the GMM at the attractor ($\boldsymbol{\mu}_{K,\boldsymbol{\xi}} = \boldsymbol{\mu}_{K,\dot{\boldsymbol{\xi}}} = 0$), and do not update this center during training. This procedure is illustrated in Fig. 3.16.

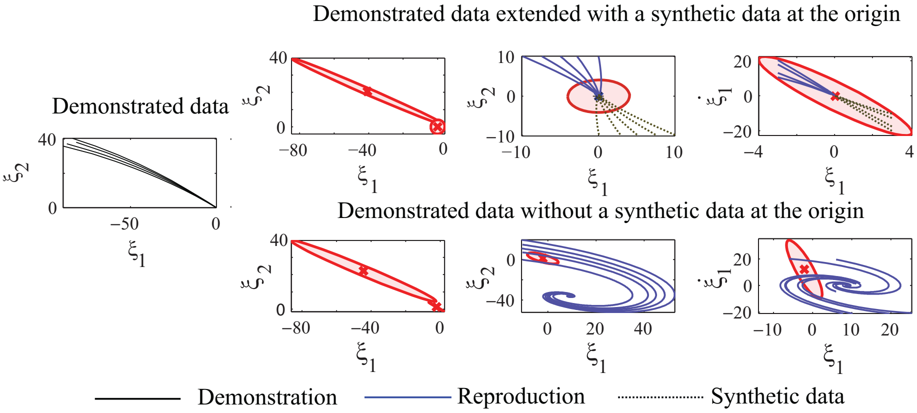


Figure 3.16: Accurate positioning of the Gaussian distribution at the attractor affects stability of the learned $f(\hat{\boldsymbol{\xi}})$. *Top*: the last Gaussian is positioned at the origin through the addition of synthetic datapoints. This modification guarantees asymptotic stability in the neighborhood of the attractor: the trajectories converge to the origin (the very right graph). *Bottom*: though the observed demonstrations converge to the origin (the very left graph), the EM training does not position the last Gaussian at the attractor automatically. Therefore, a motion generated by the learned dynamical representation $f(\hat{\boldsymbol{\xi}})$ converges to the spurious attractor (the very right graph).

   A function $f(\hat{\boldsymbol{\xi}})$ generated through this procedure is ensured to be asymptotically stable within a neighborhood around the origin. Next, we describe a procedure to empirically estimate the boundaries of the region of attraction C.

2. Empirical estimation of the region of attraction.

   As mentioned in Section 3.3.2.1, estimating dynamics in the whole state-space X is impractical. Instead, we will estimate stability locally within a subset $C \subset X$.

---

[12] If a dimensionality of a state $\boldsymbol{\xi}$ makes a design of a rotation function prohibitively complex, one can sample additional data from the last Gaussian distribution and include these data into the training set. Indeed, sampling in this case is equivalent to rotating data around the origin

C includes training data points and lies inside the robot's workspace. Initialization of C is data-driven: size of the initial C along each dimension is defined by the amplitude of the training dataset along this dimension.

After training, the initial guess regarding the boundaries of C needs to be adjusted so as to ensure that C is the region of attraction and that it does not include any other attractors. We follow a numerical procedure in which we integrate trajectories forward starting from a uniform mesh defined on the boundaries, and verify that all the trajectories converge towards the attractor.

To do this, we construct a mesh M covering boundaries of C: $M(\tau_1..\tau_N) = \{(\xi^1_{i_1}..\xi^N_{i_N}) = (i_1\tau_1..i_N\tau_N), i_1 = 1..n_1, ..., i_N = 1..n_N\}$, where $\tau_1 = c_1/n_1..\tau_N = c_N/n_N$, c_1 .. c_N – size of each of dimensions of C; n_1.. n_N – size of the mesh along each of dimensions in $\mathbb{R}^N$ (see Fig. 3.15-II). We integrate trajectories starting from each node $(\xi^1_{i_1}..\xi^N_{i_N})$ on the mesh M and verify that the velocity is zero only at the attractor, therefore it is ensured that the region of attraction C contains a single attractor. If this condition is satisfied all trajectories starting inside C will not leave the boundaries, due to the properties of differential equations.

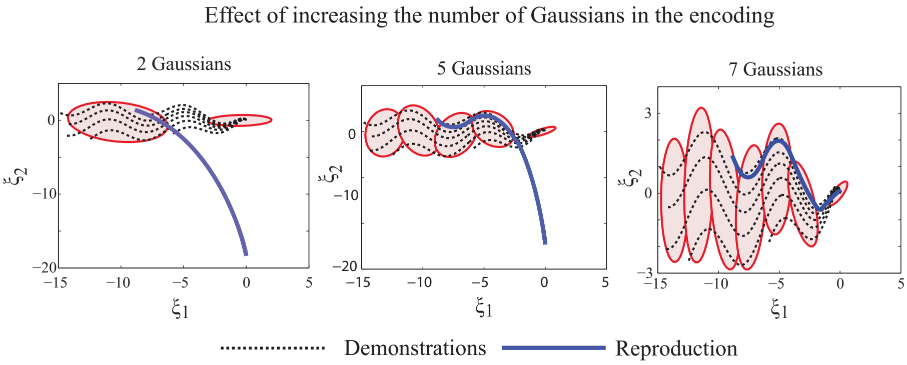


Figure 3.17: Improvement in the stability of approximation with the increase in the number of Gaussian components

To improve accuracy and extend the region of attraction, we increment the number of Gaussians K and re-estimate the system using EM; see Fig. 3.17. As instabilities often result in the motions that exit the desired trajectory (e.g. if there are sharp turns in the trajectory that have been poorly approximated by the mixture), increasing the granularity of the encoding ensures that the system will be better guided along the various non-linearities of the trajectory.

Table 3.4 summarizes the steps of the complete procedure by which we iteratively test and re-estimate the dynamical function $\hat{f(\boldsymbol{\xi})}$ so as to improve and ensure its local stability within the domain C.

3.3.3 EXPERIMENTAL RESULTS

Table 3.4: Model Training

```
Collect a dataset of demonstrations and initialize C.
Add synthetic data around the target
Choose an initial number of GMM components $K$
($K = 2$ in the experiments reported here)
LOOP until stable approximation is found
    Train the joint probability $p(\boldsymbol{\xi}, \dot{\boldsymbol{\xi}})$ with Expectation Maximization (Dempster et al., 1977):
    Verify local stability at the origin Eq. (3.32)
    IF (the origin is not asymptotically stable)
    THEN increase the number of GMM components $K = K + 1$
    ELSEIF (estimate of $C$ does not include all training trajectories) OR
    ($\exists$ spurious attractors inside the region $C$)
    THEN add training data AND retrain
    END
END
```

To validate the performance of the proposed method without blurring the results with noise inherent to human demonstrations, we first test the ability of our method to reconstruct known theoretical dynamical systems. With a known system we may generate a noise-free training set, learn an approximation of the dynamics, and compare how well the learned dynamics approximate the real one. Further, we verify the applicability of the method in robotics by teaching three different robots manipulation tasks. We report on each of these next.

3.3.3.1 LEARNING THEORETICAL DYNAMICS

The method is validated to estimate four two-dimensional dynamical systems (*Systems 1-4*) and one three-dimensional dynamical system (*System 5*), each of them contains different number of attractors and exhibits different stability properties. In each case, we generate six trajectories using the theoretical dynamics and use these for training the GMM. When the dynamical system has more than one asymptotically stable attractor, trajectories are generated only in the subpart of the state space around one of them.

Note, the legend for Figures 3.18 - 3.22 is described in Fig. 3.13. Each of the figures encompasses, in the first row, plots giving a general view of the original dynamics with vector fields (a) and three-dimensional phase plots (b-c), in the second row, a view of the GMM superimposed to the training data, and in the 3rd row, vector field (a) and phase plots (b-g) of the the estimated dynamics superimposed on the original dynamics.

System 1.

$$\begin{aligned} \dot{x}_1 &= -x_1 + 2x_1^2 x_2; \\ \dot{x}_2 &= -x_2. \end{aligned} \tag{3.33}$$

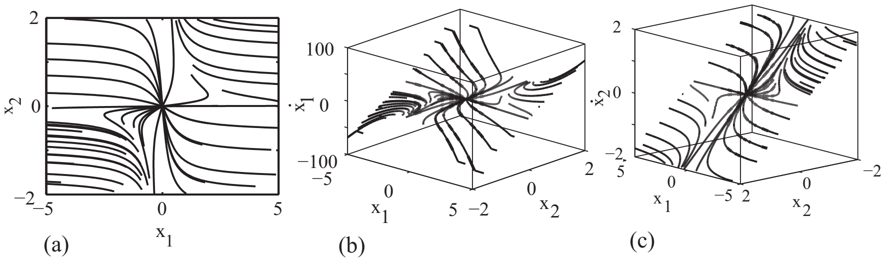
The system has a single locally asymptotically stable equilibrium point at the origin. We approximate the dynamics of this system in a region $[-4; 0] \times [0; 2]$, where it is locally asymptotically stable. Results are presented in the Fig. 3.18.

System 2

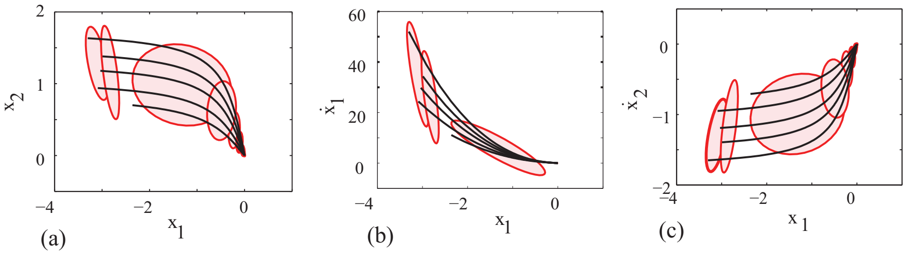
$$\begin{aligned} \dot{x}_1 &= 700 - 2x_1 + 200x_2 e^{\frac{25x_1 - 10^4}{x_1}}; \\ \dot{x}_2 &= 1 - x_2 - x_2 e^{\frac{25x_1 - 10^4}{x_1}}; \end{aligned} \tag{3.34}$$

The system has two equilibrium points – one asymptotically stable ($x_1 = 335; x_2 = 0.089$) and one unstable ($x_1 = 489;\ x_2 = 0.5$). We approximate the dynamics in the region $[0; 400] \times [-2; 2]$, where it is locally asymptotically stable. Results are presented in Fig. 3.19.

I. Actual Dynamics



II. Training data and GMM encoding (7 Gaussians)



III. Reproduction

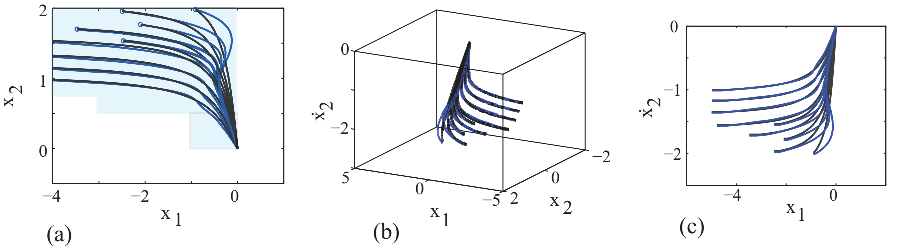
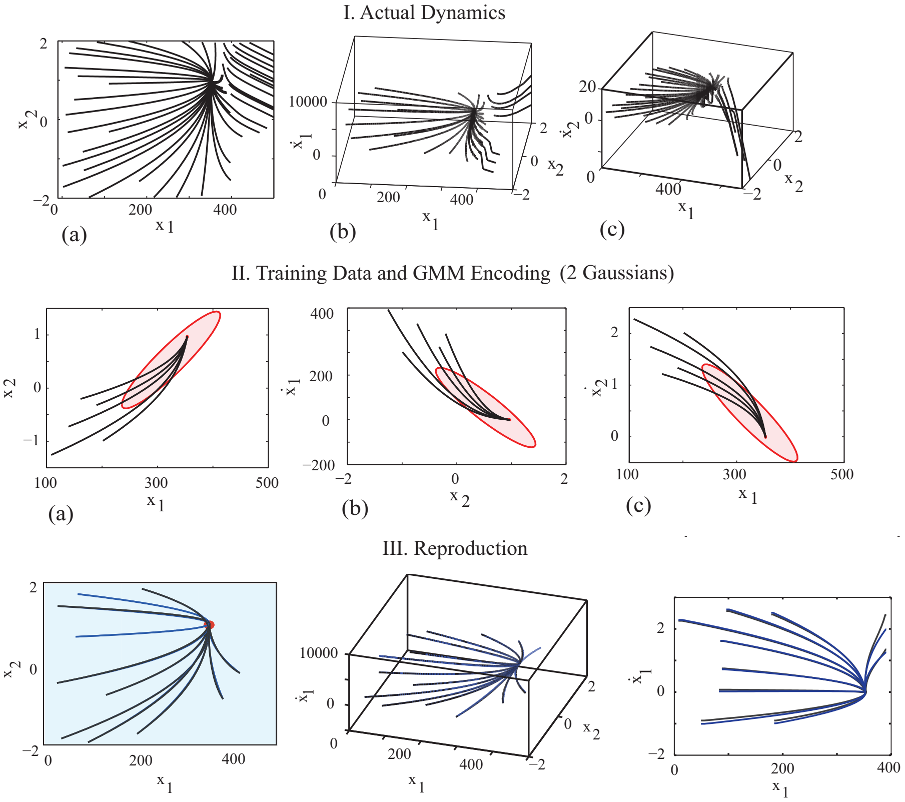


Figure 3.18: *System 1*. The proposed method encodes this system with 7 Gaussians; the learned system exhibits good precision in the area covered by demonstrations, outside this area the precision is also admissible except for a region in the direct proximity to y-axis, where actual trajectories represent an excess curvature as approaching to the equilibrium, e.g., a trajectory starting at the bound $x_2 = 2$. In this region, a flat part of trajectories is reproduced well, though the steep parts that were not demonstrated are attracted towards the region covered by the training set.



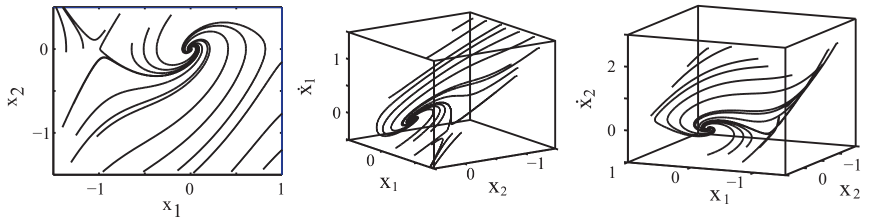
I. Actual Dynamics

II. Training Data and GMM Encoding (2 Gaussians)

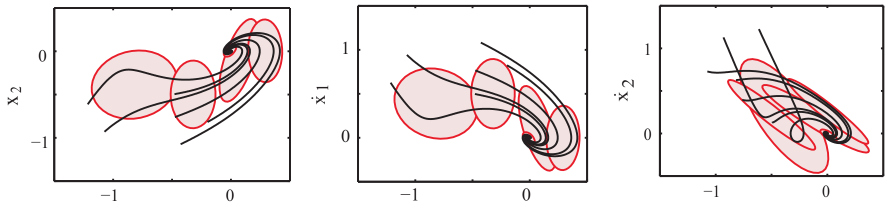
III. Reproduction

Figure 3.19: *System 2*. As the behavior of the system in the considered area is relatively simple, 2 Gaussians are sufficient to achieve the good performance, even in areas unseen during demonstration. Interestingly, the learned dynamics is extrapolated very well beyond the area covered by the training set.

I. Actual Dynamics

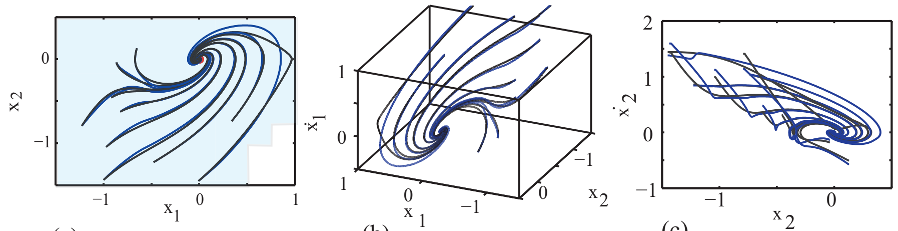


II. Training Data and GMM Encoding (6 Gaussians)



(a) (b) (c)

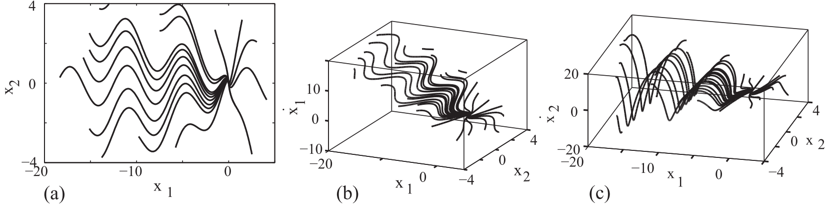
III. Reproduction



(a) (b) (c)

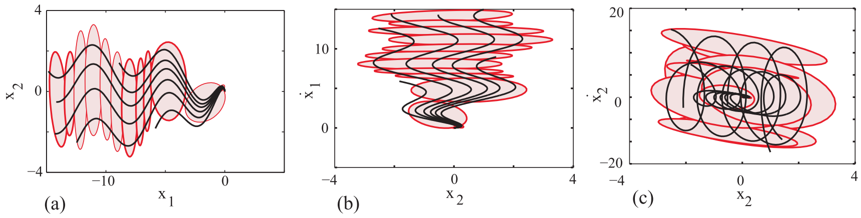
Figure 3.20: *System 3*. Despite strong non-linearities in the observed trajectories, the dynamics is successfully approximated with 6 Gaussians. Note, even unseen, circular shape trajectories (starting around $x_2 \approx 0$) are reproduced correctly in both position and velocities spaces.

I. Actual Dynamics



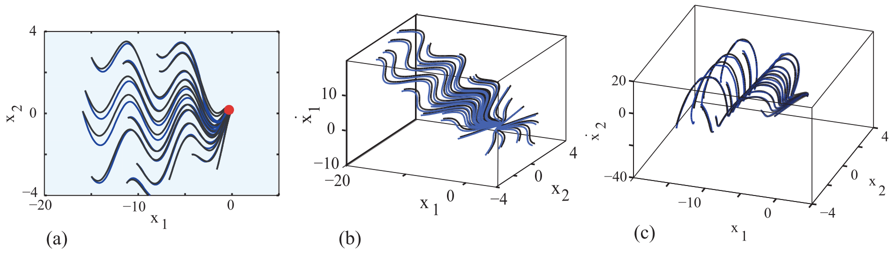
(a) (b) (c)

II. Training Data and GMM Encoding (13 Gaussians)



(a) (b) (c)

III. Reproduction



(a) (b) (c)

Figure 3.21: *System 4*. The system is strongly non-linear, 13 Gaussians are necessary to achieve a good precision in the considered region. Complex dynamics and increased number of Gaussians lead to less strong generalization abilities of the method. Indeed, trajectories started beyond the region covered by the training set tend to depart from the real trajectories generated by the dynamics, it is particularly noticeable in the velocity space, see section III-(g). However, even in this non-trivial case the system generates admissibly good results (the reproduced trajectories follow an observed motion pattern) from few demonstrations.

System 3

$$\begin{aligned}\dot{x}_1 &= -x_2; \\ \dot{x}_2 &= x_1 - x_1^3 - x_2;\end{aligned} \tag{3.35}$$

The system has three equilibrium points - two unstable ($x_1 = -1; x_2 = 0$ and $x_1 = 1; x_2 = 0$) and one asymptotically stable $x_1 = 0; x_2 = 0$. We approximate the dynamics of this system in a region $[-1.5; 1] \times [-1.5; 0.5]$, where it is locally asymptotically stable. Results are presented in Fig. 3.20.

System 4

$$\begin{aligned}\dot{x}_1 &= -x_1; \\ \dot{x}_2 &= -x_1 \cos x_1 - x_2;\end{aligned} \tag{3.36}$$

The system exhibits strong nonlinearity due to the cosine term; the system is globally asymptotically stable and converges asymptotically to the origin. We approximate the dynamics of this system in a region $[-20; 0] \times [-4; 4]$. Results are presented in Fig. 3.21.

System 5

$$\begin{aligned}\dot{x}_1 &= -x_1 - x_2 + x_3^2; \\ \dot{x}_2 &= x_1 + 10 \cos x_2 x_2 - x_3^2; \\ \dot{x}_3 &= x_1 + 2x_2 - x_3;\end{aligned} \tag{3.37}$$

Locally asymptotically stable three-dimensional dynamics with a single attractor at $[12.98; -7.75; -2.5213]$. We approximate the dynamics of this system in a region $[-20; 30] \times [-11; -5] \times [-10; 2]$. Results of the learning process are presented in the Fig. 3.22.

1. Quantification and discussion of results

   Quantification of results achieved on both theoretical systems and actual robotic motions are presented in Table 3.6; quantitative measures, that have been applied, are defined in Table 3.5. As it can be seen all systems result in a coarse representation of motion dynamics through a relatively small number of Gaussians (the *NbGaussians* column in Table 3.6). Moreover such a sparse representation achieves the good precision for both positional and velocity profiles when reproducing the actual dynamics.

   As shown in Fig. 3.18-3.22, the system can generalize (reproduce a learned pattern on unobserved states) outside the training domain (inside the stability domain as discussed below). This property is particularly useful for practical applications as it enables the prediction of the system behavior outside the region covered during training, hence reducing the amount of training data required. In the examples covered here, only 6 training trajectories were required in each

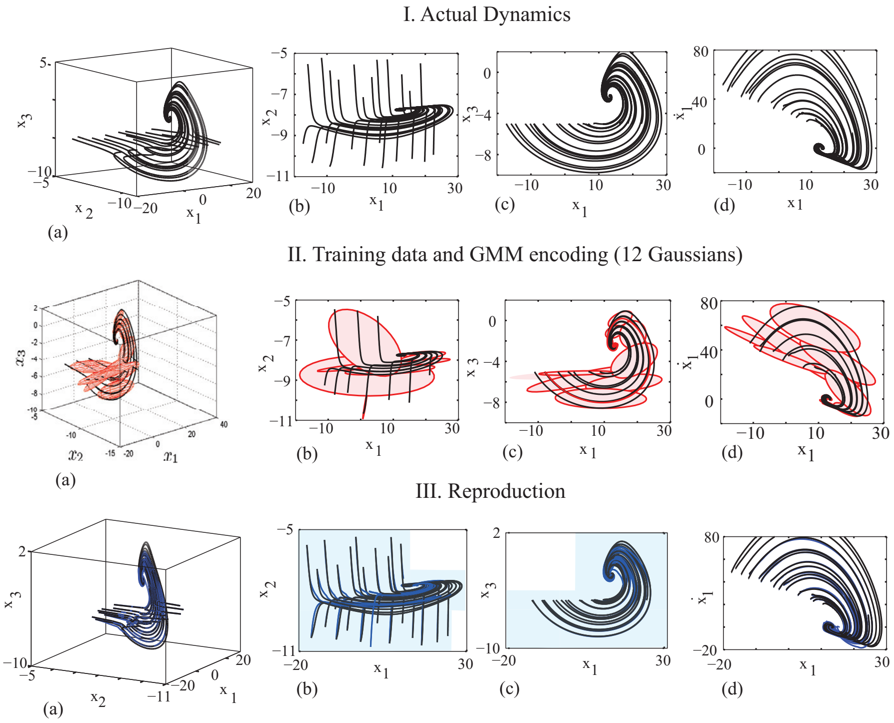
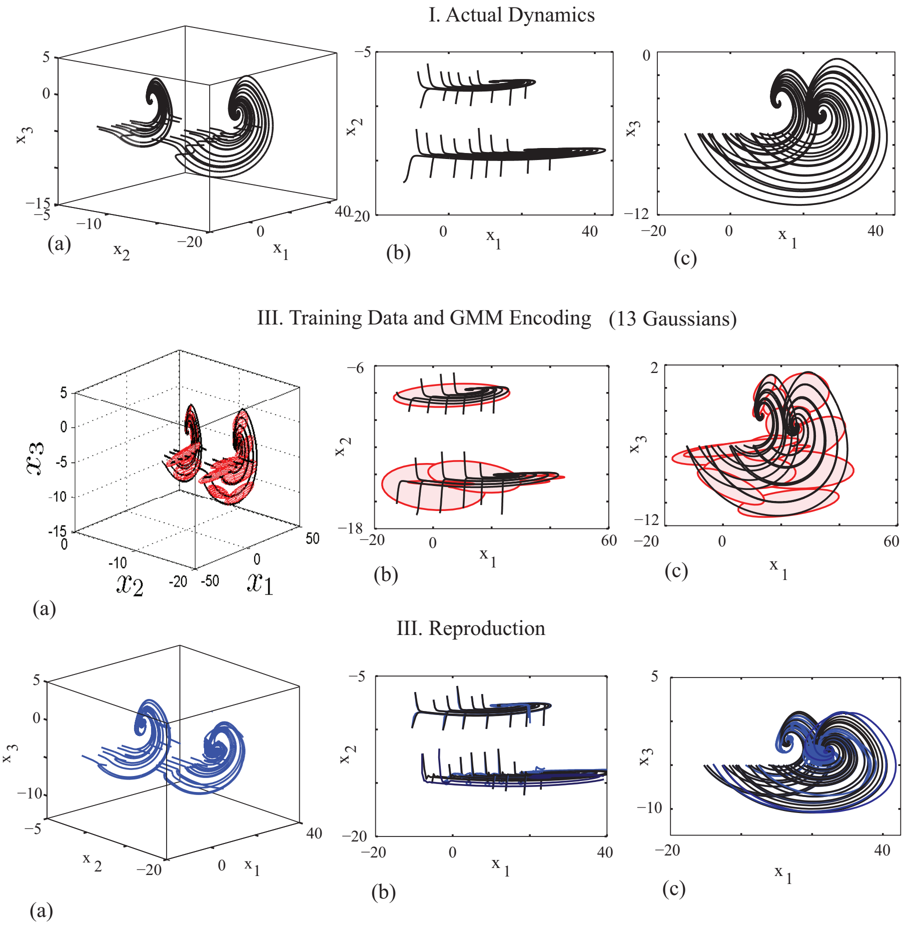


Figure 3.22: *System 5*. A strongly non-linear three-dimensional dynamical system. In this case, a slight increase in a number of demonstrations allows for the accurate approximation and generalization.



I. Actual Dynamics

III. Training Data and GMM Encoding (13 Gaussians)

III. Reproduction

Figure 3.23: Learning motion with two attractors. 3-dimensional trajectories are generated by *System 5* that displays a periodic behavior. Trajectories were demonstrated in the neighborhood of two asymptotically stable attractors. During the reproduction, the system managed to accurately reproduce dynamics around both attractors.

case.

Note that, since the dynamics is learned from data covering only a subpart of the domain, it does not necessarily have the same attractor landscape and the region of attraction across the complete domain as the original system, even if it accurately approximates the original system locally. For example, in System 3, the original dynamical system has three equilibrium points, while its approximation has a unique asymptotically stable equilibrium. To overcome this, one may provide additional demonstrations covering dynamics in the neighborhood of the other equilibriums: Fig. 3.23 presents results of learning the dynamics around the two different attractors of *System 5*. The demonstrations are provided in the neighborhood of the two asymptotically stable attractors; during learning, positions of two Gaussians are fixed on the attractors, and the algorithm runs to verify local asymptotical stability of both attractors. The regions of approximation C is analyzed separately for each attractors. The learned system manages to grasp the complex dynamics accurately; furthermore, it separates the two flows of trajectories based on where in the workspace the motion starts. In addition to stability of reproduction (see Fig.3.24), one should keep in mind that the considered region of applicability should not exceed a region where the likelihood of the input position allows for the confident inference of the velocity. In Fig. 3.25 we depict how the likelihood changes beyond the region covered by the training

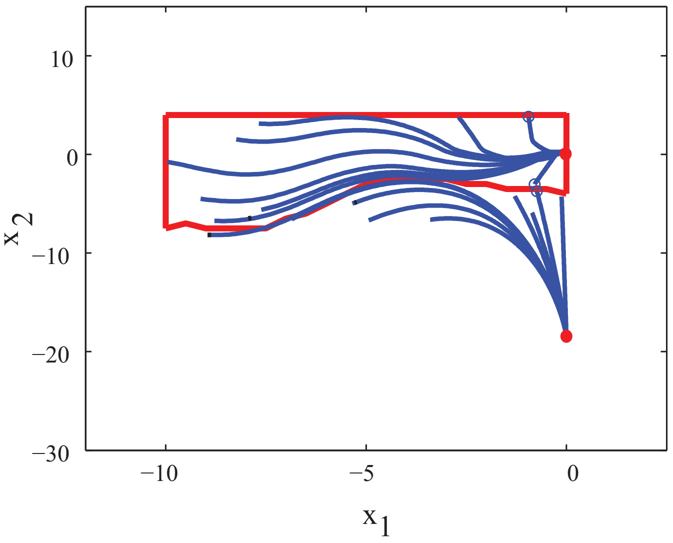


Figure 3.24: A numerically estimated region of applicability of the System 4 (the red/black(in a black-and-white version) frame). An actual and spurious attractors are highlighted with circles. Note, the numerical method estimated a lower bound that goes along a trajectory with a good precision. Other bounds were left unchanged, i.e., in the other directions the considered region does not cross boundaries of the region of applicability.

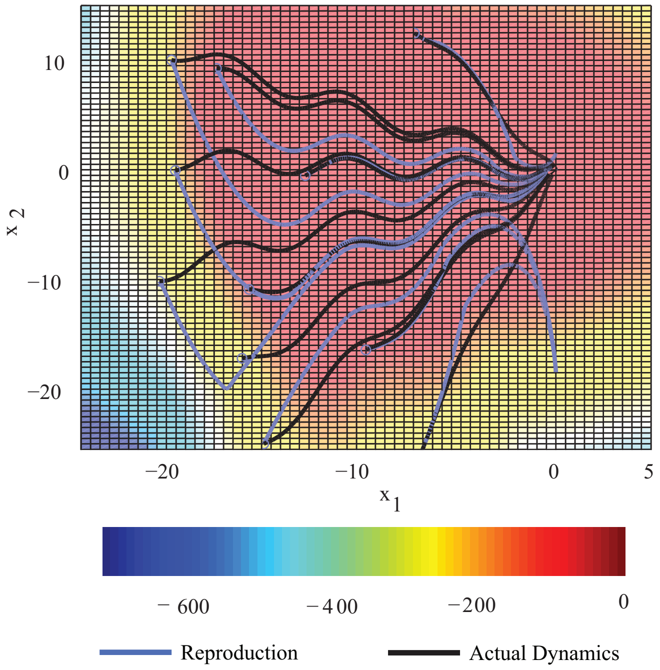


Figure 3.25: Extrapolation properties of the GMMs encoding (better see in color). A color map reflects changes in values of the the likelihood (3.38) of datapoints, the dark-red (dark grey in the center) area represents an area of the most reliable inference regarding the velocity. For reconstructed trajectories starting outside this area, the deviation from the actual dynamics may be considerable. Interestingly, in the region of attraction of the origin, trajectories are strongly attracted towards a region covered by the training set. It is a useful property for practical applications as it enables the prediction of the system behavior outside the region covered during training, hence reducing the amount of training data required.

set. Likelihood was computed as follows:

$$\mathbf{L}(\boldsymbol{\xi}) = \log\,[\max_i h_i(\boldsymbol{\xi})]. \tag{3.38}$$

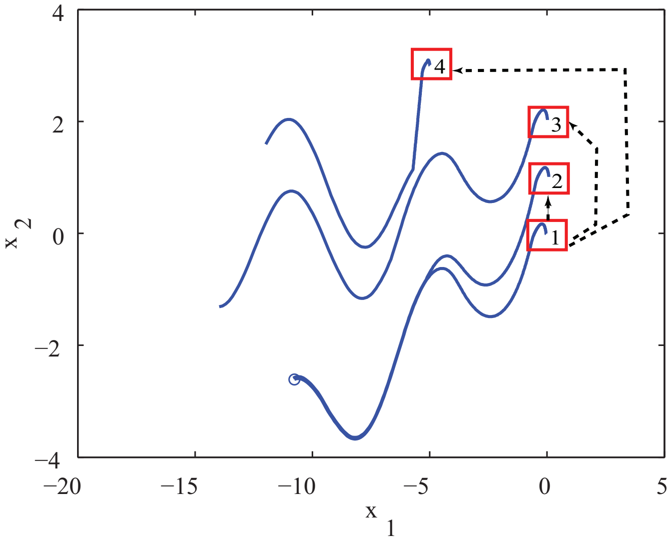


Figure 3.26: Robustness to perturbations. The target is shifted several times (to positions 2, 3, 4) after the onset of motion.

L gives a measure of the maximum probability of a point ξ to belong to any of the K Gaussians. The region where L exceeds a given threshold[13] represents the region where the system can still make a confident probabilistic inference. Note that all the trajectories that start in areas where L is too small significantly depart from the observed dynamics. This is due to the nonlinear weights h_i and their effect on a velocity direction: nearby the demonstrations, the influence of the closest Gaussian dominates that of all Gaussians, hence guiding closely the motion. However, far away from the demonstrations, the influence of all Gaussians becomes comparable and the resulting direction of velocity may point away from the signal.

---

[13] We took an empirically chosen threshold of -10.

Table 3.5: Quantification of results

Notation: R is a number of generated test trajectories ($R = 20$), each trajectory $\boldsymbol{\xi}_j = \{\boldsymbol{\xi}_j^i\}_{i=1}^{M_j}$ contains M_j datapoints; ℓ_j is the length of a jth trajectory: $\ell_j = \sum_{i=2}^{M_j} |\boldsymbol{\xi}_j^i - \boldsymbol{\xi}_{j-1}^i|$. Here and further we distinguish between $\hat{\boldsymbol{\xi}}_j$ and $\boldsymbol{\xi}_j$, which denote to a learned and a theoretical trajectory respectively.

[1] To estimate the accuracy of the proposed method in the presence of noise, we add a signal-dependent noise to a theoretical dynamics (see Fig. 3.27): $\dot{\boldsymbol{\xi}} = f(\boldsymbol{\xi}) + \eta(\boldsymbol{\xi})$, where $\eta(\boldsymbol{\xi})$ is a linear function: $\eta(\boldsymbol{\xi}) = \sigma^2\gamma\boldsymbol{\xi}$, where σ^2 is a variance of signal-dependent noise ($\sigma^2 = 20$ in our case), γ is a normal random variable with a zero mean and a unit variance.

[2] MPP, minimum positional precision: MPP= $\frac{\max_{j=1..R}^{i=1..M_j} \|\hat{\boldsymbol{\xi}}_j^i - \boldsymbol{\xi}_j^i\|}{\sum_{j=1}^{\mathrm{R}} \ell_j/\mathrm{R}}$ measures the maximum point-wise deviation of a reproduced trajectory from its exact theoretical value; MPP is normalized by the average trajectory's length. For instance, if the average length of the test trajectories is 20 cm, MPP of 0.1 means that the at each time step the deviation of a reproduced trajectory from its exact value does not exceed 2cm.

