UNIVERSITY OF CALIFORNIA, SAN DIEGO

Machine Learning for Humanoid Robot Modeling and Control

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Computer Science

by

Tingfan Wu

2013

The dissertation of Tingfan Wu is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

Chair

University of California, San Diego

2013

DEDICATION

To the robots whom I love and hate.

EPIGRAPH

What I cannot create, I do not understand.
– Richard Feynman
on his blackboard at time of death in 1988

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

Foremost, I thank my thesis advisor, Javier Movellan, for advising me during the last 6 years. Javier's clarity of insight never fails to impress me during our research discussions, and his humanity as an advisor resulted in a very rich and enjoyable graduate school career.

I thank Marian Stewart Bartlett and Gwen Littlewort-Ford for many valuable research and career discussions.

I thank Gary Cottrell for serving as my "in-house" advisor within the Department of Computer Science & Engineering, and for facilitating my interdisciplinary research at UCSD.

I thank my friends and co-authors both at the MPLab: Paul Ruvolo, Jacob Whitehill, and Nicholas Butko.

I thank my collaborators, Emanuel Todorov, Yuval Tassa, Vikash Kumar and Evangelos Theodorou for their insightful discussion and technical support..

I thank my other committee members, Serge Belongie, Thomas Bewley, Lawrence Saul, Mohan Trivedi, for their advice and for taking the time to examine my thesis.

Finally, I thank my family and friends, in particular Wei-Ting Liu and Jenny Lee, for providing mental support during my study.

Chapter 2, in full, is submitted to *Humanoids 2013* conference. The dissertation author was the primary investigator and author of this paper.

Chapter 3, in full, is a reprint of the material as it appears in "Tingfan Wu and Javier Movellan. Semi-parametric gaussian process for robot system identification. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 725–731. IEEE, 2012". The dissertation author was the primary investigator and author of this paper.

Chapter 4, in full, is a reprint of the material as it appears in "Yuval Tassa, Tingfan Wu, Javier Movellan, and Emanuel Todorov. Modeling and identification of pneumatic actuators. In *IEEE International Conference on Mechatronics and Automation*, 2013." The dissertation author was the author of this paper.

Chapter 5, in full, is a reprint of the material as it appears in "Tingfan

Wu, Nicholas J Butko, Paul Ruvulo, Marian S Bartlett, and Javier R Movellan. Learning to make facial expressions. In *IEEE 8th International Conference on Development and Learning, 2009 (ICDL 2009)*, pages 1–6. IEEE, 2009.” The dissertation author was the primary investigator and author of this paper.

Chapter 6, in full, is submitted to *Neural Information Processing Systems (NIPS) 2013* conference. The dissertation author was the primary investigator and author of this material.

Chapter 7, , in full, is a reprint of the material as it appears in “Tingfan Wu, Juan Artigas, Whitnet Mattson, Paul Ruvolo, Javier Movellan, and Daniel Messinger. Collecting a developmental dataset of reaching behaviors: First steps. In *IROS2011 Workshop on Cognitive Neuroscience Robotics*, 2011.” The dissertation author was the primary investigator and author of this material.

Support for the research in the thesis was provided by National Science Foundation grants SBE-0542013, IIS INT2-0808767, 0808653, and a DURIP grant.

VITA

2013	Ph.D. in Computer Science, University of California, San Diego – La Jolla, California
2010	M.Sc. in Computer Science, University of California, San Diego – La Jolla, California
2004	B.S. in Computer Science, National Taiwan University – Taipei City, Taiwan

PUBLICATIONS

Tingfan Wu and Javier Movellan. DNA: Dynamic network adaptation for optimal control. In *submission to Neural Information Processing Systems 2013*

Tingfan Wu, Yuval Tassa, Javier Movellan, and Emanuel Todorov. STAC: Simultaneous tracking and calibration. In *submission to Humanoids 2013*

Yuval Tassa, Tingfan Wu, Javier Movellan, and Emanuel Todorov. Modeling and identification of pneumatic actuators. In *IEEE International Conference on Mechatronics and Automation*, 2013

Tingfan Wu and Javier Movellan. Semi-parametric gaussian process for robot system identification. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 725–731. IEEE, 2012

Tingfan Wu, Nicholas J Butko, Paul Ruvolo, Jacob Whitehill, Marian S Bartlett, and Javier R Movellan. Multilayer architectures for facial action unit recognition. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics.*, 42(4):1027–1038, 2012

Tingfan Wu, Juan Artigas, Whitnet Mattson, Paul Ruvolo, Javier Movellan, and Daniel Messinger. Collecting a developmental dataset of reaching behaviors: First steps. In *IROS2011 Workshop on Cognitive Neuroscience Robotics*, 2011

Tingfan Wu, Nicholas J Butko, Paul Ruvolo, Jacob Whitehill, Marian S Bartlett, and Javier R Movellan. Action unit recognition transfer across datasets. In *IEEE International Conference on Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011*, pages 889–896. IEEE, 2011

Tingfan Wu, Marian S Bartlett, and Javier R Movellan. Facial expression recognition using gabor motion energy filters. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2010*, pages 42–47. IEEE, 2010

Tingfan Wu, Nicholas J Butko, Paul Ruvulo, Marian S Bartlett, and Javier R Movellan. Learning to make facial expressions. In *IEEE 8th International Conference on Development and Learning, 2009 (ICDL 2009)*, pages 1–6. IEEE, 2009

Tingfan Wu, Chih-Jen Lin, and Ruby C Weng. Probability estimates for multi-class classification by pairwise coupling. *The Journal of Machine Learning Research*, 5:975–1005, 2004

Jacob Whitehill, Paul Ruvolo, Tingfan Wu, Jacob Bergsma, and Javier Movellan. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. *Advances in Neural Information Processing Systems*, 22(2035-2043):7–13, 2009

Gwen Littlewort, Jacob Whitehill, Tingfan Wu, Ian Fasel, Mark Frank, Javier Movellan, and Marian Bartlett. The computer expression recognition toolbox (CERT). In *IEEE International Conference on Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011*, pages 298–305. IEEE, 2011

Marian Bartlett, Gwen Littlewort, Tingfan Wu, and Javier Movellan. Computer expression recognition toolbox. In *8th IEEE International Conference on Automatic Face & Gesture Recognition, 2008. FG'08.*, pages 1–2. IEEE, 2008

Gwen Littlewort, Jacob Whitehill, TingFan Wu, Nicholas Butko, Paul Ruvolo, Javier Movellan, and Marian Bartlett. The motion in emotion – a CERT based approach to the fera emotion challenge. In *IEEE International Conference on Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011*, pages 897–902. IEEE, 2011

Marian Bartlett, Gwen Littlewort, Jacob Whitehill, Esra Vural, Tingfan Wu, Kang Lee, Aytül Erçil, Müjdat Cetin, and Javıer Movellan. Insights on spontaneous facial expressions from automatic expression measurement. *Dynamic Faces: Insights from Experiments and Computation*, 2006

Karan Sikka, Tingfan Wu, Josh Susskind, and Marian Bartlett. Exploring bag of words architectures in the facial expression domain. In *Computer Vision–ECCV 2012. Workshops and Demonstrations*, pages 250–259. Springer, 2012

Fei Long, Tingfan Wu, Javier R Movellan, Marian S Bartlett, and Gwen Littlewort. Learning spatiotemporal features by using independent component analysis with application to facial expression recognition. *Neurocomputing*, 2012

Yiwen Wang, Tingfan Wu, Garrick Orchard, Piotr Dudek, Michele Rucci, and Bertram E Shi. Hebbian learning of visually directed reaching by a robot arm. In *Biomedical Circuits and Systems Conference, 2009. BioCAS 2009. IEEE*, pages 205–208. IEEE, 2009

Paul Ruvolo, Tingfan Wu, and Javier R Movellan. Control by gradient collocation: Applications to optimal obstacle avoidance and minimum torque control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1173–1179. IEEE, 2012

ABSTRACT OF THE DISSERTATION

Machine Learning for Humanoid Robot Modeling and Control

by

Tingfan Wu

Doctor of Philosophy in Computer Science

University of California, San Diego, 2013

Garrison Cottrell, Chair

Biologically inspired humanoid robots present new challenges for system identification and control due to the presence of many degrees of freedom, highly compliant actuators, and non-traditional force transmission mechanisms. In this thesis, we address these challenges using machine learning approaches. The key idea is to replace classical laborious manual model calibration and motion programming with statistical inference and learning from multi-modal sensory data. To this end, we develop several new parametric models and their parameter identification algorithms enabling new sensor/ actuator configurations beyond the scope of previous approaches. In addition, we also develop a semi-parametric model to learn from experiences not predicted by the parametric model. Using similar approaches grounded in machine learning, we also develop methods to allow humanoid robots to learn to make facial expressions, kick a ball, and to reach for objects while collaborating with people. We collected a unique dataset that describes development of infant reaching behavior while interacting with an adult caregiver. We compared the observed development of social reaching in human infants with the machine learning based development behavior in a complex humanoid robot.

Chapter 1

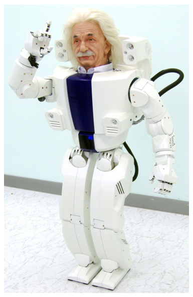
Introduction

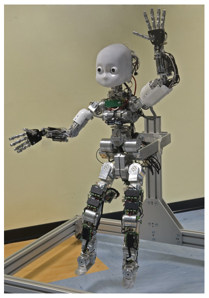
A humanoid is a robot with human-like body, appearance, and movement dynamics. This human-like design enables humanoids to operate in human-engineered environment, such as stairs, and to use tools designed for humans, such as a hammer. In addition, human-like appearance facilitates human-like interaction between humans and robots, using channels such as hand gestures and facial expressions. Besides these engineering benefits, building humanoids also contributes the science of understanding how the human brain handles the complex sensory-motor problem of controlling the human body.

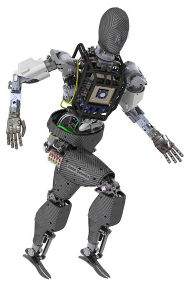
Ambitious goals have been set for future humanoid robots. They are expected to serve as companions and assistants for humans in daily life and as ultimate helpers in the case of natural disasters. The RoboCup robot soccer organization has also set a goal that in 2050, a team of humanoid robot soccer players shall compete and win against the winner of the most recent World Cup. In 2014, DARPA will hold the first Robotics Grand Challenge to promote the development of robots which do things like humans in a world made for humans, with a focus on disaster recovery challenge tasks.

Driven by these potential advantages and long-term vision, considerable progress has been made in humanoid research, resulting in a number of anthropomorphic robots. Figure 1.1 and Table 1.1 show several examples. “Diego-San” is the robot used in this thesis. Unlike traditional industrial robot arms with only a few degrees of freedom and electrical actuators, most humanoid robots are

(a) Asimo


(b) Hubo


(c) iCub


(d) Atlas

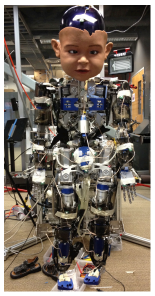

(e) Diego-San

Figure 1.1: humanoid robots

Table 1.1: List of arm and humanoid robots.