[3] APP, average positional precision: APP= $\frac{\frac{1}{R}\sum_{j=1..R}^{i=1..M_j} \|\hat{\boldsymbol{\xi}}_j^i - \boldsymbol{\xi}_j^i\|/M_j}{\sum_{j=1}^{\mathrm{R}} \ell_j/\mathrm{R}}$. APP is normalized by the average length of the test trajectories R and by a number of datapoints in the considered trajectories. APP measures the average point-wise deviation of a reproduced trajectory from its exact value. For instance, if the average length of the test trajectories is 20 cm, APP of 0.01 means that the at each time step the deviation of a reproduced trajectory from its exact value is about 2mm.

[4] MVP, minimum velocity precision: MVP= $\frac{\max_{j=1..R}^{i=1..M_j} \|\dot{\hat{\boldsymbol{\xi}}}_j^i - \dot{\boldsymbol{\xi}}_j^i\|}{\sum_{j=1}^{\mathrm{R}} \ell_j/\mathrm{R}}$ measures the maximum deviation of a reproduced velocity from its exact value. MVP is normalized by the average length of velocity trajectories.

[5] AVP, an average velocity precision: AVP $= \frac{\frac{1}{R}\sum_{j=1..R}^{i=1..M_j} \|\dot{\hat{\boldsymbol{\xi}}}_j^i - \dot{\boldsymbol{\xi}}_j^i\|/M_j}{\sum_{j=1}^{\mathrm{R}} \ell_j/\mathrm{R}}$. AVP is normalized by the average length of all considered velocity trajectories and by the number of datapoints in the considered trajectories.

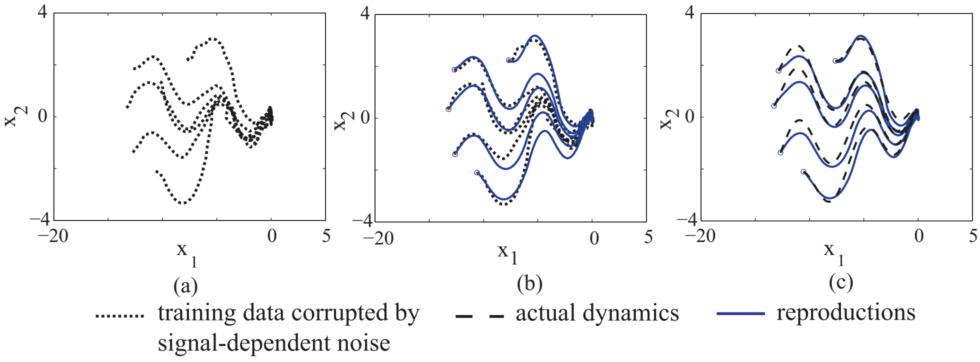


Figure 3.27: (a) Training trajectories of *System 4* are corrupted with a signal-dependent noise of variance $\sigma^2 = 20$. (b) The trajectories generated with the learned dynamics are superimposed with the training data. (c) The trajectories generated with the learned dynamics are superimposed with the trajectories of the actual dynamical system. Note, that the proposed method manages to accurately disentangle the motion dynamics from noise.

Table 3.6: Quantification of results (Continuation)

System	NbG	MPP[2]	MPP[2] noise[1]	APP[3]	APP[3] noise[1]	MVP[4]	MVP[4] noise	APV[5]	AVP[5] noise[1]
System 1	7	0.08	0.60	0.006	0.08	0.69	1.10	0.003	0.019
System 2	2	0.01	0.09	0.008	0.03	0.02	0.08	0.001	0.003
System 3	6	0.03	0.11	0.007	0.03	0.17	0.22	0.008	0.01
System 4	13	0.01	0.22	0.004	0.01	0.06	0.21	0.003	0.007
System 5	12	0.07	0.12	0.01	0.10	0.21	0.31	0.006	0.013
KATAN experiment	4		0.34	-	0.21	-	0.17	-	0.10
HOAP experiment	5	-	0.42	-	0.33	-	0.21	-	0.18

As mentioned in the introduction, an inherent property of stable dynamical systems is their robustness to spatial and temporal perturbations. Fig. 3.26 illustrates this aspect for one of the learned dynamical system, when the target is moved after the onset of the motion. As we see, the trajectories adapt smoothly to the change. Note, however, that the velocity profile may change abruptly when the perturbation occurs. To overcome this drawback it would be necessary to consider second-order dynamics.

As discussed previously, the GMMs encoding may result in spurious attractors outside the empirical stability domain C and in regions with low likelihood; see, e.g., Fig. 3.25.

There are several reasons for the emergence of spurious attractors: first, the training set gives only a partial and noisy representation of the dynamics. Providing additional data in the regions around spurious attractors usually improves greatly performance. Second, the shape of the signal influence greatly stability. For instance, if the curvature of the trajectories changes smoothly, the spurious attractors, if any, will usually lie outside of the region of the confident inference; see Fig. 3.25. However, if the system trajectories experience sharp changes in the curvature, as e.g., System 1; see Fig. 3.18, the likelihood of having spurious attractors in the considered region increases. By adding more Gaussians around the point with a sharp curvature one increases the guidance provided by the GMM and thus decreases the chances. By considering these practical shortcomings, one may improve a particular encoding to achieve the admissible performance.

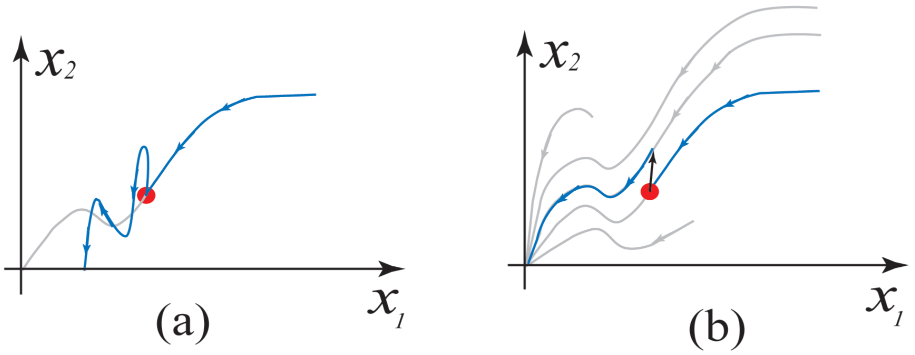
3.3.4 APPLICATION TO ROBOT CONTROL

Further, we validate the method to learn the dynamics of motion of a robot endeffector when trained through human guidance. Here, the dynamics of motion becomes the control law that iteratively moves the robot's arm along a trajectory.

3.3.4.1 ENCODING MOTION IN THE OPERATIONAL SPACE

Since the framework we defined above does not make any assumption as to the type of variables to be used for training, we are unconstrained in our choice of variables for controlling a robot. Here, we choose to describe motions according to the following variables: the translation component of motion of the end-effector is described by a vector of Cartesian coordinates $\boldsymbol{x} \in \mathbb{R}^3$.

Each demonstrated trajectory is, thus, represented by the following dataset: $D = \{\boldsymbol{x}_t, \dot{\boldsymbol{x}}_t\}_{t=1}^{M}$, where M is the number of datapoints in a trajectory. To reproduce a task, we first learn an estimate of the dynamical system using the method described in Section 3.3.2.1 and then use the Moore-Penrose pseudo-inverse to compute the corresponding joint angles. Table 3.8 summarizes the steps of the reproduction algorithm.



(a) (b)

Figure 3.28: (a) If a trajectory in the operation space passes through non-reachable joint positions IK may return velocity in the operation space that sends a robot too far from original trajectory, so linear assumptions of approximation of kinematics does not satisfy and overall trajectory tracking will fail. (b) In the case of motion encoding with a dynamical system, after perturbation the robot will not try to return to the previous trajectory violating the linear approximation of kinematics, instead the dynamical system will generate other trajectory from the point where the robot occurs.

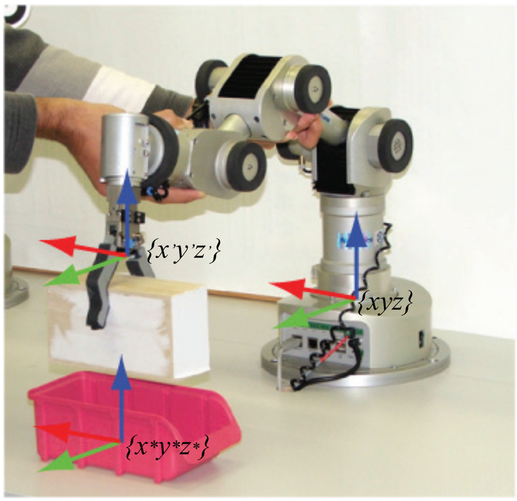


Figure 3.29: We encode tasks in a referential located at the target and moving with it$\{x^*y^*z^*\}$; this referential is expressed in the fixed global referential $\{xyz\}$(usually we choose one attached to static parts of a robot). Actually, the motion of the robot end-effector is expressed as moving a referential associated with the end-effector $\{x'y'z'\}$.

Table 3.7: On-line task reproduction

```
Assume that a controller $\hat{f}_{\boldsymbol{x}}$ has been learned,
the robot is thus ready to reproduce a task
Detect a target position in the global referential $\{xyz\}$; see Fig. 3.29: $\boldsymbol{x}^*$
Recompute the current position of an end-effector in
the target referential $\{x^*y^*z^*\}$: $\boldsymbol{x}_0$
**LOOP** until the target position is reached
    infer the velocity for the next iteration $t$ through GMR Eq.3.29: $\hat{\dot{\boldsymbol{x}}}_t$
    $\hat{\dot{\boldsymbol{x}}}_t = \sum_{k=1}^{K} h_{k,\boldsymbol{x}}(\boldsymbol{\mu}_{k,\dot{\boldsymbol{x}}} + \boldsymbol{\Sigma}_{k,\dot{\boldsymbol{x}}\boldsymbol{x}}\boldsymbol{\Sigma}_{k,\boldsymbol{x}}^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_{k,\boldsymbol{x}}))$
    solve the Inverse Kinematics problem to find: $\dot{\boldsymbol{x}}_t, \dot{\boldsymbol{q}}_t$
    compute a new position $\boldsymbol{x}_t, \boldsymbol{q}_t$
**END**
```

3.3.4.2 SET-UP

We validated the above method in three practical tasks; see Fig. 3.30, 3.35. We also implemented the theoretical 3-dimensional *System* 5, as a motion generation policy for the robot. To highlight the generic character of the approach we ran experiments with three different robotic platforms: a 6 degree of freedom industrial-like KATANA arm from Neuronics, a 4 degree of freedom robot arm of the humanoid robot HOAP-3 from Fujitsu, and a 7 degree of freedom humanoid platform i-Cub which 7 degree of freedom of the right arm have been used; see Fig. 3.35-(a).

For KATANA and Hoap demonstration is accomplished through kinesthetic teaching. In the case of iCub, as the motors are not back-drivable, demonstration is accomplished via teleoperation of the robot arm by a human teacher. The simultaneous control of all 7 degrees of freedom is conducted through a joint recording system placed on the human. A mapping from human to robot arm allows the human to directly control the motion of the robot arm. Measurements from the motion sensors are mapped in real-time into the robot joint commands, therefore, the human teacher is getting immediate visual feedback regarding accuracy of demonstrations he/she provided. While moving the robot records observations, taken from its own sensors. In detail, sensing units from commercial $XSens$ joint recording system are placed on the upper and lower arm, and back of the palm, of the human; see Fig. 3.35. During the reproduction i-Cub was controlled in real-time at the frequency of 50Hz. An external color-blob tracking vision system was used to detect the position of the ping-pong ball.

3.3.4.3 EXPERIMENTS WITH KATANA

The first experiment consists in the KATANA putting an object into a container. Here, the KATANA arm was taught to put a rectangular wooden brick into a rectangular container; see Fig.3.30-left.

In the second experiment, the KATANA was controlled with *System 5* with the origin of the system positioned on an arbitrary object. This experiment meant to test the ability of the learned system to *generalize* to context unseen during training and to quickly adapt to perturbations.

This experiment meant to show that the theoretical dynamical system could be of practical use to guide robot motions for a simple reaching task. It also demonstrate that, as shown in simulation, the system is stable and follows the trained (and known) dynamics of motion.

3.3.4.4 EXPERIMENTS WITH HOAP-3

The clench of the HOAP-3 is rather small, therefore it can grasp only thin objects. In this task the robot had to grasp a box which is thin along one dimension, so the robot should follow a specific path to properly position its hand; see Fig.3.30-right.

During training, the robots were shown the tasks 5 times by a human user guiding their arms. Values of the robots joints were recorded during this passive motion and used for reconstructing the position of the end-effector.

3.3.4.5 EXPERIMENTS WITH ICUB

The experiments with iCub aim to demonstrate the abilities of our proposed approach to (1) *generalize* a motion to unseen conditions (e.g., if the ball is placed at locations different than these seen during training), and (2) adapt to temporal perturbations (the varying duration of a motion). To emphasize the importance of time-independency and of the state-space representation of motions, we compare the robot performance when its trajectories are learned with our proposed approach and when these are encoded by Dynamic Movement Primitives (Hoffmann, Pastor, et al., 2009; Ijspeert, Nakanishi, & Schaal, 2001; Pastor et al., 2009). We choose Dynamic Movement Primitives for comparison, as it appears to be the closest Robot Learning method that exploits the dynamical system view on the motion production. It is also the only approach that is proved to be asymptotically stable at the target.

An iCub robot learns how to reach a ping pong ball with a forehand motion and to stop at the target. The experiment deliberately replicates the task of reaching for a ball with a tennis racket from the original paper of Ijspeert, Nakanishi, & Schaal (2001) (where the velocity at the target is also zero).

3.3.4.6 RESULTS OF LEARNING DYNAMICS FROM MOTION DATA

After training, the robots is requested to reproduce the tasks under different spatial conditions. The results of the experiments are summarized in Fig. 3.32-3.31. To test the generalization abilities and the robustness to perturbations, we conduct experiments

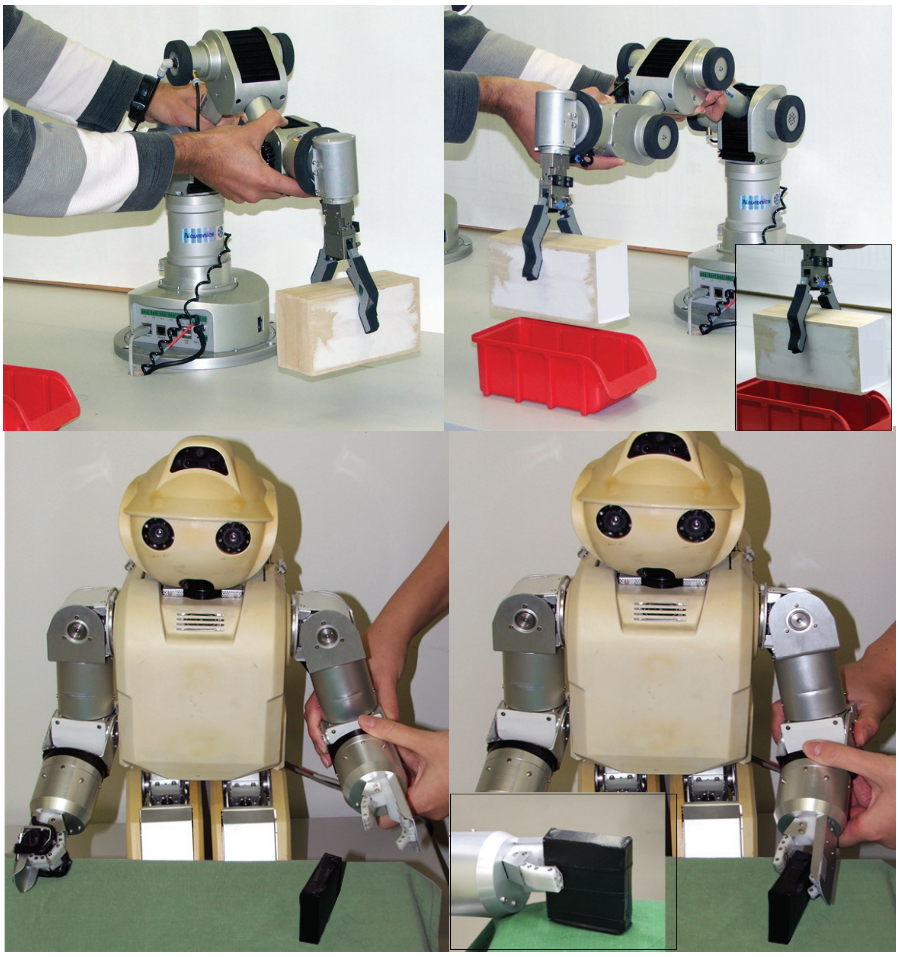


Figure 3.30: Set-up of the experiments. Left: KATANA puts a wooden brick into the container, to achieve the task the robot should lift the brick and move it following an elevating trajectory. Right: HOAP-3 grasps a box, to accomplish this task HOAP should approach the box with a specific orientation and than lower its arm, as the clench is small, see small figure in the corner.

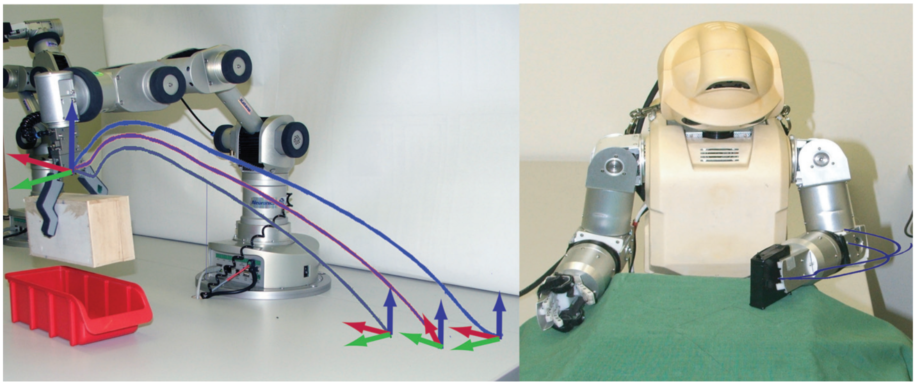


Figure 3.31: The results of reproduction of dynamically generated trajectories on the robots. To check the generalization abilities of the learned dynamics the trajectories were reproduced from different initial positions.

starting from different initial positions of the robots. Also, the locations of the the container (for the KATANA's experiment) and of the box (for the HOAP-3's experiment) have been modified. Results are presented in Fig. 3.31; in both experiments learning of position control is successful: the robots reach the targets accurately and accomplish the tasks. For the second experiment, where KATANA reproduces *System 5*, results of generalizing the task to the unseen context are presented in Fig. 3.33 - II. The area where demonstrations are provided is in red. Our method further reproduces the motion starting from anywhere in the sub-space highlighted in gray. Note, that even when learned from a few demonstrations, the algorithm enables the good generalization. The ability to generate a trajectory from an arbitrary initial position with a relevant velocity profile is an advantage of encoding motions with dynamical systems. Furthermore such a state-space encoding provides online adaptation to the target perturbations. Fig. 3.33-I presents the results of tracking a marked object, which position is mapped into the attractor of the learned task model. Even after the target has been shifted several times, the robot still reaches the object, and this while following a demonstrated spatial and velocity profile.

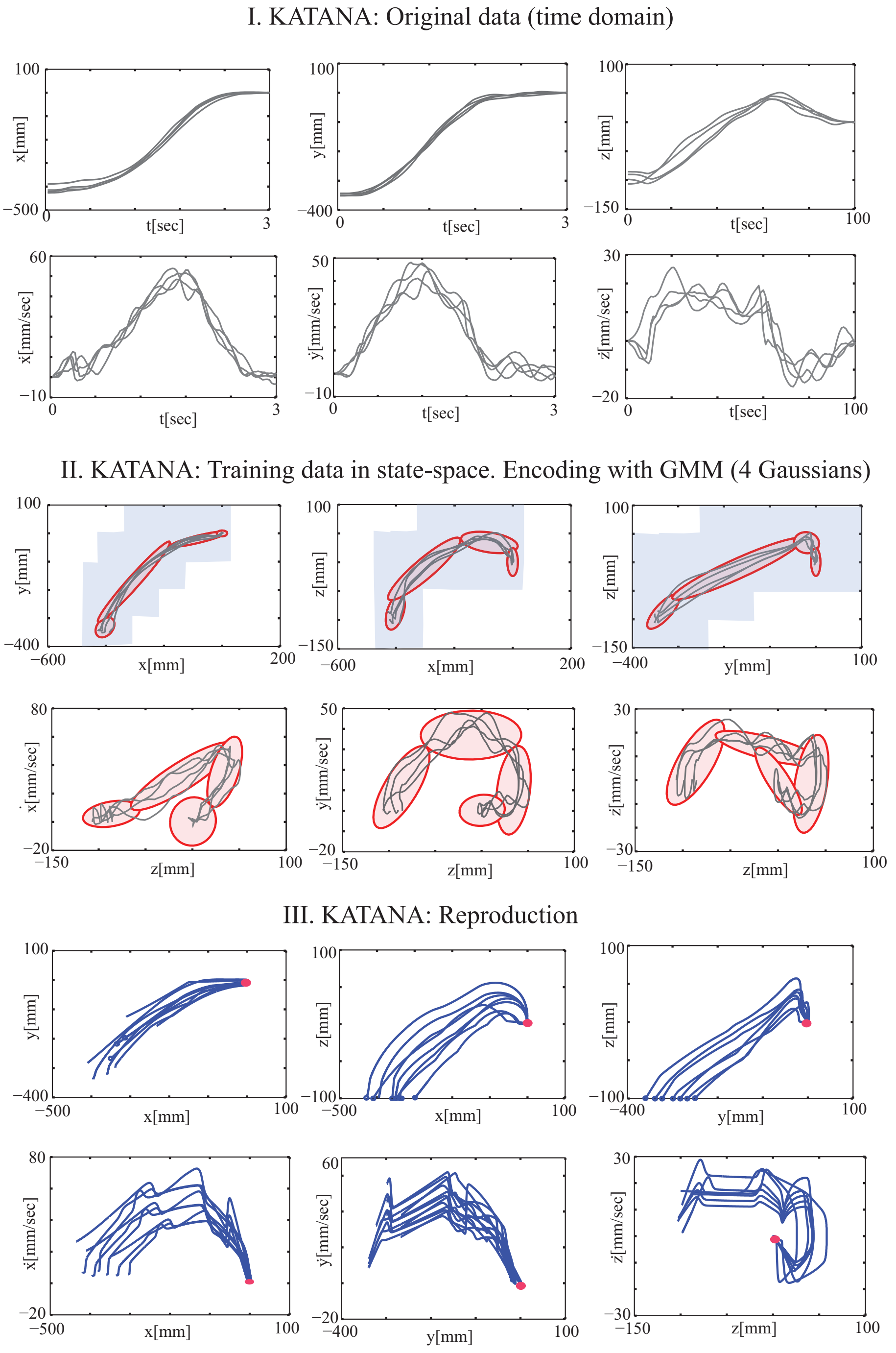
3.3.4.7 COMPARISON WITH DYNAMIC MOVEMENT PRIMITIVES

In addition to the theoretical comparison in Appendix II, we experimentally compare the performance of our method with that of Dynamic Movement Primitives (DMP). The strength of DMP and of its recent modifications (Hoffmann, Pastor, et al., 2009; Pastor et al., 2009) consists in the ability of the method to learn a motion representation from a single demonstration. A considerable deformation of the motion, if it starts from an unobserved location, is one of the pitfalls. DMP proceeds by modulating a *linear* stable dynamical system with a learned acceleration profile. The modulation function is dependent on the internal clock (the canonical variable s; see Appendix II for details). This implicit time dependency makes the system sensitive to perturbation as we demonstrate here. Note that improvements offered on the method by Pastor et al. (2009) and Hoffmann, Pastor, et al. (2009) do not resolve the time-dependency issue that we tackle in our approach.

Before proceeding with our discussion, we shall emphasize theoretical differences between our method and DMP, so as to motivate our choice of qualitative criteria for comparison.

In DMP, a modulation function $\hat{f}(s)$ is learned either using LWR (Atkeson et al., 1997) or LWPR (Vijayakumar & Schaal, 2000) from a *single* demonstration. In contrast, our approach learns a task model from *several* demonstrations, encoding these in the *state-space*. Learning in the state-space is particularly an important difference between the two methods. Being controlled with a state-space task model, a robot receives a feedback signal that enables it to continuously adapt the velocity depending on the current position.

Another difference relates to how the methods process the training set. In DMP,

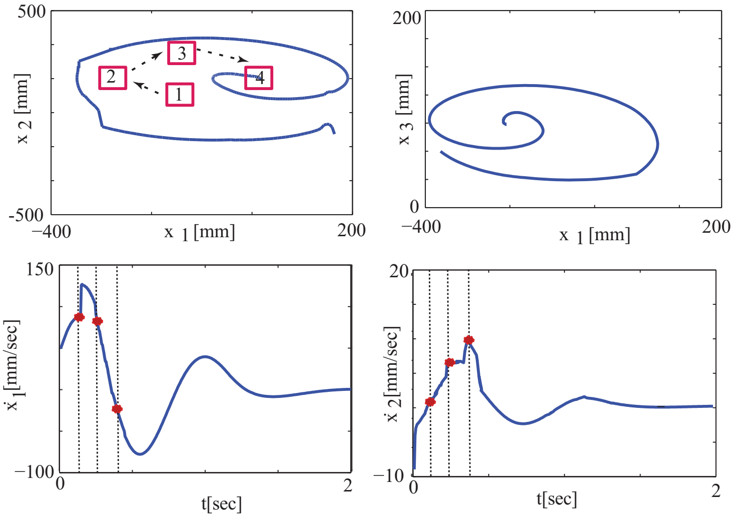

I. KATANA: Original data (time domain)

II. KATANA: Training data in state-space. Encoding with GMM (4 Gaussians)

III. KATANA: Reproduction

Figure 3.32: KATANA experiment 1: Results of encoding and reproduction of the experiment where KATANA had to put a brick into a container.

I. KATANA experiment 2: real-time adaptation to perturbation



II. KATANA experiment 2: generalization to the unseen context

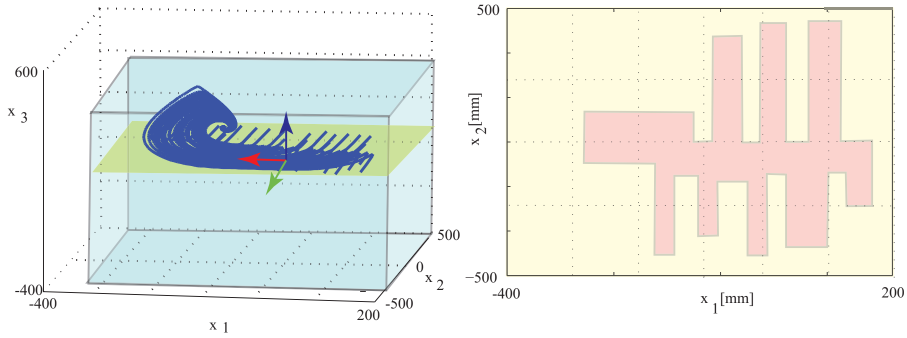
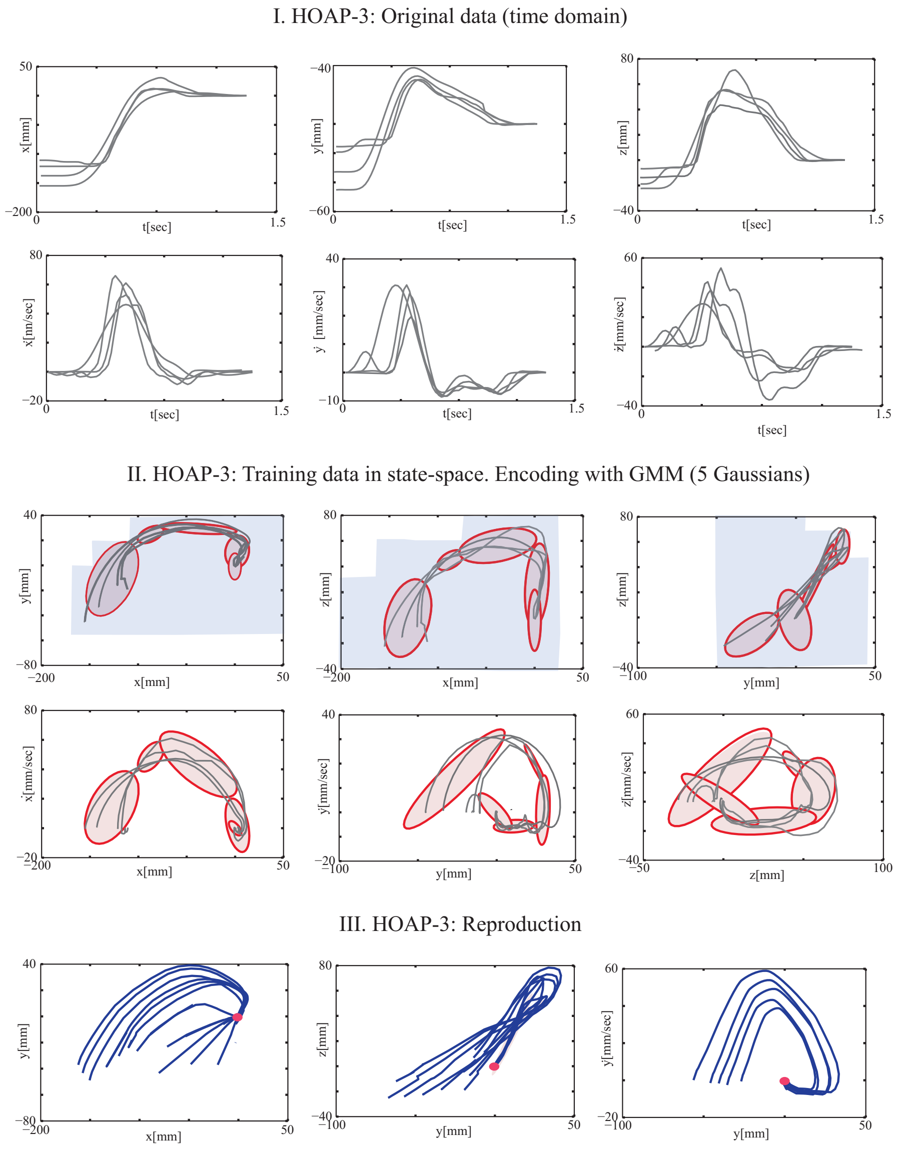


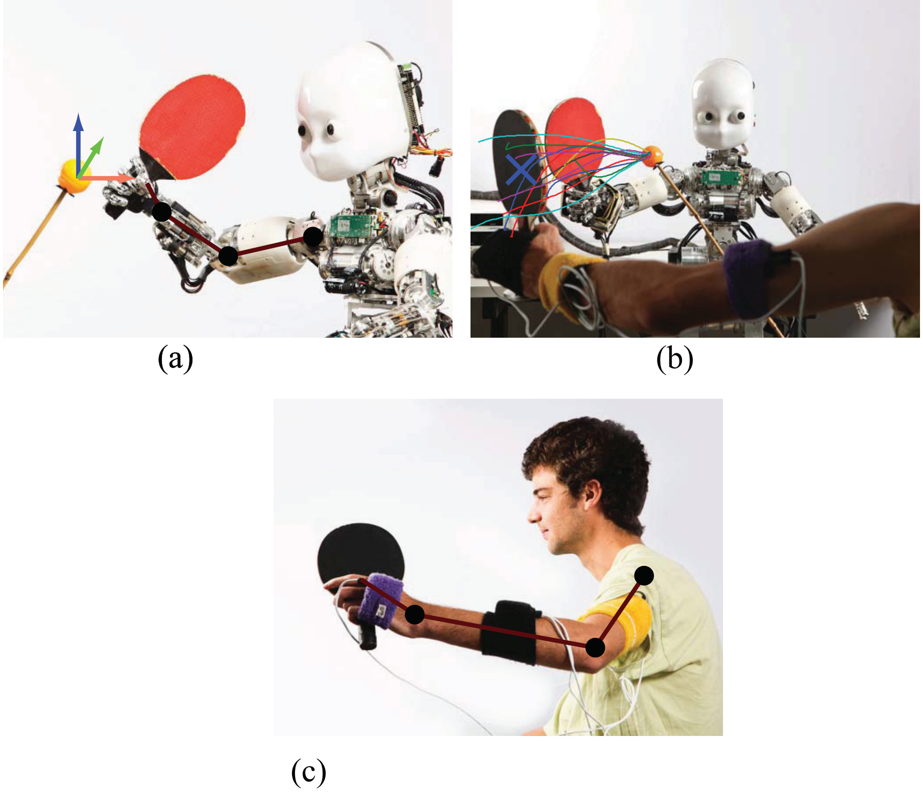
Figure 3.33: KATANA experiment 2: *I. Real-time adaptation to perturbations*. The target is shifted several times from the position 1 to the position 4. First row: trajectory of the robot's end-effector; second row: velocity profile. *II. Generalization to the unseen context*. (a) The approximate workspace of KATANA is highlighted by the blue box (light grey in a black-and-white version), the reproduction is systematically tested starting the robot from positions on the starting plane (yellow/darker grey). (b) The robot is required to reproduce the motion from points monotonically covering the yellow/light-grey sub-part. For comparison, the part of space where the demonstrations are provided is in pink/dark-grey. Note, the demonstrations are sparse, but the system manages to generalize to other parts of the workspace.


I. HOAP-3: Original data (time domain)

II. HOAP-3: Training data in state-space. Encoding with GMM (5 Gaussians)

III. HOAP-3: Reproduction

Figure 3.34: HOAP-3 experiment: Results of encoding and reproduction of the experiment where HOAP-3 had to grasp a box.



(a) (b) (c)

Figure 3.35: (a) The humanoid robot iCub used in the experiments. We encode tasks in the frame of reference located at and moving with the target; this frame of reference is expressed in the fixed global frame of reference (we choose the one attached to static parts of a robot). (b) A human teacher demonstrates a ping-pong motion to the robot. (c) Three XSens motion capture sensors are attached to the hand, forearm, and upper arm of the demonstrator and allow the reconstruction of the motion of each joint.

task models are learned from a single demonstration, if the data contain noise the task model can fit noise as a part of the true signal. In contrast, our method combines several demonstrations to build a more accurate estimate of an actual underlying dynamics (Coates et al., 2008). This does not necessarily lead to exact trajectory fitting if the data are noisy.

Therefore, next, we compare the two methods in terms of their qualitative performance in the case of (1) changing of an initial position (generalization to the unseen context); (2) spatial perturbations (changes in a target position *after the onset* of a motion); (3) temporal perturbations (changes in the target position that considerably *change the time* of reaching the target).

Recently, the discussion on generalization abilities of DMP has been relaunched by Bitzer & Vijayakumar (2009). The proposed approach is theoretically sound and demonstrates appealing results. We do not include this method into our comparison, but outline its main idea for the completeness of our discussion. Bitzer & Vijayakumar (2009) suggest that modulation problems of DMP are partially related to the choice of a space where the DMP is applied. Specifically, their hypothesis suggests that there exists a latent space such that if trajectories, generated by DMP in this space, are projected back to the original space, they closely follow actual demonstrations. The considered experiments include, for instance, full-body punching motions recorded in the joint space. A training set consists of several demonstrations, only one of which is used for training a DMP. The remaining part of the training set is applied to learn a latent space. Learning of the latent space is implemented as an extension of Gaussian Process Latent Variable Models (GPLVM) (Lawrence, 2005). The method is illustrated with two experiments and it will be highly interesting to further investigate whether such a latent space can be estimated for an arbitrary task.

We follow the most recent formulation of DMP by Pastor et al. (2009); see Appendix II. The proportionate and derivative coefficients K_p and K_v are chosen so as to guarantee the critical damping. Note that we have implemented DMP as defined in Pastor et al. (2009); that is, without a heuristic to re-index the canonical variable s so as to handle perturbations[14].

For illustrative purposes, we first compare how well the two methods learn a theoretical two-dimensional dynamics. We then move to the performance comparison in the ping-pong task.

1. Comparison of generalization abilities

   In the considered experiments, DMP exhibits limited capacities to generalize: when trajectories are to be reproduced when starting from unseen parts of the workspace the algorithm generates only *scaled versions* of a demonstrated trajectory. In the case depicted in 3.37-(b) we see an example of the task reproduction when the motion onset is located in the middle of the demonstrated trajectory. Instead of following the remainder of the motion, DMP forces the robot to re-

---

[14] The authors suggest that this formulation is sufficient for adaptation of robot's trajectories to a moving target g

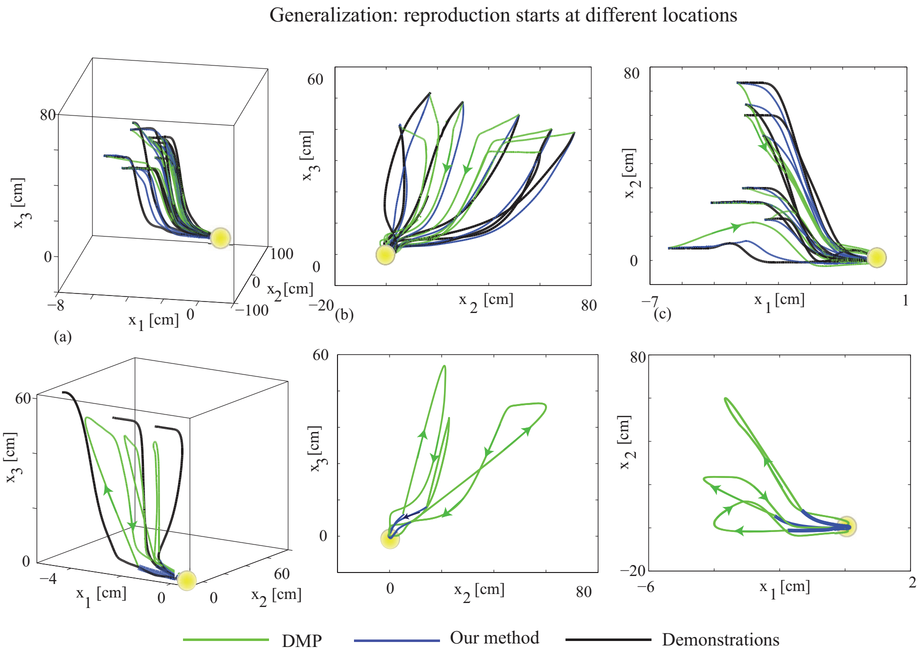


Figure 3.36: Task generalization: our method vs. DMP (Hoffmann, Pastor, et al., 2009) in the ping-pong experiments. Here, due to the noise in the training data, our system tries to extract a generic pattern and, therefore, the reproduced trajectories do not follow the demonstrations exactly. However, in comparison with DMP generated trajectories, the trajectories produced by our method exhibit more similarity with demonstrations (in terms of the trajectory shape). The difference between the two method is particularly obvious when the robot starts its motion from locations unobserved during demonstration; DMP tends to generate unexpected swinging motions.

produce the whole trajectory, scaled so as to fit into the distance to the target. In contrast, our system guides the robot along the remaining segment of the motion. The scaling is even more evident in the three-dimensional case of the ping-pong task; see Fig. 3.38.

The sole scaling of trajectories, instead of regenerating a new motion that follows a coordination pattern, can fail the reproduction. One of the reasons why human movements are often curved consists in their intention to satisfy coordination constraints caused, for instance, by the geometrical constraints of manipulated objects. The demonstrations provided by a human teacher are implicitly encode these constraints in the form of a specific curvature (Petreska & Billard, 2009). A motion representation should be able to encode these coordination constraints and allow their accurate reproduction. Note, the demonstrated trajectories in Fig. 3.37-3.39 are strongly curved around the target, this can be, for instance, due to the presence of an obstacle that should be avoided or such a path might be dictated by the particular shape of a manipulated object. Therefore, it is crucial for the robot to coordinate its motion so as to satisfy the constraint. The trajectories generated by our system follow the demonstrated approach direction, while DMP runs considerable risk of violating the constraints and bumping into obstacles.

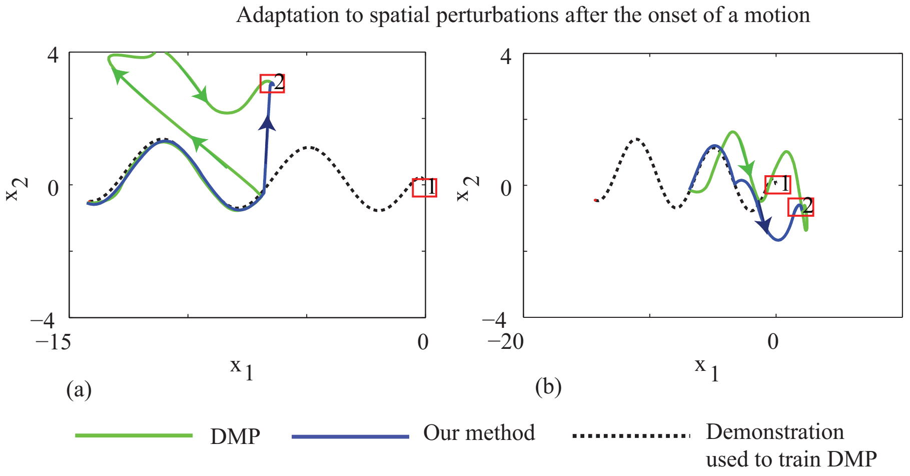


Figure 3.37: Robustness to spatial perturbations: our method vs. DMP (Hoffmann, Pastor, et al., 2009), learning a theoretical noise-free dynamics. Due to scaling that DMP performs for adapting a learned acceleration profile to the conditions after perturbation, a generated motion may have an unexpectedly excessive curvature (a) or can overshoot the target (b).

2. **Robustness to Spatial Perturbations**

   We compare the robustness to spatial perturbations; that is, to displacements of a manipulated object or of a robot's arm that occur after the motion onset. Here, we consider perturbations that do not necessarily cause considerable variation of the motion duration. As both, our method and DMP, ensure that a robot reaches

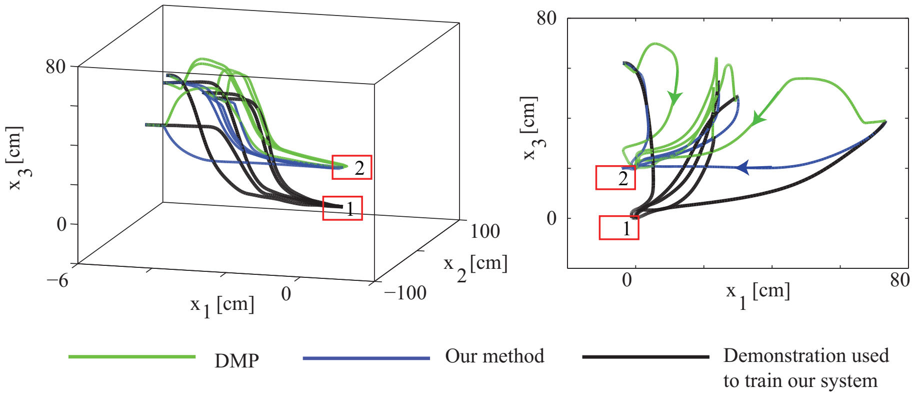


Figure 3.38: Robustness to spatio-temporal perturbations: our method vs. DMP (Hoffmann, Pastor, et al., 2009) in the ping-pong experiment. The ball is moved up (from position (1) to position (2)) after the onset of the motion, DMP trajectories produce strong swings and tend to overshoot the target.

a target (due to asymptotic stability of the learned task models), we concentrate solely on qualitative aspects of the robustness. Precisely, we look at whether both systems can reproduce key characteristics of the motion, such as the curvature. We compare performance both in the noise-free case and in the real -world ping-pong task; see Fig. 3.37 and 3.38 respectively. DMP does not adapt the shape of the trajectory when moved to an arbitrary location in the workspace. The lack of adaptation can bring about the excessive curvature of the trajectories and overshoot at the target.

3. **Robustness to Temporal Perturbations**

   We consider spatial perturbations happened after the motion onset that result in significant variation of the motion duration. Results are shown in Figures 3.38, 3.39. In Fig. 3.39-(a) the target is moved from position (1) to position (2), *farther* from the robot's end-effector, DMP takes the shortest path to the initial position of the target and stretches it to reach the shifted target position. This results in an almost straight line trajectory which may violate external constraints implicitly encoded in the demonstrations. The deformation of a motion pattern also occurs in the case of the ping-pong experiment when the ball is moved away from the robot; see Fig. 3.38. DMP fail to reproduce the demonstrated slope of trajectories; see particularly Fig. 3.38-(b).

   In Fig. 3.39-(b) the target is moved from position (1) to position (2), *closer* to the robot's end-effector. DMP tries to fit the learned trajectory into the new spatial interval and produces jerky motion. Our algorithm, in contrast, drives the trajectory directly to the target, so that the robot reaches the target faster than if controller with DMP.

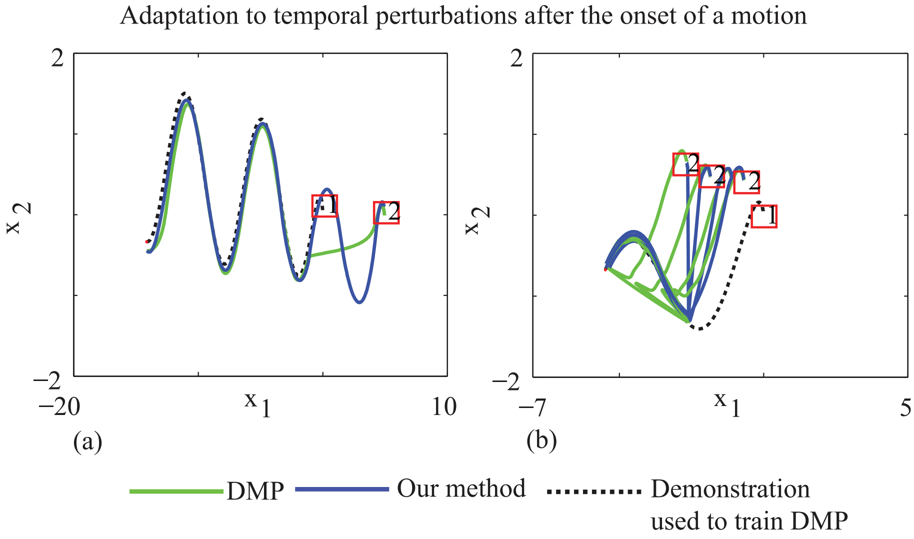


Figure 3.39: Robustness to temporal perturbations: our method vs. DMP (Hoffmann, Pastor, et al., 2009). The target has been shifted so that the duration of motion has been increased (a) or decreased (b). (a) The target is moved from position (1) to position (2), farther from the robot's end-effector, DMP takes the shortest path to the initial position of the target and stretches it to reach the shifted target position, this results in an almost straight line trajectory which potentially may violate external constraints implicitly encoded in the demonstrations. (b) The target shifted so as to decrease the duration of motion, in this case DMP scale the trajectory and produce the jerky motion right after the perturbation; in this case DMP require more time to reach the target than our system.

4. **Conclusion**

   DMP provides an efficient tool for learning a stable estimate of a motion dynamics from a single demonstration. It allows for adaptive scaling of the demonstrated acceleration profile and ensures global convergence to the target. DMP, hence, is particularly useful if the robot is expected to operate in a vicinity of the demonstration, and if the robot is required to replicate the demonstrated motion exactly; see Fig. 3.40.

   However, this solution comes with its drawbacks: depending on the starting position and perturbations along a motion, a resultant trajectory might not satisfy coordination constraints encoded in the demonstrations. As DMP does not address learning state-space dependencies, the temporal robustness cannot be guaranteed as demonstrated in the comparison. Note that this implicit time-dependency remains even in the recent reformulation of DMP suggested by Hoffmann, Pastor, et al. (2009); D.-H. Park et al. (2008); Pastor et al. (2009). The time dependency is implemented using a canonical variable s, that acts as a clock for the system[15]. To adapt to changes in the motion's duration associated with different initial positions or significant spatial perturbations, one must use a heuristic to re-set the canonical variable. Failing this, the canonical variable forces the modulation term $f(s)$ to reproduce the same acceleration profile irrespective of where in the workspace the robot's arm locates. Furthermore, once the canonical variable s decays to zero, it ultimately cancels the modulation terms $f(s)$. When it happens, the system is left to be driven solely by a linear dynamical system.

   We should note that these undesirable responses of the system are avoidable if one can find a means to rescale the phase variable. It is, however, not easy to engineer such a heuristic, especially, if the motion duration is unknown, e.g., after perturbations. In our method, the time dependency is removed entirely, hence eliminating the requirement of searching for the heuristic. To conclude, DMP represents a major step towards introducing dynamical systems as a means for flexible and robust robot motion learning. Additionally, the reformulation proposed by Hoffmann, Pastor, et al. (2009), Pastor et al. (2009), and D.-H. Park et al. (2008) offers a means to perform obstacle avoidance, a problem to which we do not offer a solution here.

   As discussed above, the method, which we present in this Section, ensures local asymptotic stability at the attractor within the region of attraction(as per *Definitions* 6 and 7 of Table 3.2). However, a secondary mechanism should be employed to bring the robot back in to the region of attraction, if a perturbation sends it outside of this region. For instance, one can assume that the motion outside of the region of attraction is driven, is driven by the linear, globally stable, dynamics defined by the last Gaussian. Note that, in our experiments, the region of attraction is sufficiently large; see Fig. 3.32-3.34. From a practical point

---

[15] Specifically, this is the variable θ in Equation 11 of (D.-H. Park et al., 2008), which is equivalent to the variable s in the original DMP formulation (Ijspeert, Nakanishi, & Schaal, 2001) and its another reformulation (Hoffmann, Pastor, et al., 2009), see also Appendix II of this manuscript.

of view, ensuring only local stability in many cases is not so restrictive. Nonlinear motions are often driven by local coordination constraints, e.g., caused by the shape of a manipulated object. Therefore, it might make little sense if a robot reproduces these constraints in an arbitrary part of its workspace. Statistical learning is local by nature; hence, one cannot ensure that inference far from the demonstrations will be relevant in a statistical sense (as the likelihood of the input data is negligible). Though DMP ensures the global stability, the method cannot guarantee the generation of meaningful or even feasible trajectories far from demonstrations; see Fig. 3.44.

We note that DMP is directly extendable to learning periodic movements (Ijspeert et al., 2002b; Schaal et al., 2007). Our approach also can be applied to learning rhythmic motions, however, the problem of ensuring stability of a learned dynamics needs to be addressed. As we concentrate on coordinated goal-directed movements, we do not compare performance of our method and DMP when applied to learning periodic motions.

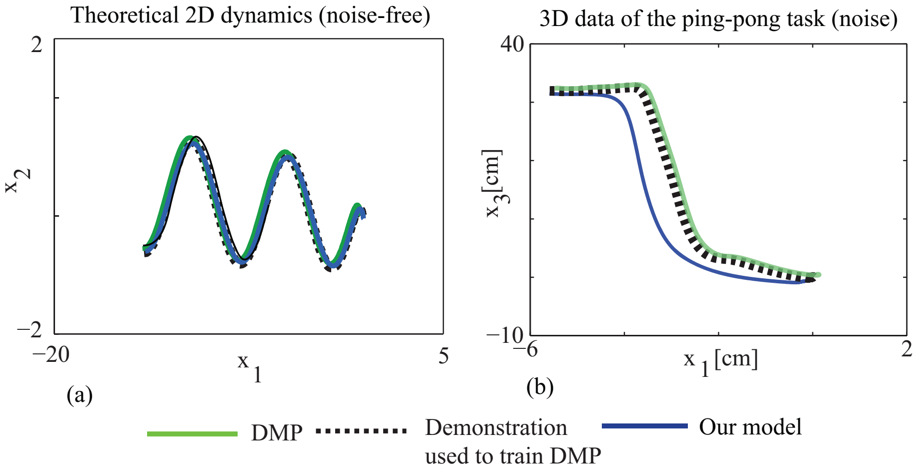


Figure 3.40: (a) Illustration of the high accuracy of DMP at reproducing trajectories that start in a small neighborhood of a demonstrated trajectory. When starting the motion at the same location as that demonstrated, reproduction fits the original signal very accurately. (b) When reproducing the noisy training data from the ping-pong task. DMP accurately fits each demonstration separately. This leads to over-fitting as each trajectory contain noise inherent to the physical world.

3.3.5 CONCLUSION

In this Section, we proposed a method for learning a non-linear multi-dimensional dynamics of motion through statistically encoding demonstrated data with Gaussian Mixtures. Further, we addressed the problem of ensuring stability of a resultant control law: first, we formulated conditions that parameters of GMMs should satisfy to guarantee local asymptotical stability of an attractor, then we proposed a numerical procedure to verify boundaries of the region of applicability where the control law can be securely applied.

To test the method, we conducted two types of experiments: 1) learning theoretical dynamics with known mathematical forms to estimate the accuracy of approximation and 2) learning dynamics of manipulation tasks recorded with two different robotic platforms to assess the applicability of the approach to the noisy data. In all experiments the system demonstrated good results in terms of the high accuracy during reproduction, ability to generalize motions to unseen contexts, and ability to adapt on-the-fly to spatio-temporal perturbations. We also showed how the system could encode more than a single attractor and could successfully reproduce each of the dynamics around a corresponding attractor.

3.4 LEARNING ONLINE MOTION GENERATION FOR INTRA-LIMB COORDINATION IN MANIPULATION TASKS

The method presented in this section has been published in:

Gribovskaya E. and Billard, A. G.. Learning Nonlinear Multi-Variate Motion Dynamics for Real- Time Position and Orientation Control of Robotic Manipulators. *Proceedings of IEEE International Conference on Humanoid Robots*. 2008.

3.4.1 INTRODUCTION

In this section, we extend the dynamical system approach proposed in Section 3.3 to consider intra-limb coordination between the position and orientation of a robot's hand.

The motion of a human hand during a manipulation task consists of two phases: a *transport* phase followed by a *grasping* phase (Jeannerod, 1981; Smeets & Brenner, 1999). The interconnection between these phases has been actively debated. Early works (Jeannerod, 1981) suggest that the two phases are controlled separately, while more recent works (Haggard & A., 1997; Smeets & Brenner, 1999) provide evidence that they are intertwined in the sense that the grip's aperture can be dependent on the position of the hand. It is however agreed that the two phases are tightly interconnected at least through an *approach vector* (a vector that defines the motion direction prior to grasping) (Brenner & Smeets, 1995). That is, already during the transport phase, the hand's orientation is being adjusted so as to get aligned with the approach vector.

We further adopt this assumption and suggest an algorithm to endow a robot with the similar coordination skill. For this, we extend our dynamical system framework so as to account for learning of the complete transport phase. That is, the robot encodes

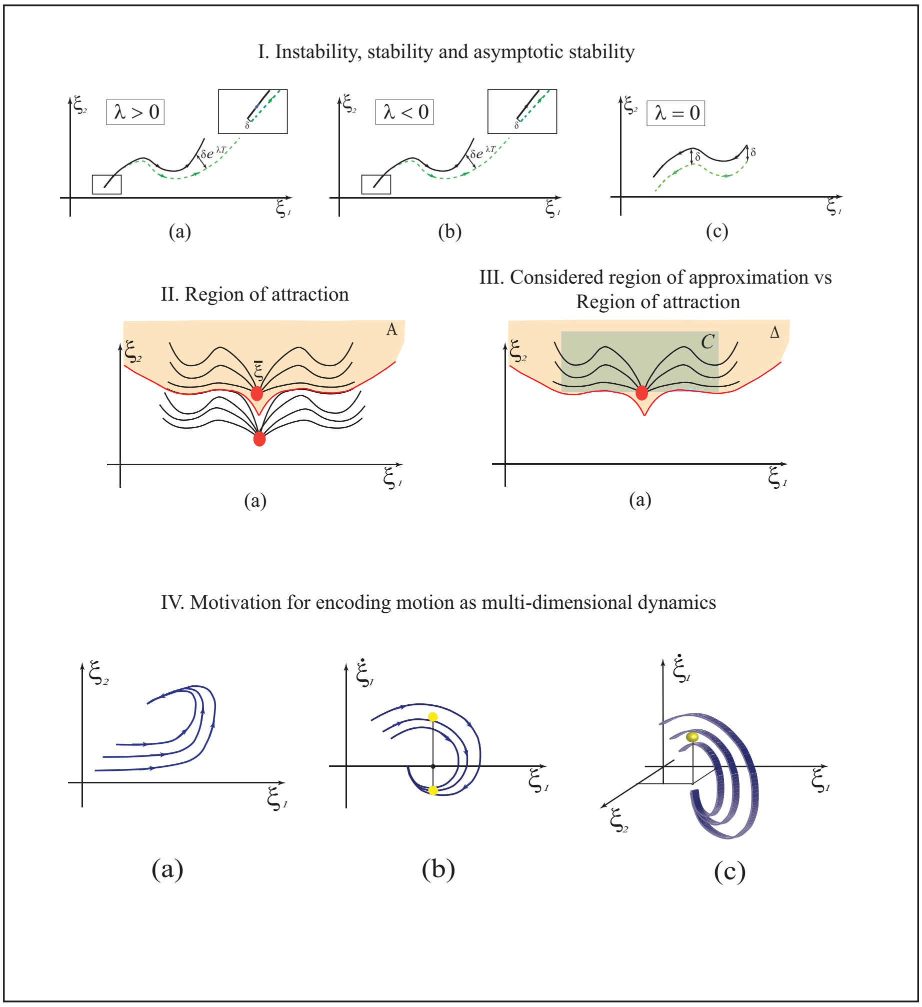

I. Instability, stability and asymptotic stability
ξ₂
λ > 0
δe^{λT}
δ
ξ₁
(a)
λ < 0
(b)
λ = 0
(c)
II. Region of attraction
A
ξ
(a)
III. Considered region of approximation vs Region of attraction
C
Δ
(a)
IV. Motivation for encoding motion as multi-dimensional dynamics
(a)
(b)
(c)

Figure 3.41: Geometrical illustration of stability and multi-dimensional correlation in the state-space. I. ***Stability problem***: stability of a dynamical system is defined by a maximum value of its *Lyapunov exponent* λ (in the linear case, it coincides with eigenvalues of a control matrix). (a) In systems with *negative* Lyapunov exponents volume between trajectories contracts; (b) In systems with *positive* Lyapunov exponents two arbitrary near trajectories diverge from each other exponentially fast. In the linear case, one may easily find Lyapunov exponents and estimate the global behavior of the overall system. In the non-linear case, the system may have different Lyapunov exponents in different parts of the state-space, moreover, non-linearities make analytical investigation of properties particularly tedious. IV. ***Multi-dimensional dynamics*** Analyzing dynamics of vector-valued timeseries requires their encoding in multi-dimensional state-spaces. Generally, one cannot unambiguously decouple dynamics of each dimension. Consider a simple 2D motion in Fig. II-(a), the phase-space of this motion in $\{\dot{x}_1, x_1\}$ is in Fig. II-(b): for each value x_1 there exist two different values of velocity, therefore, it is not possible to unambiguously encode dynamics of motion as two decoupled system $\dot{x}_1 = f_1(x_1)$, $\dot{x}_2 = f_2(x_2)$. However, if one look at the dependency $\dot{x}_1 = f(x_1, x_2)$ depicted at Fig. II-(c) this ambiguity can be easily eliminated. This problem is know in the literature on Dynamical Systems as a problem of searching for a *minimum embedding dimension*. In this particular example, the minimum embedding dimension is 4 (x_1, $\dot{x}_1$, x_2, $\dot{x}_2$). Alternatively, one may argue that in this case we may avoid an ambiguity and separate dimensions encoding $\ddot{x}_1 = f_1(x_1, \dot{x}_1)$, though it is possible in this particular case, it will lead to the necessity to analyze 5 state variables (x_1, $\dot{x}_1$, $\ddot{x}_1$, x_2, $\dot{x}_2$). Furthermore, to preserve a spatial correlation pattern between x_1 and x_2 the decoupled systems should be synchronized by an external mechanism.

and reproduces the position and orientation components of the motion, replicating a coordination pattern. The conducted experiments demonstrate that the proposed algorithm allows the robot to recover from spatio-temporal perturbations affecting both the position and orientation of a manipulated object.

From the review of the literature on motion planning for manipulation tasks (Section 2.2), one can conclude that there is a variety of methods for generating task trajectories. Indeed, traditional planners offer a powerful means for motion generation if all information about an environment is known and modeled prior to the onset of a motion. However, the planners may fail to accomplish a task if the environment changes rapidly and/or unpredictably (as when interacting with humans). Under perturbations, a motion has to be replanned, and this process may be too slow to be computed in real time if perturbations are frequent. Furthermore, some approaches to path generation also assume that a wrist axis is aligned with the direction of a motion, while the alignment with a desired approach vector is performed once the arm is already in the vicinity of an object. As an alternative to the use of planners for adjusting a hand configuration before grasping, methods for automatic grasping (Buss & Hashimoto, 1994; Ekvall & Kragic, 2007; Tegin et al., 2009) address a problem of control of wrist orientation and fingers closure. Once a hand is brought into a proximity of an object to be manipulated, such methods generate a motion that aligns the wrist's axis with an approach vector and arrange fingers into a grasping posture. The two-step process increases a total motion; it also makes the adaptation to perturbations cumbersome, requiring finely tuned heuristics.

The coupled generation of trajectories for position and orientation through dynamical systems, as we propose in this section, helps to overcome some of these problems. The robot's motions look smoother and might be more predictable for humans working with it.

3.4.1.1 LEARNING POSITION AND ORIENTATION CONTROL

To construct $\hat{f}$ from the set of demonstrated trajectories, we follow a learning approach described in Section 3.2 and define $\hat{f}$ using *Gaussian Mixture Models* (GMM). The presented method (Section 3.2) makes no assumptions regarding a type of variables to be used for training, thus we are unconstrained in our choice of variables for motion learning. Here, we choose that: 1) a translational motion of a hand is described by a vector of Cartesian coordinates $\boldsymbol{x} \in \mathbb{R}^3$; 2) an orientation of the hand is described by a pair of variables $\{\boldsymbol{s}, \phi\}$ (an axis and an angle of rotation (Altmann, 1986)). According to the representation $\{\boldsymbol{s}, \phi\}$, an orientation of a moving referential $x'y'z'$ with respect to a fixed referential xyz (see Fig. 3.29) is described by a rotational axis $\boldsymbol{s} \in \mathbb{R}^3$ and an angle $\phi \in [0; 2\pi]$. This representation is similar to the quaternionic representation and can be easily converted into the latter. But for the use with our algorithm for learning motion dynamics, the axis/angle representation is more convenient, as it does not require renormalization at each time step, in contrast, for instance, to the quaternionic representation, and has a more compact form than rotational matrices.

Table 3.8: On-line Task Reproduction: Control over Position and Orientation

```
1  Learn the estimates $\hat{f}_{\boldsymbol{x}}, \hat{f}_{\boldsymbol{o}}$ of the dynamics underlying the position
   and orientation of the end-effector's motion:

2  Detect a target position in the global referential $\{xyz\}$,
   see Fig. 3.29: $\{\boldsymbol{x}^*, \boldsymbol{s}^*, \phi^*\}$

3  Recompute the current position of the end-effector in
   the target referential $\{x^*y^*z^*\}$: $\{\boldsymbol{x}_0, \boldsymbol{s}_0, \phi_0\}$

4  LOOP from $t = 0$ until the target position is reached

6      Infer the velocity at the next time step through GMR (Eq. 3.29):
       $\dot{\hat{\boldsymbol{x}}}_t = \sum_{k=1}^{K} h_{\boldsymbol{x}}^k (\boldsymbol{\mu}_{\dot{\boldsymbol{x}}}^k + \boldsymbol{\Sigma}_{\dot{\boldsymbol{x}}\boldsymbol{x}}^k (\boldsymbol{\Sigma}_{\boldsymbol{x}}^k)^{-1} (\boldsymbol{x}_{t-1} - \boldsymbol{\mu}_{\boldsymbol{x}}^k))$
       $[\dot{\hat{\boldsymbol{s}}}_t; \dot{\hat{\phi}}_t] = \sum_{k=1}^{K} h_{\boldsymbol{s}_t,\phi}^k (\boldsymbol{\mu}_{\dot{\boldsymbol{s}},\phi}^k +$
       $+ \boldsymbol{\Sigma}_{\dot{\boldsymbol{s}},\dot{\phi},\boldsymbol{s},\phi}^k (\Sigma_{\boldsymbol{s},\phi}^k)^{-1} ([\boldsymbol{s}_{t-1}; \phi_{t-1}] - \boldsymbol{\mu}_{\boldsymbol{s}_{t-1},\phi_{t-1}}^k))$

8      Solve the Inverse Kinematics problem (Eq. 3.41) to find: $\dot{\boldsymbol{q}}_t$

9      Send command $\boldsymbol{q}_t$ to a robot and get motors feedback

10     Compute the actual position and orientation of the end-effector $\boldsymbol{x}_t, \boldsymbol{s}_t, \phi_t$

10 END
```

A robot learns the following functions from demonstrations:

$$\dot{\boldsymbol{x}} = \hat{f}_{\boldsymbol{x}}(\boldsymbol{x}), \ [\dot{\boldsymbol{s}}; \dot{\phi}] = \hat{f}_o(\boldsymbol{s}, \phi); \tag{3.39}$$

where $C_{\boldsymbol{x}} \subset \mathbb{R}^3, \ C_o \subset \mathbb{R}^3 \times [0; 2\pi]$.

3.4.1.2 OPTIMIZED INVERSE KINEMATICS

Traditional methods for inverse kinematics that focus on trajectory following might lead to unfavorable joint postures and poor performance near singularities. A number of recent works propose to reformulate the inverse kinematics as an optimization problem (Hersch et al., 2008; Peters et al., 2005). We follow the same approach and we aim to 1) follow a generated trajectory as closely as possible (given by the learned dynamical systems); and 2) find a joint space solution closest to the center of the joint space $\boldsymbol{q}_0$. To avoid the joint limit problem, we additionally impose hard boundary constraints on our optimization problem.

In the previous work of ours (Billard et al., 2006), we define a metric of imitation $\mathbb{H}(\dot{\boldsymbol{q}}_t) : \mathbb{R}^{N_j} \to \mathbb{R}$ (N_j is a number of degrees of freedom (DOFs) in a manipulator) that now control the trade-off between trajectory following and reproducing a desired orientation:

$$\begin{aligned}\mathbb{H}(\dot{\boldsymbol{q}}_t) = {} & (\boldsymbol{J_x}\dot{\boldsymbol{q}}_t - \dot{\hat{\boldsymbol{x}}}_t)^T(\hat{\boldsymbol{\Sigma}}_{\boldsymbol{x}})^{-1}(\boldsymbol{J_x}\dot{\boldsymbol{q}}_t - \dot{\hat{\boldsymbol{x}}}_t) \\ & + (\boldsymbol{J_\omega}\dot{\boldsymbol{q}}_t - \hat{\boldsymbol{\omega}}_t)^T(\hat{\boldsymbol{\Sigma}}_{\boldsymbol{\omega}})^{-1}(\boldsymbol{J_\omega}\dot{\boldsymbol{q}}_t - \hat{\boldsymbol{\omega}}_t) \\ & + (\dot{\boldsymbol{q}}_t - (\boldsymbol{q}_0 - \boldsymbol{q}_{t-1}))^T\boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1}(\dot{\boldsymbol{q}}_t - (\boldsymbol{q}_0 - \boldsymbol{q}_{t-1}));\end{aligned} \tag{3.40}$$

where $\dot{\hat{\boldsymbol{x}}}_t$ and $\hat{\boldsymbol{\omega}}_t$ are the translational and rotational velocities generated by the learned dynamical system $\hat{f}_{\boldsymbol{x}}(\boldsymbol{x}), \hat{f}_{\boldsymbol{s}}(\boldsymbol{s}), \hat{f}_\phi(\phi)$; $\hat{\boldsymbol{\Sigma}}_{\dot{\boldsymbol{x}}}^{-1}, \hat{\boldsymbol{\Sigma}}_{\boldsymbol{\omega}}^{-1} = \hat{\boldsymbol{\Sigma}}_{\dot{\phi}}^{-1}\hat{\boldsymbol{\Sigma}}_{\dot{\boldsymbol{s}}}^{-1}$ are the estimated variance at a point $\{\dot{\hat{\boldsymbol{x}}}_t, \hat{\boldsymbol{\omega}}_t\}$; $\boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1}$ is a weight matrix that is built so as to have maximum on the joint boundaries and to rapidly decrease as the robot approaches to the center of its workspace; $\boldsymbol{J_x}, \boldsymbol{J_\omega}$ are respectively position and orientation Jacobian matrices of a robot's arm.

Discretizing in time and assuming a local linear approximation of the derivative at each time step: $\dot{x}_t = x_t - x_{t-1}$, we then minimize $\mathbb{H}$ on a set $[\boldsymbol{q}_{\min}; \boldsymbol{q}_{\max}]$, where $\boldsymbol{q}_{\min}, \boldsymbol{q}_{\max}$ are the lower and upper joint limits.

A solution of the minimization problem in Eq.(3.40) has the following form:

$$\dot{\boldsymbol{q}} = (\tilde{\boldsymbol{J}}_{\boldsymbol{x}} + \tilde{\boldsymbol{J}}_\omega + \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1})^{-1}(\tilde{\boldsymbol{J}}_{\boldsymbol{x}}\dot{\hat{\boldsymbol{x}}}_t + +\tilde{\boldsymbol{J}}_\omega\hat{\omega}_t + \boldsymbol{\Sigma}_{\boldsymbol{q}}^{-1}(\boldsymbol{q}_0 - \boldsymbol{q}_{t-1})) \tag{3.41}$$

where $\tilde{\boldsymbol{J}}_{\mathbf{x}} = (\boldsymbol{J}_{\mathbf{x}}^T\hat{\boldsymbol{\Sigma}}_{\mathbf{x}}^{-1}\boldsymbol{J}_{\mathbf{x}}), \tilde{\boldsymbol{J}}_\omega = \boldsymbol{J}_\omega^T\hat{\boldsymbol{\Sigma}}_\omega^{-1}\boldsymbol{J}_\omega$.

If the IK problem is not over-constrained (the rank of a Jacobian is not less than a number of controlled dimensions in the operational space) and if the robot operates within the workspace, far from singularities, the optimization problem will produce a solution that coincides with the desired values generated by the dynamical systems. However, when controlling an under-constrained manipulator or if it is in the proximity of singularities, the generalized IK presented above may be considered as a source of intrinsic perturbations on the dynamical system generating the motion. The dynamical system will be still able to generate a trajectory and will enable a robot to reach a target, given that a final position and orientation are reachable by the robot (the IK solution exists).

Table 3.8 summarizes the steps the robot follows during the task reproduction.

3.4.2 EXPERIMENTAL RESULTS AND DISCUSSION

3.4.2.1 SET-UP

We validate the above method in two experiments; see Fig. 3.30, using a four degree of freedom arm of a humanoid robot HOAP-3 and a six degree of freedom industrial-type KATANA arm from Neuronics.