Name	#Joint	Actuator	Face
Barret Arm	7	electric motor	N/A
Asimo	34	electric motor	N/A
Hubo-2	40	electric motor	actuated realistic face
iCub	32	electric motor	LED light animation
Baxter	16	SEA electric motor	rendered computer animation
ATLAS	28	hydraulic	N/A
Diego-San	38∗	pneumatic	18-servo actuated realistic face

Table 1.2: hardware and software specification of Diego-San humanoid robot

Full Name	Diego-San
Body Creator	Kokoro Corporation, Japan.
Face Creator	Hanson Robotics, US
Height	130cm
Weight	35kg
Sensors	2 DragonFly-2 camera in the eyes, 2 microphones and 1 speaker in the ears and the mouth, 38 potentiometers, 88 pressure sensors, 2 Inertial measurement units
Actuators	44 pneumatic actuators for body joints. 18 electric RC servos for the face
Computing	The computer is a Intel Xeon 2.8Ghz 12-core machine. The analog sensors and pneumatic actuators are connected to the computer using a National Instruments Data Acquisition System (DAQ)
Degrees of Freedom	**38 independently controlled pneumatic body joints** (HeadNeck-4, Trunk-4, Each Arm-8 , Each Leg 7). And 6 pneumatic actuators on the hands controlling all 26 finger joints with tendon coupling. Each joint has 2 degrees of freedom (position and compliance). 18 electric servos for facial expression

designed to match human skeleton complexity and actuator capabilities. This results in high degree of freedom and the wide adoption of novel biologically inspired actuators. In addition, humanoids designed to interact with humans are usually equipped with animated faces capable of posing facial expressions. These special designs pose new challenges in modeling and controlling humanoid robots, which is the primary target of this thesis.

1.1 Design of Diego-San Humanoid Robot

Humanoid robots stand out from general robots in the following aspects: rich variety of sensors, high degrees of freedom, and novel biologically inspired actuators. In this section, we discuss our design choices in these aspects for our humanoid robot, Diego-San. For a list of the software and hardware specification of Diego-San, please refer to Tbl. 1.2.

1.1.1 Pneumatic Joint Actuators

Actuators are the mechanism allowing a robot to act. Most humanoid robots utilize electric motors; others use hydraulic or pneumatic actuators. Electrical motors are popular due to the easiness to control as well as simple wiring. However their torque output is weak and therefore, electrical motors are typically coupled with geared transmission to amplify their torque. In turn the gearing makes the joints stiff, and diagonalizes the robot's inertial matrix. While this simplifies the robot control problem, it results on stiff non-human movement that we identify as robotic motion. Hydraulic actuators provide strong force but they are not as compliant as pneumatic ones since liquid is not compressible.

In this thesis, we study the problem of system identification and control of a complex humanoid robot (named Diego-San) that uses pneumatic actuators (see Figure 1.2 for a diagram of a pneumatically actuated joint). Pneumatic actuators are becoming increasingly popular in robotics due to their high speed and force capability, which makes the use of complex transmission gears unnecessary, as well as relatively low price and overall robustness. From the perspective of bio-robotics, pneumatic actuators are further desirable because they have many of the essential properties of biological muscle at the mechanism level. In particular: (i) as the joint moves in the direction of actuated force, the chamber volume increases and thus the pressure/force drops, resulting in stiffness; (ii) this stiffness can be tuned by activating opposing cylinders, much like a biological limb becomes stiffer when antagonist muscles co-contract; (iii) the actuator has an internal activation state (air mass in the case of pneumatics, calcium concentration in the case of muscles)

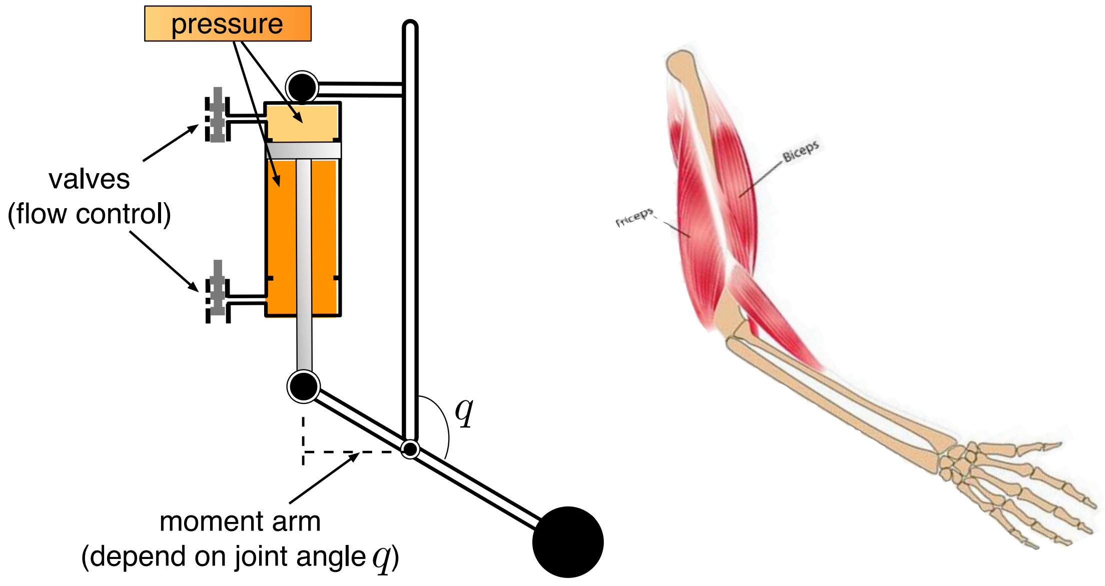


Figure 1.2: Pneumatic actuator and muscle for robot and human arms respectively.

whose dynamics make the entire system 3rd-order. Fig. 1.3 compares the response of air pressure and calcium concentration given a step command; The third order dynamics effectively introduce a low-pass filter between command signals and forces, with similar time-constants for muscles and pneumatic cylinders; (iv) since the actuators are often linear they can be mounted in a way reminiscent of muscle attachment to the skeleton, resulting in moment arms which vary with joint angle; (v) the high force output makes gears and other amplification mechanisms unnecessary, which in turn results in uniquely compliant systems capable of dynamic interactions with the environment. It can be argued that some of these properties are unnecessary complications that could be avoided by using stiff electrical actuators. However, if we are serious about understanding the principles of biological control and replicating those principles in synthetic systems, it would be a mistake to ignore the fact that biological control has evolved in the context of the musculo-skeletal plant and is profoundly shaped by the properties of this plant.

The valve-cylinder connection on Diego-San humanoid robot uses a unique

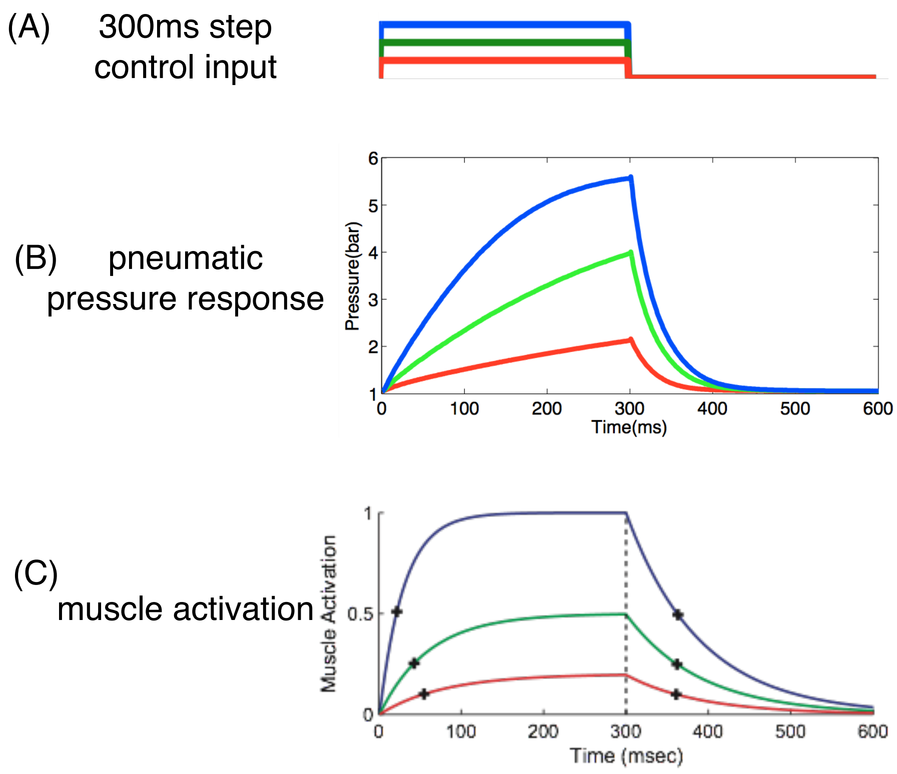


Figure 1.3: Given a step input as in panel (a), the temporal response of pneumatic pressure in panel (b) and muscle calcium concentration in panel (c)

design. Each pneumatic cylinder has two outlets. Each outlet is connected to a valve adjusting the pressure. In Chapter 4, we propose a novel pneumatic dynamic model of our new design and identified model parameters from real data.

1.1.2 Sensors

As in the human body, the sensors on a robot enable it to perceive its surrounding environment (exteroception) as well as the state of its own body (proprioception). The sensors on a humanoid robot are typically tailored to approximate the capability of the human sensory system. In humans, the proprioceptive sense is composed of information from sensory neurons located in the inner ear (motion and orientation) and in the stretch receptors located in the muscles and the joint-supporting ligaments (stance). The corresponding sensors in humanoids robots are inertial measurement units, actuator-length/joint-angle sensor and actuator force sensors. On Diego-San, joint angle information is provided by linear (on the actuator) and rotary (on the joint) potentiometers; joint forces are measured by pressure sensors on each chamber of the pneumatic actuators. While these proprioceptive sensors measure the state of the robot, other sensors are responsible for sensing the environment, including video cameras in each eye, and microphones in each ear.

The information provided by these sensors generally needs to be further processed to be useful. For example, a facial expression recognition system can use the information provided by the cameras to extract emotional status of people the robot interact with. The position, velocity and acceleration of the different parts of the robot can be calculated from the history of joint angles using the equations of kinematics and dynamics.

1.1.3 Articulated Body Kinematics and Dynamics

An articulated body is a collection of rigid bodies interconnected by joints. The kinematics and dynamics of articulated bodies study relationship between motion of these bodies and forces applied on them. Kinematics is the branch of classical mechanics that describes the motion of the bodies under constraints, such as joints and contacts. Dynamics refers to the branch of classical mechanics that focuses on the relationship between forces acting on the bodies (friction, gravity, contact forces, actuator forces, Coriolis forces), the inertial properties of the bodies (the moment of inertia matrix), and the resulting motion. The computational complexity of kinematics and dynamics greatly depends on the structure of the robot. In general, the complexity grows quadratically in the number of bodies since one needs to compute the forces between each pair of the bodies. Fortunately, the way these bodies (links) connect on a humanoid robot fits a special class called "tree structure", whose kinematics and dynamics can be calculated efficiently in a recursive manner [71]. Being able to scale up with the number of degree of freedom (essentially the number of bodies for tree structures) is crucial for high degree-of-freedom robots.