The KATANA arm is taught to put a rectangular wooden brick into a container slightly bigger than the brick; see Fig. 3.30-top. Before releasing the brick, the robot has to accurately adapt its orientation and position with respect to the container.

The clench of the HOAP-3 is small, therefore, it can grasp only thin objects. In this task the robot has to grasp a box that is thin only along one dimension; see Fig. 3.30-bottom.

The tasks were chosen as (1) the objects are asymmetrical, and, hence, for the objects to be grasped, both tasks require a particular orientation of the robot's hand, (2) robot's trajectories in each task can be described by a dynamical system with a single attractor, and (3) robot's success or failure can be easily estimated.

In both experiments, we demonstrate the tasks to the robots four times by guiding their arms. The position and orientation of the manipulated objects are tracked with a stereovision system that uses Augmented Reality Toolkit (ARToolKit) markers.

3.4.2.2 RESULTS

After training, we test our algorithm by letting the robots to manipulate the objects at different locations in their workspaces (to demonstrate generalization capacities of the learned dynamics) and by producing spatial perturbations (to demonstrate the ability of our method to recover from perturbations in both position and orientation).

The results of the experiments are summarized in Fig. 3.31, 3.42-3.43.

The dynamical dependencies between encoded variables and their derivatives exhibit strong non-linearities; see Fig.3.42; however, our method manages to encode them accurately with relatively few GMM components. Therefore, the memory requirements for storing motion models are much less in comparison with these of memory-based approaches. Furthermore, the correlations between the positional and orientational variables $\mathbf{x}$, $\{\mathbf{s}, \phi\}$ are strong, therefore, learning and reproduction of these correlations leads to more accurate and faithful motions.

To test the generalization abilities and the robustness to perturbations we performed experiments in different conditions: we vary starting positions of the robots and shift the objects to be grasped while the robots are moving. The results are presented in Fig. 3.31, 3.26. From multiple starting locations, the robots accurately reach the desired objects and grasp them with a correct orientation

The KATANA arm is a 6DOF manipulator. Hence, when both the location and orientation of an end-effector are specified, the arm's jacobian has a full rank. However, due to a particular geometry of the KATANA's workspace, the robot tends to bump into joint limits. The use of the inverse kinematics controller that brings the arm closer to the center of the workspace enables the robot to avoid unfavorable configurations. Indeed, if one just simply assigns a joint value to its limit each time the robot tries to leave an admissible interval, this joint most likely will stick to the limit for the rest of a motion, which is undesirable.

In contrast, for the HOAP-3 robot, the inverse kinematics problem is under-defined. Therefore, one cannot ensure that a precise solution will be found and this will produce a desired learned dynamics along a motion. In this case, the proposed optimization algorithm balances between the constraints on position and orientation, while taking

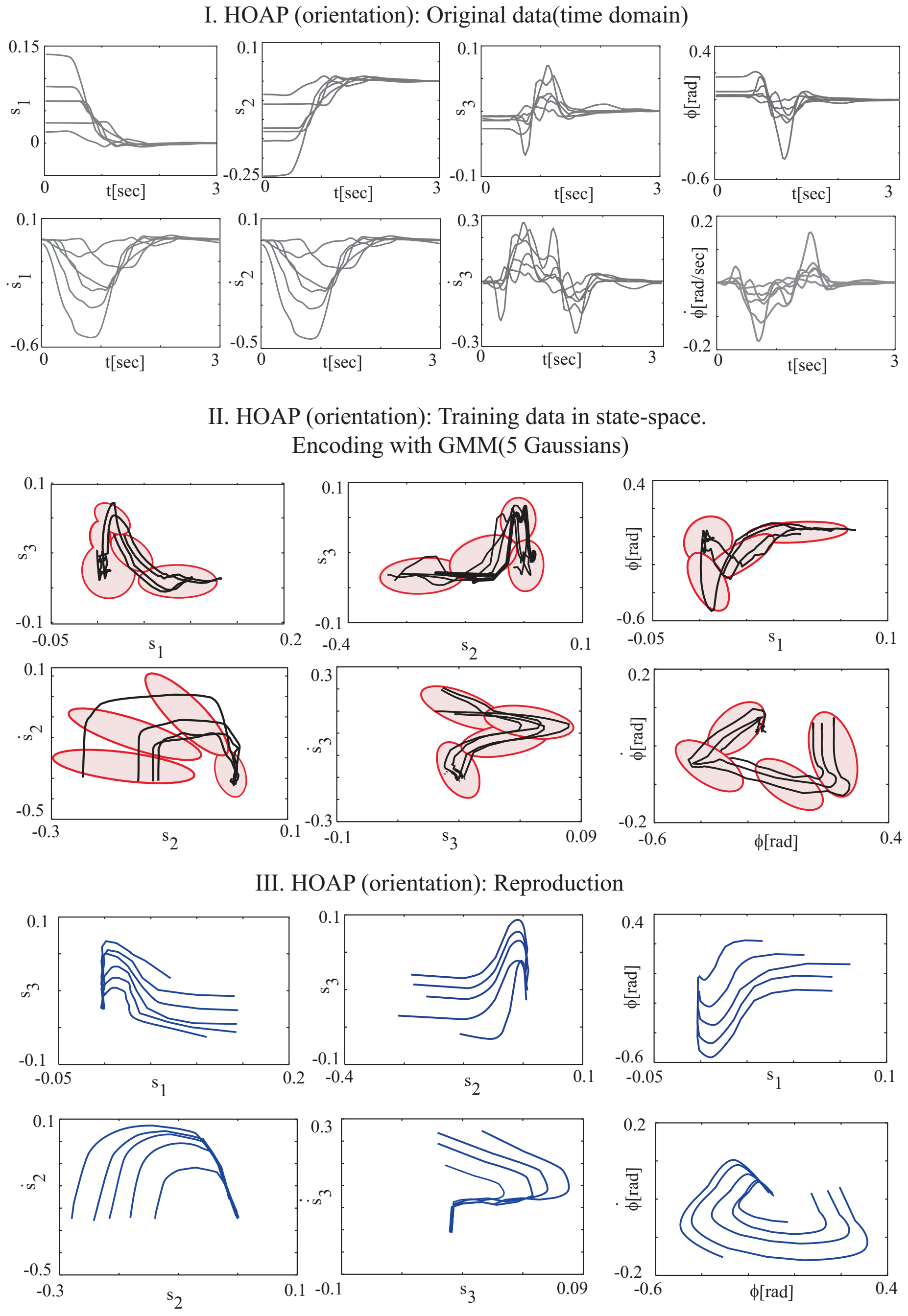
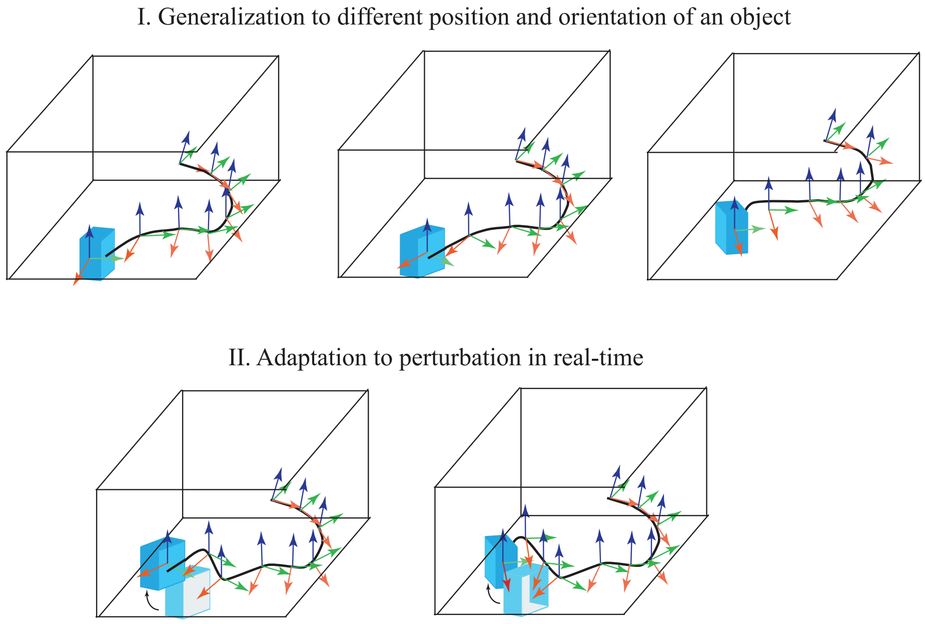


Figure 3.42: Results of encoding the orientation phase of demonstrations in an experiment with HOAP-3. Note, the existence of non-linear correlation between an axis and an angle of rotation.



I. Generalization to different position and orientation of an object

II. Adaptation to perturbation in real-time

(a) Translational perturbation

(b) Translational and orientational perturbation

Figure 3.43: Experiments with the humanoid robot HOAP-3. Referentials display the change in the orientation of the robot's end-effector along the motion. Starting positions of the hand are highlighted by yellow circles. I. Generalization abilities of the method: the robot successfully grasped a box placed in different positions in the workspace. These configurations of the box have not been observed by the robot during demonstration. II. Real-time adaptation to perturbations: while the robot was moving towards the box its position was perturbed (a), both position and orientation were perturbed (b). Control of position and orientation through dynamical systems enables the smooth adaptation to both types of perturbations.

into account the variance across the demonstrations. Such a trade-off helps to find a solution and leads to the successful task accomplishment.

3.4.2.3 DISCUSSION AND CONCLUSION

In this section, we presented a motion generation system that is based on the dynamical system representation. The method allows a robot to successfully accomplish manipulation tasks even in contexts unvisited during demonstration. Learning of the coordination between the position and orientation of a hand offered a "pre-shape" control strategy similarly to a way humans approach objects (i.e. by adjusting a hand orientation along a motion (Christel & Billard, 2001)). As the orientation was getting aligned with the desired approaching vector already during the motion, the tasks were accomplished faster than they would be in the case of subsequent translational and rotational positioning. Encoding orientation and position in two separate dynamical systems (while coupling them through the generalized IK) endowed the robot with an ability to adapt to perturbations that might affect either of these two constraints, separately or simultaneously. If a manipulated object was only shifted from its original location, the robot's hand did not change its orientation and remained aligned with the desired approach vector.

In future work, we consider to investigate alternative types of coupling between position and orientation control (e.g., by learning them as a single dynamical system or by introducing a hierarchy, where the orientation is position-dependent, but not the other way around).

3.5 DISCUSSION

For scientific completeness, we now revisit each of the hypotheses underlying our approach and suggest alternatives solutions.

3.5.1 MULTI-DIMENSIONAL SYSTEMS, FIRST ORDER DYNAMICS

The method proposed here allows for learning of non-linear multivariate dynamics where the correlation between the variables is important. Other works on learning control with dynamical systems consider each degree of freedom separately, and, hence, discard an information pertaining to the correlation across the joints. While storing correlations across the joints is computationally expensive, it is advantageous as correlations contain important motion features.

For instance, in bimanual coordination tasks, left and right arms might follow different motions while doing so in coordination. Embedding correlation between the arms motion into a representation ensures a correct synchronization between the arms. Furthermore, learning of a correlation decreases a number of Gaussians required for an accurate approximation.

Despite these advantages, the complexity of learning grows with the number of degrees of freedom. Building an accurate model of a multi-body motion requires a considerable amount of training data. This problem is fundamental and affects many statistical approaches. McLachlan & Peel (2000) argues that global learning methods (e.g., GMR), utilize all available information for building an output prediction for a given input. In contrast, local non-parametric methods (e.g., LWR), use only data points close to an input stateLocal learning methods are less suitable for tackling multi-dimensional problems, where distances between the training data points are large and where an input state may not have close neighbors. . LWPR offers an appealing solution. Keeping a flexibility of a local method, LWPR deals with the "curse of dimensionality" by learning data not in an original space but in a subspace of lower dimensionality. We should note that though GMR is a global learning method it still faces computational difficulties when applied to highly multi-dimensional data. One needs to either increase an amount of training data or use dimensionality reduction techniques prior to learning.

An alternative approach, which is frequently pursued in motion learning, is to encode a trajectory of each dimension as a function of a time index and hence learn them independently. Such univariate learning has a strong advantage: it poses fewer requirements on an amount of training data. Dynamical Movement Primitives (DMP), against which we compare our method in Section 3.3.4.7, follows this approach. However, independent learning of trajectories along each dimension significantly increases a number of free parameters. Furthermore, information about a correlation pattern across dimensions is being discarded, which leads to a distortion of an original motion pattern during reproduction.

It is likely that nature has taken ways to resolve the problem of multi-body control. One can cite two important observations from human motor control. Firstly, the human motor system tends to decouple control of multiple degree of freedom so as to coordinate only subsets of these (d'Avella et al., 2003; d'Avella & Tresch, 2002). For instance, depending on the type of task, a researcher might decide to separate task control of upper and lower parts of a robot's body. Secondly, (Giszter et al., 1993; J. A. S. Kelso, 1995) observe that even within such a subset, degrees of freedom are not controlled independently, but rather in a synergy, which again decreases the number of actual control parameters. For instance, (Wang. et al., 2008) explore this concept in robotics and suggest to learn whole-body swinging motions by projecting trajectories into a subspace of lower dimensionality. Similarly, (Bitzer & Vijayakumar, 2009) follows a similar path to overcome some limitations of the DMP approach.

While we start with the hypothesis that a motion is governed by a first order dynamics, the method proposed here might be extended so as to learn higher-order dynamical systems (higher-order systems can always be expressed in the canonical form, i.e., as a set of first-order systems). This ability is particularly relevant in applications where the acceleration profile needs to be controlled.

3.5.2 TIME INDEPENDENCY *vs* TIME DEPENDENCY

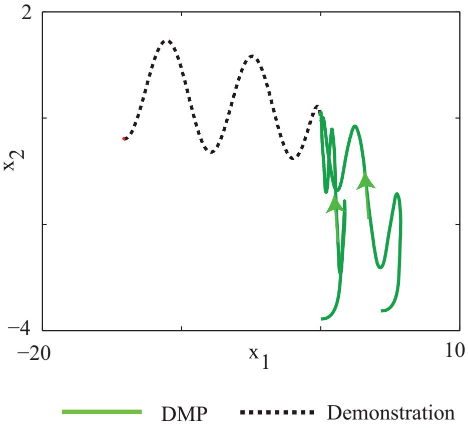


Figure 3.44: An example of a trajectory generated with DMP and starting far from an original demonstration. Note, that although the motion is globally asymptotically stable, the resulting trajectory makes little sense.

In this chapter, we advocate that time-independent encodings in the state-space offer the more robust representation as compared with time-dependent encodings. The presented results confirm that the state-space representation is, indeed, highly robust to spatial and temporal perturbations.

Yet, some motions, such as those that require synchronization with an external dynamics (e.g., catching motion, where a robot has to synchronize with a ball to be caught), should be encoded using a time-dependent representation or, if the external dynamics is known, using an explicit parametrical coupling of two time-independent dynamics, for instance, as it is done by Ijspeert, Nakanishi, & Schaal (2001). Another limitation of the time-independent representation is the impossibility to encode compound motions: once the robot reaches a target (the attractor of a dynamical system), its velocity drops to zero and it stops. To learn compound motions, we suggest to segment the motion into a set of primitives, each of which is governed by a single attractor. Each primitive motion then can be represented by a dynamical system. The transition between these primitives can be controlled by an external algorithm that switches between the primitives. Note, that we already discussed a related problem in Section 3.2: the transition between the primitives is learned and controlled through a Hidden Markov Model.

3.5.3 KINEMATIC CONTROLLER

In the experiments reported here, control of the robot is purely kinematical: we encode the desired kinematic trajectories, but do not take into consideration the dynamical properties (actual torques) of the robot arms. An additional control step is necessary to convert positions into motor commands by means of the inverse dynamics or a PID controller.

We should emphasize that the proposed method can be coupled with operation space control (Hsu et al., 1989; Khatib, 1987; Nakanishi et al., 2005): one of objectives of operational space control is to execute desired trajectories defined in the task space of an end-effector. Our algorithm provides an input for the operation space control by generating the desired kinematic trajectory in real-time.

Learning the inverse dynamics and operational space control (Peters & Schaal, 2008a), while a highly valuable topic in itself, is beyond the scope of the present paper. Further, considering that many of the current robotic platforms are controlled in joint position or velocities, the proposed approach combined with the inverse kinematics is thus valid for a large set of applications.

The proposed approach essentially compensates for the robot's hardware limitations (joint velocity and torque limits) that can lead to deviations from original commands: e.g, if a robot is not able to reach a particular position in a given time span due to angular velocity limits, the system at each time step will recompute the next motor command based on an actual position of the robot. Therefore, while the hardware limits can slow down the motion, but the real-time dynamical controller still allows the robot to follow a desired path.

A problem of the overall stability of a system that consists of a low- and a high-level controllers may arise if the low-level controller does not support a control frequency necessary for the high-level controller to be stable: if the frequency of the low-level controller is too low, the dynamical planner at the high-level will tend to overshoot a target and may fail to converge. However, the state-of-the art robotic platforms operate at a frequency that is sufficiently high to generate stable trajectories given a stable dynamical controller at the high-level.

3.5.4 CHOICE OF STATISTICAL FRAMEWORK

Being a global statistical techniques[16], GMMs is a proven mean for estimating functions from sparse demonstrations, which are typical of PbD applications. However, neither GMMs nor LWPR or GPR ensure stability of a learned *dynamical* function. Here, we propose an algorithm that ensures local asymptotical stability and gradually improves the quality of the approximation. Potentially, the same procedure may be adopted for other statistical frameworks. However, the accuracy of the approximation may vary significantly depending on a particular choice.

One should note that EM is more computationally expensive than LWPR: the number of iteration steps for training GMM is of order $O(K \cdot M \cdot N)$ in comparison to of $O(N)$ for LWPR. Both of these numbers, however, remain small in comparison to GPR. Similarly to LWPR and in contrast to the GPR-based methods, GMR's computational costs for the retrieval procedure are low and increase only linearly with the number of parameters. Importantly, GMM-based models usually contain much less parameters due to the coarse representation.

A part of computational complexity of the proposed method comes from the itera-

---

[16] in contrast to local non-parametric methods such as LWPR, GPR

tive estimation of the region of applicability, which requires $n_1 \cdot .. \cdot n_N \cdot M$ iteration steps ($n_1..n_N$ are the respective sizes of the mesh along the N dimensions, M is the number of data points). In our experiments, estimation of the region of applicability has not exceeded 100-120sec. As learning can be performed offline and because, once learned, the model allows the task reproduction without heavy computations. Hence, the computational complexity of training is counter-balanced by the low computational cost during the retrieval.

3.5.5 REAL-TIME ADAPTATION TO PERTURBATIONS

One of the strengths of the proposed approach is its ability to cope with perturbations in real-time. Under a *perturbation* we mean an unexpected change in the positions of a manipulate object or a robot's arm during motion. We demonstrate that a learned motion dynamics with the position of an object mapped into the attractor allows the robot to successfully track the object even in case of perturbations. This adaptability, combined with the guarantee of ultimately reaching the object, is one of important advantages of the proposed method over analytical planners that we discuss in Section 2.1.

The planners are definitely powerful tools for trajectory generation as they also provide mechanisms for obstacle avoidance. However, for planner require accurate information about a robot's environment, which is not always available. In contrast, our method allows the robot to accomplish tasks if the environmental information is limited or inaccurate.

3.5.6 SINGLE *vs* SEVERAL ATTRACTORS

A further hypothesis pertaining to the work presented here is the idea that a dynamical system to be discovered has a single or several known fixed point attractors. This can be considered as a limitation, as a dynamics may be governed by the existence of more complex orbits than merely fixed points. For example, a dancing motion may have a curve as an attractor. The applicability of the proposed method in this case will mostly depend on the quality of training data; further no stability can be guarantee. Procedures for ensuring stability of complex orbits may substantially widen the class of motion under consideration, covering dancing or sport motions that are usually characterized by the existence of certain curves to which all trajectories converge.

3.5.7 TRAINING DATA

The generalization properties of dynamical controllers directly depend on the quality of training data; the aspect common to all statistical learning methods. To improve the quality of the training, one can: 1) provide an exhaustive set of accurate demonstrations; 2) permit a robot to explore on its own (Reinforcement Learning (Guenter et al., 2007)); 3) provide more variability in between demonstrations (the problem has been discussed in (Calinon & Billard, 2007c)). The first option does not agree with the requirement of user-friendliness of teaching interfaces: a number of demonstrations should be kept bearable for a user. The computation time of the second approach is sometime impediment, if a robot should quickly react to perturbations. Furthermore, all existing RL approaches are time-dependent. Therefore, we concentrate on improving quality of demonstrations by introducing more variability into a small set of demonstrations.

3.5.8 KINESTHETIC TEACHING

For task demonstration we use the kinesthetic teaching approach: a robot is observing a task through its own body (the motors are set in the passive mode). One of the advantages of kinesthetic teaching is that the human can perceive limitations of the robot's architecture. Therefore, he/she can adapt the intuition about an optimal or efficient motions accordingly. Although we actively exploit the kinesthetic teaching paradigm, other approaches such as vision-based learning can be equally applied. That is, the proposed algorithm that can be applied to the motion data obtained through different modalities.

3.5.9 PRACTICAL CONSIDERATIONS

From the practical point of view, mapping the position of a manipulated object into an attractor of a dynamical system improves motion precision at the target and, therefore, enables learning of prehensile tasks. This is a important improvement, as many existing programming by demonstration algorithms are applied for teaching large-scale motions.

We show that our approach is generic in that it does not make assumptions regarding variables to be learned nor nor regarding a robot's arm geometry. Indeed, the method can be applied for controlling robotic arms of different geometries and for learning dynamics of different motion variables.

3.6 CONCLUSION

This chapter outlined our approach to learning motion coordination through a dynamical system representation. From the stance of *motion production*, two approaches were presented. The *first method* was built upon an existing dynamical model of human reaching movements. As we demonstrated in Section 3.2, among its advantages were robustness against perturbations, precision, and global stability. It was discussed that the linear form of the model facilitated coupling of several systems. In the proposed algorithm, the coupling was used to coordinate the two arms of a robot. That is, the coupling ensured reproduction of the learned bimanual constraints. However, we emphasized that, due to a simple linear form of dynamics, the method fell short if manipulation required curved motions. Therefore, we further suggested a more generic approach to motion learning. The *second method*, proposed in Section 3.3, made only a broad assumption about the type of dynamics underlying demonstrated trajectories (i.e., that it was an autonomous dynamics) and, therefore, flexibly accounted for a multitude of coordinated motions. We developed a method that essentially learned a dynamical motion representation of arbitrary nonlinear motions from training data provided by a human. The experiments further demonstrated that the robot could reproduce sophisticated motion patterns and was adaptable to external environment. To emphasize strengths of our work, we experimentally compared our method with another state-of-the-art approach.

From the stance of *robot learning*, we addressed the problem of coordination at the low (trajectory) level and at high (task) level. In literature review in Chapter 2, we provided the evidence that learning coordination at the *low trajectory level* was fundamental for developing robot's manipulation skills. The algorithms proposed in Section 3.3 and 3.4 enabled a robot to acquire task models with nonlinear correlation between parameters. Encoding and reproduction of nonlinear correlation was demonstrated to be the crucial condition for successful task accomplishment. Particularly, in Section 3.3.4.7, we compared our method with Dynamic Movement Primitives (DMPs). As we explained: DMPs were essentially an uni-variate model, that did not consider coordination between the variables explaining the motion. The results of comparison clearly highlighted the importance of coordination, especially, under perturbations in the environment.

Bimanual tasks that we considered in Section 3.2 were compound tasks (i.e., consisted of several subtasks); learning, hence, occurred at *the high level*. The robot learned underlying discrete spatio-temporal constraints that characterized a subpart of a trajectory. We applied Hidden Markov Models to uncover a generic sequence of the constraints and to learn transitions within this sequence. We suggested that combining the two levels of representation, i.e., using the proposed dynamical motion representations (Section 3.3) within the bimanual framework (Section 3.2), would enhance the performance: the robot would be capable of learning nonlinear continuous motion patterns and ensure resolution of discrete task-level constraints. We assumed such an extension as a part of future work.

In the next chapter, we will discuss learning for physical human-robot interaction. Teaching motion coordination to a robot so that it can physically interact with humans is a unique challenge for robot learning. Indeed, physical interaction requires to resolve a number of issues that have not been explored sufficiently neither in analytical robotics nor in robot learning. Between these issues are online trajectory regeneration, strong action-perception coupling (to make the robot responsive to force applied by a human), and efficient generalization of task models (human partners vary the pace as well other parameters of the motion). We will demonstrate how the use of the dynamical motion representation developed in Section 3.3 facilitates the learning of physical human-robot interaction and to achieve good performance.

Chapter 4

LEARNING PHYSICAL HUMAN-ROBOT COORDINATION

The method presented in this section has been previously published in:

Gribovskaya E., Kheddar A., and Billard, A.. Motion Learning and Adaptive Impedance for Robot Control during Physical Interaction with Humans. *Proceedings of IEEE International Conference on Robotics and Automation*. 2011.

4.1 OVERVIEW

ONE key component of *physical coordination* between humans is *haptic* communication – an information exchange through force signals. In this chapter, we demonstrate how to combine learning of dynamical motion representations (presented in Section 3.3) and impedance control to teach a robot physical collaboration.

In some works on control of physical interaction (see Section 2.4.4), external forces are considered as random disturbances and therefore the control objective is to compliantly reject them so that a robot can proceed with trajectory tracking. This objective is suitable for autonomous manipulation tasks, however, if a robot needs to coordinate with a human, the control requirements should be reconsidered. Here, we address a problem that goes beyond that of a robot reacting to random disturbances and focus on a continuous prediction and adaptation to the dynamics of the partner.

To motivate our approach, we start by discussing preliminary experiments [1] where a robot HRP-2 collaborates with a human in a lifting task (Section 4.2). In these experiments, the robot learns how to execute the task by observing it through teleoperation and then uses a learned task model to infer suitable actions while collaborating with the human autonomously. We revisit the results of these experiments and highlight open issues that we further address in our method (Sections 4.3 to 4.8). Specifically, in the preliminary experiments, the robot's performance would be poor if, during reproduction, the human partner is moving at a pace different from a demonstrated one. Moreover, if the human attempts to stop abruptly or bring the object to a different lo-

---

[1] The experiments are preliminary with respect to the work presented in the manuscript.

cation, the robot is unable to infer a suitable action from the learned task model. In this case, the robot behaves unsafely by counteracting the human motion.

While analyzing the results of these experiments, we also observe that it is difficult to assess the algorithm's performance while the robot is interacting with a human. Unaided manipulation tasks often allow for an intuitive performance criterion: a task is deemed to be accomplished successfully if an overall objective is achieved (e.g., if the object is brought to a desired location). Assessing performance in collaborative tasks is more intricate; force distribution between partners should be taken into account. To check whether the robot adapts to the human partner, one way would be to somehow "freeze" the behavior on the human side and verify whether the task still can be accomplished. Controlling a human's behavior in real-world experiments is difficult, therefore, we propose a simulation set-up for testing our algorithm.

In Sections 4.3 to 4.8, we describe our approach to learning physical interaction and validate it *in simulation*. We demonstrate that *learning* a dynamical task model allows the robot to anticipate the partner's intentions and adapt its motion according to perceived forces. In comparison to the preliminary experiments with the HRP-2 robot discussed in Section 4.2, the use of our algorithm helps to improve performance in situations where the partner changes a motion pace. To compensate for unmodelled uncertainties, in addition to learning, we propose an *adaptive control* algorithm that tunes the impedance parameters, so as to ensure accurate reproduction.

4.2 PRELIMINARY EXPERIMENTS WITH A HRP-2 ROBOT

In this section, we briefly summarize experiments that have been conducted on a humanoid robot HRP-2. The work presented below is a result of collaboration with Prof. A. Kheddar, Dr. S. Calinon, and Dr. P. Evrard and appears in the following publications.

- Evrard P., Gribovskaya E., Calinon S., Billard A., and Kheddar, A. Teaching Physical Collaborative Tasks: Object-Lifting Case Study with a Humanoid. In *Proceedings of IEEE International Conference on Humanoid Robots*, 2009.
- Calinon S., Evrard P., Gribovskaya E., Billard A., and Kheddar A. Learning collaborative manipulation tasks by demonstration using a haptic interface. In *Proceedings of the International Conference on Advanced Robotics*, 2009.

E. Gribovskaya's contribution in the above mentioned works consists in participation in the data collection experiments discussed in (Calinon, Evrard, et al., 2009) and (Evrard et al., 2009), in conducting simulation experiments (Calinon, Evrard, et al., 2009), and in preparing task models and developing software for testing the algorithm on the HRP-2 robot (Evrard et al., 2009).

The algorithm described in this section and its experimental validation provide a background important for motivating our method that is discussed further in this chapter. The current section is organized as follows: Section 4.2.1 describes the hardware set-up and the data acquisition process, Section 4.2.2 outlines the proposed learning algorithm, and Section 4.2.3 discusses the reproduction set-up and experimental results.

4.2.1 HARDWARE SET-UP

In (Calinon, Evrard, et al., 2009; Evrard et al., 2009), a collaborative lifting task is demonstrated to a full-sized humanoid robot HRP-2. Fig. 4.1 presents the experimental setup used for demonstration. A human operator (teacher) teleoperates the robot through a six degree of freedom haptic device. The teacher perceives a complete interaction wrench measured by a force sensor mounted on the robot's wrist. The second operator (human partner) lifts a beam together with the teleoperated robot.

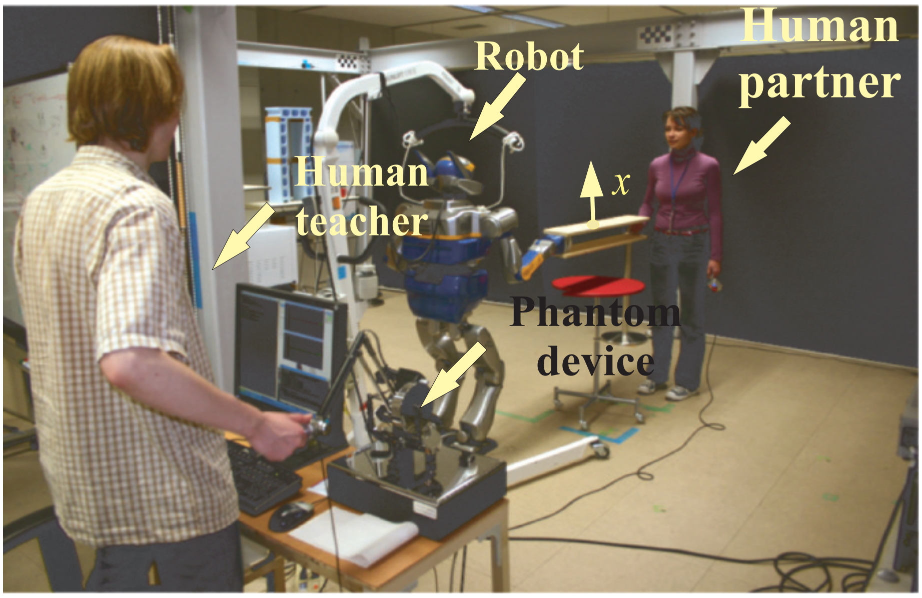


Figure 4.1: We consider a task where a human and a robot lift a rigid beam in collaboration (Evrard et al., 2009). Training is accomplished through teleoperation of the HRP-2 robot through a haptic device.

The robot performs the task in the upright standing position using solely its right arm. The robot's wrist is constrained to move along the vertical direction. The position and velocity of the robot's gripper and the force measured by the sensor are recorded at the frequency of 200Hz.

The robot is provided with two sets of demonstrations. In the first set, the teacher is blind-folded [2] and the partner initiates and terminates the motion. In the second set, the roles are exchanged: the teacher leads the motion's onset and the offset. The data recorded during demonstration are depicted in Fig.4.2. Note that the two sets of demonstrations contain different force-velocity patterns.

---

[2] Being blind-folded helps a human to rely on his/her haptic perception rather than on visual cues.

4.2.2 ALGORITHM

Demonstrated data are collected into a training set $\mathcal{D}$, which consists of M demonstrated trajectories of a length N^k, $k = 1..M$. Each trajectory is a sequence of wrist positions $\boldsymbol{x}_t^k$, velocities $\dot{\boldsymbol{x}}_t^k$, and force measurements $\boldsymbol{f}_t^k$: $\mathcal{D} = \{\boldsymbol{x}_t^k, \dot{\boldsymbol{x}}_t^k, \boldsymbol{f}_t^k\}_{t=1..N^k}^{k=1..M} = \{\boldsymbol{x}, \dot{\boldsymbol{x}}, \boldsymbol{f}\}$. The joint probability of the data $\mathcal{D}$ is encoded with Gaussian Mixture Models:

$$p(\boldsymbol{x}, \dot{\boldsymbol{x}}, \boldsymbol{f}) = \sum_{k=1}^{K} \pi_k \mathcal{N}_k(\boldsymbol{x}, \dot{\boldsymbol{x}}, \boldsymbol{f}), \tag{4.1}$$

where K is the number of Gaussian components in the mixture, $\mathcal{N}_k$ is a Gaussian distributions with the mean value and the covariance matrix $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$, $k = 1..K$.

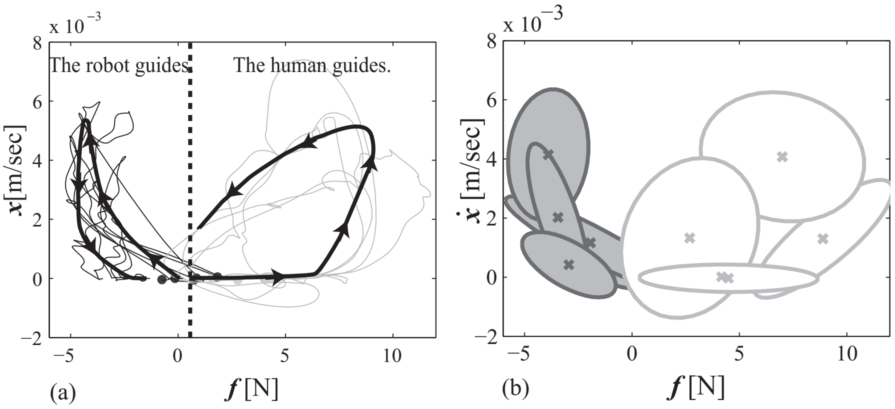


Figure 4.2: [Taken from (Calinon, Evrard, et al., 2009)] The two sets of demonstrations are provided. In the first set, the teacher is blind-folded and the partner initiates and terminates the motion. In the second set, the roles are exchanged: the teacher leads the motion's onset and the offset. Note that the two set produce different force-velocity patterns. (a) Collected data are plotted in fine lines. An average demonstration in each set is plotted as a wide line with arrows. Right: when the human partner initiates the motion, the robot perceives positive interaction force, when the robot starts moving the force gradually decreases to zero by the end of the movement. Left: in contrast, when the robot (guided by the teacher) initiates the motion, the robot perceives negative force, when the human partner starts moving the force gradually increases to zero by the end of the movement. (b) The GMM encoding of the training sets.

Once learned, the task model given by Eq.4.1 is used for generating a kinematic command for the robot's wrist. At each time step, the robot infers both desired position and desired velocity from the position, velocity, and force measured at a previous time step. The desired acceleration is then calculated using a PD-type controller:

$$\begin{aligned} \dot{\boldsymbol{x}}_d &= \mathbb{E}\left[\, p(\dot{\boldsymbol{x}}|\boldsymbol{x}, \boldsymbol{f}) \,\right], \quad \boldsymbol{x}_d = \mathbb{E}\left[\, p(\boldsymbol{x}|\dot{\boldsymbol{x}}, \boldsymbol{f}) \,\right] \\ \ddot{\boldsymbol{x}}_d &= (\dot{\boldsymbol{x}}_d - \dot{\boldsymbol{x}})\kappa^{\nu} + (\boldsymbol{x}_d - \boldsymbol{x})\kappa^{p}, \end{aligned} \tag{4.2}$$

where $\boldsymbol{x}_d, \dot{\boldsymbol{x}}_d$ are robot's position and velocity estimated from the learned model using Gaussian Mixture Regression (see Chapter 3.2); $\boldsymbol{x}, \dot{\boldsymbol{x}}$ are the actual position and velocity. The gains κ^{ν} and κ^{p} are analogous to the gains of a PD controller (see Section 2.1); an approach suggested in (Calinon, Evrard, et al., 2009) sets these parameters proportional to the likelihood of the current position and velocity of the robot's wrist. The desired acceleration $\ddot{\boldsymbol{x}}_d$ is numerically integrated to the positional command, which is sent to the robot.

The controller given in Eq. 4.2 exploits the GMMs' ability to infer the expected value of any variable encoded into a joint distribution $p(\boldsymbol{x}, \dot{\boldsymbol{x}}, \boldsymbol{f})$ conditioned on other known variables. Note that the perceived force $\boldsymbol{f}$ enters into the task model in Eq. 4.1 as a control input.

4.2.3 REPRODUCTION RESULTS

During reproduction, the robot is no longer teleoperated, but acts autonomously. The desired velocity and position for the robot's wrist are computed at each control iteration by integrating the acceleration given in Eq. 4.2.

Several human subjects are asked to lift a rigid beam together with the robot. They are explained how to execute the task, but otherwise are not given any specific instructions, whether to adapt to robot's motion or impose their own pace. In many reproduction trials, the object is lifted, but the demonstrated dynamics is not followed. In Fig.4.3, one can see that the force-velocity pattern is stretched along the force dimension. Such a deformation can be explained by the increase in the perceived force (e.g., a human tries to move faster) to which the robot does not react (does not change its velocity accordingly). Evrard et al. (2009) emphasize that the overall behavior of the system is qualitatively similar to the demonstrated one in the sense that the force-

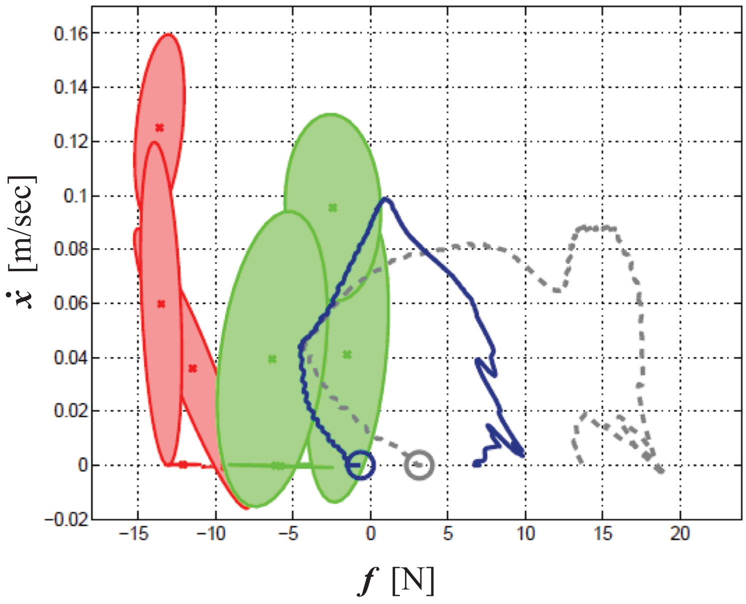


Figure 4.3: [Taken from (Evrard et al., 2009)] Dashed grey and solid blue lines show force-velocity patterns in failed reproduction trials where a human partner tries to stop the robot abruptly. The green and pink ovals represent Gaussian components of the learned task model and correspond to the these in Fig. 4.2-(b).

velocity correlation is still present. Therefore, the authors advocate that one way to improve robot performance during interaction is to rescale observed force-velocity patterns. However, one may argue that the scaling is a heuristic solution. Furthermore, finding suitable scaling factors is highly nontrivial. Instead, in this work, we propose an encoding which allow the robot to accommodate to force profiles different from the observed ones.

Evrard et al. (2009) also test the algorithm in *unobserved* situations by asking the human partner to depart from the original task motion, for instance, by stopping the motion abruptly or by trying to bring the object to a different location. These experiments provide useful insights into limitations of this learning approach. In particular, while the human is trying to stop the motion, the robot keeps moving upwards. This behavior is due to the properties of the chosen encoding: once the perceived force deviates from the observed values, its influence on the estimated velocity gets small and the velocity tends to be driven by the position signal (see Eq. 4.1). As a result, the robot keeps following the demonstrated trajectory and ignoring the partner's intentions. Some participants also report that they perceive the robot trying to accelerate upwards as they push the beam down.

To prevent such an undesirable behavior, it would be necessary to endow the robot with the ability to adapt to the partner's motion by generalizing learned interaction forces. In the next section, we propose an algorithm that addresses this problem and allow a robot to adapt to the pace of its partner.

4.3 MODEL INTRODUCTION AND A SIMULATION SET-UP

We further outline a method that combines task learning and impedance control so as to enhance robot's adaptive skills. We suggest that the ability to generalize and anticipate the perceived forces is highly important for generating an appropriate kinematic/dynamic response. Furthermore, as generalization and prediction may not account for all perturbations, for instance, for those induced by changes in human behavior (sudden deviations from the motion plan or varying arm impedance), additional mechanisms are required to mitigate such non-modeled effects.

When reviewing methods control of physical interaction in Section 2.1.4.2, we observe two major research directions: active following, where a robot is provided with task knowledge, and variable impedance, where the robot's impedance is adjusted so as to decrease efforts on the human side. In our approach, we pursue both directions; the proposed algorithm consists of two parts.

First, we learn a *task model* from demonstrations. The learned task model is used to generate a reference kinematic and feedforward control signals in response to perceived forces. The learned task model also allows the robot-follower to predict the perceived forces. This ability to anticipate incoming forces helps to adapt its motion on-line.

Second, we propose an *adaptive impedance controller* that compensates for non-

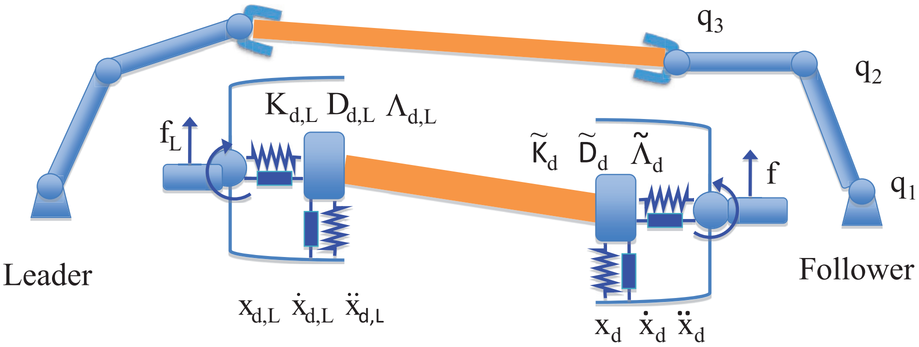


Figure 4.4: Two planar robots lift a beam in collaboration. For successful task completion the two robots have to coordinate and adapt their motions so as to avoid tilting the beam. The robot-leader substitutes the human. The desired kinematic plan $\boldsymbol{x}_{d,L}, \dot{\boldsymbol{x}}_{d,L}, \ddot{\boldsymbol{x}}_{d,L}$ of the robot-leader is pre-defined. The robot-follower anticipates the motion intentions of the robot-leader and adapts accordingly. During demonstration, the robot-follower learns to generate a desired kinematic command $\boldsymbol{x}_d, \dot{\boldsymbol{x}}_d, \ddot{\boldsymbol{x}}_d$ in response to the perceived force $\boldsymbol{f}$. The two robots are controlled by impedance control laws with desired stiffness, damping, and inertia. During task execution, the robot-follower adapts its desired stiffness $\tilde{\boldsymbol{K}}_d$ and inertia $\tilde{\boldsymbol{\Lambda}}_d$, so as to ensure accurate reproduction of a learned task model.

modeled effects. We extend the method proposed by Ganesh, Albu-Schaffer, et al. (2010) so as to encapsulate the force feedback error into the adaptation laws. We consider a full impedance with stiffness, damping, and inertia. Some studies on human motor control report qualitative analogy between hand stiffness and viscosity (damping) (Tsuji et al., 1995, 2004). Therefore, we additionally assume that the damping is correlated with the stiffness and develop adaptation laws for the stiffness and inertia.

We evaluate the performance of the proposed algorithm in controlled situations using a *physical simulation* of a pair of planar robots; see Fig. 4.4. The *robot-leader* mimics the role played by the human in the real-world experiments from Section 4.2, the *robot-follower* mimics the HRP-2 robot.

In our simulation set-up, both robots, the leader and the follower, are controlled by an impedance control law, which implementation is discussed in more detail in Section 4.4. The leader is to impose its kinematic behavior to the follower. Therefore, its stiffness is set to be much higher than that of the follower.

We introduce *delays*, *signal-dependent* noise, as well as change in the impedance of the controller of the follower.

The remainder of this section is organized as follows. Section 4.4 formulates an impedance control law that we use to control both simulated robots during data acquisition (Section 4.5) and task reproduction. Section 4.6 provides an overview of the control algorithm for the robot-follower (the robot-leader is controlled by a predefined impedance control law with fixed impedance parameters). Sections 4.6.1 and 4.6.2 discuss methods for learning task models and feedforward control respectively. Section 4.6.3 presents an adaptive algorithm for tuning impedance parameters. Section 4.7 illustrates the control algorithm through several experiments.

4.4 IMPEDANCE CONTROL

The goal of impedance control is to implement a desired dynamical relationship between the robot motion and the external forces/torques (Hogan, 1985). We consider the impedance of contact points defined in the Cartesian space, specifically, at the robot's end-effector.

We write the rigid body dynamics in the task coordinates (Khatib, 87)[3] as:

$$\boldsymbol{\Lambda}(\boldsymbol{x},\dot{\boldsymbol{x}})\ddot{\boldsymbol{x}}+\boldsymbol{\mu}(\boldsymbol{x},\dot{\boldsymbol{x}})\dot{\boldsymbol{x}}+\boldsymbol{J}^{-T}\boldsymbol{g}(\boldsymbol{x})=\boldsymbol{J}^{-T}\boldsymbol{\tau}+\boldsymbol{f} \tag{4.3}$$

Where the matrices $\boldsymbol{\Lambda}(\boldsymbol{x})$ and $\boldsymbol{\mu}(\boldsymbol{x},\dot{\boldsymbol{x}})$ are given by:

$$\boldsymbol{\Lambda}(\boldsymbol{x})=\boldsymbol{J}^{-T}\boldsymbol{M}\boldsymbol{J}^{-1} \quad \boldsymbol{\mu}(\boldsymbol{x},\dot{\boldsymbol{x}})=\boldsymbol{J}^{-1}(\mathbf{C}-\boldsymbol{M}\boldsymbol{J}^{-1}\dot{\boldsymbol{J}})\boldsymbol{J}^{-1}, \tag{4.4}$$

where $\boldsymbol{M}=\boldsymbol{M}(\boldsymbol{q})$ is the inertia matrix ($\boldsymbol{q}\in\mathbb{R}^{N_q}$ is the vector of joint angles), $\boldsymbol{C}=\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$ is the Coriolis/centrifugal matrix, $\boldsymbol{g}=\boldsymbol{g}(\boldsymbol{q})$ is the vector of gravity torques, $\boldsymbol{f}$ is the vector of external forces, $\boldsymbol{\tau}$ are the applied joint torques, and $\boldsymbol{J}$ is the Jacobian.

Impedance control in the task space sets the following control objective (Hogan, 1985):

$$\begin{aligned}&\boldsymbol{\Lambda}_d\ddot{\boldsymbol{e}}_x+\boldsymbol{D}_d\dot{\boldsymbol{e}}_x+\boldsymbol{K}_d\boldsymbol{e}_x=\boldsymbol{e}_f \qquad (4.5)\\&\text{where } \boldsymbol{e}_x=\boldsymbol{x}-\boldsymbol{x}_d \text{ and } \boldsymbol{e}_f=\boldsymbol{f}-\boldsymbol{f}_d.\end{aligned}$$

where $\boldsymbol{e}_x$ is the position error between the actual position $\boldsymbol{x}$ and the reference position $\boldsymbol{x}_d$; $\boldsymbol{e}_f=\boldsymbol{f}-\boldsymbol{f}_d$ measures how much the actual perceived force $\boldsymbol{f}$ deviates from the reference force $\boldsymbol{f}_d$. $\boldsymbol{\Lambda}_d$, $\boldsymbol{D}_d$, and $\boldsymbol{K}_d$ are the symmetric positive definite matrices of desired inertia, damping, and stiffness, respectively. The external forces defined in the Cartesian space are projected onto the joint torques according to: $\boldsymbol{\tau}_{ext}=\boldsymbol{J}^T\boldsymbol{f}$.

Substituting $\ddot{\boldsymbol{x}}$ from Eq. 4.5 into Eq. 4.3, we can implement the Cartesian impedance controller via the joint torques $\boldsymbol{\tau}$ as follows:

$$\boldsymbol{\tau}=\boldsymbol{u}+\boldsymbol{J}^T\tilde{\boldsymbol{K}}\boldsymbol{e}_x+\boldsymbol{J}^T\tilde{\boldsymbol{D}}\dot{\boldsymbol{e}}_x+\boldsymbol{J}^T\tilde{\boldsymbol{\Lambda}}_d\boldsymbol{e}_f, \tag{4.6}$$

$$\begin{aligned}&\text{where } \boldsymbol{u}=\boldsymbol{g}+\boldsymbol{J}^T(\boldsymbol{\Lambda}\ddot{\boldsymbol{x}}_d+\boldsymbol{\mu}\dot{\boldsymbol{x}}_d) \qquad (4.7)\\&\tilde{\boldsymbol{K}}_d=\boldsymbol{\Lambda}\boldsymbol{\Lambda}_d^{-1}\boldsymbol{K}_d,\ \tilde{\boldsymbol{D}}_d=\boldsymbol{\Lambda}\boldsymbol{\Lambda}_d^{-1}\boldsymbol{D}_d+\boldsymbol{\mu},\ \tilde{\boldsymbol{\Lambda}}_d=\boldsymbol{\Lambda}\boldsymbol{\Lambda}_d^{-1}-\boldsymbol{I}.\end{aligned}$$

Eq. 4.6 can be rewritten as a sum of the feedforward and feedback components. The feedback signal can be decomposed into the kinematic feedback $\boldsymbol{v}_x$ and the force feedback $\boldsymbol{v}_f$.

$$\begin{aligned}&\boldsymbol{\tau}=\boldsymbol{u}+\boldsymbol{v},\ \ \boldsymbol{v}=\boldsymbol{v}_x+\boldsymbol{v}_f \qquad (4.8)\\&\boldsymbol{v}_x=\boldsymbol{J}^T(\tilde{\boldsymbol{K}}_d\boldsymbol{e}_x+\tilde{\boldsymbol{D}}_d\dot{\boldsymbol{e}}_x),\ \boldsymbol{v}_f=\boldsymbol{J}^T\tilde{\boldsymbol{\Lambda}}_d\boldsymbol{e}_f.\end{aligned}$$

---

[3] The notation $^{-T}$ refers to pseudo-inverse of a transposed matrix. The notation $^{-1}$ refers to pseudoinverse if a matrix is noninvertible.

We will use this decomposition on kinematic and force feedback when discussing adaptive impedance in Section 4.6.3.

Finally, in our experiments, most variation occurs along the vertical axis. Therefore only the impedance parameters along the vertical axis are taken into account. Following common practice, we further set $\tilde{D}^j = \lambda\sqrt{\tilde{K}^j}$; here we set $\lambda = 2$.

4.5 TWO-STAGE TRAINING PROCEDURE

As we test our algorithm in simulation, we need a special procedure to obtain a training set. We suggest to generate training data through a two-stage training procedure: in each demonstration, the robot-follower alternates between "*passive*" and "*active*" stages. This procedure relates to the way humans incrementally learn to synchronize with each other. During the passive observation, the follower is not aware of intentions of the leader and tracks its own kinematic plan, however, with low stiffness. The leader therefore can correct the follower, but needs to apply efforts. During the active observations, the leader repeats the same motion, while the follower tracks the kinematic plan recorded during the passive observation. Such "proactive" adaptation on the follower's side allows for better alignment of both kinematic and force profiles between the two robots.

Before discussing these two stages in more detail, we first outline several assumptions built-in into the simulation set-up.

During training the robots are controlled with the impedance control law according to Eq. 4.6 with the pre-defined impedance parameters and the zero reference force $\boldsymbol{f}_d$. The choice of desired impedance parameters for the leader and the follower is discussed further in this section. The robots's inertia $\boldsymbol{\Lambda}(\boldsymbol{q})$, the Coriolis matrix $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$, and the gravity torques $\boldsymbol{g}(\boldsymbol{q})$ are computed analytically.

For both robots, the reference kinematics are generated by a dynamical system parameterized with a multiplicative parameter. Specifically, we use the VITE dynamical model of human reaching motions discussed in Section 3.2: $\ddot{\boldsymbol{x}}_d = \alpha(-\dot{\boldsymbol{x}}_d + 4(\boldsymbol{x}_{\text{tar}} - \boldsymbol{x}_d))$, where α is the multiplicative parameter, $\boldsymbol{x}_{\text{tar}}$ is the given target location.

To provide examples of adaptation to different velocities, the parameter α of the leader is varied from one demonstration to another, so as to generate motions with a maximum desired velocity of 0.2-0.5m/s and a duration of 0.4-0.8s. The α of the follower is fixed so to generate the reference motion of 0.8s with the maximum velocity 0.2m/s. In total, 15 demonstrations at different velocities are acquired.

The dynamical system produces the same task space trajectories but with different velocity profiles[4]. Hence, the reference kinematics profiles share the same goal (i.e. bring the beam in a specified location), but have dissimilar timing (due to different reference velocity profiles and sensory delays).

We simulate two types of sensorimotor limitations of the human motor system: a signal-dependent noise and sensory delay. A *reaction* delay (a time span between the

---

[4] A model parameterized with $\alpha = 10$ produces the same trajectories in the state-space $\{x_1; x_2\}$ as a model parameterized with $\alpha = 20$, however the latter converges to the target twice as fast.

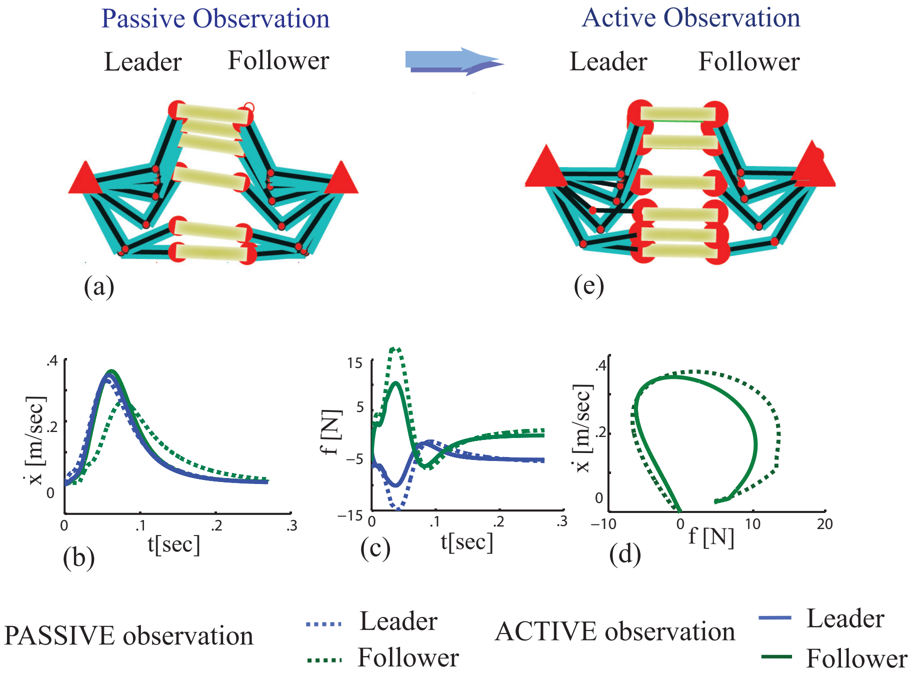


Figure 4.5: TWO-STAGE TRAINING PROCEDURE. To simulate real-world training, where the robot is teleoperated by a human, we adopt a two stage training procedure. Figs. (a), (e) present the robots' configurations during training. Desyncronization between the partners is greatly reduced during active observation, as expressed by the reduced tilting of the beam. *PASSIVE OBSERVATION*: The stiffness of the robot-follower is set to be low ($5N/m$) and the stiffness of the robot-leader is high ($50N/m$). This allows the robot-leader to impose its kinematic plan; see Fig. (b). The actual velocity of the robot-follower is higher than its reference signal and coincides with the actual and reference velocities of the robot-leader. Such a forced adaptation is achieved at the cost of considerable energy injection; see Figs. (c)-(d). The robot-follower perceives high positive external forces that are due to the effort of the robot-leader. After observing the task "passively", the robot-follower stores the kinematic information and discards the force signals. *ACTIVE OBSERVATION*: The stiffness of both partners is medium ($15N/m$). The robot-leader repeats the same reference kinematical profile as at the previous stage, while the robot-follower utilizes the kinematic profile acquired during passive observation. Improved synchronization decreases the magnitude of the forces perceived by both partners; see Figs. (c)-(d), solid line. The final training set is composed of the velocity signal recorded during passive observation, and the external forces/applied torques recorded during active observation.

moments when the follower starts perceiving that the force is changing and when it actually starts moving) is assigned to be 150ms; the *perception delay* along the motion (delay between perception of a force and reaction) is 3ms.

4.5.1 PASSIVE OBSERVATION

The two robots are controlled to track their reference kinematic profiles generated with the VITE system as discussed above. The stiffness of the robot-follower is set to be low (5N/m) and the stiffness of the robot-leader is high (50N/m). The robot-leader imposes its motion plan to the partner; see Fig. 4.5-(b). This requires the leader to inject a considerable amount of energy; see Fig. 4.5-(c),(d), dashed line. Therefore, even though the follower tracks the motion of the leader (the actual velocity of the follower is close to that of the leader), the perceived forces are different from those that would be observed if the follower reproduced a correct velocity profile intentionally. To observe these forces, the follower reproduces the observed kinematics in the next training step.

4.5.2 ACTIVE OBSERVATION

The robot-leader tracks the same reference trajectory as during the passive stage. The robot-follower utilizes as the reference signal the actual kinematics $\boldsymbol{x}, \dot{\boldsymbol{x}}, \ddot{\boldsymbol{x}}$ recorded during passive observation (as it better matches the reference kinematics of the leader; see Fig. 4.5-(b)).

By deliberately reproducing this imposed kinematic profile, the follower generates forces that are better aligned with those of the leader. The leader injects less energy and, therefore, the forces perceived by the follower are smaller than those measured during passive observation; see Fig. 4.5-(c),(d), solid line. Fig. 4.5-(a),(e) highlights improvements in synchronization across the two stages[5]. The collected data (actual velocity, forces, and the feedforward commands of the follower) are further used to learn the task model $\hat{\boldsymbol{h}}$ and the feedforward control $\boldsymbol{u}$.

4.6 APPROACH

In this section, we discuss how a task model is learned from a set of demonstrations, and present the adaptive impedance control law that accounts for compensation of effects not captured by the learned model. Fig. 4.6 shows the control flow of our model. After acquiring a *set of demonstrations* $\mathcal{D}$, the robot *learns* the task model $\dot{\boldsymbol{\xi}} = \hat{\mathbf{h}}(\boldsymbol{\xi})$ and a forward control signal $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{\xi})$ that maps the desired state $\boldsymbol{\xi}$ of the task model to actual motor commands. The dynamical system representation of the task model allows the robot to generate *reference signals* on-line adapting to the force applied by a human. The robot is controlled through an *impedance control law* so as to compensate for non-modeled aspects of the external dynamics. The *desired stiffness* $\tilde{\mathbf{K}}_d$ and *inertia* $\tilde{\mathbf{\Lambda}}_d$ are *adapted* during task execution.

4.6.1 LEARNING A TASK MODEL

A training set $\mathcal{D}$ consists of M demonstrated trajectories of length N^k, for $k = 1..M$. Each trajectory is a sequence of states, $\boldsymbol{\xi}_t^k$, state derivatives $\dot{\boldsymbol{\xi}}_t^k$, and control inputs $\boldsymbol{u}_t^k$, $t = 1..N^k$. The control input $\boldsymbol{u}_t^k$ is the vector of joint torques. The task state is an *augmented state* $\boldsymbol{\xi} = [\dot{\boldsymbol{x}}_d, \boldsymbol{f}_d]^T$ that consists of the reference velocity $\dot{\boldsymbol{x}}_d$, and the reference force $\boldsymbol{f}_d$, to be measured by the force sensor mounted at the robot's wrist.

---

[5] During active observation the decrease in the perceived forces is mainly caused by an improved synchronization. But the lowered stiffness of the robot-leader also contributes to this decrease. Indeed, the robot-leader does not forcefully guide the robot-follower.

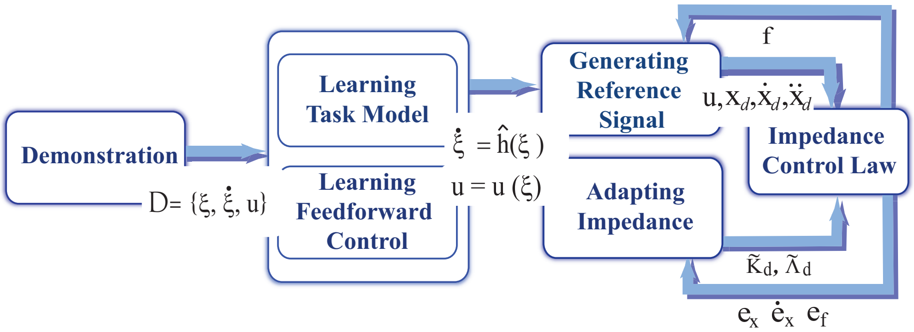


Figure 4.6: After acquiring a *set of demonstrations* $\mathcal{D}$, the robot *learns* the task model $\dot{\boldsymbol{\xi}} = \hat{\mathbf{h}}(\boldsymbol{\xi})$ and a forward control signal $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{\xi})$ that maps the desired state $\boldsymbol{\xi}$ of the task model to actual motor commands.The dynamical system representation of the task model allows the robot to generate *reference signals* on-line adapting to the force applied by a human. The robot is controlled through an *impedance control law* so as to compensate for non-modeled aspects of the external dynamics. The *desired stiffness* $\tilde{\mathbf{K}}_d$ and *inertia* $\tilde{\mathbf{\Lambda}}_d$ are *adapted* during task execution.

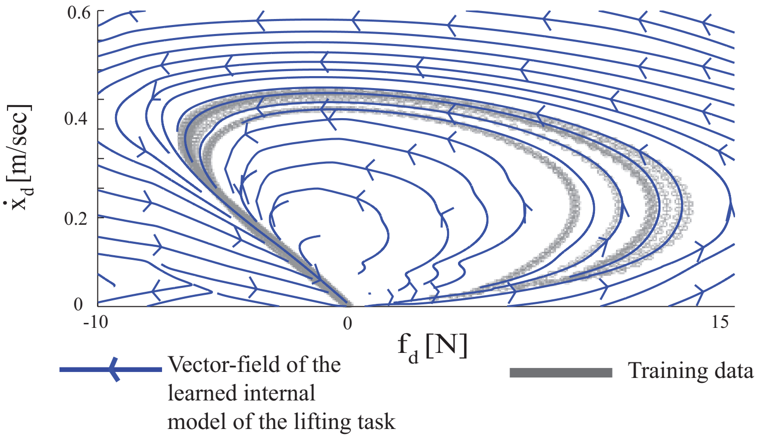


Figure 4.7: The task model is represented by a dynamical system $\dot{\boldsymbol{\xi}} = \hat{\mathbf{h}}(\boldsymbol{\xi}),\ \ \boldsymbol{\xi} = [\dot{\boldsymbol{x}}_d;\boldsymbol{f}_d]$ and estimated from the training data. At each time step, the velocity $\dot{\boldsymbol{x}}_d$ and force $\boldsymbol{f}_d$ are inferred from these observed at the previous step. Their dynamical relationships follow vector fields displayed in blue. Dark gray lines show the demonstrations. One can observe an accurate fit between the inferred and demonstrated dynamics. Statistical inference extends prediction of the force-velocity pattern to ranges of these variables not observed during training. This offers a greater robustness during adaptation to a new human partner.

Following the method proposed in Section 3.3, we assume that the task model is governed by an autonomous (time-independent) dynamical system, where the acceleration of the robot's end-effector is a function of its velocity and the perceived force:

$$\dot{\boldsymbol{\xi}} = \mathbf{h}(\boldsymbol{\xi}) + \boldsymbol{\eta}(\boldsymbol{\xi}) \sim \hat{\mathbf{h}}(\boldsymbol{\xi}) = \mathbb{E}[\ p(\dot{\boldsymbol{\xi}}|\boldsymbol{\xi})\ ] \tag{4.9}$$

where $\mathbf{h}(\boldsymbol{\xi})$ is the dynamic function governing the temporal evolution of the motion, and $\boldsymbol{\eta}(\boldsymbol{\xi}) \sim N(0, \boldsymbol{\Sigma}_\eta(\boldsymbol{\xi}))$ is the signal-dependent noise.

Given a training set $\mathcal{D}$ (see Section 4.5 for a description of the data acquisition procedure), we learn the task model $\hat{\mathbf{h}}(\boldsymbol{\xi})$ by estimating the joint density $p(\boldsymbol{\xi}, \dot{\boldsymbol{\xi}})$ through Gaussian Mixture Models; given the joint density, we compute the conditional expectation from Eq. 4.9. We apply the incremental EM procedure that we developed in Section 3.3 to ensure that the estimate of the velocity is asymptotically stable at the origin of the system. In this context, the stability ensures that when the force perceived at the end-effector is null, the robot stops moving, as shown in Fig. 4.7.

Note that we introduce the augmented state $\boldsymbol{\xi}$ that encapsulates both the kinematic command $\dot{\boldsymbol{x}}$ and the haptic input $\boldsymbol{f}$. In contrast to the formulation in Eq.4.2 where the interaction force is considered only as a control input, Eq. 4.9 includes the force as a state variable. Including the force as a state variable allows learned task models to account for a richer class of velocity-force correlations. Specifically, data acquired during experiments with the HRP-2 robot shows that the dependency between velocity and perceived force is non- functional: a single value of force corresponds to different velocities; see Fig. 4.2. Therefore, assuming the one-to-one function dependency, as it is done in Eq.4.2, distorts an actual coordination pattern and leads to undesirable behaviors discussed in Section 4.2.3.

Learning a task model as a dynamical function of the augmented state captures a temporal evolution of the reference velocity correlated with the perceived external force. An advantage of this formulation is that the robot can switch across different reference velocity profiles in response to a change in the partner's intentions (i.e. different interaction forces).

Another advantage of this encoding is that the task model can be used to mitigate sensory delays by predicting the perceived force. Specifically, to generate a reference kinematics, the robot does not need to get actual value of the force at each time step, it can predict the perceived force from the task model. Later, once it gets the actual value of the force, the robot may offset its prediction so as to switch to a different velocity profile if necessary.

4.6.2 LEARNING FEEDFORWARD CONTROL

Note that the analytical computation of a feedforward control $\boldsymbol{u}$ from Eq. 4.7 requires an accurate model the robot's dynamics. For the planar robots that we use in our simulation, the dynamical model can be computed analytically, however for real robots such a model is rarely available. As we discuss in Section 2.4.4, there exist different algorithms for approximating the feedforward control, here we learn $\boldsymbol{u}$ from demonstrations.

The feedforward control input $\boldsymbol{u}$ is generally a function of the robot's desired and actual states. Given a task model $\hat{\mathbf{h}}(\boldsymbol{\xi})$ that defines the nonlinear dependency between these parameters, $\boldsymbol{u}$ becomes a function of the task state $\boldsymbol{\xi}$. Following Ganesh, Albu-Schaffer, et al. (2010), we learn an estimate of the feedforward command $\mathbf{u}$ in the linear form:

$$\boldsymbol{u} = [\boldsymbol{\Phi}(\boldsymbol{\xi})^T \boldsymbol{\Theta}]^T, \tag{4.10}$$

where $\boldsymbol{\Phi} \in \mathbb{R}^{K(N_x+N_f)}$ is a vector of G basis functions and $\boldsymbol{\Theta} \in \mathbb{R}^{K(N_x+N_f)\times N_q}$ is a matrix of the tunable parameters (each column $\boldsymbol{\theta}^i$, $i = 1..N_q$, of the matrix $\boldsymbol{\Theta}$ corresponds to one degree of freedom in the joint space). N_x, N_f, N_q refer to the dimensionality of the Cartesian space of the robot's end-effector, perceived force, and the joint space, respectively. Linear control parametrization given in Eq. 4.10 is commonly used in the adaptive control literature (Astrom & Wittenmark, 1989; Burdet & Codourey, 1998). The basis $\boldsymbol{\Phi}(\boldsymbol{\xi})$ consists of G Gaussian functions:

$$\begin{aligned} \boldsymbol{\Phi}_j = \boldsymbol{\Phi}(\boldsymbol{\xi})_j &= \frac{\pi_j(\boldsymbol{\xi})\ \boldsymbol{\xi}}{\sum_{g=1}^{G} \pi_g(\boldsymbol{\xi})},\ \ j = 1..G \qquad (4.11) \\ \text{where } \pi_j(\boldsymbol{\xi}) &= \exp\left(-0.5(\boldsymbol{\xi} - \boldsymbol{\mu}_{\boldsymbol{\xi},j})^T \boldsymbol{\Sigma}_j^{-1} (\boldsymbol{\xi} - \boldsymbol{\mu}_{\boldsymbol{\xi},j})\right) \end{aligned}$$

with $\boldsymbol{\mu}_{\boldsymbol{\xi},j},\ \boldsymbol{\Sigma}_j$ are the mean and the diagonal covariance of a jth Gaussian kernel. Given the training set $\mathcal{D}$, we learn the parameterization in Eq.4.10-4.11 through Linear Weighted Regression (Atkeson et al., 1997).

4.6.3 ADAPTIVE IMPEDANCE

In this section we describe an adaptive control algorithm for tuning the impedance parameters and feedforward control; the algorithm is an extension of a bio-mimetic approach for tuning the stiffness (Ganesh, Albu-Schaffer, et al., 2010) (see summary in Appendix III). We write an adaptation law for the feedforward parameters $\boldsymbol{\theta}^i$ ($\forall i = 1..N_q$) from Eq. 4.10 as follows:

$$\Delta\boldsymbol{\theta}^i = \frac{\beta}{2}((1-\chi)\boldsymbol{\Phi}\epsilon^i + (1+\chi)\boldsymbol{\Phi}|\epsilon^i|) - \gamma\boldsymbol{1} \qquad (4.12)$$

where ϵ is the error function, that will be defined later and $\beta,\ \chi$, and γ are empirical constants. Ganesh, Albu-Schaffer, et al. (2010) use a purely kinematic error function, we extend the definition to include the force error.