Consider a humanoid robot of n joints. The robot's kinematics describes the relationship between the position of a point $x \in \mathbb{R}^3$ on the robot in Cartesian coordinate system and generalized joint angles $q \in \mathbb{R}^n$. The relationship can be further divided into forward and inverse kinematics. Inverse kinematics refers to the problem of inferring the position and orientation of the robot links, from the joint angles

$$x = h_x(q); \tag{1.1}$$

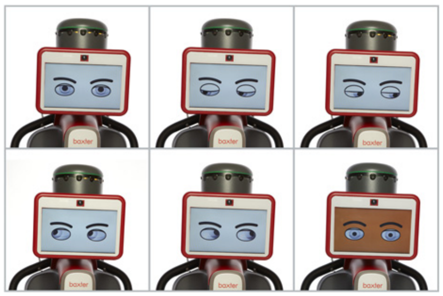
the latter is about finding a feasible configuration of joint angles that move the point x to a given goal position x_g

$$q = h_x^{-1}(x_g). \tag{1.2}$$

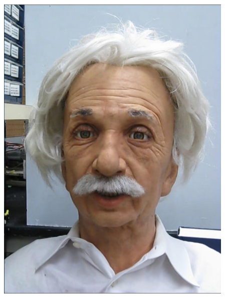
Inverse kinematics is typically solved by numerical optimization minimizing the distance between the given point and target position. The optimization is usually relies on the Jacobian of x with respect to joint angles,

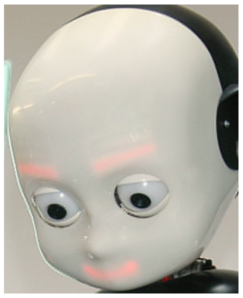
$$J_x = \frac{\partial h_x(q)}{\partial q} \in \mathbb{R}^{3\times n}. \tag{1.3}$$

Typically, under one-to-one joint-sensor setting, a joint angle q can be inferred directly from sensor measurement, such as inferring joint angle from the corresponding potentiometer. However, in Diego-San and other complex humanoid robots


(a) Baxter

(b) Diego-San


(c) Einstein


(d) iCub

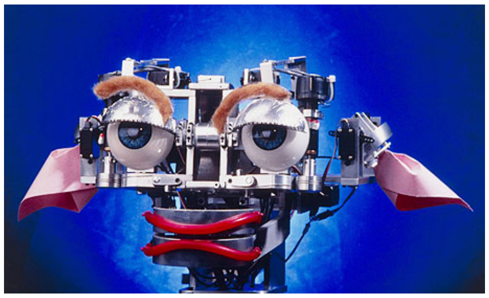

(e) Kismet

Figure 1.4: Faces in Humanoids.

a sensor measurement may depend on multiple joint angles, such as the length measurement of a tendon/actuator span over multiple joints. In addition the relationship between joint angles and sensor measurements may not be one-to-one. In this case, more sophisticated inference is needed (see Chapter 2).

Dynamics in humanoids is typically formulated in joint angle space with forces expressed as torques applied on the joints. These forces/torques include actuator torque $\tau \in \mathbb{R}^n$, Coriolis (virtual) torques $C \in \mathbb{R}^{n\times n}$, Coulomb friction $\tau^c \in \mathbb{R}^n$, viscosity $\tau^v \in \mathbb{R}^n$ and gravity induced torques $\tau^g \in \mathbb{R}^n$. Their relationship can be described by the joint-space robot equation :

$$M\ddot{q} + C(q,\dot{q})\dot{q} = \tau + \tau^c + \tau^v(\dot{q}) + \tau^g(q), \tag{1.4}$$

where $M \in \mathbb{R}^{n\times n}$ is the robot inertial matrix.

1.1.4 Deformable Bodies – Face

Facial expression plays an important role in human non-verbal communication. Robots' ability to produce and perceive facial expression may help improve the human-robot interaction. For this reason many modern humanoid robots have human-like faces and facial expressions. Figure 1.4 show a small collection of them.

Some robot faces are very realistic to the extent it is hard to judge whether they are human or robot faces. However, building such realistic faces is a complicated task requiring special material and dedicated plastic skills. In addition, a slightly imperfect realistic face may produce adversarial so effect that it is perceived as creepier than a non-realistic face. The phenomena is called the "uncanny valley" effect. Therefore, many robot designers choose to stay on the safer side of the uncanny valley employing non-realistic faces, such as movable mechanical parts (Kismet), programmable LED array (iCub) or even a display panel (Baxter). Despite the fact of these faces are not human-like, they still serve good purpose of communication.

Diego-San has a complex realistic face modeled after a child. The face is made of a special rubber-like material designed to deform in a manner similar to the way the facial tissue deforms in humans to produce facial expressions. Diego-San has 18 electric servo motors mounted behind the face. The motors are connected to the facial tissue via tendons. The deformations produced by these tendons produce human-like facial expressions. Precise control of these servos is crucial to make realistic facial expressions, which are typically manually tuned. However, this manual tuning is a very time-consuming task. In Chapter 5, we develop an automatic tuning approach to expedite the tuning process.

1.2 Machine Learning Approaches to Robot Control

Robot control refers to the problem of how to send control signals to the robot actuators so as to accomplish goals. For example the goal may be for the

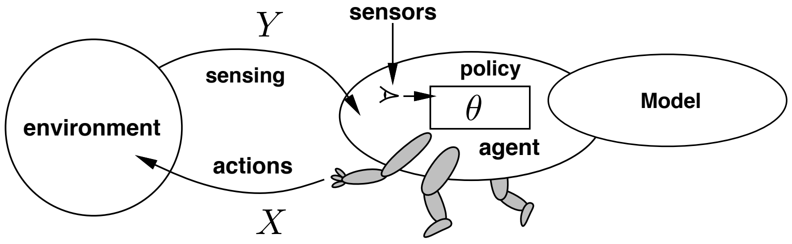


Figure 1.5: The optimal control algorithm utilizes sensor inputs, plans for actions that gathers the optimal rewards

robot to grab a distant object, or to kick a ball as high as possible, or to make a facial expressions that humans perceive as joyful. Stochastic optimal control provides a useful formalism and a set of powerful mathematical tools to frame the robot control problem. In optimal control the goal is expressed as a scalar function of the trajectories of the different robot parts and the relevant objects in the environment. This scalar function is typically called the reward if the objective is to maximize it, or the cost, if the objective is to minimize it. For example, if the goal is to reach for an object as quickly as possible, we could use as the reward function the negative Euclidean distance integrated over time, between the robot's right hand and the target object. Stochastic optimal control provides a range of methods for finding exact solutions to some control problems, namely problems with linear dynamics, Gaussian sensor noise, and quadratic cost functions. Unfortunately most robot control problems do not satisfy these assumptions and thus approximate methods are required. One class of methods that is becoming popular in recent years originated from the machine learning literature, and are known as Reinforcement Learning (RL) methods.

Figure 1.5 illustrates a typical Reinforcement Learning cycle, illustrating the relationship between action, sensing, reward and learning. The robot has a control policy parameterize by a vector θ. The policy maps the history of sensor information into moment to moment signals sent to the robot actuators. As a consequence of its actions the robot gets a scalar reward value. The RL algorithm then fine tunes the policy parameter based on the observed reward. While RL is an expanding field, the currently the available RL algorithms can be divided into two categories: Model Free and Model Based.

1.2.1 Model-Free Reinforcement Learning

Model-Free RL represents a class of RL algorithms that do not require a model of the robot dynamics. These algorithms operate exclusively on the raw actuator control, sensor values, and observed rewards. Typically the reward is a real-valued function of the sensor values. For example, the proximity between the robot's hand and target is an ideal reward for a reaching task which can be measured directly with a proximity sensor.

The simplicity makes model-free RL algorithms applicable to a wide range of problems, including problems that involve interaction with humans. Unfortunately, also because of their simplicity, model-free RL algorithms often exhibit extremely slow learning in complex problems. One way to speed this up is to create virtual experiences by having a model of the environment and running these algorithms inside the model.

1.2.2 Model Based Reinforcement Learning

In model-based RL, as in the basic model-free approach, the primary goal of learning is the improvement of a behavioral policy in order to maximize a numerical reward signal. However, the approach differs from model-free RL in that the agent simultaneously attempts to learn a model of its environment. Having such as model is useful: it allows the agent to predict the consequences of actions before they are taken, allowing the agent to generate virtual experience, as well as mental search through a problem space to locate an efficient solution. Thus, model-based RL integrates both learning on the basis of past experiences and planning future actions. Further, a model of the environment is not directly tied to the task one is currently performing. For example, consider a humanoid learning to draw a circle on the whiteboard. While performing this task, the robot could acquire the kinematics and dynamics model of its arm. If the task goals change, for example, from drawing a circle to toss a ball, the previously learned policy (the sequences of joint angles to draw the circle) is of no help, but the learned arm model could still aid the robot in achieving its new goal. Finally, in addition to learning the model from experiences, it is also possible to directly construct an analytical model from prior knowledge of the environment. For example, one can build a geometric kinematics model directly as the prior knowledge of the robot.

The downside of the model based approach is that it increases the complexity of the learning algorithm. In addition, a poorly designed model would provide inaccurate prediction and may mislead the learning process.

1.3 Contributions

1.3.1 Models and Simulators for Humanoid Robots

Simulators are crucial for robot control development, which allow researchers to try new algorithms before running on physical robots. Simulation eliminates the risk of debugging code on physical robots which may be harmful. It also allows us to execute algorithms in super-realtime which in turn speeds up the development. In addition, simulators also open up the possibility of model-predictive control.

We collaborated with Emanuel Todorov to develop a customized simulator named Mujoco [80]. The kinematic model and rigid body dynamic model parameters were extracted from CAD model of Diego-San. We developed new kinematic identification approach, STAC in Chapter 2, to identify the sensor parameters. We built a new pneumatic model and identified its parameters for all 38 actuators for the valve-cylinder actuator dynamics in Chapter 4.

Further, we find that a simple parametric model for rigid body dynamics cannot capture the actual dynamics observed. There is unmodeled noise created by cables and tubes going across joints. Ideally, this can be solved by data-driven non-parametric regression. However, the high degree of freedom suggests that regression in high dimensional space would require huge amount of data to perform well. To take advantage of both parametric and non-parametric approach, we develop a semi-parametric Gaussian process approach, SGP, described in Chapter 3.

1.3.2 Learning to Control in Humanoid Robots

Robot Facial Expression: In Chapter 5, we tackle the problem of controlling 18 servos on Diego-San's face to produce facial expressions. The problem is difficult as the specification of facial expressions is hard to obtain. To provide a quantitative reward signal, we make use of computer expression recognition toolbox (CERT) to evaluate the goodness of an expression posed by the robot during random exploration. CERT essentially learns the visual definition of expressions from thousands of examples of human facial expressions. With this visual feedback, the robot learns to pose 6 basic emotion expression as well as individual action units from facial action coding scheme (FACS). Prior to this automatic learning approach, an expert had to spend weeks to manually tune these servo motors for each facial expression. Our algorithm reduces the effort to just a few minutes.

Diffusion Network Adaptation for Control: In Chapter 6, we developed a new reinforcement learning algorithm, named Diffusion Network Adaptation (DNA), for robot control problems. We tested the DNA algorithm on several model-free learning tasks in simulation, achieving state-of-the-art performance comparable to other leading methods. Next, we evaluated the DNA on real-world humanoid learning tasks: kick a ball and reaching. In both tasks, our robot learned effective policies to achieve the task goals.

1.3.3 Developmental Psychology

Learning The Correspondence Between Facial Muscle Activation And Facial Appearance: It is unknown how humans learn to control their own facial muscles to generate expressions to match those they observe. Such facial gesture imitation requires the cognitive system to equate the seen-but-unfelt with the felt-but-unseen. Researchers are uncertain of the source of correspondence between visual and proprioceptive representation of facial actions. Rival accounts propose that this is either innate or learned. In Chapter 5, we demonstrate that learning such correspondence is actually possible with very small amount of experience, which verifies the feasibility for the "learned" hypothesis. Interestingly, human study [17] also suggests that visual feedback is necessary for facial expression imitation learning.

Learning To Develop Symbolic Behavior As part of this thesis we studied how infants learn to reach for objects in the presence of a caretaker. We first collected a dataset of infant-mother interaction in the presence of objects. We found that infants generate fast and large limb movements when the toy is out of the infant's arm reaching distance. On the other hand, the infants exhibit slow, small and precise, movements when they have the toy in hand. We hypothesize that the large movements serve as social gestures for mothers to pass the toy. This is known in the infant development literature as "protoimperative" symbols. We studied whether Diego-San could learn to develop this type of proto-symbolic behavior while interacting with humans. The results are presented in more detail in Chapter 6.

1.3.4 Software

We utilized many off-the-shelf software packages developed by the robotics community, such as Robotic Operating System (ROS) [61] and OpenCV [11]. In return, we contributed our non-robot-specific software back to the community. These software packages are available at `https://code.google.com/p/mplab-ros-pkg/`, where we received more than 70 downloads in the past 6 months.

Matlab-ROS Bridge: *Matlab* is one of the most popular programming languages in engineering for its fast-prototyping and visualization capability. ROS is the de-facto standard for integrating heterogeneous robot sensors and actuators across multiple computers. We develop Matlab-ROS bridge to glue the best of these two worlds together, making it possible to turn research ideas into complicated real-world robotic experiment in short time.

ROS-National Instrument Driver (NIDAQ-IO): Digital analogue converters (DACs) and analogue digital converters (ADCs) are essential interfacing hardware for computer-based control using analogue sensors/actuators such as potentiometer and electrical motors. *National Instrument* is one of the major brand. However, their hardware were unsupported in ROS. We develop the driver to fill in the gap.

Part I

Modeling Humanoid Robots

Chapter 2

STAC: Simultaneous Tracking and Calibration

Abstract: System identification is an essential first step in robotic control. Here we focus on the calibration of kinematic sensors, such as joint angle potentiometers, tendon/actuator extension sensors and motion capture markers, on complex humanoid robots.

Manual calibration with protractors and rulers does not scale to complex humanoids like the ones studied here. Classic automatic approaches cross-calibrate multiple sensor systems on the same robot by exploiting their redundancy. However, these approaches make the strong assumption that the observed joint angles are functions of the sensor measurements plus observation noise. This assumption is too restrictive on modern humanoids where linear actuators and tendons span multiple joints.

Here we formulate the calibration problem as a Bayesian inference process on a generative model where hidden joint-angles generate sensor observations. A novel alternating optimization approach is developed to simultaneously track space-time joint angles and calibrate parameters (STAC). Explicit estimation of joint angles makes it possible to calibrate sensors that otherwise cannot be handled by classical approaches, such as tendons wrapping on complicated surfaces and spanning multiple joints. We evaluate STAC to calibrate joint potentiometer, tendon length sensor and motion capture marker positions, on a 38-DoF humanoid

robot with 24 optical markers, and a 24 DoF tendon driven hand with 12 markers. We show that STAC can be applied to problems that cannot be handled with classical approaches. In addition we show that for simpler problems STAC is more robust than classical approaches and other probabilistic approaches such as the Extended Kalman Filter.

2.1 Introduction

System identification (ID) is an essential first step in robotic control, and can be divided into kinematic and dynamic system identification. Dynamic ID deals with quantities which emerge when there is movement, like moments-of-inertia and friction coefficients. Kinematic ID deals with parameters which are relevant even when the velocity is zero, i.e. geometric properties. For example, a pick-and-place robot with an inaccurate kinematics model will do a poor job, regardless of how slow it moves. In this paper, we focus on kinematic ID, including calibrating joint angle and cylinder extension sensors which are typically measured by potentiometers, magnetic or optical encoders. Some of these sensors are directly mounted on the joints; others are connected through a transmission mechanism, such as cranks, cables, tendons or gears. These mechanisms while useful, may change their dynamic range, linearity, and even accuracy of the sensors, all of which essentially contribute additional parameters to be identified.

Manual calibration approaches usually rely on ground-truth joint angle measurements using protractors, and the corresponding joint angle sensor readings. Regression models are then used to learn a function that maps sensor reading into joint angle estimates. We found this approach to be inaccurate and inefficient when applied to complex robot platforms. Accurate measurements are hard to obtain with a protractor, our robot's arms are covered with tubes and their surface is uneven; making it difficult to align a protractor to a joint. Empirically, measurement precision can be as bad as 5 degrees. The approach is also very time-consuming especially on a large number of joints. One of the humanoid platforms studied in this paper has 38 joints and many of them come with non-linear transmission mechanisms which require multiple measurements over different angles to fully identify the underlying parameters. Furthermore, calibration values can change after each repair or intensive use. The development of a fully automatic calibration system is necessary to keep the robot fully functional.

More sophisticated kinematic ID approaches cross-calibrate multiple sensor systems on the same robot by exploiting their redundancy [32]. For example, consider a robot with potentiometers measuring joint angle and motion capture markers attached to some of the bodies. Assuming both sensors are calibrated, one can infer the pose of the robot using either system, thus the redundancy. However, before the calibration, neither of the systems is accurate. Both come with unknown parameters: the gain and bias for potentiometer and the positions of the markers. To calibrate these parameters, one first collects synchronized measurement from both sensors for multiple frames, and then tries to find the optimal values, such that the two sensor systems agree with each other.

However current kinematic ID approaches assume that the observed joint angles are a function of the sensor measurement plus some observation noise. This assumption raises two issues: First, the assumption would only work for simple one-joint-to-one-sensor sensor types as in Fig. 2.3abc. Pose sensor measurement on modern humanoids may depend on multiple joint angles. For example, modern dexterous hands are driven by tendons where the length of a tendon is a linear function of multiple joint angles (Fig. 2.3e). For hydraulic or pneumatic systems, it is common to apply linear actuators to multiple DoF joints such as the 2-DoF rotational gimbal as in Fig. 2.3d and the Stewart platform [25]. In these cases, it is generally not trivial to write joint angle as a function of the sensor measurement. More importantly, in some cases the noise-free mapping between sensors and joint angles may be a multi-valued function. In this work, we formulate the relationship between sensors and joint angles as a Bayesian generative model in which sensor measurements are noisy observations generated from joint angles. This contrasts with the classical regression-based approach in which the joint angles are treated as noisy observations generated by noiseless sensor readings. This approach lets us calibrate robots in which the relationship between joint angles and sensors is very complex and cases in which the observed angles are not a single-valued function of the sensor readings.

2.2 Multi-sensor Parameter and Joint Angle Estimation

Consider a tree-structured robot of n-joints with a known (skeletal) kinematics model, including link lengths, joint positions and types. The robot is equipped with different types of pose sensors which measure some quantities as *parameterized* functions of one or more joint angles and derived quantities, such as body positions or orientations. The goal is to identify these (fixed) sensor parameters from multiple synchronized measurements from the multiple sensors.

2.2.1 Classical Methods

The past decades have seen the development of kinematic calibration methods that do not require ground truth knowledge of joint angles [32]. Let the forward kinematics function h of a robot arm be as follows

$$\hat{\mathbf{x}} = h(q, \theta) \tag{2.1}$$

where $\hat{\mathbf{x}}$ is the end-effector pose, $q \in R^n$ are the joint angles and θ are parameters to be identified including potentiometer gains θ_{gain} and biases θ_{bias}. Typically, potentiometer readings are linear functions of the joint angles,

$$q_j = \theta_{gain,j} \cdot p_j + \theta_{bias,j} \tag{2.2}$$

where p_j is the reading from the corresponding potentiometer. Therefore we can rewrite the (2.1) as

$$\hat{\mathbf{x}} = h(\theta_{gain,j} p_j + \theta_{bias,j}, \theta) = h(p_j, \theta). \tag{2.3}$$

Under this framework the system ID problem can be formulated as a non-linear regression problem. Given a sufficient number synchronized measurement of $(\mathbf{x}_t, p_t)$,

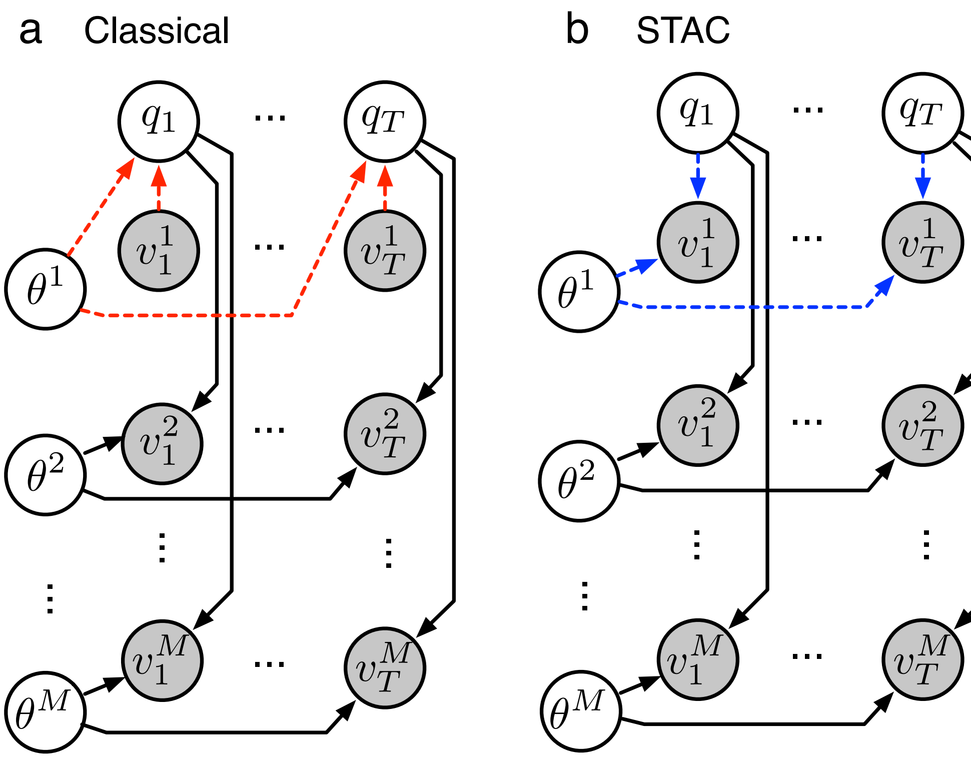


Figure 2.1: The graphical model of the (a) classical approach (b) our approach. Joint angles q and sensor parameters θ are hidden and sensor measurements v are observed (shaded).

the goal is to find a parameter vector θ that minimizes a sum of squared errors cost function

$$L(\theta; \mathbf{x}_{1:T}, p_{1:T}) = \sum_{t=1}^{T} ||\hat{\mathbf{x}}_t - \mathbf{x}_t||_2^2 = \sum_{t=1}^{T} ||h(p_t, \theta) - \mathbf{x}_t||_2^2. \tag{2.4}$$

Note that explicit estimation of joint angle q is unnecessary. Such formulation is convenient. However regression approaches rely on the assumption that the observed joint angles q are a function of the sensor readings plus some observation noise. This assumption is often violated in complex, biologically-inspired humanoid robots.

2.2.2 Generative Model

While regression models assume that joint angles are a function of sensor readings, here we use the much weaker assumption that sensor reading are a function of joint angles. Figure 2.1 illustrates the probabilistic graphical models corresponding to the classical regression based approach and to the approach we propose (STAC). On the top row, each unshaded node q_t represents the *hidden* joint angles in snapshot t while on the left column, each unshaded node θ^i represents *unknown* parameters of sensor i. In the middle, the *observed* sensor measurements are organized into the array of shaded nodes, where the rows are measurements from the same sensor and columns are measurements in the same snapshot. The measurement from sensor i in snapshot t is then denoted as v_t^i. The left panel (a) shows the model for classical regression based approach. Joint angles $q_{1:T}$ are generated from potentiometer reading $v_{1:T}^1$ and parameters θ^1 as shown in dashed arrows. For STAC on right panel (b), we re-assign these arrows that all the sensor measurement (whether it is from potentiometer or not) are generated from joint angles and sensor parameters.