We introduce two feedback terms, $\boldsymbol{e}_f$ and $\boldsymbol{e}_m$ in addition to the original formulation by Ganesh, Albu-Schaffer, et al. (2010). These terms denote an error between the predicted and the actual perceived force $\boldsymbol{e}_f$ and a kinematic error $\boldsymbol{e}_m$, which is caused by the discrepancies in the learned feedforward control $\boldsymbol{u}(\boldsymbol{\xi})$:

$$\begin{aligned} \boldsymbol{\epsilon} &= J^T(\boldsymbol{e}_m + \rho_3\boldsymbol{e}_f) \qquad (4.13) \\ \boldsymbol{e}_m &= \ell \min(\boldsymbol{f}^T\dot{\boldsymbol{e}}_x, 0)(\rho_1\boldsymbol{e}_x + \rho_2\dot{\boldsymbol{e}}_x) \end{aligned}$$

where $\ell = \|\boldsymbol{f}^T\dot{\boldsymbol{e}}_x\|^{-1}$ is the normalization factor. The parameters $\rho_1, \rho_2, \rho_3 \in \mathbb{R}$ weight the importance given to errors in the trajectory, the velocity, and the force.

Kinematic Error

During free-space motions, the main cause of kinematic errors $\boldsymbol{e}_x$ and $\dot{\boldsymbol{e}}_x$ is inaccuracies in the feedforward control. In contrast, during physical coordination, the kinematic errors might be also due to adaptation to partner's intentions. For instance, when the robot-follower departs from a learned task model to accommodate changes in a partner's motion. If the errors are due to inaccuracies in feedforward control, the kinematic errors should be corrected by increasing stiffness and adjusting $\boldsymbol{u}(\boldsymbol{\xi})$. If the kinematic errors are due to adaptation, we suggest that the stiffness should not be increased or the robot would be forced to counteract the partner.

We use a simplified geometrical reasoning to write down the kinematic error in Eq. 4.13. We assume that only the part of the actual kinematic error that is collinear and oppositely directed to the perceived force $\boldsymbol{f}$ should be used for the adaptation of stiffness and the feedforward control. If the kinematic error is co-directed with the interaction force (the kinematic error is assumed to be caused by adaptation), the stiffness is not increased.

Adaptation law

Next, we use the definition of the feedback error $\boldsymbol{\epsilon}$ to infer adaptation rules for impedance and feedforward parameters. Substituting Eq.4.13 into Eq. 4.12, we can rewrite:

$$\begin{aligned}\Delta\boldsymbol{\theta}^i = & \frac{\beta}{2}(1-\chi_1)\boldsymbol{\Phi}J^T\boldsymbol{e}_m + \frac{\beta}{2}(1+\chi_1)\boldsymbol{\Phi}|J^T\boldsymbol{e}_m| + \\ & \frac{\beta}{2}(1-\chi_2)\boldsymbol{\Phi}J^T\boldsymbol{e}_f + \frac{\beta}{2}(1+\chi_2)\boldsymbol{\Phi}|J^T\boldsymbol{e}_f| - \gamma\boldsymbol{1}.\end{aligned} \tag{4.14}$$

Next, we discuss the components in Eq. 4.14 and propose the algorithm for adaptation of the impedance parameters during physical coordination with a human.

In Eq. 4.14, the terms $\frac{\beta}{2}(1-\chi_1)\boldsymbol{\Phi}\epsilon_m^i$ and $\frac{\beta}{2}(1-\chi_2)\boldsymbol{\Phi}\epsilon_f^i$ correspond to the regular feedback adaptation terms. Specifically, $\frac{\beta}{2}(1-\chi_1)\boldsymbol{\Phi}\epsilon_m^i$ generates a force in the direction opposite to the kinematic error $\boldsymbol{e}_m$, and updates the feedforward signal $\boldsymbol{u}$. $\frac{\beta}{2}(1-\chi_2)\boldsymbol{\Phi}\epsilon_f^i$ compensates for the deviation of the external force from its reference value and contributes to the adaptation of the desired inertia.

The terms dependent on the absolute values of the errors aim at tuning the stiffness. $\frac{\beta}{2}(1+\chi_1)\boldsymbol{\Phi}|\epsilon_m^i|$ increases stability in response to kinematic perturbation, while $\frac{\beta}{2}(1+\chi_2)\boldsymbol{\Phi}|\epsilon_f^i|$ decreases the stiffness if the deviation of the external force is increasing. Indeed, a sudden increase in the force error $\boldsymbol{e}_f$ means that the human is attempting to impose a different motion plan, and hence the robot should decrease the stiffness so as to maintain stable interaction.

Analogous to the method of Ganesh, Albu-Schaffer, et al. (2010), the update mechanism emulates automatic relaxation through the term $\gamma\boldsymbol{1}$. This is similar to a motor behavior observed in humans who, in the absence of motion errors, tend to relax muscles so as to minimize energy consumption (Franklin et al., 2008). Ganesh, Albu-Schaffer, et al. (2010) advocate that the components involving absolute values of the errors are responsible for muscle "co-activation" and should affect adaptation of impedance in robots. Following this reasoning, we rearrange the terms in Eq. 4.14; the update procedure for the forward signal and the impedance parameters then can be written as follows:

$$\Delta\boldsymbol{\theta}^i = \kappa_x \boldsymbol{\Phi}(J^T \boldsymbol{e}_m)^i - \boldsymbol{\gamma}_\theta, \quad \kappa_x > 0, i = 1..N_q \tag{4.15}$$

$$\Delta\tilde{K}_d^j = \beta_x |e_m^j| - \beta_f |e_f^j| - \gamma_{\tilde{K}}, \quad \beta_x, \beta_f, \gamma > 0, \tag{4.16}$$

$$\Delta\tilde{\Lambda}_d^j = \kappa_f e_f^j - \gamma_{\tilde{\Lambda}}, \;\; j = 1..N_x \quad \kappa_f > 0. \tag{4.17}$$

Eq. 4.6 together with Eq. 4.10,4.15-4.17 represent the control algorithm that enables the simultaneous on-line adaptation of the feedforward signal, the desired stiffness, and the inertia. Note, to avoid instabilities, we consider adaptation only within predefined boundaries: $2 < K_d^j < 100$.

In our experiments we choose β_x in Eq. 4.16 to be smaller than β_f: the robot's stiffness increases slowly with an increasing kinematic error and drops fast as the force error grows. The proposed treatment of the kinematic error prevents the robot from increasing stiffness while it adapts to the partner. However, the stiffness is still affected by the force error and cannot grow much. This is a consequence of the proposed adaptation rules in Eq.4.15-4.17.

4.7 RESULTS

We assess our method in simulations where the robot-follower interacts with the robot-leader, see Fig. 4.4. To highlight different types of adaptation handled by our algorithm, we simulate different conditions that may arise during execution of the collaborative tasks, and that would require on-the-fly adaptation of the robot's control law.

4.7.1 LEARNING A TASK MODEL

The acquired training data are depicted in Fig. 4.8-(a),(b). Note that the data exhibit a force-velocity correlation, which is similar to the one observed in real-world data acquired with the HRP-2 robot (see (Evrard et al., 2009)). Importantly, the force-velocity dependency defines a task model: how to generate a motion that is consistent with perceived force; i.e. for a given value of the perceived force, the robot estimates the relevant velocity. After data acquisition, the robot learns the task model $\dot{\boldsymbol{\xi}} = \hat{\mathbf{h}}(\boldsymbol{\xi})$ and the forward control model $\boldsymbol{u} = \boldsymbol{u}(\boldsymbol{\xi})$. During reproduction, the learned models are fed into the control law in Eq. 4.6. The results of the task execution are depicted in green in Fig. 4.8. The robot-follower successfully adapts its velocity and synchronizes with the robot-leader. Note that because of a non-functional dependency between velocity and force, the algorithm by Evrard et al. (2009) would not accommodate to such a task.

4.7.2 ADAPTATION TO PERTURBATIONS

We tested the ability of the learned model to adapt to changing intentions of the leader during task execution. We simulated uncertainties about the motion objectives by varying the target position $\boldsymbol{x}_{\text{tar}}$ in the motion plan of the leader. Three cases are considered where the leader changes the motion plan during task execution. It decides to move the

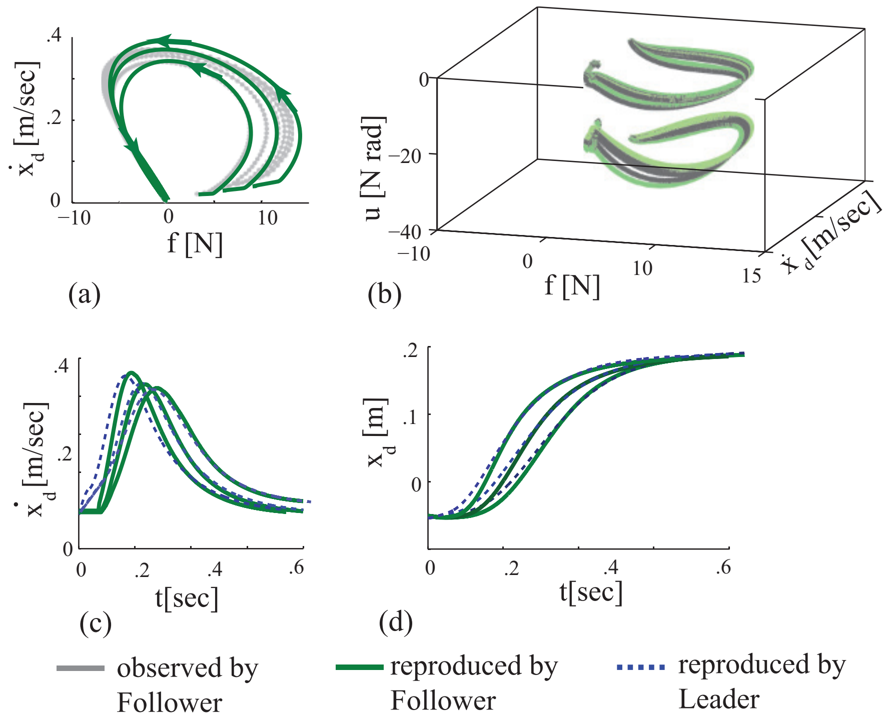


Figure 4.8: TASK LEARNING AND REPRODUCTION. In this experiment, the robot-follower learns the lifting task by observing demonstrations performed with different velocity profiles imposed by the leader. During reproduction, the robot-leader varies its kinematics plan from one attempt to another; it does so by changing motion duration and maximum velocity. The robot-follower, governed by the learned task and control model, adapts its motion and successfully accomplishes the task. (a) The state-space view of the data used for training a task model (gray line) and the reproduction attempts (green line). (b) The forward control signal generated by the follower during demonstration (gray line) and reproduction (green line). The two datasets correspond to the two joints of the planar robot-follower. (c)-(d) The time-series of the Cartesian velocity and trajectory of the robot-follower during reproduction. The follower (green line) adjusts its kinematics and synchronize with the leader (dashed blue line).

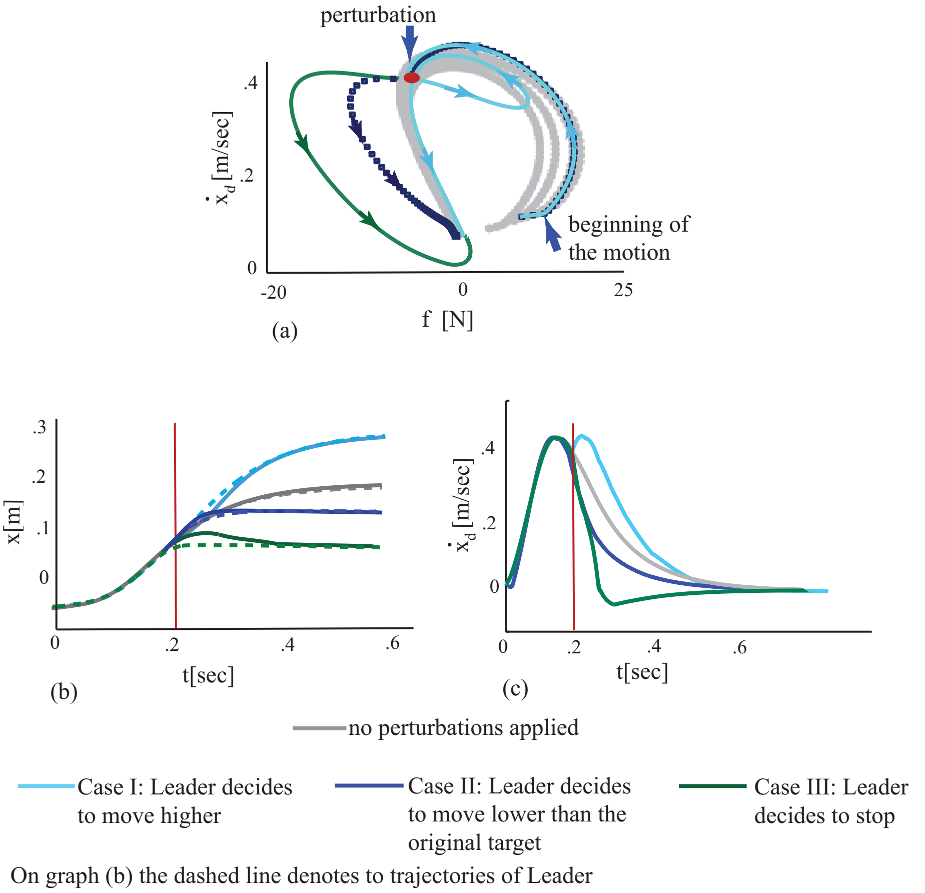


Figure 4.9: ADAPTATION TO PERTURBATIONS. Three cases are analyzed: the robot-leader changes the motion plan on-line and move the beam (1) higher than initially planned, (2) lower, but higher than the actual position of the beam at the moment the change decision is taken, and (3) stops at a position lower than the actual position at the decision-taking moment. In case (1), the robot-follower manages to re-accelerate (see the two peaks in the velocity profile). In cases (2) the robot-follower pro-actively decelerates. In case (3), the robot also manages to smoothly drop velocity below zero and lower the beam.

beam (1) higher than it has planned initially, (2) lower, but still higher than the actual position of the robots at the moment of taking the decision, and (3) lower than both the original target position and the actual position. Fig. 4.9 shows that the follower succeeds in bringing the beam to the new desired location in each case. Such an adaptation is possible because the learned model generalizes the force and velocity patterns to values not observed during training (as illustrated in Fig. 4.7).

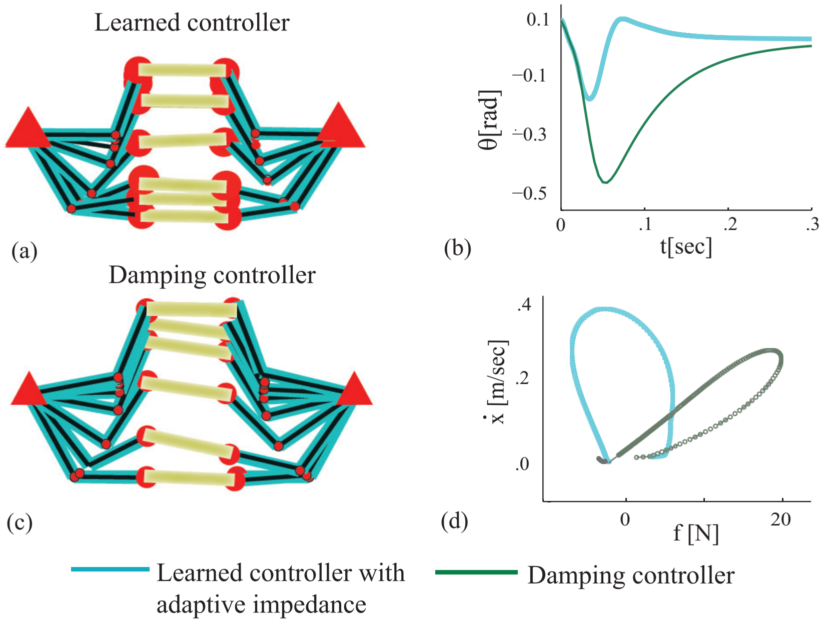


Figure 4.10: COMPARISON WITH A DAMPING CONTROLLER. We compare our system versus the damping controller. (a), (c) The planar robots performing lifting; the follower is controlled by our controller (a), and the damping controller (c). One may notice the improvements in coordination between the partners when the robot-follower adapts its kinematic profile (a): the beam is kept horizontal all along the motion. It is persistently tilted when the follower is controlled with the damping (c). (b) θ characterizes deviation of the beam from the horizontal orientation. (d) The leader has to apply considerably higher forces to make the system move.

4.7.3 COMPARISON WITH A DAMPING CONTROLLER

To highlight the advantages of the proposed approach for controlling a robot during collaborative manipulation tasks, we also implemented a damping controller according to (Maeda et al., 2001). This easy to implement and computationally cheap method is often used to control robots during physical collaboration with people. The only free parameter in this controller is the damping coefficient; the reference kinematics and all other impedance parameters are set to zero. The damping control has been proven to be useful and efficient in many applications; however, it puts additional workload on the human and is not adequate to control for fast motions.

We compare our system versus the damping controller in Fig. 4.10. The forces perceived by the robot are much higher than those observed with our learned system (Fig. 4.10-(d)). This is due to the higher forces that the leader has to apply to maintain the interaction. The beam undergoes stronger rotation (Fig. 4.10-(b)) due to the imbalanced forces on its two sides. This will be highly undesirable if the beam is loaded with unfixed objects that may slip down. However, we should emphasize, that our controller possesses local knowledge and far from the demonstrated data, its prediction is irrelevant. In this case, we suggest switching to a damping controller to ensure safety.

4.7.4 IMPEDANCE ADAPTATION

In the previous experiments we reused the impedance parameters that the two robots had during training. However, in general the impedance parameters of the robot-follower are unknown, e.g., if the demonstrations are provided through teleoperation as discussed in Section 4.2. We now assume that the robot-follower has no information about the impedance it should apply at the end-effector; hence it should adapt the impedance parameters on-line, during task reproduction.

As discussed in Section 4.6.3, we consider adaptation of stiffness $\tilde{\boldsymbol{K}}_d$ and inertia $\tilde{\boldsymbol{\Lambda}}_d$, while assuming that the damping is correlated with the stiffness. If the stiffness $\tilde{\boldsymbol{K}}_d$ of the follower is too high, the robot would reject discrepancies in coordination with the leader. If, in response, the parter is sufficiently stiff, the interaction would be unstable. The effect of unadjusted inertia $\tilde{\boldsymbol{\Lambda}}_d$ is less intuitive. To motivate further discussion, we illustrate the impact of unadjusted inertia parameters. Fig.4.11 compares three cases, where the follower executed the task with: 1) the same impedance parameters as used during demonstration; 2) the desired inertia $\tilde{\boldsymbol{\Lambda}}_d$ is set higher than during demonstration; and 3) the desired inertia $\tilde{\boldsymbol{\Lambda}}_d$ is set lower than during demonstration. Note that the changes in the inertia affects the perceived force and prediction of the desired velocity: the unadjusted inertia leads to either the underestimation or overestimation of the velocity in comparison to this followed by the leader (green line).

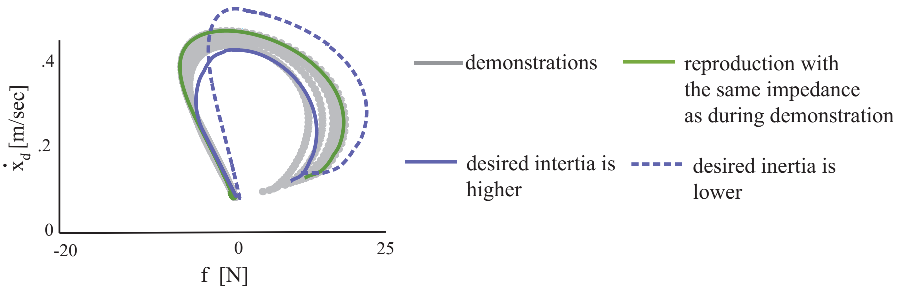


Figure 4.11: AN IMPACT OF DESIRED INERTIA. We demonstrate three cases, where the follower executes the task with: 1) same impedance parameters as used during demonstration; 2) the desired inertia $\tilde{\boldsymbol{\Lambda}}_d$ is set higher than during demonstration; and 3) the desired inertia $\tilde{\boldsymbol{\Lambda}}_d$ is set lower than during demonstration. Note that changes in the inertia affects the perceived force and prediction of the desired velocity.

In the next experiment, we change both desired stiffness $\tilde{\boldsymbol{K}}_d$ and inertia $\tilde{\boldsymbol{\Lambda}}_d$. The experiment starts with the follower's stiffness and inertia set to mid-range values of $\tilde{\boldsymbol{K}}_d = 35$N/m and $\tilde{\boldsymbol{\Lambda}}_d = 0$ respectively (in contrast, during demonstration the two parameters were $\tilde{\boldsymbol{K}}_d = 10\text{N/m}$ and $\tilde{\boldsymbol{\Lambda}}_d = 0.7$). This is an important difference in the parameter values. One can see in Fig. 4.12-(a), dashed line, that reproduction without adaptation of the impedance parameters leads to an overestimated reference velocity and difficulties to stop at the target (oscillations in the velocity profile). Adaptation allows for tuning of the parameters so as to ensure stable interaction; see Fig. 4.12-(a), green line. After slightly growing in the beginning of the motion, to enable for smooth acceleration, the stiffness gradually decreases due to the relaxation term and errors in tracking the desired force profile; see Fig. 4.12-(d). The inertia, in turn, decreases in the beginning, to ensure the stable onset of the motion. It further increases to endow the robot-follower with greater reactivity to the partner's intentions; see Fig. 4.12-(c).

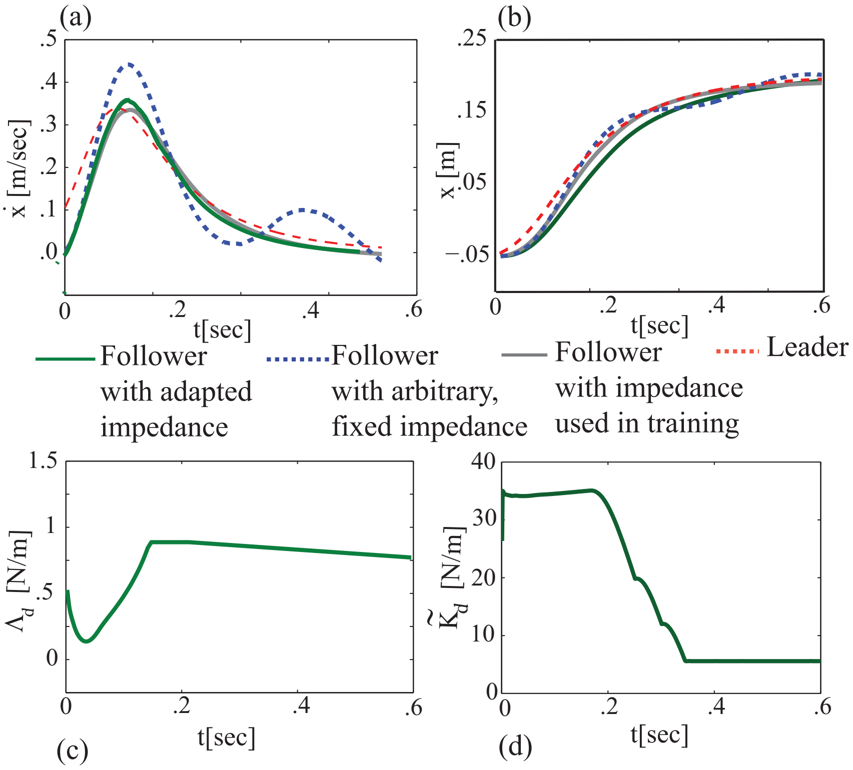


Figure 4.12: ADAPTATION OF UNKNOWN IMPEDANCE. In general, the impedance parameters that would be optimal for the task are unknown. Therefore, the follower should tune its stiffness and inertia during task execution. Arbitrary impedance parameters may cause undesirable effects, e.g. overestimated reference kinematics and contact instabilities (blue dashed line). Our algorithm provides an adaptation law, to tune the parameters and to ensure accurate reproduction (green lines in (a)-(b) show the results with impedance adaptation).

4.8 Discussion

Next we revisit some assumptions taken in the proposed algorithm.

Active Following

In robot learning, the few existing works consider a single learned trajectory as a reference signal for a hard-coded impedance controller. In (D. Lee et al., 2010), a robot is taught to clap hands with a human. The robot utilizes Hidden Markov Models (HMM) to recognize the human behavior from motion data and generate a reference trajectory. This trajectory is further incorporated into a hard-coded impedance controller to compensate for a potential physical impact. The considered scenario does not require continuous interaction and haptic signals do not effect the reference kinematics. A hand-shaking robot is presented by Z. Wang et al. (2009). The authors encode motion trajectories with an HMM, where the hidden variables represent the human impedance. Such encoding requires the robot to measure human impedance and to recognize which motion model to use. Recognition happens at the onset of the motion and governs the robot through the rest of the task without adaptation.

It might be argued that interaction with a human requires continuous proactive adaptation. We demonstrate how the use of dynamical systems for encoding task models can enhance coordination skills of a robot. Specifically, incorporating the perceived force as one of state variables of the dynamical system allows for action-perception coupling. A side effect of such formulation is the requirement that a robot should be equipped with a force sensor, which is not always available.

Variable Impedance

As discussed in Section 4.4, recent attempts to implement physical human-robot interaction introduced the notions of variable impedance. For instance in work of Duchaine & Gosselin (2007) a robot adjusts its damping depending on the perceived force. Although the validity of the approach is confirmed by successful experimental results, so far no generic framework for tackling both task learning and the variable impedance during physical interaction with a human has been reported in the literature.

Recent work by C. Yang et al. (2011) presents a biomimetic learning controller able to adapt to unknown dynamic environments. The algorithm combines between-trial trajectory learning and adaptive impedance. The algorithm endows a robot with the capacity to increase impedance if operating in an unstable environment and to act compliantly otherwise. The authors develop a rigorous adaptation mechanism that ensures stability of the control law. However, in the reported experiments, a human is considered as a source of external perturbations, rather than a partner.

Our work suggests one step towards implementing the variable impedance for collaboration with a human. We hypothesize that during collaboration impedance might adapt differently than in situations when the robot is required to counteract all external forces. This assumption would need to be further investigated.

Stability

To ensure safety during interaction, we incorporate security mechanisms that bound robot impedance and generated forces. The presence of a human in the control loop makes a formal stability analysis particularly challenging. The proof of stability would have to demonstrate that the errors $\boldsymbol{e}_x$, $\dot{\boldsymbol{e}}_x$, and $\boldsymbol{e}_f$ converge to zero with time. However, the human brings uncertainty into errors and, therefore, it is difficult to show convergence unless one adopts hypotheses about human behavior.

4.9 CONCLUSION

In this Chapter, we presented an approach to learning robot control during physical interaction with humans. The method addressed the problem of controlling a robot so that it could coordinate its motions with that of a human in collaborative tasks, and this while relying solely on haptic and proprioceptive feedback (no vision or verbal commands was involved).

We demonstrated that, in contrast to other works on physical human-robot interaction, our method allowed the adaptation within an execution trial, and not only from trial to trial. It was argued that due to the presence of a human in the loop, this characteristic was essential. We could not ensure that at the next trial the person would repeat the task identically and provide the robot with time to tune its controller.

Here, we considered non-redundant robots; however, the method would be also applicable to redundant set-ups; for instance, one could follow a projection based approach (Ott, 2008) to assign a null-space impedance matrices.

To conclude, the proposed system endowed the robot with two fundamental features of human motor control that emerged during physical interaction: learning haptic communication in a natural manner, and continuous adaptation to incoming forces during task execution. Additionally, the simulator developed to validate our approach provided an efficient means to study physical interactions between two agents for which we had yet very few models. Simulation offered a framework for systematical assessment and performance comparison of different algorithms for control of human-robot interaction.

Chapter 5

CONCLUSION

In this chapter, we revisit the contributions of the thesis and assess major limitations of the proposed algorithms (Section 5.1), as well as identify promising directions of future work (Section 5.2).

5.1 MAIN CONTRIBUTIONS AND LIMITATIONS

In the Introduction, we announced the major contributions of this manuscript. Next, while discussing the theoretical background of our work in Chapter 2, we emphasized current challenges for the state of the art in robotics. Here, we consolidate this information so as to provide a global outlook of our work and conclude the advancements done in this thesis.

Learning Bimanual Tasks

As we learned from the review on human motor control provided in Section 2.3: in humans, the two arms were not independent in bimanual tasks. It has been shown that the arms were coupled through spatial and temporal constraints. These constraints were argued to carry a practical function of facilitating manipulation. In robots, as in humans, the coordination constraints could improve the motion dexterity. However, these constraints were revealed to be task-dependent. For analytical approaches, the task-dependency of the constraints meant that for each particular task, a robotic controller should have been handcrafted. We explained this difficulty in Sections 2.1.5 and 2.3.4. Another concern was also raised: even if one knew the constraints, how to ensure that the robot would adhere to them, especially, if it was perturbed (e.g., pushed away) while moving.

We stated in the Introduction that our work would contribute to the resolution of these difficulties. In Section 3.2, we further showed that by adopting the coordination dynamics view of the problem (namely, the hypothesis that the constraints between the two arms could be expressed through a *collective variable*), we were able to suggest a method to explicitly learn constraints from training data. Next, a dynamical system view of motion production allowed us to propose an algorithm that maintained the learned constraints in a changing environment.

However, it was important to note that, by choosing the collective variable to be the relative position between the two arms, we were not able to explain all the diversity of bimanual constraints that may arise in manipulation tasks. Therefore, we suggested that for some tasks coupling between the hands necessarily took a nonlinear form (the collective variable could be a nonlinear function of the handsŠ motions). Our model, hence, should be extended to befit such tasks.

Learning a Motion Representation for Coordinated Tasks

Across all the disciplines that we reviewed – analytical robotics (Section 2.1), human motor control (Section 2.3), and robot learning (Section 2.4) – the challenge of explaining and generating trajectories of coordinated motions was fundamental. Several factors caused the complexity of the problem: the redundancy in the ways that a task might be accomplished (hence, the prerequisite to assume a heuristic to pick up a single strategy), the multi-dimensionality of a motion space (the increased computational costs), and continuous coordination constraints (difficult to engineer by hand or learn from data). Finally, the coordination constraints were also task-dependent. In Section 2.4, we described how some of these issues had been tackled within robot learning. For instance, redundancy and learning constraints were partly resolved through imitation of a human teacher (Programming by Demonstration) or through optimization of a reward function (Reinforcement Learning). Yet, a number of issues were left unaddressed, to name a few: learning *generative* representations for coordinated motions, encoding multivariate correlations within a motion, and robustness of the learned motions to perturbations.

We investigated these problems in Section 3.3. Again, a dynamical system view of motion generation was adopted, however, instead of fitting a pre-defined dynamical model as was typically done in related works, we suggested a method to learn dynamical motion policies directly from several demonstrations. The dynamical system formulation allowed us to build generative motion policies: trajectories could be produced in parts of a robot's workspace unobserved during training. Furthermore, the generated trajectories satisfied a continuous coordination constraint defined by a learned dynamical system.

Our method was built upon an assumption that a motion could be represented by an *autonomous* dynamical system. Though we showed that this assumption was relevant for many manipulation tasks, for some tasks the time-independency did not hold; for instance, if a robot had to synchronize with external objects (e.g., consider a robot playing tennis). Therefore, we clarified that the performance of our method was not guaranteed in such tasks and the method should be extended with a time-keeping mechanism.

Learning Task Models for Physical Human-Robot Interaction

In this thesis, we argued that motion coordination during direct physical interaction between two peers added another level of complexity to robot control as compared to motion coordination during autonomous task execution. Several reasons could explain this complexity. We particularly pinpointed the following reasons: motion control required continuous adaptation to sensory (haptic) information, and, in addition to controlling its own motion, the robot needed to predict the behavior of the partner. In the review of human motor control in Section 2.2, we discussed that one of the advantages of dynamical systems for motion representation was the technical ease with which such such a representation incorporated the sensorimotor integration. In Sections 3.2, 3.3, and 3.4 of this thesis, we explored sensorimotor integration of visual information – we mapped a target position provided by a vision system into a dynamical motion representation (a target position is mapped to the attractor of the dynamical system).

In Chapter 4, we extended the learning method from Section 3.3 so as to be able to integrate *continuous* haptic information. Moreover, due to the generative abilities of the learned dynamical representation, the robot was able to predict the kinematic profile of its partner and, therefore, to synchronize with him, while conditioning its actions solely on the incoming haptic information.

From the learning point of view, we observed several difficulties related to the acquisition of training data for teaching physical Human-Robot Interaction. We explored two types of teaching modalities: teleoperation through a haptic device and direct demonstration by a more Ťknowledgable" partner (as we used in simulation). Teleoperation seemed to be a more natural way for teaching physical interaction, however, it required complex hardware and software. Also we noted that the workspace of haptic devices was limited and, hence, unsuitable for the demonstration of large motions. Direct demonstration worked well for symmetric tasks but did not not support tasks where the robot should have behaved differently from its partner. Therefore, we stressed that the demonstration of physical interactions proved to be an important problem from both a technical and algorithmic point of view.

5.2 DIRECTIONS FOR FUTURE WORK

We can identify the following promising research directions that stem from the work conducted in this thesis.

Combining Continuous and Discrete Task Representations

The method for learning bimanual coordination that we present in Chapter 3.2 encoded a task at the discrete level. At the continuous level, the trajectories are generated through a pre-defined dynamical system. Though the use of predefined dynamics has been proven to be efficient for some tasks, extending the method, to also learn the task trajectories, would improve the method performance in more complex movements. One way to do so is to combine the task level learning with learning motion dynamics as presented later in Chapter 3.3. To address the problem of linking the task level learning of the bimanual constraints and the trajectory level learning, one might use Hidden Markov Models (HMMs) of a special structure that encapsulates both levels of abstraction. Namely, coupled HHMs are specifically designed to model several interacting processes that operate at different time scales. This method, therefore, is relevant for the integrated learning of high-level and low-level features.

Developing a Reinforcement Learning Approach for Training Dynamical System Motion Representations

In the current version of our algorithm (as described in Chapter 3.3), dynamical motion representations are learned from demonstrations. However, it often appears that self-practice is essential, and, therefore, the refinement of a dynamical system representation through reinforcement learning is desirable. As we review in Section 2.3.3, one recent trend in reinforcement learning is probabilistic policy search. The methods of this group support learning structured policies (e.g., our dynamical system motion representation can be seen as a structured policy: a parameterized mixture of dynamics). However, for now, to the best of our knowledge, none of the existing reinforcement learning algorithms permits learning of time-independent policies.

We believe that further investigations on how one can use Reinforcement Learning to train policies represented as autonomous dynamical systems is a necessary step to further improve robot coordination skills. To support our statement, we emphasize that, throughout this thesis, the dynamical system motion representation has been shown to deliver several advantages. Time-independent policies and particularly indispensable to control robot motion during interaction with the human, as, in this case, a robot is relieved from a necessity to continuously reestimate motion duration.

Learning to combine interaction and task constraints: moving a dynamically changing load

In the experiments on collaborative manipulation presented in this thesis, a robot and a human are moving a single solid object with fixed dynamical properties (e.g., the center of mass, the moment of inertia). When the dynamical parameters are fixed, the robot concentrates solely on learning and reproduction of interaction constraints, e.g. synchronization with the human. However, one often seeks partnerŠs help to move a table or tray; often in such cases, there are other objects piled on top the table of the tray. Consider for instance a scenario, where the human and the robot should move a plate with a ball rolling on top. The ball rolls on the plate (or the piled objects slip on the table) and can fall down eventually if the partners do not respect the task constraints. We might envisage two ways to learn such tasks: (1) directly encode data observed while the constraints were varying or (2) learn a task under static constraints and then impose the constraints on rolling the objects analytically.

Chapter 6

APPENDICES

Table 6.1: **Appendix I**: Derivation of constraint-consistent velocities for bimanual coordination

	We start by defining a metric of imitation (a functional that measures how accurately a robot reproduces a learned behavior)
(A-I-1)	$\mathbb{H}(\mathbf{q}^R, \mathbf{q}^L, \mathbf{x}^R, \mathbf{x}^L) = (\mathbf{q}^R - \mathbf{q}_d^R)^T W_{\mathbf{q}}^R (\mathbf{q}^R - \mathbf{q}_d^R) + (\mathbf{x}^R - \mathbf{x}_d^R)^T W_{\mathbf{x}}^R (\mathbf{x}^R - \mathbf{x}_d^R) + (\mathbf{q}^L - \mathbf{q}_d^L)^T W_{\mathbf{q}}^L (\mathbf{q}^L - \mathbf{q}_d^L) + (\mathbf{x}^L - \mathbf{x}_d^L)^T W_{\mathbf{x}}^L (\mathbf{x}^L - \mathbf{x}_d^L),$
	The constraint optimization of the metric of imitation $\mathbb{H}$ under spatial coordination and robot-body constraints:
(A-I-2)	$\min \mathbb{H}_{\mathbf{q}^R, \mathbf{q}^L}$
	u.c.
(A-I-3)	$\dot{\mathbf{x}}^R - J^R \dot{\mathbf{q}}^R = 0$
(A-I-4)	$\dot{\mathbf{x}}^L - J^L \dot{\mathbf{q}}^L = 0$
(A-I-5)	$\dot{\mathbf{x}}^R - \dot{\mathbf{x}}^L = 0$
	Let us redefine error terms from Eq. (A-I-1), $(\mathbf{q}^R - \mathbf{q}_d^R)$, $(\mathbf{x}_t^R - \mathbf{x}_{d,t}^R)$, as follows (note that here we use time indices to express errors at time t through the velocity at time t and the robot's position at time $t-1$):
(A-I-6)	$\mathbf{q}_t^R - \mathbf{q}_d^R = \dot{\mathbf{q}}^R + \mathbf{q}_{(t-1)}^R - \mathbf{q}_t^{Rd} = \dot{\mathbf{q}}_t^R - c_1;$
(A-I-7)	$\mathbf{q}_t^L - \mathbf{q}_{d,t}^L = \dot{\mathbf{q}}_t^L + \mathbf{q}_{(t-1)}^L - \mathbf{q}_{d,t}^L = \dot{\mathbf{q}}_t^L - c_2;$
(A-I-8)	$\mathbf{x}_t^R - \mathbf{x}_{d,t}^R = \dot{\mathbf{x}}_t^R + \mathbf{x}_{(t-1)}^R - \mathbf{x}_{d,t}^R = \dot{\mathbf{x}}_t^R - c_3;$
(A-I-9)	$\mathbf{x}_t^L - \mathbf{x}_{d,t}^L = \dot{\mathbf{x}}_t^L + \mathbf{x}_{(t-1)}^L - \mathbf{x}_{d,t}^L = \dot{\mathbf{x}}_t^L - c_4.$
	For the constrained optimization of Eq.(A-I-1)-(A-I-5), we define the Lagrangian $\mathbb{L}$:
(A-I-10)	$\mathbb{L}(\dot{\mathbf{q}}^R, \dot{\mathbf{q}}^L, \dot{\mathbf{x}}^R, \dot{\mathbf{x}}^L, \lambda_1, \lambda_2, \lambda_3) = \mathbb{H} + \lambda_1^T(\dot{\mathbf{x}}^R - J^R \dot{\mathbf{q}}^R) + \lambda_2^T(\dot{\mathbf{x}}^L - J^L \dot{\mathbf{q}}^L) + \lambda_3^T(\dot{\mathbf{x}}^R - \dot{\mathbf{x}}^L).$

Table 6.2: **Appendix I**: Derivation of the constraint-consistent velocities for bimanual coordination (continuation)

	To find an analytical solution of the optimization problem Eq.(A-I-1)-(A-I-5), we differentiate the Lagrangian $\mathbb{L}$ with respect to all variables:
(A-I-11)	$\frac{\partial \mathbb{L}}{\partial \dot{\mathbf{x}}^R} = 2W_{\mathbf{x}}^R(\dot{\mathbf{x}}^R - c_3) + \lambda_1 + \lambda_3 = 0;$
(A-I-12)	$\frac{\partial \mathbb{L}}{\partial \dot{\mathbf{q}}^R} = 2W_{\mathbf{q}}^R(\dot{\mathbf{q}}^R - c_1) - (J^R)^T \lambda_1 = 0;$
(A-I-13)	$\frac{\partial L}{\partial \dot{\mathbf{x}}^L} = 2W_{\mathbf{x}}^L(\dot{\mathbf{x}}^L - c_4) + \lambda_2 - \lambda 3 = 0;$
(A-I-14)	$\frac{\partial \mathbb{L}}{\partial \dot{\mathbf{q}}^L} = 2W_{\mathbf{q}}^L(\dot{\mathbf{q}}^L - c_2) - (J^L)^T \lambda 2 = 0;$
(A-I-15)	$\frac{\partial \mathbb{L}}{\partial \lambda 1} = \dot{\mathbf{x}}^R - J^R \dot{\mathbf{q}}^R = 0;$
(A-I-16)	$\frac{\partial \mathbb{L}}{\partial \lambda_2} = \dot{\mathbf{x}}^L - J^L \dot{\mathbf{q}}^L = 0;$
(A-I-17)	$\frac{\partial \mathbb{L}}{\partial \lambda_3} = \dot{\mathbf{x}}^R - \dot{\mathbf{x}}^L = 0$
	Substituting Eq.(A-I-11), (A-I-13)-(A-I-17) into Eq.(A-I-12):
(A-I-18)	$W_{\mathbf{x}}^R(J^R \dot{\mathbf{q}}_t^R - c_2) + (J^R)^{-T} W_{\mathbf{q}}^R(\dot{\mathbf{q}}_t^R - c_1) + W_{\mathbf{x}}^L(J^R \dot{\mathbf{q}}_t^R - c_4) + (J^L)^{-T} W_{\mathbf{q}}^L((J^L)^{-1} J^R \dot{\mathbf{q}}_t^R - c_3) = 0;$
	After arranging in the left part the terms that contain $\dot{\mathbf{q}}^R$, we obtain:
	$\dot{\mathbf{q}}^R = (M_1)^{-1} M_2;$ $\dot{\mathbf{q}}^L = [(J^L)^{-1} J^R] \dot{\mathbf{q}}^R;$ $M_1 = W_{\mathbf{x}}^R J^R + (J^R)^{-T} W_{\mathbf{q}}^R + W_{\mathbf{x}}^L J^R + (J^L)^{-T} W_{\mathbf{q}}^L (J^L)^{-1} J^R$ $M_2 = W_{\mathbf{x}}^R c_3 + (J^R)^{-T} W_{\mathbf{q}}^R c_1 + W_{\mathbf{x}}^L c_4 + (J^L)^{-T} W_{\mathbf{q}}^L c_2$

Table 6.3: **Appendix II**: Comparison of the proposed method with Dynamical Movement Primitives, as defined in Hoffmann, Pastor, et al. (2009)

The method proposed in this paper:

a single multidimensional system is running to control several DOFs

(A-II-1) $$\dot{\mathbf{x}} = \hat{f}(\mathbf{x})$$

(A-II-2) $$\hat{f}(\mathbf{x}) = \sum_{k=1}^{K} h_k(\mathbf{x})(\mu_{k,\dot{\mathbf{x}}} + \Sigma_{k,\dot{\mathbf{x}}\mathbf{x}}\Sigma_{k,\mathbf{x}}^{-1}(\mathbf{x} - \mu_{k,\mathbf{x}}))$$

where $\mathbf{x} \in \mathbb{R}^N$; $\Sigma_{k,\dot{\mathbf{x}}\mathbf{x}},\ \Sigma_{k,\mathbf{x}} \in \mathbb{R}^{N \times N}$ are estimated matrices
$\mu_{k,\dot{\mathbf{x}}},\ \mu_{k,\mathbf{x}} \in \mathbb{R}^N$ are estimated vectors

---

Dynamic Movement Primitives:

the acceleration along each DOF $\dot{x}$ is defined by according to:

(A-II-3) $$\tau\ddot{x} = -K_v\dot{x} + K_p(g - x) - K_p(g - x_0)s + K_p\hat{f}(s)$$

(A-II-4) $$\hat{f}(s) = \frac{\sum_{k=1}^{K} s\Psi_k(s)\omega_k}{\sum_{k=1}^{K} \Psi_k(s)}$$

where $x,\ s, \in \mathbb{R}$
$\Psi_k(s) = \text{exp}\frac{(s-c_k)^2}{2\sigma_k^2}, \omega_k \in \mathbb{R}$.

The canonical variable s is governed by a dynamical system:

(A-II-5) $$\tau\dot{s} = -\alpha_s s, \quad s \in [0..1];\ s(0) = 1$$

where $g,\ \alpha_s,\ K_p,\ K_v \in \mathbb{R}$ are the known proportionate and derivative coefficients

---

Comparison between $\hat{f}(\mathbf{x})$ from our approach and $\hat{f}(s)$ from DMP:

The function $f(\mathbf{x})$ aims at encoding an actual dependency between position and velocity of along a motion; it therefore simultaneously provides the feedback loop for motion adaptation. On the other hand, the function $\hat{f}(s)$ encodes a particular acceleration profile as a function of the internal counter s. The variable s is not linked with positional information and, hence, $\hat{f}(s)$ does not provide the necessary feedback to adapt the motion. Due to such a choice of learning variables, DMP perform the scaling of a demonstrated trajectory, but do not provide the actual robust adaptation to perturbations that can be generated by our system.

Table 6.4: **Appendix III**: Bio-mimetic adaptive impedance (Ganesh, Albu-Schaffer, et al., 2010)

We summarize the major steps of the bio-mimetic adaptive algorithm presented in Ganesh, Albu-Schaffer, et al. (2010). Let us consider the following cost function that penalizes a feedback cost $(v^i)^2$ and a cost of activation of the feedforward command $\sum_{k=1}^{K} \theta_k^i$:

(A-III-1) $\min_{\boldsymbol{\theta}^i} \; R^i(\boldsymbol{\theta}^i) = 0.5\beta \; (v^i)^2 + \gamma \sum_{k=1}^{K} \theta_k^i, \; \forall \, i = 1 \cdots N_q.$

where $\beta > 0, \gamma > 0$ are empirical constants controlling the influence of the feedback and feedforward components. To derive an adaptation policy, Ganesh, Albu-Schaffer, et al. (2010) suggest to use a special form of the feedback signal v^i:

(A-III-2) $v^i = 0.5[(1-\chi)\epsilon^i + (1+\chi)|\epsilon^i|], \;\; \epsilon^i = \rho_1 e^i + \rho_2 \dot{e}^i.$

Here, e^i is the deviation of the controlled signal from its desired value, $\chi, \rho_1, \rho_2 > 0$ are empirical constants. To compensate for the feedback error, one can adjust the parameters of the feedforward control $\boldsymbol{\Theta}$ in Eq.4.10 by optimizing the cost function R_i using the gradient descent:

(A-III-3) $\Delta\boldsymbol{\theta}_t^i = -\frac{dR^i}{d\boldsymbol{\theta}^i} = -\beta(\frac{\partial v^i}{\partial \boldsymbol{\theta}_k^i})^T v^i - \gamma \boldsymbol{1},$

The control $\boldsymbol{\tau}$, feedforward $\boldsymbol{u}$, and feedback signal $\boldsymbol{v}$ are linked: $\boldsymbol{\tau} = \boldsymbol{u} + \boldsymbol{v}$; see Eq.4.8. $\boldsymbol{\tau}$ represents the environment being learned and is assumed to be independent of $\boldsymbol{\Theta}$, therefore the adaptation law in Eq.(A-III-3) can be rewritten as:

(A-III-4) $\Delta\boldsymbol{\theta}^i = \beta(\frac{\partial u^i}{\partial \boldsymbol{\theta}_k^i})^T v^i - \gamma \boldsymbol{1} = \beta \boldsymbol{\Phi} v^i - \gamma \boldsymbol{1}.$

REFERENCES

Abbeel, P., & Ng, A. (2004). Apprenticeship learning via inverse reinforcement learning. In *Proceedings of International Conference on Machine Learning.*

Admiraal, M., & Kusters, M. (2004). Modeling kinematics and dynamics of human arm movements. *Motor Control*, *5*, 312Ű-338.

Aguirre, L., & Billings, S. (1995). Retrieving dynamical invariants from chaotic data using NARMAX models. *International Journal of Bifurcation and Chaos*, *5*, 449-474.

Albus, J. (1975). A new approach to manipulator control: The cerebellar model articulation controller CMAC. *Transactions of the ASME, Journal of Dynamic Systems Measurement and Control*, 220-227.

Aleotti, J., & Caselli, S. (2006). Robust trajectory learning and approximation for robot programming by demonstration. *Robotics and Autonomous Systems*, *54(5)*, 409Ű413.

Aleotti, J., Caselli, S., & Reggiani, M. (2005). Evaluation of virtual fixtures for a robot programming by demonstration interface. *Transactions on Systems, Man, and Cybernetics*, *35(4)*, 536-545.

Altmann, S. L. (1986). *Rotations, quaternions, and double groups*. Claredon Press, Oxford.

Amari, S. (1998). Natural gradient works efficiently in learning. *Neural Computation*, *10*, 251Ű276.

An, C., Atkeson, C., & Hollerbach, J. (1988). *Model-based control of a robot manipulator*. MIT Press.

Anderson, B., & Moore, J. (1979). *Optimal filtering*. Prentice-Hall.

Andersson, L., & Rantzer, A. (1999). Robustness of equilibria in nonlinear systems. In *Proceedings of world congress of IFAC.*

Andersson, R. (1989, Feb). Aggressive trajectory generator for a robot ping-pong player. *IEEE Control Systems Magazine*, *9*(2), 15-21.

Arai, H., Takubo, T., Hayashibara, Y., & Tanie, K. (2000). Human-robot cooperative manipulation using a virtual nonholonomic constraint. In *Proceedings of IEEE International Conference on Robotics and Automation.*

Argall, B., Chernova, S., Veloso, M., & Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, *57*, 469-483.

Arimoto, A. (1993). *Learning control* (C. Abdallah, Ed.). IEEE Press.

Arulampalam, S., Maskell, S., Gordon, N., & Clapp, T. (2002). A tutorial on particle filters for on-line nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 174–88.

Astrom, K., & Wittenmark, B. (1989). *Adaptive control* (K. Astrom & B. Wittenmark, Eds.). Addison Wesley.

Astrom, K., & Wittenmark, B. (2008). *Adaptive control* (K. Astrom & B. Wittenmark, Eds.). Dover.

Atkeson, C. (1992). Nonlinear modeling and forecasting. In M. Casdagli & S. Eubank (Eds.), (chap. Memory-based approaches to approximating continuous functions).

Atkeson, C., Moore, A., & Schaal, S. (1997). Locally weighted learning. *Artificial Intelligence Review*, *11*, 11-73.

Attias, H. (1999). Inferring parameters and structure of latent variable models by variational bayes. *Uncertainty in Artificial Intelligence*.

Aylward, E., Parillo, A., & Slotine, J. (2008). Stability and robustness analysis of nonlinear systems via contraction metrics and SoS programming. *Automatica*, *44*, 2163-2170.

Baake, E., Baake, M., Bock, H., & Briggs, K. (1992). Fitting ordinary differential equations to chaotic data. *Physical Review A*, *45*, 5524-5529.

Baginski, B. (1998). *Motion planning for manipulators with many degrees of freedom – the BB method*. Unpublished doctoral dissertation, Technical University of Munich.

Bagnell, J., Kadade, S., Ng, A., & Schneider, J. (2003). Policy search by dynamic programming. In *Advances in Neural Information Processing Systems.*

Barraquand, J., & Latombe, J.-C. (1991). Robot motion planning: A distributed representation approach. *International Journal of Robotic Research (IJRR)*, *10(6)*, 628-649.

Bar-Shalom, Y., Li, X., & Kirubarajan, T. (2001). *Estimation with applications to tracking and navigation*. Wiley, NY.

Berg, J. van den, Miller, S., Duckworth, D., Hu, H., Fu, X., Goldberg, K., et al. (2010). Superhuman performance of surgical tasks by robots using iterative learning from human-guided demonstrations. In *Proceedings of IEEE International Conference on Robotics and Automation.*

Bernstein, N. (1967). *The coordination and regulation of movements*. Pergamon Press.

Berret, B., Darlot, C., Jean, F., Pozzo, T., Papaxanthis, C., & Gauthier, J.-P. (2008). The inactivation principle: Mathematical solutions minimizing the absolute work and biological implications for the planning of arm movements. *Journal of Computational Biology*, *4(10)*.

Bezdek, J., & Pal, N. (1998). Some new indexes of cluster validity. *IEEE Transaction on System, Man, and Cybernetics*, *28*, 301-315.

Billard, A., Calinon, S., Dillman, R., & Schaal, S. (2008). Handbook of robotics. In B. Siciliano & O. Khatib (Eds.), (chap. Robot programming by demonstration). Springer New York.

Billard, A., Calinon, S., & Guenter, F. (2006). Discriminative and adaptive imitation in uni-manual and bi-manual tasks. *Robotics and Autonomous Systems*, *54*, 370-384.

Bitzer, S., & Vijayakumar, S. (2009). Latent spaces for dynamic movement primitives. In *Proceedings of IEEE international conference on humanoid robotics.*

Boggs, P., Byrd, R., & Schnabel, R. (1987). A stable and efficient algorithm for nonlinear orthogonal distance regression. *SIAM Journal of Scientific Statistical Computation*, *8*, 1052-1078.

Boyd, S., Ghaoui, L., Feron, E., & V., B. (1994). Linear Matrix Inequalities in system and control theory. *SIAM Studies in Applied Mathematics.*, *15*.

Brenner, E., & Smeets, J. (1995). Moving one's finger to a visually specified position: Target orientation influences the finger's path. *Experimental Brain Research*, *105*, 318-320.

Brock, O., & Khatib, O. (1997). Elastic Strips: Real-time path modification for mobile manipulation. In *Proceedings of International Symposium of Robotic Research.*

Brock, O., & Khatib, O. (2002). Elastic Strips: A framework for motion generation in human environments. *International Journal of Robotic Research (IJRR)*, *21(12)*.

Brock, O., Kuffner, J., & Xiao, J. (2008). Handbook of robotics: Motion for manipulation tasks. Springer-Verlag.

Buchli, J., Theodorou, E., Stulp, F., & Schaal, S. (2010). Variable impedance control - a reinforcement learning approach. In *Proceedings of Robotics Science and Systems.*

Buerger, S., & Hogan, N. (2007). Complementary stability and loop shaping for improved human-robot interaction. *IEEE Transactions on Robotics*, *23*, 232-244.

Buhmann, M. D. (2003). *Radial basis functions: Theory and implementations*. Cambridge University Press.

Bullock, D., & Grossberg, S. (1988). Neural dynamics of planned arm movements: Emergent invariants and speed-accuracy properties during trajectory formation. *Psychological Review*, *95*, 49-90.

Burdet, E., & Codourey, A. (1998). Evaluation of parametric and nonparametric nonlinear adaptive controllers. *Robotica*, *16*, 59-73.

Burdet, E., Codourey, A., & Rey, L. (1998). Experimental evaluation of nonlinear adaptive controllers. *IEEE Control System Magazine*, *18*, 39-47.

Burridge, R., Rizzi, A., & Koditschek, D. (1999). Sequential composition of dynamically dexterous robot behaviors. *International Journal of Robotic Research (IJRR)*, *vol. 18(6)*, 534-555.

Buss, M., & Hashimoto, H. (1994). Manipulation skill modeling for dexterous hands. In *Proceedings of IEEE International Conference on Robotics and Automation.*

Calinon, S. (2009). *Robot programming by demonstration: A probabilistic approach* (S. Calinon, Ed.). EPFL/CRC Press.

Calinon, S., & Billard, A. (2007a). Active Teaching in Robot Programming by Demonstration. In *Proceedings of the IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)* (pp. 702–707).

Calinon, S., & Billard, A. (2007b). Incremental learning of gestures by imitation in a humanoid robot. In *Proceedings of the ACM/IEEE International Conference on Human-Robot interaction.*

Calinon, S., & Billard, A. (2007c). What is the teacher's role in robot programming by demonstration? - Toward benchmarks for improved learning. *Interaction Studies. Special Issue on Psychological Benchmarks in Human-Robot Interaction*, *8(3)*, 441-464.

Calinon, S., & Billard, A. (2009). Statistical learning by imitation of competing constraints in joint space and task space. *Advanced Robotics*, *23*, 2059–2076.

Calinon, S., D'halluin, F., Caldwell, D., & Billard, A. (2009). Handling of multiple constraints and motion alternatives in a robot programming by demonstration framework. In *Proc. IEEE International Conference on Humanoid Robots* (pp. 582–588).

Calinon, S., D'halluin, F., Sauser, E. L., & Caldwell, A., D.and Billard. (2010). Learning and reproduction of gestures by imitation: An approach based on hidden Markov model and Gaussian mixture regression. *IEEE Robotics and Automation Magazine*, *17*(2), 44–54.

Calinon, S., Evrard, P., Gribovskaya, E., Billard, A., & Kheddar, A. (2009). Learning collaborative manipulation tasks by demonstration using a haptic interface. In *Proceedings of the International Conference on Advanced Robotics.*

Calinon, S., Guenter, F., & Billard, A. (2007). On learning, representing and generalizing a task in a humanoid robot. *IEEE Transactions on Systems, Man and Cybernetics, Part B. Special issue on robot learning by observation, demonstration and imitation*, *37*(2), 286–298.

Calinon, S., Sardellitti, I., & Caldwell, D. G. (2010). Learning-based control strategy for safe human-robot interaction exploiting task and robot redundancies. In *Proceedings IEEE International Conference on Intelligent Robots and Systems.*

Canny, J., Rege, A., & Reif, J. (1991). An exact algorithm for kinodynamic planning in the plane. *Discrete and Computational Geometry*, *6*, 461Ű484.

Cheng, P., Shen, Z., & LaValle, S. (2001). RRT-based trajectory design for autonomous automobiles and spacecraft. *Archives of Control Sciences*, *11(3-4)*, 167Ű194.

Chow, C., & Jacobson, D. (1971). Studies of human locomotion via optimal programming. *Mathematical Bioscience*, *10*, 239-306.

Christel, M., & Billard, A. (2001). How monkeys morphology constrain natural prehension kinematics. unconstrained conditions and simulation. *Behavioural Brain Research*, *131*, 169-184.

Coates, A., Abbeel, P., & Ng, A. (2008). Learning for control from multiple demonstrations. In *Proceedings of International Conference on Machine Learning.*

Colgate, J., & Hogan, N. (1988). Robust control of dynamically interacting systems. *International Journal of Control*, *48*, 65-88.

Comer, D., Rizzi, A., & Choset, H. (2003). Composition of local potential functions for global robot control and navigation. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems.*

Corteville, B., Aertbelien, E., Bruyninckx, B., De Schutter, J., & Van Brussel, H. (2007). Human-inspired robot assistant for fast point-to-point movements. In *Proceedings of ieee international conference on robotics and automation.*

Cremers, J., & Huebler, A. (1987). Construction of differential equations from experimental data. *Naturforsch*, *42a*, 797-802.

Crutchfield, J., & McNamara, B. (1987). Equation of motion from data series. *Complex Systems*, *1*, 417–451.

d'Avella, A., Saltiel, P., & Bizzi, E. (2003). Combinations of muscle synergies in the construction of a natural motor behavior. *Nature Neuroscience*, *6*, 300Ű-308.

d'Avella, A., & Tresch, M. (2002). Modularity in the motor system: decomposition of muscle patterns as combinations of time-varying synergies. In *Advances in Neural Information Processing.*

Dempster, A., Laird, N., & Rubin, D. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of Royal Statistic Society*, *39*, 1–38.

Diankov, R., Kanade, T., & Kuffner, J. (2009). Integrating grasp planning and visual feedback for reliable manipulation. In *Proceedings of IEEE International Conference on Humanoid Robots.*

Dixon, K., & Khosla, P. (2004). Trajectory representation using sequenced linear dynamical systems. *Proceedings of IEEE International Conference on Robotics and Automation*, *4*, 3925–3930.

Domselaar, B. van, & Hemker, P. (1975). *Nonlinear parameter estimation in initial value problems* (Tech. Rep.). Mathematical Centre Amsterdam.

Duchaine, V., & Gosselin, C. (2007). General model of human-robot cooperation using a novel velocity based variable impedance control. In *Proceedings of the joint eurohaptics conference and symposium.*

Ekvall, S., & Kragic, D. (2007). Learning and evaluation of the approach vector for automatic grasp generation and planning. In *Proceeding of the IEEE International Conference on Robotics and Automation.*

Evrard, P., Gribovskaya, E., Calinon, S., Billard, A., & Kheddar, A. (2009). Teaching Physical Collaborative Tasks: Object-Lifting Case Study with a Humanoid. In *Proceedings of IEEE International Conference on Humanoid Robots.*

Farmer, J., & Sidorovich, J. (1988). *Predicting chaotic dynamics* (J. Kelso, Ed.). World Scientific.

Faverjon, B., & Tournassoud, P. (1987). A local based approach for path planning of manipulators with a high number of degrees of freedom. In *Proceedings of IEEE International Conference on Robotics and Automation.*

Flash, T., & Hogan, N. (1985). The coordination of arm movements: an experimentally confirmed mathematical model. *Journal of Neuroscience*, *5*, 1688-1703.

Franklin, D., Burdet, E., Tee, K., Osu, R., Chew, C.-M., Milner, T., et al. (2008). CNS learns Stable, Accurate, and Efficient Movements Using a Simple Algorithm. *Journal of Neuroscience*, *28(44)*, 11165–11173.

Frazzoli, E., Dahleh, M., & Feron, E. (2002). Real-time motion planning for agile autonomous vehicles. *Journal of Guidance, Control, and Dynamics*, *25(1)*, 116-129.

Ganesh, G., Albu-Schaffer, A., Haruno, M., Kawato, M., & Burdet, E. (2010). Biomimetic motor behavior for simultaneous adaptation of force, impedance and trajectory in interaction tasks. In *Proceedings of IEEE International Conference on Robotics and Automation.*

Ganesh, G., Haruno, M., Kawato, M., & Burdet, E. (2010). Motor memory and local minimization of error and effort, not global optimization, determine motor behavior. *Journal of Neurophysiology*, *104*, 382-390.

Gentry, S. (2005). *Dancing cheek to cheek: Haptic communication between partner dancers and swing as a finite state machine.* Unpublished doctoral dissertation, Massachussetts Institute of Technology.

Ghahramani, Z., & Roweis, S. (1999). Learning nonlinear dynamical systems using an EM algorithm. In *Advances in Neural Information Processing Systems.*

Giszter, S. F., Mussa-Ivaldi, F. A., & Bizzi, E. (1993). Convergent force fields organized in the frogŠs spinal cord. *The Journal of Neuroscience*, *13(2)*, 467-Ű491.

Glassman, E., & Tedrake, R. (2010). A Quadratic Regulator-based heuristic for rapidly exploring state-space. In *Proceedings of IEEE International Conference on Robotics and Automation.*

Gonzalez-Serranoa, F., Figueiras-Vidalb, A., & Artes-Rodriguez, A. (1998). Fourier analysis of the generalized CMAC neural network. *Neural Networks*, *11*, 391-396.

Guckenheimer, J., & Holmes, P. (1993). *Nonlinear oscillations, dynamical systems, and bifurcations of vector fields*. Springer, New York.

Guenter, F., Hersch, M., Calinon, S., & Billard, A. (2007). Reinforcement Learning for Imitating Constrained Reaching Movements. *RSJ Advanced Robotics, Special Issue on Imitative Robots*, *21*(13), 1521–1544.

Haggard, P., & A., W. (1997). On the hand transport component of prehensile movements. *Journal of Motor Behavior*, *29*, 282-287.

Haken, H. (1993). *Advanced synergetics: Instability hierarchies of self-organizing systems and devices*. Springer, New York.

Haken, H., Kelso, J., & Bunz, H. (1985). A theoretical model of phase transitions in human hand movements. *Biological Cybernetics*, *51*, 347-356.

Harris, C., & Wolpert, D. (1998). Signal-dependent noise determines motor planning. *Nature*, *394*, 780Ű-784.

Hartmann, J. (2005). *Counter-intuitive behavior in locally optimal solar sail escape trajectories.* Unpublished doctoral dissertation, University of Illinois.

Hegger, R. (1998). Dynamical properties of a ferroelectric capacitor observed through nonlinear time series analysis. *Chaos*, *8*, 727-736.

Hegger, R., & Kantz, H. (1999). Improved false nearest neighbor method to detect determinism in time series. *Physical Review E*, *60*, 4970-4973.

Hersch, M., & Billard, A. (2008). Reaching with multi-referential dynamical systems. *Autonomous Robots*, *25*, 71-83.

Hersch, M., Guenter, F., Calinon, S., & Billard, A. (2008). Dynamical system modulation for robot learning via kinesthetic demonstrations. *IEEE Transactions on Robotics*, *24*, 1463-1467.

Hill, D., & Moylan, P. (1976). The stability of nonlinear dissipative systems. *IEEE Transaction on Automatic Control*, *21*, 708 - 711.

Hinrichsen, D., & Pritchard, A. (2000). *Mathematical systems theory*. Springer Berlin.

Hirata, Y., & Kosuge, K. (2000). Distributed robot helpers handling a single object in cooperation with a human. In *Proceedings of IEEE international conference on robotics and automation.*

Hirata, Y., Takagi, T., Kosuge, K., Asama, H., Kaetsu, H., & Kawabata, K. (2001). Manipulation of a large object by multiple DR helpers in cooperation with a human. In *Proceedings of IEEE international conference on intelligent systems and robotics.*

Hoff, B., & Arbib, M. (1993). Models of trajectory formation and temporal interaction of reach and grasp. *Journal of Motor Behavior*, *25*, 175-192.

Hoffmann, H., Pastor, P., Park, D.-H., & Schaal, S. (2009). Biologically-inspired dynamical systems for movement generation: automatic real-time goal adaptation and obstacle avoidance. In *Proceedings of IEEE International Conference on Robotics and Automation.*

Hoffmann, H., Schaal, S., & Vijayakumar, S. (2009). Local dimensionality reduction for nonparametric regression. *Neural Processing Letters*, *29*, 109-131.

Hogan, N. (1985). Impedance control: An approach to manipulation, part I-III. *ASME Journal of Dynamic Systems, Measurements, and Control*, *107*, 1-24.

Hopfield, J. J. (1984). Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences*, *81*, 3088-3092.

Horbelt, W., Muller, T., Timmer, J., Melzer, W., & Winkler, K. (2000). Analysis of nonlinear differential equations: parameter estimation and model selection. medical data analysis, Lecture Notes in Mathematics. In R. Brause & E. Hanisch (Eds.), (p. 152-158). Springer, Berlin.

Howard, M., Klanke, S., Gienger, M., Goerick, C., & Vijayakumar, S. (2008a). Behaviour generation in humanoids by learning potential-based policies from constrained motion. *Applied Bionics and Biomechanics*, *5*, 195-211.

Howard, M., Klanke, S., Gienger, M., Goerick, C., & Vijayakumar, S. (2008b). Learning potential-based policies from constrained motion. In *Proceedings of IEEE international conference on humanoid robots.*

Howard, M., Klanke, S., Gienger, M., Goerick, C., & Vijayakumar, S. (2009a). A novel method for learning policies from variable constraint data. *Autonomous Robots*, *27*, 105-121.

Howard, M., Klanke, S., Gienger, M., Goerick, C., & Vijayakumar, S. (2009b). Robust constraint consistent learning. In *Proceedings of IEEE international conference on intelligent robotics and systems.*

Howard, M., Klanke, S., Gienger, M., Goerick, C., & Vijayakumar, S. (2010). From motor learning to interaction learning in robots. In O. Sigaud & J. Peters (Eds.), (chap. Methods for learning control policies from variable-constraint demonstrations). Springer New York.

Hsu, P., Hauser, J., & Sastry, S. (1989). Dynamic control of redundant manipulators. *Journal of Robotic Systems*, *6(2)*, 133Ű148.

Hubert, L., & Arabie, P. (1985). Comparing partitions. *Journal of Classification*, *2*, 193-218.

Hunt, K., Sbarbaroa, D., Bikowskia, R., & Gawthropa, P. (1992). Neural networks for control systems-a survey. *Automatica*, *28*, 1083-1112.

Hwang, J., Arkin, R., & Kwon, D. (2003, Oct.). Mobile robots at your fingertip: Bezier curve on-line trajectory generation for supervisory control. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems* (Vol. 2, p. 1444-1449 vol.2).

Ijspeert, A., Nakanishi, J., & Schaal, S. (2001). Trajectory formation for imitation with nonlinear dynamical systems. In *Proceedings of IEEE International Conference on Intelligent Robots and System.*

Ijspeert, A., Nakanishi, J., & Schaal, S. (2002a). Learning rhythmic movements by demonstration using nonlinear oscillators. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems* (pp. 958–963).

Ijspeert, A., Nakanishi, J., & Schaal, S. (2002b). Learning rhythmic movements by demonstration using nonlinear oscillators. In *Proceedings of ieee international conference on intelligent robots and systems.*

Ijspeert, A., Nakanishi, J., & Schaal, S. (2003). Learning attractor landscapes for learning motor primitives. In *Advances in Neural Information Processing Systems.*

Ijspeert, A., Nakanishi, J., Shibata, T., & Schaal, S. (2001). Nonlinear dynamical systems for imitation with humanoid robots. In *Proceedings of IEEE International Conference on Humanoid Robotics.*

Ikemoto, S., Ben Amor, H., Minato, T., Ishiguro, H., & Jung, B. (2009). Physical interaction learning: Behavior adaptation in cooperative human-robot tasks involving physical contact. In *Proceedings of IEEE Internation Conference on Robot and Human Interactive Communication.*

Inamura, T., Nakamura, Y., & Toshima, I. (2004). Embodied symbol emergence based on mimesis theory. *International Journal of Robotics Research*, *23(4)*, 363–377.

Ivry, R., Spencer, R., Zelaznik, H., & Diedrichsen, J. (2002). The cerebellum and event timing. *Annals of New-York Academy of Science*, *978*, 302-317.

Jaeger, H., Luko, M., & Popovici, D. (2007). Optimization and applications of echo state networks with leaky integrator neurons. *Neural Networks*, *20(3)*, 335-352.

Jeannerod, M. (1981). Atention and performance. In J. Long & A. Baddeley (Eds.), (p. 153-169). Hillsdale NJ:Erlbaum.

Jirsa, V., & Kelso, J. (2005). The excitator as a minimal model for the coordination dynamics of discrete and rhythmic movement generation. *Journal of Motor Behavior*, *37(1)*, 35-51.