Standard graphical model machinery can be used to compute the negative log likelihood function used by STAC

$$LL(q_{1:T}, \boldsymbol{\theta}) = -\sum_{m=1}^{M} \sum_{t=1}^{T} \log p(v_t^m | q_t; \theta^m). \tag{2.5}$$

where v represents sensor observations, $\boldsymbol{\theta}$ represents kinematic parameters and $q_{1:T}$ represents joint angles.

2.2.3 Alternating Descent Optimization

The standard maximum likelihood approach to estimate θ is to directly minimize $LL(q, \theta)$. However, direct optimization over all the parameters and joint angles is difficult due to large number of parameters. Consider a 38-DoF humanoid with 2 parameters for each joint and a collection of 100 snapshots would easily amount to 3876 parameters!

We found that the optimization process could be greatly accelerated by using an approach that alternated between two phases: a p-phase that estimates sensor parameters while keeping joint angle fixed and a q-phase that estimates joint angles using the updated sensor parameters. The optimization in each phase can be further divided into multiple simpler and parallelizable sub-problems taking advantage of the special structure of the graphical model. The idea is justified by the following two key observations:

- joint angles q_t's of different time frames are conditionally independent from each other when θ^i's are known;
- on the other hand, the parameter θ^i's of different sensors are conditionally independent from each other when joint angles are known.

Using this property we can re-write the maximum likelihood problem as

$$\begin{aligned}&\min_{\boldsymbol{\theta},q_{1:T}} \sum_{m=1}^{M}\sum_{t=1}^{T} \log p(v_t^m|q_t;\theta^m)\\ &= \sum_{m} \min_{\theta^m}\left(\sum_{t=1}^{T} \min_{q_t} \log p(v_t^m|q_t;\theta^m)\right). \end{aligned} \tag{2.6}$$

Notice that the maximization subproblems inside the summations can be optimized independently. In many cases, the subproblems solving each q_i or θ_i are simple enough to have closed-form solutions. Even when numerical optimization is necessary, these subproblems can be optimized in parallel over multiple processors.

2.2.4 Sensor Observation Noise Model

Typically, sensor measurement is assumed to be contaminated by additive zero-mean σ_v^2 variance Gaussian noise,

$$P(v_t^m|q_t;\theta) = \mathcal{N}\left(v|\hat{v}^m(q_t;\theta),\sigma_v^2\right). \tag{2.7}$$

Then, the maximum likelihood problem is equivalent to a least squares problem. In other words, the likelihood function can be written as

$$LL(q,\boldsymbol{\theta}) = \frac{1}{2} r(\boldsymbol{\theta},q)^T r(\boldsymbol{\theta},q) + \frac{TM}{2}\log 2\pi \tag{2.8}$$

where the residual vector is

$$r(\boldsymbol{\theta}, q) = \begin{bmatrix} r(\boldsymbol{\theta}, q_1) \\ r(\boldsymbol{\theta}, q_2) \\ \vdots \\ r(\boldsymbol{\theta}, q_T) \end{bmatrix}, \; r(\boldsymbol{\theta}, q_t) = \frac{1}{\sigma_v} \begin{bmatrix} v_t^1 - \hat{v}_t^1(\boldsymbol{\theta}, q_t) \\ v_t^2 - \hat{v}_t^2(\boldsymbol{\theta}, q_t) \\ \vdots \\ v_t^M - \hat{v}_t^M(\boldsymbol{\theta}, q_t) \end{bmatrix}. \tag{2.9}$$

Here we put the variance σ_v at sensor level as the same type of sensor typically share similar noise variance. One can certainly use a per-sensor variance option when necessary. There are off-the-shelf tools solving non-linear least squares problem, such as Gauss-Newton or Levenberg-Marquardt approaches.

2.3 Case Study: Calibrating Joint/Tendon Potentiometer Against Motion Capture

Motion capture systems, are becoming standard measuring tools in robotic labs for various control and identification tasks [2,41,47]. In this section, we study the case of using motion capture system to calibrate other sensors.

There are two type of pose sensors on the robot: rotary potentiometers for 1-DoF rotary joints and linear potentiometers for 2-DoF Gimbal joints. To identify the potentiometer parameters, we attach M motion capture markers to some links of the robot as auxiliary sensors. In total, the sensor parameters to be identified are potentiometer gains and bias, the translation and rotation of motion capture coordinate frame from the robot baselink frame and the marker positions on the links, or

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_{gain} & \theta_{bias} & R & T_0 & d_{1:M} \end{bmatrix}. \tag{2.10}$$

Next, we describe how the observations of the various types of sensors are generated and how to solve for the parameters analytically in the "p-phase". For notational clarity, we further split the observation variable $v : \{\mathbf{x}, p\}$ into $\mathbf{x}$ for motion capture markers and p for potentiometers.

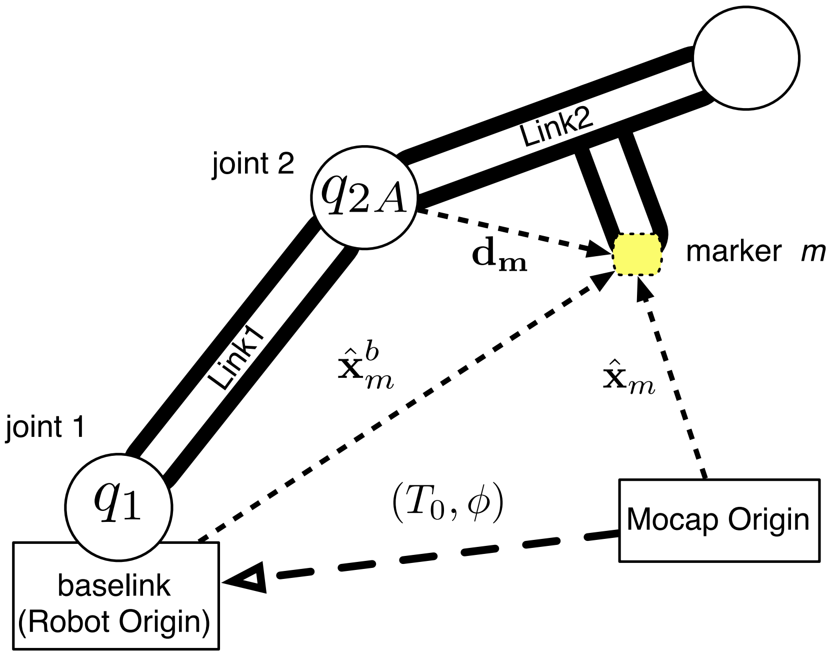


Figure 2.2: markers on kinematics chain

2.3.1 Motion Capture Markers

Figure 2.2 shows the spatial relationship of a marker m and the parent link (link 2) it is attached to. Let d_m be the unknown marker local position in the parent link frame. Then the marker position in the robot (baselink) frame is,

$$\hat{\mathbf{x}}_m^b(d_m, q_t) = h_r(q_t)d_m + h_p(q_t), \tag{2.11}$$

where $h(\cdot)$ is the forward kinematics function that calculates the position (h_p) and orientation (h_r) of the parent link, to transform d_m into the baselink frame.

The motion capture system measures 3-dimensional position of the markers $x_{m,t} \in \mathbb{R}^3, m = 1, 2, \ldots, M$ in the motion capture frame. We denote the transformation from baselink- to motion-capture-frame by rotation matrix $R \in \mathbb{R}^{3\times3}$ and translation $T_0 \in \mathbb{R}^3$. Then the prediction of marker position in the motion capture frame is

$$\hat{\mathbf{x}}_m(R, T_0, d_m, q_t) = R\hat{\mathbf{x}}_m^b(d_m, q_t) + T_0. \tag{2.12}$$

During the "p-phase" of alternating descent, we solve for the parameters R, T_0, d_m while fixing the joint angles $q_{1:T}$. The transformation parameters (R, T_0), which affect all markers at all times, can be solved for using Procrustes analysis of rigid body motion problem [74]: First we calculate the center of the markers in each coordinate system,

$$\bar{\mathbf{x}}^b = \frac{1}{MT}\sum_{m=1}^{M}\sum_{t=1}^{T}\hat{\mathbf{x}}^b_{m,t}, \quad \bar{\mathbf{x}} = \frac{1}{MT}\sum_{m=1}^{M}\sum_{t=1}^{T}\mathbf{x}_{m,t}. \tag{2.13}$$

Next, the rotation matrix can be obtained through singular-value-decomposition of the "covariance matrix",

$$\sum_{m,t}(\hat{\mathbf{x}}^b_{m,t} - \bar{\mathbf{x}}^b)(\mathbf{x}_{m,t} - \bar{\mathbf{x}})^T \overset{\text{SVD}}{=} U\Sigma V^T. \tag{2.14}$$

Then,

$$R = \text{sign}(\det(\Sigma))VU^T \tag{2.15}$$

$$T_0 = \bar{\mathbf{x}} - R\bar{\mathbf{x}}^b \tag{2.16}$$

Finally, the local position of the markers in the corresponding link can be identified by taking the average of observed marker position in the link coordinate (2.12) :

$$\bar{d}_m = \frac{1}{T}\sum_{t=1}^{T} h_r(q_t)^{-1}\underbrace{\left(R^{-1}(\mathbf{x}_{m,t} - T_0) - h_p(q_t)\right)}_{\mathbf{x}^b_{m,t}} \tag{2.17}$$

2.3.2 Potentiometers

There are several popular types of potentiometer mountings as shown in Fig.2.3. Although potentiometers are typically linear in the rotation angle or linear in the displacement, the transmission mechanism can be either linear or non-linear.

For linear (fixed gearing or direct driving) transmission, such as a hinge joint with one rotary potentiometer (Fig.2.3a) or a sliding joint with a linear potentiometer (Fig.2.3b), the output voltage $\hat{p}$ is linear in the joint angle.

$$\hat{p}(\theta, q) = \theta_{\text{gain}}q + \theta_{\text{bias}} \tag{2.18}$$

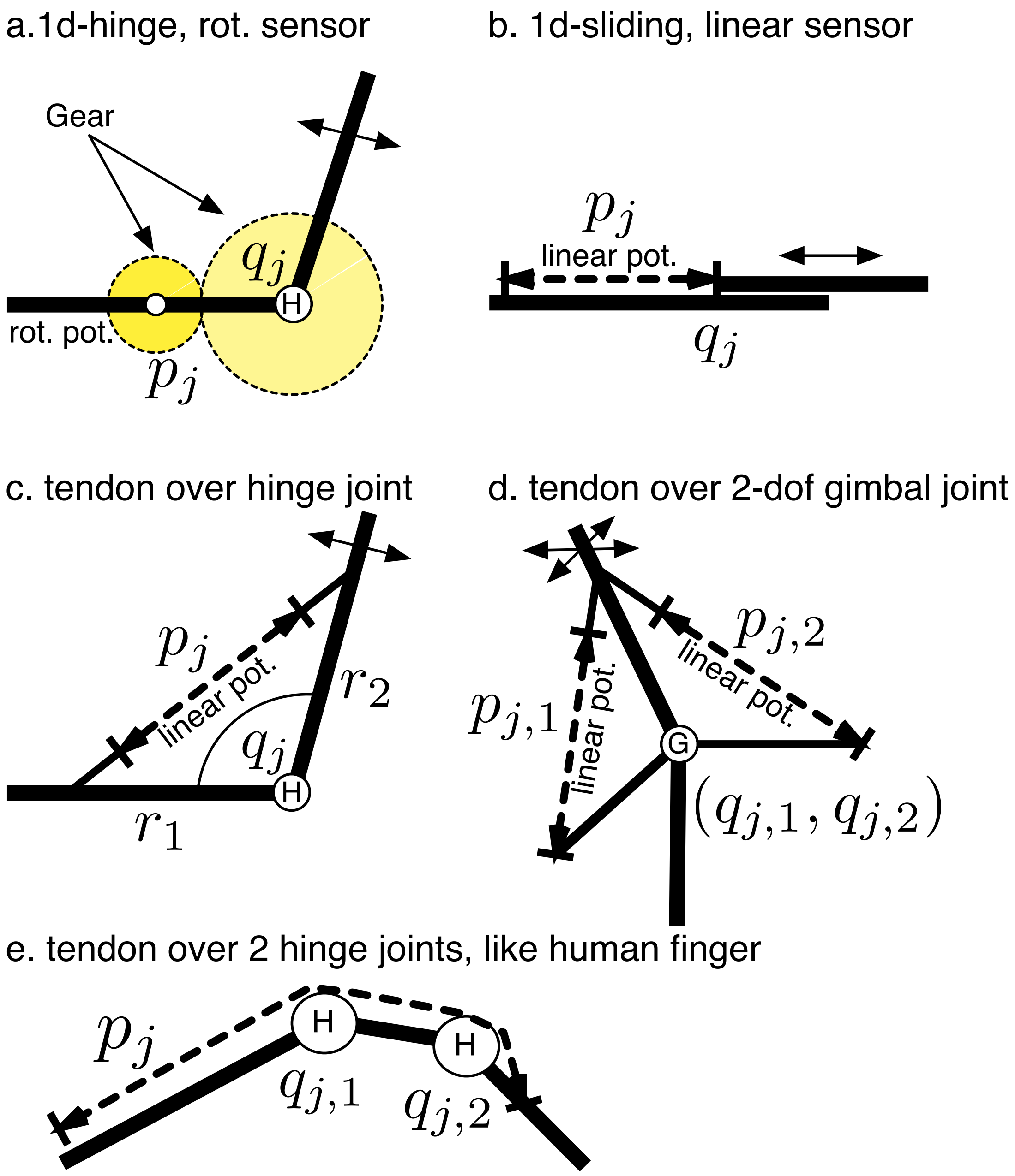


Figure 2.3: Exemplar joint types where q_j are generalized joint angle / displacement, p_j are sensor readings.

2.3.3 Tendons

Tendons are force transmission mechanism connecting two links. The length of a tendon can be used to determine the joint angles between the two links. Classical regression based approaches cannot handle this case because typically there are many joint angle combinations that yield the same tendon length i.e., the mapping from tendon lengths to joint angles is not a single-valued function. Figure 2.3e shows the tendon used in an anthropomorphic tendon-driven finger. Another type of tendon setup uses linear actuators/sensors on 2-DoF rotation joints. As it is not easy to attach rotary potentiometers on the 2DoF Gimbal structure, two linear potentiometers are attached across the two links on distinct pairs of points (Fig.2.3d) to measure the joint angles. The anchor points are available from the CAD model. For convenience, the measured voltage and calculated length of these potentiometers are referred to as p_1, p_2 and L_1, L_2 while the corresponding joint angles are q_1 and q_2. The predicted measurement is then

$$\hat{p}_1(\theta, q) = \theta_{\text{gain}} L_1(q_1, q_2) + \theta_{\text{bias}} \tag{2.19}$$

$$\hat{p}_2(\theta, q) = \theta_{\text{gain}} L_2(q_1, q_2) + \theta_{\text{bias}}, \tag{2.20}$$

Note that it is easy to calculate $L(\cdot)$ from q as part of the analytical forward kinematics routine but inferring q from L would require numerical inverse kinematics.

When solving for the potentiometer parameter given the joint angle q_t and the measured voltage p_t at snapshot t, whether the transmission is linear or non-linear, the equations (2.18)-(2.20) are always linear in θ_{gain} and θ_{bias}, and thus can be estimated using linear least squares methods.

2.3.4 Space-Time Joint Angles

Once the sensor parameters are updated, we turn to the “q-phase”: optimizing for the joint angles. The goal is to maximize the log-likelihood,

$$q_t^* = \max_{q_t} \frac{\sum_m ||\hat{x}_{mt} - x_{mt}||_2^2}{\sigma_x^2} + \frac{\sum_j ||\hat{p}_{jt} - p_{jt}||_2^2}{\sigma_p^2}. \tag{2.21}$$

All we need is the Jacobian:

$$\frac{\partial \hat{p}_j(\theta, q_t)}{\partial q_t} = \begin{cases} \theta_{gain} & \text{rotary pots.} \\ \theta_{gain}\frac{\partial L(q_t)}{\partial q_t} & \text{linear pots.} \end{cases} \tag{2.22}$$

$$\frac{\partial \hat{x}_m(\theta, q_t)}{\partial q_t} = \frac{\partial h_m(q_t; d_m)}{\partial q_t}, \tag{2.23}$$

where both $\frac{\partial L(q_t)}{\partial q_t}$ and $\frac{\partial h_m}{\partial q_t}$ are standard kinematics Jacobian available in almost every kinematics packages such as [80].

2.3.5 Dynamic Variance Adjustment

Distinct types of sensors produce residual of different dynamics range so the variances $\sigma_v^2 : \{\sigma_x^2, \sigma_p^2\}$ have to be adjusted to the right scale. We initialize these parameter to the corresponding sensors' dynamic range and re-estimate the variance in each iteration based on the residuals.

2.4 Related Work

We compare our formulation to two related approaches here and present the experimental comparison in the next Section.

2.4.1 Kalman Filter for Tracking and Calibration

The Extended Kalman filter (EKF) is an algorithm for tracking the state of a system with known dynamics and observation function from a noisy time series of observations. It has been applied to human skeleton tracking and kinematics identification [82]. Figure 2.4 gives an example how the EKF can be used to solve our tracking and calibration problem.

In this method, the state space consists of both joint angles q_t and parameters θ_t. Since the control sequence applied to the robot is assumed unknown, the dynamics equation contains only a drift term w with large variance Σ_q for

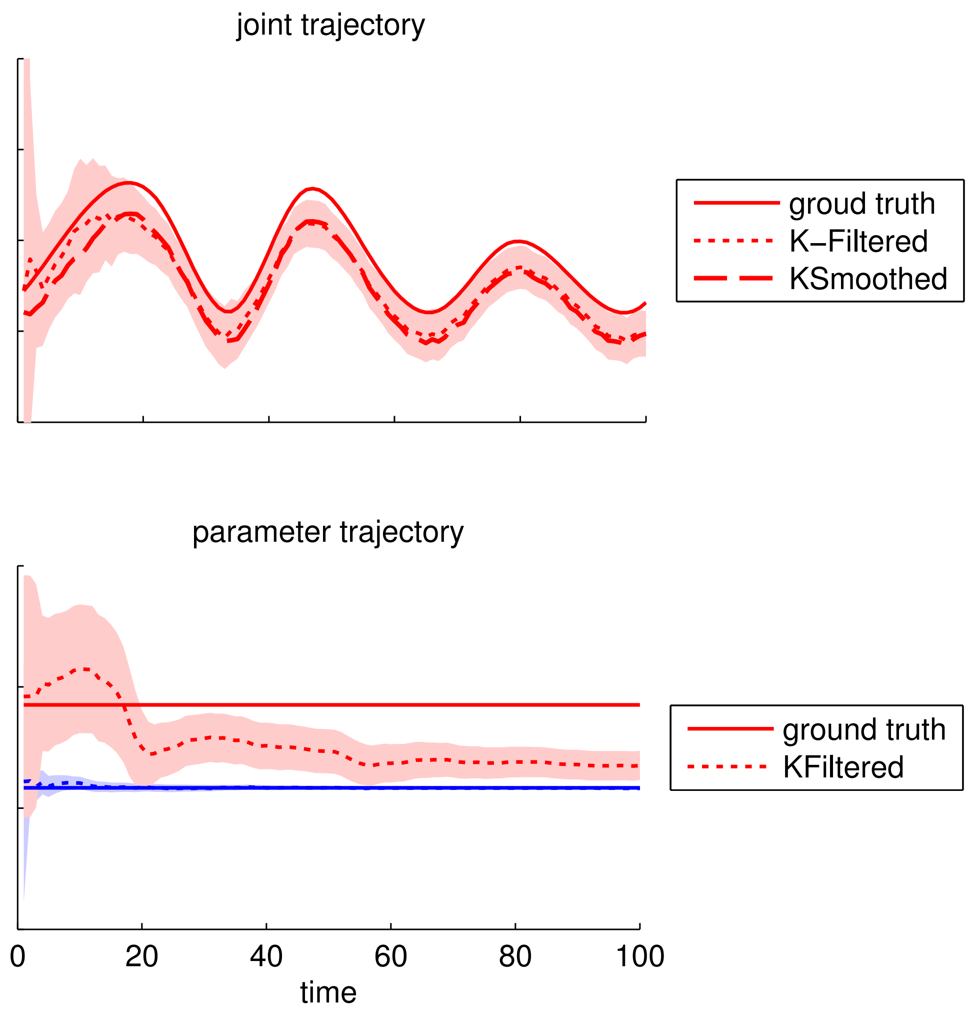


Figure 2.4: Extended Kalman Filter for simultaneous calibration and tracking. The variance (shaded band) of parameter shrinks more data is seen.

time-varying joint angles and zero-variance for fixed parameters,

$$\begin{bmatrix}\theta_{t+1}\\ q_{t+1}\end{bmatrix} = \begin{bmatrix}\theta_t\\ q_t\end{bmatrix} + w, \quad w \sim \mathcal{N}(\mathbf{0}, \begin{bmatrix}\mathbf{0} & \\ & \Sigma_q\end{bmatrix}) \tag{2.24}$$

The observation function is same as what we use in STAC, see Sec.2.3. For both potentiometers $\hat{p}$ and motion capture markers $\hat{x}$ with observation noise z

$$v_t = [\hat{p}(\theta_t, q_t); \hat{x}(\theta_t, q_t)] \tag{2.25}$$

$$v_{t+1} = v_t + z, \quad z \sim \mathcal{N}(0, \sigma_v^2). \tag{2.26}$$

The EKF updates require linearization v_t around current state which needs the Jacobian

$$\frac{\partial v_t}{\partial(\theta_t, q_t)} = \begin{bmatrix}\frac{\partial \hat{p}}{\partial \theta_t} & \frac{\partial \hat{p}}{\partial q_t}\\ \frac{\partial \hat{x}}{\partial \theta_t} & \frac{\partial \hat{x}}{\partial q_t}\end{bmatrix} \tag{2.27}$$

The derivatives with respect to q are in (2.22) and (2.23) and those with respect to θ are also analytical. In practice, because the linearized observation function is only valid at a local neighborhood around the current state, the EKF is prone to divergence without proper seeding of initial state.

Even with proper starting seed, once EKF loses track of the target due to observation noise at time t, the estimation will typically remain off after t. This is one of the major difference between STAC and EKF: while EKF tracks the parameters and space-time joint-angles sequentially in time, STAC jointly optimizes for them across time. Therefore, an erroneous estimate at one time frame would not propagate as in EKF. We will compare the robustness of EKF to STAC under various seeding and observation noise in Sec.2.6.2.