Julier, S., Uhlmann, J., & Durrant-Whyte, H. (2000). A new method for the nonlinear transformation of. *IEEE Transaction on Automatic Control*, *45*, 477-482.

Kakade, S. (2002). Natural policy gradient. In *Advances in Neural Information Processing Systems.*

Kalman, R. (1960). A new approach to linear filtering and prediction problems. *Transactions of ASME Journal of Basic Engineering Series*, *82*, 35-46.

Kalman, R., & Bucy, R. (1960). New results in linear filtering and prediction theory. *Transactions of ASME Journal of Basic Engineering Series*, *83*, 95-108.

Kang, T., He, J., & Tillery, S. (2005). Determining natural arm configuration along a reaching trajectory. *Journal of Experimental Brain Research*, 352-361.

Kavraki, L., Svestka, P., Latombe, J., & Overmars, M. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, *12(4)*, 566–580.

Kelso, J. (1995). *Dynamic patterns: The self-organization of brain and behavior*. MIT Press.

Kelso, J., Putnam, C., & Goodman, D. (1983). On the space-time structure of human interlimb co-ordination. *Quarterly Journal of Experimental Psychology Section A*, *35(2)*, 347-375.

Kelso, J. A. S. (1995). *Dynamic patterns: The self-organization of brain and behavior*. Cambridge: MIT Press.

Khatib, O. (1986). Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotic Research (IJRR)*, *5(1)*, 90-98.

Khatib, O. (1987). A unified approach to motion and force control of robot manipulators: The operational space formulation. *International Journal of Robotic Research (IJRR)*, *3(1)*, 43-53.

Khatib, O. (87). A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal of Robotics and Automation*, *3(1)*, 43Ű53.

Kim, S., Gribovskaya, E., & Billard, A. (2010). Learning motion dynamics to catch a moving object. In *Proceedings of IEEE international conference on humanoid robotics.*

Kitagawa, G. (1996). Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of Computer Graphics and Statistics*, *5*, 1-25.

Kober, J., & Peters, J. (2010). Policy search for motor primitives in robotics. *Maching Learning*, *OnlineFirst*.

Koditschek, D. (1987). Exact robot navigation by means of potential functions: Some topological considerations. In *Proceedings of International Conference on Robotics and Automation.*

Koeber, J., & Peters, J. (2008). Learning motor primitives in robotics. In *Advances in Neural Information Processing Systems.*

Koga, M., Kosuge, K., Furuta, K., & Nosaki, K. (1992). Coordinated motion control of robot arms based on the virtual internal model. In *Proceedings of IEEE international conference on robotics and automation.*

Koga, Y., Kondo, K., Kuffner, J., & Latombe, J.-C. (2004). Planning motions with intentions. In *Proceedings ACM SIGGRAPH.*

Konda, V., & Tsitsiklis, J. (2000). Actor-critic algorithms. In *Advances in Neural Information Processing Systems.*

Kormushev, P., Calinon, S., & Caldwell, D. (2010). Robot motor skill coordination with EM-based reinforcement learning. In *Proceedings of the IEEE International Conference on Intelligent Robots and Systems.*

Kormushev, P., Calinon, S., Saegusa, R., & Metta, G. (2010). Learning the skill of archery by a humanoid robot iCub. In *Proceedings of IEEE International Conference on Humanoid Robotics.*

Kostelich, E. (2001). Bootstrap estimates of chaotic dynamics. *Physical Review*.

Kosuge, K., & Kazamura, N. (1996). Control of a robot handling an object in cooperation with a human. In *Proceedings of ieee international symposium on robot and human communication.*

Kosuge, K., Yoshida, H., & Fukuda., T. (1993). Dynamic control for robot-human collaboration. In *Proceedings of IEEE International Symposium on Robot and Human Interactive Communication.*

Kuffner, J., Nishiwaki, K., Kagami, S., Inaba, M., & Inoue, H. (2001). Motion planning for humanoid robots under obstacle and dynamic balance constraints. In *Proceedings of IEEE International Conference on Robotics and Automation.*

Kulic, D., Takano, W., & Nakamura, Y. (2008). Incremental learning, clustering and hierarchy formation of whole body motion patterns using adaptive hidden markov chains. *International Journal of Robotics Research*, *27(7)*, 761-784.

Lancaster, P., & SalKauskas. (1986). *Curve and surface fitting*. Academic Press, New York.

Lane, S., Handehnan, D., & Gelfand, J. (1992). Theory and development of higher-order CMAC neural networks. *IEEE Control Systems Magazine*, 23-30.

La Salle, J., & Lefschetz, S. (1961). *Stability by Lyapunov direct method*. Academic Press, New York.

Latombe, J.-C. (1991). *Robot motion planning*. Kluwer Academic Publishers.

Laumond, J.-P., Sckhavat, S., & F., L. (1998). Guidelines in nonholonomic motion planning for mobile robots. In J.-P. Laumond (Ed.), (p. 1-53). Springer-Verlag.

LaValle, S. (2006). *Planning algorithms*. Cambridge University Press.

LaValle, S., & Kuffner, J. (2001). Randomized kinodynamic planning. *International Journal of Robotic Research (IJRR)*, *20(5)*, 378-400.

Lavalle, S. M. (1998). *Rapidly-exploring Random Trees: A new tool for path planning* (Tech. Rep.). Computer Science Department, Iowa State University.

Lawrence, N. (2005). Probabilistic nonlinear principal component analysis with gaussian process latent variable models. *Journal of Maching Learning Research*, *6*, 1783-1816.

Lee, D., Ott, C., & Nakamura, Y. (2009). Mimetic communication with impedance control for physical human-robot interaction. In *Proceedings of IEEE International Conference on Robotics and Automation.*

Lee, D., Ott, C., & Nakamura, Y. (2010). Mimetic communication model with compliant physical contact in humanŰhumanoid interaction. *International Journal of Robotics Research*, *OnlineFirst*.

Lee, E. (1986). Comments on some B-spline algorithms. *Computing (Springer-Verlag)*, *36*, 229-238.

Lee, S., & Kil, R. (1991). A Gaussian potential function network with hierarchically self-organizing learning. *Neural Networks*, 207-214.

Li, P., & Horowitz, R. (2001a). Passive velocity field control (pvfc). *IEEE Transactions on Automatic Control*, *46*, 1346-1359.

Li, P., & Horowitz, R. (2001b). Passive velocity field control (pvfc): Part ii - application to contour following. *IEEE Transactions on Automatic Control*, *46*, 1360-1371.

Li, W. (1990). *Adaptive control of robot motion*. Unpublished doctoral dissertation, Massachusetts Institute of Technology.

Lieberman, J., & Breazeal, C. (2004). Improvements on action parsing and action interpolation for learning through demonstration. In *Proceedings of IEEE International Conference on Humanoid Robots* (Vol. 1, p. 342- 365).

Loeb, G., & Levine, J., W.and He. (1990). Understanding sensorimotor feedback through optimal control. *Cold Spring Harbor Symposium Quantitative Biology*, *55*, 791-803.

Lohmiller, W., & Slotine, J.-J. (1998). On contraction analysis for nonlinear systems. *Automatica*, *34*, 671-682.

Lozano-Perez, T. (1983). Spatial planning: A configuration space approach. *IEEE Transactions on Computers*, *32(2)*, 108-120.

Lyapunov, A. (1992). The general problem of the stability of motion. *Translation of the original publication by A. T. Fuller, London: Taylor & Francis..*

MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of Berkeley Symposium on Mathematical Statistics and Probability.*

Madansky, A. (1959). The fitting of straight lines when both variables are subject to error. *Journal of American Statistics Association*, *54*, 173-205.

Maeda, Y., Hara, T., & Arai, T. (2001). Human-robot cooperative manipulation with motion estimation. In *Proceedings on international conference on intelligent robots and systems.*

Mason, M. (1985). The mechanics of manipulation. In *Proceedings of IEEE International Conference on Robotics and Automation.*

Masoud, A. (2010). Kinodynamic motion planning. *IEEE Robotics and Automation Magazine*, *17(1)*, 85 - 99.

Masoud, S., & Masoud, A. (2002). Motion planning in the presence of directional and obstacle avoidance constraints using nonlinear anisotropic, harmonic potential fields: A physical methaphor. *IEEE Transactions on System, Man, and Cybernetics*, *32(6)*, 705-723.

Mataric, M. (2000). Imitation in animals and artifacts. chapter: Sensory-motor primitives as a basis for imitation: Linking perception to action and biology to robotics. MIT Press. Available from `citeseer.ist.psu.edu/mataric00sensorymotor.html`

McLachlan, G., & Peel, D. (2000). *Finite mixture models* (G. McLachlan & D. Peel, Eds.). Wiley Series in Probability and Statistics.

McLean, A., & Cameron, S. (1996). The virtual springs method: Path planning and collision avoidance for redundant manipulators. *International Journal of Robotic Research (IJRR)*, *15(4)*, 300-319.

McSharry, P., & Smith, L. (1999). Better nonlinear models from noisy data: Attractors with maximum likelihood. *Physical Review Letters*, *83*, 4285-4288.

Michel, A., & Wang, K. (1993). Robust stability: perturbed systems with perturbed equilibria. *System and Control Letters*, *21(2)*, 155Ű162.

Miller, W., Glan, F., & Kraft, F. (1987). Application of a general learning algorithm to the control of robotic manipulator. *International Journal of Robotic Research*, *6*, 84-98.

Mitrovic, D., Klanke, S., & Vijayakumar, S. (2008). Adaptive optimal control for redundancy actuated arms. In *Proceedings of the international conference on simulation of adaptive behavior.*

Mitrovic, D., Klanke, S., & Vijayakumar, S. (2010). Learning impedance control of antagonistic systems based on stochastic optimization principles. *International Journal of Robotics Research*, *OnlineFirst*.

Molina-Tanco, L., & Hilton, A. (2000). Realistic synthesis of novel human movements from a database of motion capture examples. In *Proceedings IEEE Workshop on Human Motion.*

Nakamura, Y., Takano, W., & Yamane, K. (2005). Mimetic communication theory for humanoid robots interacting with humans. In *Proceedings of International Symposium on Robotics Research.*

Nakanishi, J., Cory, R., Mistry, M., Peters, J., & Schaal, S. (2005). Comparative experiments on task space control with redundancy resolution. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems.*

Nehaniv, C., & Dautenhahn, K. (2002). The correspondence problem. In C. Nehaniv & K. Dautenhahn (Eds.), (chap. Chapter 2). MIT Press.

Nelson, W. (1983). Physical principles for economies of skilled movements. *Biological Cybernetics*, *46*, 135-147.

Ng., A., & Jordan, M. (2000). PEGASUS: A policy search method for large mdps and pomdps. In *Proceedings of interational conference on uncertainty in artificial intelligence.*

Nguyen-Tuong, D., & Peters, J. (2010a). Model learning with local gaussian process recognition. *Advanced Robotics*.

Nguyen-Tuong, D., & Peters, J. (2010b). Using model knowledge for learning inverse dynamics. In *Proceedings of IEEE International Conference on Robotics and Automation.*

Ott, C. (2008). *Cartesian impedance control of redundant and flexible-joint robots* (B. Siciliano, O. Khatib, & F. Groen, Eds.). Spinger.

Ott, C., Bauml, B., Borst, C., & Hirzinger, G. (2005). Employing cartesian impedance control for the opening of a door: A case study in mobile manipulation. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems.*

Ott, C., Eiberger, O., Friedl, W., Bauml, B., Hillenbrand, U., Borst, A., C. Albu-Schaffer, et al. (2006). Humanoid two-arm system for dexterous manipulation. In *Proceedings of IEEE International Conference on Humanoid Robotics.*

Ott, C., Mukherjee, R., & Nakamura, Y. (2010). Unified impedance and admittance control. In *Proceedings of IEEE international conference on robotics and automation.*

Parillo, P. (2003). Semidefinite programming relaxations for semialgebraic problems. *Mathematical Progress*, *96*, 293-320.

Park, D.-H., Hoffmann, H., Pastor, P., & Schaal, S. (2008). Movement reproduction and obstacle avoidance with dynamic movement primitives and potential field. In *Proceedings of IEEE International Conference on Humanoid Robots.*

Park, J., & Sandberg, I. (1991). Universal approximation using radial-basis-function networks. *Neural Computation*, *3(2)*, 246 - 257.

Pastor, P., Hoffmann, H., Asfour, T., & Schaal, S. (2009). Learning and generalization of motor skills by learning from demonstration. In *Proceedings of IEEE International Conference on Robotics and Automation.*

Peifer, M., Timmer, J., & Voss, H. (2002). Nonparametric identification of nonlinear oscillating systems. *Journal of Sound and Vibration*.

Peters, J., Mistry, M., Udwadia, F., & Schaal, S. (2005). A unifying methodology for the control of robotic systems. *Proceedings of IEEE International Conference on Intelligent Robots and Systems*.

Peters, J., & Schaal, S. (2007). Reinforcement learning by reward-weighted regression for operational space control. In *Proceedings of International Conference on Machine Learning.*

Peters, J., & Schaal, S. (2008a). Learning to control in operational space. *International Journal of Robotics Research*, *27*, 197-212.

Peters, J., & Schaal, S. (2008b). Natural actor-critic. *Neurocomputing*, *71*, 1180-1190.

Petersen, D., & Middleton, D. (1962). Sampling and reconstruction of wave number limited functions in n-dimensional euclidian spaces. *Information and Control*, *5*, 279-323.

Petreska, B., & Billard, A. (2009). Movement curvature planning through force field internal models. *Biological Cybernetics*, *100*, 331-350.

Phillips, J., Bedrosian, N., & Kavraki, L. (2003). Spacecraft rendezvous and docking with real-time randomized optimization. In *Proceedings AIAA Guidance, Navigation and Control Conference.*

Pineda, F. J. (1989). Recurrent backpropagation and the dynamical approach to adaptive neural computation. *Neural Computation*, 161-172.

Popovic, J., Seitz, S., Erdmann, M., Popovic, Z., & Wiktin, A. (2002). Interactive manipulation of rigid body simulations. In *Proceedings ACM SIGGRAPH.*

Quinlan, S. (1994). *Real-time modification of collision-free paths*. Unpublished doctoral dissertation, Stanford University.

Rabiner, L. (1989). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. , *77*, 257-286.

Rahman, M., Ikeura, R., & Mizutani, K. (2002). Investigation of the impedance characteristic of human arm for development of robots to cooperate with humans. *International Journal of Mechanical Systems, Machine Elements and Manufacturing*, *45*, 510-518.

Raibert, M., & Craig, J. (1981). Hybrid position/force control of manipulators. *ASME Journal of Dynamical Systems, Measurement and Control*, *105*, 126-133.

Rasmussen, C., & Williams, C. (2006). *Gaussian processes for machine learning* (C. Rasmussen & C. Williams, Eds.). MIT Press.

Rasmussen, J., Damsgaard, M., & Voigt, M. (2001). Muscle recruitment by the min/max criterion–a comparative numerical study. *Journal of Biomechanics*, *34*, 409-415.

Ratliff, N., Bagnell, A., & Zinkevich, M. (2006). Maximum margin planning. In *Proceedings of International Conference on Machine Learning.*

Ratliff, N., Silver, D., & Bagnell, J. (2009). Learning to search: Functional gradient techniques for imitation learning. *Autonomous Robots*, *27(1)*, 25-53.

Reed, K., & Peshkin, M. (2008). Physical collaboration of human-human and human-robot teams. *IEEE Transactions on Haptics*, *1(2)*, 108-120,.

Reed, K., Peshkin, M., Hartmann, M., Grabowecky, M., Patton, J., & Vishton, P. (2007). Haptically linked dyads: Are two motor-control systems better than one? *Psychological Science*.

Righetti, L., Buchli, J., & Ijspeert, A. (2006). Dynamic hebbian learning in adaptive frequency oscillators. *Physica D*, *216*(2), 269–281.

Rizzi, A. (1998). Hybrid control as a method for robot motion programming. In *Proceedings of IEEE International Conference on Robotics and Automation.*

Rizzolatti, G., & Craighero, L. (2004). The mirror-neuron system. *Annual Review of Neuroscience*, *27*, 169-192.

Roberts, P., & Bell, C. (2000). Computational consequences of temporally asymmetric learning. *Journal of Computational Neuroscience*, *9*, 67-83.

Roweis, S., & Ghahramani, Z. (2001). Learning nonlinear dynamical systems using the Expectation-Maximization algorithms. *Kalman Filtering and Neural Networks*, 175-220.

Sahar, G., & Hollerbach, J. M. (1986). Planning minimum time trajectories for robot arms. *International Journal of Robotic Research (IJRR)*, *5(3)*, 97Ű-140.

Sakoe, H., & Chiba, S. (1978). Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustic, Speech, and Signal Processing*, 43-49.

Sallnas, E., & Zhai, S. (2003). Collaboration meets fittsŠ law: Passing virtual objects with and without haptic force feedback. In *Proceedings of conference on human-computer interaction.*

Sanner, R., & Slotine, J.-J. (1992). Gaussian networks for direct adaptive control. *IEEE Transaction on Neural Networks*, *3*, 837-863.

Schaal, S., & Atkeson, C. (1994). Robot juggling: implementation of memory-based learning. *IEEE Control Systems Magazine*, *14*, 57-71.

Schaal, S., & Atkeson, C. (1998). Constructive incremental learning from only local information. *Neural Computation*, *10*(8), 2047-2084.

Schaal, S., Ijspeert, A., & Billard, A. (2003). Computational Approaches to Motor Learning by Imitation. *Philosophical Transactions: Biological Sciences*, *358*, 537–547.

Schaal, S., Mohajerian, P., & Ijspeert, A. (2007). Dynamical Systems vs. Optimal Control – a unifying view. *Progress in Brain Research*, *165*, 425-445.

Schaal, S., Vijayakumar, S., & Atkeson, C. (1998). Local dimensionality reduction. In *Neural information processing.*

Schittkowski, K. (1994). Parameter estimation in systems of nonlinear equations. *Numerical Mathematics*, *68*, 129-142.

Schmidt, R. (1975). A schema theory of discrete motor skill learning. *Psychological Review*, *82*, 225-260.

Schoner, G. (1990). A dynamic theory of coordination of discrete movements. *Biological Cybernetics*, *63*, 257-270.

Shadmehr, R., Smith, M., & Krakauer, W. (2010). Error correction. sensory prediction, and adaptation in mirror control. *Annual Reviews of Neuroscience*.

Shimansky, Y., Kang, T., & He, J. (2004). A novel model of motor learning capable of developing an optimal movement control law online from scratch. *Biological Cybernetics*, *90*, 133-145.

Siciliano, B., & Khatib, O. (Eds.). (2008). *Handbook of robotics*. Springer-Verlag.

Slotine, J., & Coetsee, J. (1986). Adaptive sliding controller synthesis for nonlinear systems. *International Journal of Control*, *43*, 1631-1651.

Slotine, J.-J., & Li, W. (1991). *Applied nonlinear control*. Prentice Hall.

Smeets, J., & Brenner, E. (1999). A new view on grasping. *Motor Control*, *3*, 237-271.

Sternad, D., & Schaal, D. (1999). Segmentation of end-point trajectories does not imply segmented control. *Experimental Brain Research*, *124*, 118-136.

Strogatz, S. (1994). *Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering*. Addison-Wesley.

Sundar, S., & Shiller, Z. (1997). Optimal obstacle avoidance based on the Hamilton-Jacobi-Bellman equation. *IEEE Transactions on Robotics and Automation*, *13(2)*, 305-310.

Sutton, R., McAllester, D., Singh, S., & Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems.*

Takeda, T., Kosuge, K., & Hirata, Y. (2005). HMM-based dance step estimation for dance partner robot – MS DanceR. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems.*

Tanizaki, H. (1993). *Nonlinear filters: Estimation and Applications*. Springer, Berlin.

Tedrake, R. (2009). LQR-trees: Feedback motion planning on sparse randomized trees. In *Proceedings of Robotics, Science and Systems.*

Tedrake, R., Manchester, I., Tobenkin, M., & Roberts, J. (2010). LQR-trees: Feedback motion planning via Sums-of-Squares verification. *International Journal of Robotic Research (IJRR)*, *OnlineFirst*.

Tee, K., Franklin, D., Kawato, M., Milner, T., & Burdet, E. (2010). Concurrent adaptation of force and impedance in the redundant muscle system. *Biological Cybernetics*, *102*, 31-44.

Tegin, J., Ekvall, S., Kragic, D., Wikander, J., & Iliev, B. (2009). Demonstration-based learning and control for automatic grasping. In *Intelligent service robotics* (p. 1861-2784). Springer Berlin / Heidelberg.

Theiler, J. (1990). Statistical precision of dimension estimators. *Physical Review A*, *51*, 3038-3051.

Theiler, J., & Smith, L. (1995). Anomalous convergence of Lyapunov exponent estimates. *Physical Review E*, *51*, 3738-3741.

Theodorou, E., Buchli, J., & Schaal, S. (2010). A generalized path integral control approach to reinforcement learning. *Journal of Machine Learning Research*, *1*, 1-48.

Timmer, J., Rust, H., Horbelt, W., & Voss, H. (2000). Parametric, nonparametric and parametric modelling of a chaotic circuit time series. *Physical Letters A*, *274*, 123-134.

Tobenkin, M., Manchester, I., Wang, J., Megretski, A., & Tedrake, R. (2010). *Convex optimization in identification of stable non-linear state space models* (Tech. Rep.). MIT.

Todorov, E. (1998). *Studies of goal directed movements*. Unpublished doctoral dissertation, Massachussetts Institute of Technology.

Todorov, E. (2004). Optimality principles in sensorimotor control. *Nature Neuroscience*, *7(9)*, 907-915.

Todorov, E., & Jordan, M. (1998). Smoothness maximization along a predefined path accurately predicts the speed profiles of compex arm movements. *Journal of Neurophysiology*, *80*, 696-714.

Todorov, E., & Jordan, M. (2003). A minimal intervention principle for coordinated movement. In *Proceedings of Advances in Neural Information Processing.*

Todorov, E., & Jordan, M. I. (2002). Optimal feedback control as a theory of motor coordination. *Nature Neuroscience*, *5(11)*, 1226Ű-1235.

Tomohisa, H., Haddad, W. M., & Naira, H. (2008). Neural network adaptive control for a class of nonlinear uncertain dynamical systems with asymptotic stability guarantees. *IEEE Transactions on Neural Networks*, *19*, 80-90.

Toussaint, M., & Goerick, C. (2007). Probabilistic inference for structured planning in robotics. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems.*

Travis, D., Thumati, B. T., & Jagannathan, S. (2009). Optimal control of unknown affine nonlinear discrete-time systems using offline-trained neural networks with proof of convergence. *Neural Networks*, *22*, 851-860.

Tsuji, T., Morasso, P., Kazuhiro, G., & Ito, K. (1995). Human hand impedance characteristics during maintained posture. *Biological Cybernetics*, *72*, 475-485.

Tsuji, T., Takeda, Y., & Tanaka, Y. (2004). Analysis of mechanical impedance in human arm movements using a virtual tennis system. *Biological Cybernetics*, *91*, 295-305.

Tsumugiwa, T., Yokogawa, R., & Hara, K. (2001). Variable impedance control with regard to working process for man-machine cooperation-work system. In *Proceedings of IEEE International Conference on Intelligent Systems and Robotics.*

Ude, A. (1993). Trajectory generation from noisy positions of object features for teaching robot paths. *Robotics and Autonomous Systems*, *11(2)*, 113–127.

Ude, A., Atkeson, C., & Riley, M. (2004). Programming full-body movements for humanoid robots by observation. *Robotics and Autonomous Systems*, *47*, 93-108.

Ude, A., Gams, A., Asfour, T., & Morimoto, J. (2010). Task-specific generalization of discrete and periodic dynamic movement primitives. *IEEE Transactions on Robotics*, *26(5)*, 800-815.

Valency, T., & Zacksenhouse, M. (2003). Accuracy/robustness dilemma in impedance control. *ASME Journal of Dynamic Systems, Measurement, and Control*, *125*, 310Ű319.

Van Huffel, S., & Vandewalle, J. (1991). *The total least squares problem: Computational aspects and analysis*. SIAM, Filadelfia.

Vijayakumar, S., D'Souza, A., & Schaal, S. (2003). Incremental online learning in high dimensions. *Neural Computation*, *17(12)*, 2602-2634.

Vijayakumar, S., D'Souza, A., & Schaal, S. (2005). *Incremental online learning in high dimensions* (Tech. Rep.). University of Edinburgh.

Vijayakumar, S., & Schaal, S. (2000). Locally Weighted Projection Regression: Incremental Real Time Learning in High Dimensional Space. In *Proceedings of International Conference on Machine Learning* (pp. 1079–1086).

Vivani, P., & Terzuolo, C. (1982). Trajectory determines movement dynamics. *Neuroscience*, *7*, 431Ű-437.

Voss, H., & Kurths, J. (1997). Reconstruction of nonlinear time delay models from data by the use of optimal transformations. *Physical Letters A*, *234*, 336-344.

Wang., A., Fleet, D., & Hertzmann, A. (2008). Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *30(2)*, 283-298.

Wang, J., Fleet, D., & Hertzmann, A. (2006). Gaussian process dynamical models. In *Advances in Neural Information Processing Systems.*

Wang, Z., Peer, A., & Buss, M. (2009). An HMM approach to realistic haptic human-robot interaction. In *Proceedings of the joint eurohaptics conference and symposium.*

Wei, H., & Amari, S. (2008). Dynamics of learning near singularities in radial basis function networks. *Neural Networks*, *21*, 981-1005.

Werbos, P. (1980). Backpropagation through time: what it does and how to do it? *Proceedings of IEEE*, *78*, 1550 - 1560.

Willems, J. (1972). Dissipative dinamical systems-Part I: General Theory. *Archives of Rational Mechanics and Analysis*, *45*, 321-351.

Yamane, K., Kuffner, J., & Hodgins, J. (2004). Synthesizing animations of human manipulation tasks. *ACM Transactions on Graphics*, *23(3)*.

Yang, B., & Asada, H. (1996). Progressive learning and its applications to robot impedance learning. *IEEE Transactions on Neural Networks*, *7(4)*, 941-952.

Yang, C., Ganesh, G., Haddadin, S., Parusel, S., Albu-Schaeffer, A., & Burdet, E. (2011). Human-like adaptation of force and impedance in stable and unstable interactions. *IEEE Transactions on Robotics*, *27(5)*, 918-930.

Yun, X., & Tan, K. (1997). A wall-following method for escaping local minima in potential field based motion planning. In *Proceedings of IEEE International Conference on Advanced Robotics.*

Zefran, M., & Burdick, J. (1998). Stabilization of systems with changing dynamics by means of switching. In *Proceedings of IEEE Conference on Robotics and Automation.*

Ziebart, B., Maas, A., Bagnell, J., & Dey, A. (2008). Maximum entropy inverse reinforcement learning. In *Proceeding of AAAI International Conference on Artificial Intelligence.*

Zollner, R., Afour, T., & R., D. (2004). Programming by demonstration: Dual-arm manipulation tasks for humanoid robots. In *Proceedings of IEEE International Conference on Intelligent Robots and Systems.*

Curriculum Vitae

Elena Gribovskaya

Date of birth: 11.10.1983

Private address :
Rue Baulacre 9
c/o Eric Sauser
Geneve, Switzerland
elena.gribovskaya@yahoo.com

+41 78 78 57 349

Professional address :
EPFL-STI-I2S-LASA, Station 9
CH 1015, Lausanne, Switzerland
elena.gribovskaya@epfl.ch

+41 21 69 35 956

Areas of expertise

Robot Learning	Dynamical Systems	Active Learning
Human-Robot Interaction	Machine learning	Haptic Interaction

My work aims at endowing robots with advanced *motor skills* through *learning* these skills from humans. On the application side, I am particularly interested in teaching to the robots *kinematic* and *haptic* aspects of *coordination*. On the theoretical side, I believe in *machine learning* and *dynamical systems* as promising means for boosting robots's *motion planning* abilities.

Research Experience and Collaboration

Research assistant
Learning Algorithms and Systems Laboratory — November 2006 - now
EPFL, Lausanne, Switzerland

Duties

- research and development of algorithms for statistical learning in robotic applications
- development of software for robot control (programming on C/C++ under Linux OS)
- supervision of Master students

Extra Duties

- organization of joint experiments with partner research groups
- preparation of deliverables and presentations on conducted work for the European Commission
- reviewing for several robotics journals and conferences
- student activities co-chair in IEEE RAS

Visiting Researcher
Joint Robotics Laboratory, — March 2009, October 2009
Advanced Institute of Science and Technology,
Tsukuba, Japan

Duties

- specifying research objectives for the collaboration
- conducting joint experiments

Visiting Researcher
Joint French-Japanese Robotic Laboratory (JRL), — April 2008
Centre National de Recherché Scientifique (CNRS),
Toulouse, France

Duties

- conducting experiments on human motion capturing with a vision-based system

Education

PhD in robot learning (to be submitted)
LASA Laboratory, EPFL, Switzerland.
Imitation Learning of Motion Coordination in Robots: A Dynamical System Approach
Supervisor: Prof. Aude Billard

MSc in Applied Mathematics (2006) (*diploma with honors, GPA 5/5*)
(specialization in Mathematical Modeling)
Voronezh State University, Voronezh, Russia
Fuzzy adaptive control of nonlinear systems

BSc in Applied Mechanics (2004) (*diploma with honors, GPA 5/5*)
Voronezh State University, Voronezh, Russia

Publications

International Journals

Learning Nonlinear Multivariate Dynamics of Motion in Robotic Manipulators
E. Gribovskaya , S. M. Khansari Zadeh , Aude Billard
International Journal of Robotic Research, (2011).

International Conferences

Motion Learning and Adaptive Impedance for Robot Control during Physical Interaction with Humans
E.Gribovskaya, A. Kheddar, and A.Billard

Proceeding of the IEEE-RAS International Conference on Robotics and Automation, (2011).

Learning Motion Dynamics to Catch a Moving Object
Seungsu Kim, Elena Gribovskaya, Aude Billard

Proceedings of the IEEE-RAS International Conference on Humanoid Robots, (2010).

Learning Nonlinear Multi-Variate Motion Dynamics for Real- Time Position and Orientation Control of Robotic Manipulators
E. Gribovskaya and A. Billard
Proceedings of 9th IEEE-RAS International Conference on Humanoid Robots, (2009).

Teaching Physical Collaborative Tasks: Object-Lifting Case Study with a Humanoid
P. Evrard, E. Gribovskaya, S. Calinon, A. Billard and A. Kheddar
Proceedings of IEEE International Conference on Humanoid Robots, (2009).

Learning collaborative manipulation tasks by demonstration using a haptic interface
S. Calinon, P. Evrard, E. Gribovskaya, A. Billard and A. Kheddar
Proceedings of the International Conference on Advanced Robotics (ICAR), (2009).

Combining Dynamical Systems Control and Programming by Demonstration for Teaching Discrete Bimanual Coordination Tasks to a Humanoid Robot
E. Gribovskaya and A. Billard
Proceedings of 3rd ACM/IEEE International Conference on Human-Robot Interaction (HRI) (2008).

A Model of Acquisition of Discrete Bimanual Coordination Skills for a Humanoid Robot
E. Gribovskaya and A. Billard
Proceedings of the International Conference in Epigenetic Robotics, (2007)

International Workshops

An Active Learning Interface for Bootstrapping Robot's Generalization Abilities in Learning from Demonstration
E. Gribovskaya and A. Billard *Presented at:* Robotics Science and Systems (RSS), Workshop "Towards Closing the loop: Active Learning for Robotics*",* (2010).

Learning the Nonlinear Multivariate Dynamics of Motion of Robotic Manipulators
E. Gribovskaya and A. Billard *Presented at:* International Conference in Neural Information Processing Systems, Workshop "Women in Machine Learning*"* (2009).

Combining Task-Level and Trajectory-Level Learning for Teaching Robots Bimanual Coordinated Tasks
E. Gribovskaya and A. Billard *Presented at:* Robotics Science and Systems (RSS). (2009)

Human Motion Prediction in a Human-Robot Joint Task
E. Gribovskaya and A. Billard *Presented at:* IEEE International Conference on Intelligent Robots and Systems (IROS), Workshop on Robotics Challenges for Machine Learning.
(2008)

Thesis

Fuzzy adaptive control of nonlinear systems *[in Russian]*
E. Gribovskaya. MSc thesis. Mathematical Modeling Laboratory, Voronezh State University, Voronezh, Russia (2006)

Invited Talks

Motion Learning and Adaptive Impedance for Robot Control during Physical Interaction with Humans
Given at: IEEE International Symposium on Robot and Human Interactive Communication, Workshop: "Haptic Joint Actions by Humans and Robots", (2010).

Robot Programming by Demonstration
Given at: LAAS, CNRS, (2008).

Teaching Experience

Supervision of MSc theses and projects

Implementation of a learning-by-imitation algorithm on the Katana Robot
Richard, M. Ecole Polytechnique Fédérale de Lausanne(EPFL), Lausanne, Switzerland (2007).

Studying coupling of planning position and orientation of a robot's hand in manipulation tasks
Garcia Fernandez, D. Universidad Carlos III de Madrid(UC3M), Madrid, Spain (2010).

Implementation of an Algorithm for Learning Vision Guided Grasping
Bachman, P. Ecole Polytechnique Fédérale de Lausanne(EPFL), Lausanne, Switzerland (2009).

Imitational learning of a human-robot joint task
Bozyd, C. Ecole Nationale Supérieure de Techniques Avancées (ENSTA), Paris, France (2008).

Implementation of a bi-manual coordinated task on Katana robotic arms
Alexander, L. Ecole Polytechnique Fédérale de Lausanne(EPFL), Lausanne, Switzerland (2007).

External expert
at the exam "Biological and artificial intelligent systems" (graduate course), EPFL 2007, 2008

Technical Skills

Programming: C/C++, C#, HTML
Computational engines: MATLAB, Maple
Robotic platforms: HOAP-3, Katana, HRP-2, i-Cub, WAM Barrett Arm
OS: Windows, Linux

Languages

Russian:	Mother tongue
English:	Fluent (TOEFL CBT, 272/300 (2006))
French:	Intermediate
German:	Basic

Awards and Grants

2010	Travel grant for attending Workshop "Human-Robot Interaction (HRI) Pioneers" (1500 USD)
2009	Travel grant for attending Workshop "Women in Machine Learning" (1000 USD)
2009	Best poster award among doctoral students, the EPFL's research day.
2008	Travel grant for attending Conference "Human Robot Interaction" (1000 EUR)
2006	One-year scholarship for the doctoral studies in EPFL (~40000 CHF, selectivity: 10/100)
2004	One-year federal scholarship for excellent students, Russia (selectivity: 20 / 1000)
2004-2006	Federal scholarship for master studies, Russia
2000-2004	Federal scholarship for bachelor studies, Russia

Academic Services

Organizational activities

2010-2011	Local chair of the workshop HRI Pioneers 2011
2008	IEEE RAS Student Activity Committee, Co-chair

Affiliations to Research Projects

European Project **Robot@CWE** (Robots in the future collaborative working environments)
http://www.robot-at-cwe.eu

European Project **Feelix Growing** (Feel, interact, express: a global approach to development with interdisciplinary grounding)
http://www.feelix-growing.org.

Referring

International Journal of Social Robotics, 2011

IEEE Transactions on Robotics (TRO), 2008, 2009.

Autonomous Robots (AR), 2008 and 2009.

IEEE International Conference on Intelligent Robots and Systems (IROS) 2010

IEEE International Conference on Robotics and Automation (ICRA) 2010, 2011

IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN) 2008, 2009, 2010, 2011