2.4.2 Classical Regression-Based Approaches

Classical approaches (Sec. 2.2.1) can be seen as a special case of STAC with zero joint potentiometer sensor noise $\sigma_p = 0$. In this way, the potentiometer term in the likelihood function (2.21) approaches infinity, effectively making it a constraint. Then the overall least squares problem can be simplified as

$$\begin{aligned} \max_{\theta,d} \quad & \textstyle\sum_t \max_{q_t} \sum_m ||\hat{\mathbf{x}}_{mt}(\boldsymbol{\theta}, q_t) - \mathbf{x}_{mt}||_2^2/\sigma_x^2 \\ \text{subject to} \quad & \hat{p}_{jt}(\boldsymbol{\theta}, q_t) = p_{jt} \end{aligned} \tag{2.28}$$

If the constraints are all about simple one-joint-to-one-potentiometer, such that the joint angles can be inferred from measurement analytically, we can re-write the constraints as $q_j = q_j(\mathbf{p}_j, \theta_j)$. Plugging q_j into the objective function in place of $q_j(\cdot)$, we obtain

$$\max_{\boldsymbol{\theta},d} \sum_t \sum_m ||\hat{\mathbf{x}}_{mt}(q(\hat{p}_t, \boldsymbol{\theta})) - \mathbf{x}_{mt}||_2^2, \tag{2.29}$$

which is identical to (2.4)

2.5 Experiment Setup

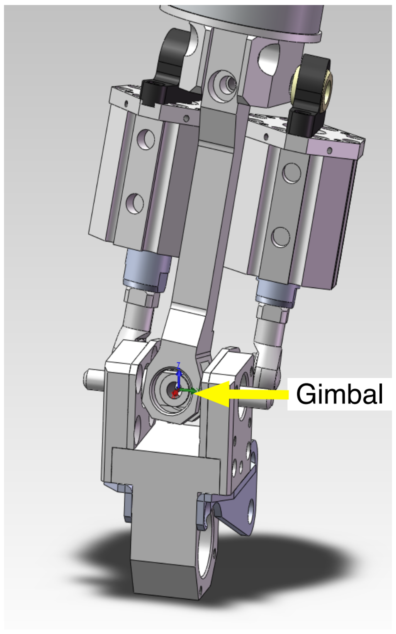
We performed experiments on a complex pneumatics-based humanoid robot [83] named "Diego San" as well as a dexterous tendon-driven hand [39].

2.5.1 Humanoid

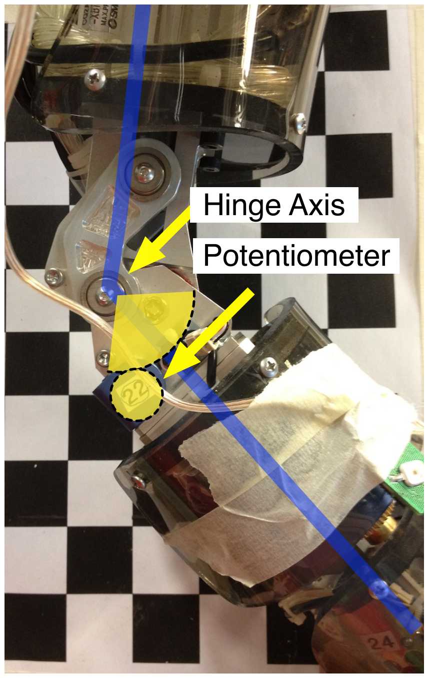
Figure 2.5c shows a picture of Diego San. It is a pneumatic humanoid with body parts proportional to that of a 1-year old human body. Among 38 joints, 4 are 2-DoF Gimbal joints (as in Fig. 2.3(d)) with two linear potentiometers measuring the length of the two pneumatic cylinders (*e.g.,* neck, Fig. 2.5a); the rest of the joints are hinge type (as in Fig. 2.3(a)) with gear transmitted rotary potentiometers (*e.g.,* elbow, Fig. 2.5b).

We attached 24 markers to every other link counting from the baselink taking advantage of the fact that the rotation axes of adjacent joints are mostly non-parallel. In this way, we were able to perform full-body sensor calibration without putting markers to every link. During motion capture, the robot was driven by a simple PID controller to move the joints randomly. Both marker positions and potentiometer readings were captured synchronously at 100Hz for 250 seconds. The traces are visualized in Fig.2.5d.

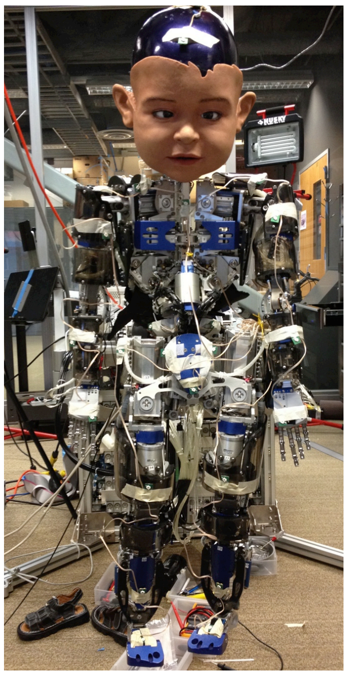
The kinematic model was extracted from the CAD file provided by the manufacturer (Kokoro Robotics), which includes link lengths, joint locations and orientations. The baselink of the robot is the waist, which was hung from a stable



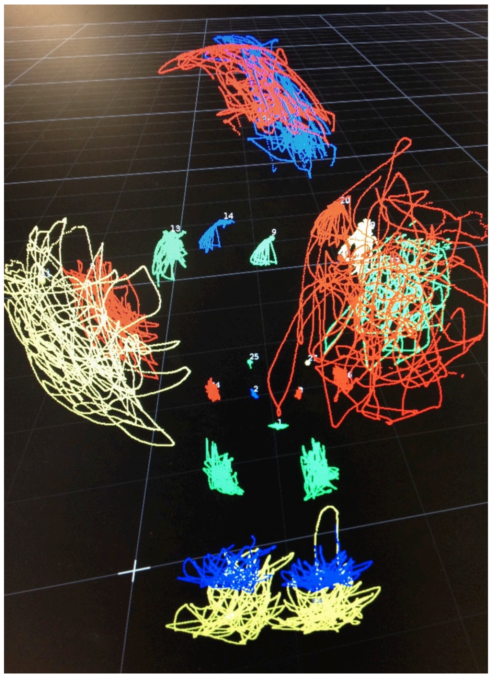
(a) 2-DoF Gimbal joint for Neck



(b) Hinge joint for elbow



(c) The robot



(d) Marker trajectories

Figure 2.5: Diego San – the humanoid robot used in this study.

crane. Therefore we could safely assume that the transformation between the robot baselink and motion capture coordinate systems was constant.

2.5.2 Tendon Driven Hand

The dexterous hand by *Shadow Robot* is a human sized hand with 24 joints [39] (Fig.2.7(c)). The joints are actuated by pairs of tendons with the pneumatic pistons mounted at the fore-arm. Each joint has a Hall-effect joint angle sensor and the tendon lengths are also measured at the pistons. The tendon lengths are functions of one or more joint angles depending on the anchor points.

2.6 Simulation-Based Experiments

To evaluate how precise our methods can recover the unknown parameters, we started with a set of random ground-truth parameters internally, and then generated simulated noisy sensor observations.

2.6.1 Selection of Optimization Algorithms for q-phase

Here we explore different strategies for optimizing the joint angles in q-phase. The simulation experiments were performed using synthesized 200 random joint angles and corresponding noisy observations from the Hand robot model. The initial seeding parameters and joint angles are all zero except for the rotation matrix which is set to the identity.

LM-batch: To start, we optimize for the joint angles until convergence and then solve for sensor parameters until convergence. This gives the Levenberg-Marquardt method enough steps to find the right step size. However, we observed that allowing full LM convergence tended to drive the optimization process into local minima before the sensor parameters had a chance to settle into the correct region.

LM-iter: In addition, we observed that after few alternations, LM converged in less than 5 iterations without much progress on the objective value. To address this problem we tested a second approach in which we alternated between joint angles and sensor parameters after each LM step iteration.

BFGS: In addition to LM we also evaluated another popular optimization algorithm, BFGS. When computing the gradient $\partial LL/\partial q$, we first optimize for the parameters until convergence.

Figure 2.6a and 2.6b show the negative likelihood of the objective function and mean-relative-error between the estimated parameters and ground truth parameters. We use relative error because the range of different parameters is quite different.

It is observed that BFGS performs best immediately followed by LM-iter. LM-Batch converged slower and the parameter actually diverged in the first 10 iterations but recovered later.

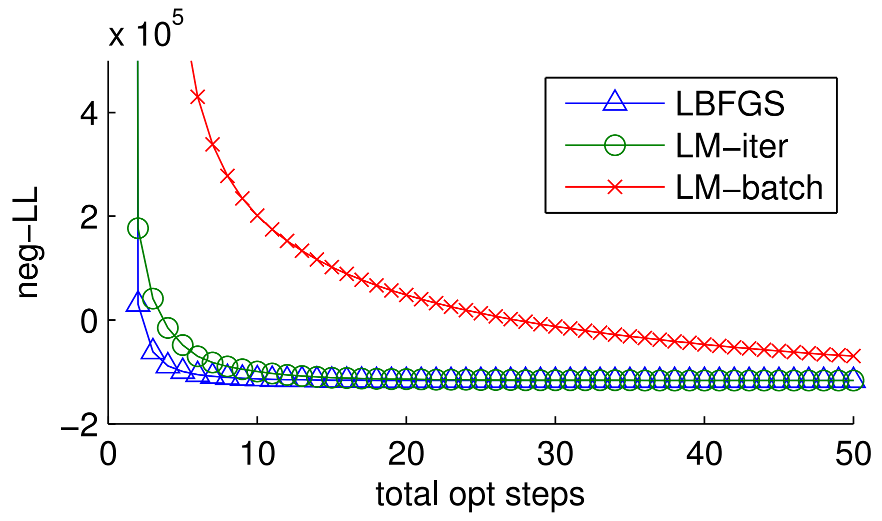
2.6.2 STAC vs Kalman Filter - Resistance to Noise

We analyzed the sensitivity of the different algorithms to the quality of seeding parameter as well as observation noise. Typically the seeding parameters are from manual calibration. While precise manual calibration is time-consuming, rough eyeballing measurement is generally enough to get the algorithm to converge to the correct local minimum.

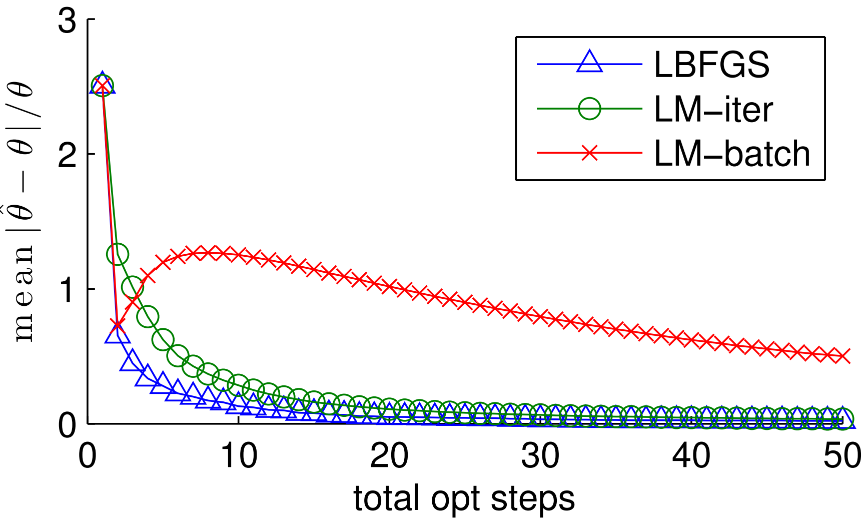
We first synthesized smooth random robot movement traces for 500 time steps within the nominal joint limits. Next we added various amount of Gaussian noise to both seeding parameter and the simulated sensor readings (3d markers and generalized potentiometers).

Then the data was fed to both STAC and EKF to evaluate how well they tracked the joint angles and parameters over time. The seeding and observation noise σ_v in EKF and in STAC were set to match the injected noise.

Note that for EKF, the initial state consists of both initial parameter and joint angle for the first frame. We set EKF initial joint angle q_0 to ground truth as it diverges immediately otherwise. On the other hand, STAC required seeding for both parameter and the entire joint angle trajectory. We therefore initialized every frame to the first frame given to EKF.



(a) negative likelihood



(b) mean relative error between estimated parameter $\hat{\theta}$ and ground truth θ

Figure 2.6: Performance comparison of optimization algorithms in “q-phase”

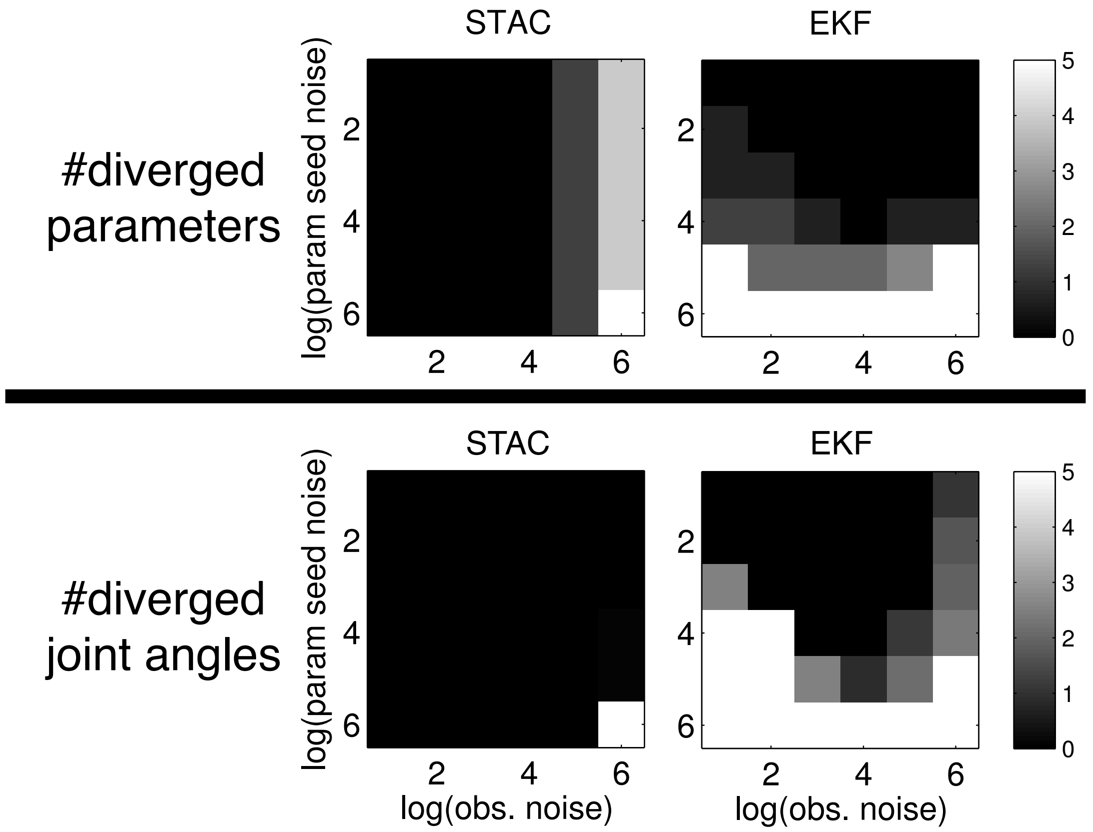


Figure 2.7: The number of diverged parameters (top row) and space-time joint angles (bottom row) under various parameter seeding error and sensor observation noise. The first column is the seeding parameter error for comparison.

The performance evaluation was separated into two parts: the accuracy of estimated parameter and space-time joint angles. Figure 2.7 plots the number of diverged variables as gray level images. Divergence is quantified by thresholding the distance between the estimated and ground truth variable. With regard to estimation of sensor parameters we found that EKF's performance is quite robust to the presence of sensor noise but it is very sensitive to noise in the seeding parameter values. STAC was much more robust than EKF to noise in the seeding parameter values but slightly more sensitive than EKF when large amount of sensor noise was present. With regard to the tracking of joint angles, STAC greatly outperformed EKF.

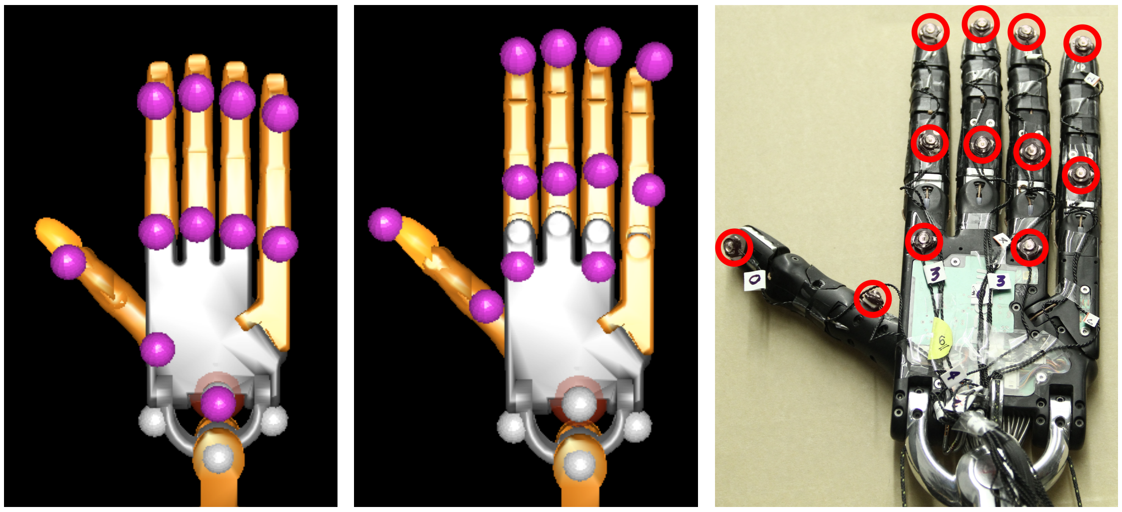
2.7 Experiments with Physical Robots

We evaluated the models learned by STAC using two different robots (Diego San and the tendon driven dexterous hand). The evaluation criteria were based on the precision of the estimated marker positions, and the ability to fit novel data.

2.7.1 Identified Marker Position

As we saw in the previous section, STAC is quite robust to errors in the parameter initialization. In both robots, we initialized the marker positions to the origin of the link it is attached to. Typically, the origin of a link is on the joint connection to its parent link. Fig.2.9(a) and Fig.2.8(a) shows the initial position.

With Diego San we used manual calibration of joint angles as seed values while for the dexterous hand we simply initialized all the parameter to zero. After optimization, we compared the estimated marker position to the pictures of the robots side-by-side in Fig. 2.8(b) and Fig.2.9(b) The close correspondence again verified that our calibration procedure worked properly.



(a) initial pos (b) calibrated (c) ground truth

Figure 2.8: Hand robot: marker positions before/after calibration

2.7.2 Cross-Sensor Prediction on Novel Data

As ground truth parameters of to-be-calibrated robots are unknown, we employee a training/testing data approach to evaluate model accuracy. The collected traces were split into training and testing sets. First we estimated system parameters from the training set and then use these parameter to predict values of one type of sensor in the test set given other sensor readings. We report the mean absolute error in Tbl. 2.1 and Tbl. 2.2. The prediction is obtained by first running the joint optimization algorithm given the selected subset of sensors (left column), and then use the obtained joint angles to infer the target sensor value (column header). For example, the first column "3D-marker" reports the mean distance between predicted and measured marker positions given individual or combination of other redundant sensors. The values in parentheses predict the same type of sensors, which can be seen as fitness of the model to the data.

For the hand robot (Tbl. 2.1), the movement of a joint is observed by all three types of sensors simultaneously. In fact, it is possible to predict joint angle given only one type of sensor. Comparing to the manual calibration done by manufacturer, STAC reduced the error by half in all sensor types. For Diego

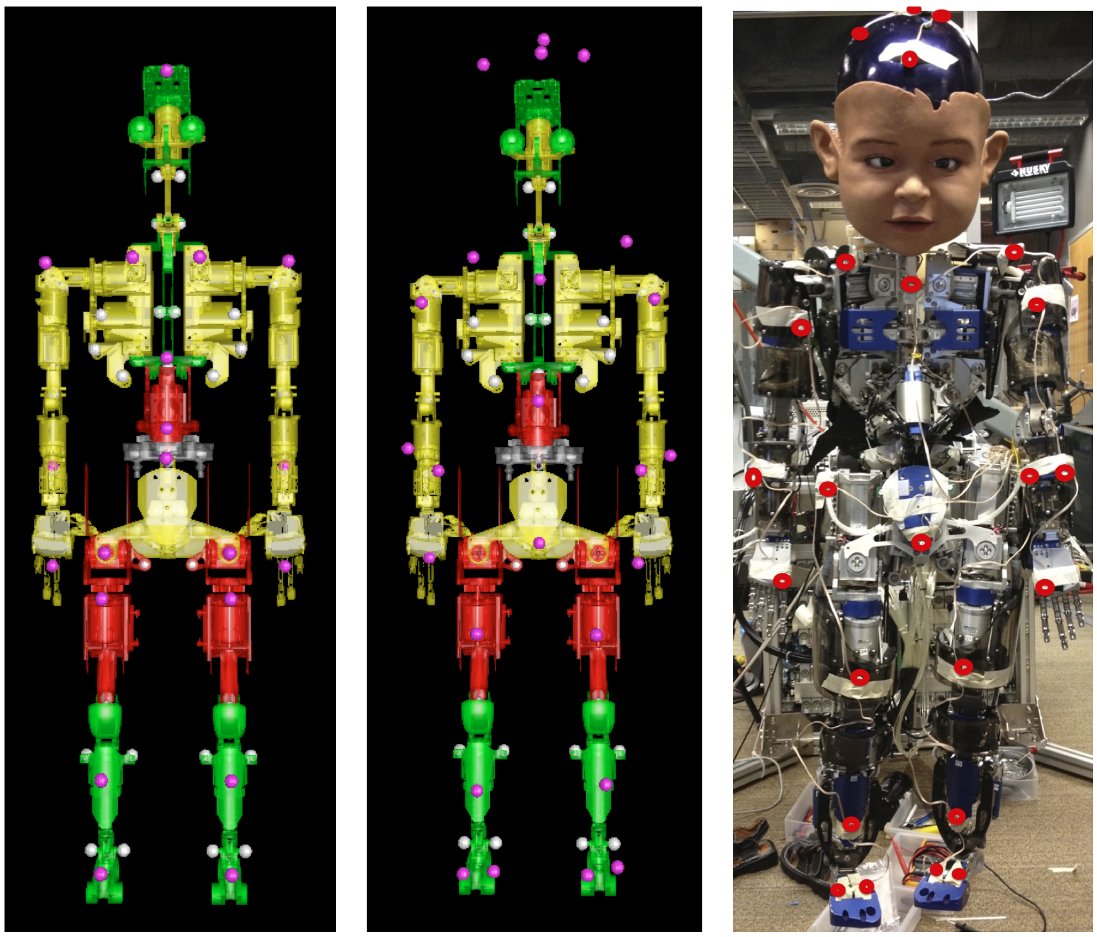

(a) initial pos (b) calibrated (c) ground truth

Figure 2.9: Diego San Humanoid: marker positions before/after calibration

Table 2.1: Hand: cross sensor prediction on novel data. (MAE)

Sensor Set	3D Marker(m)	Joint (rad)	Tendon (m)
Joint(manual)	0.0201	0.2610	0.0022
Marker (stac)	(0.0026)	0.1384	0.0014
Joint (stac)	0.0119	(0.0020)	0.0010
Tendon (stac)	0.0077	0.0895	(0.0007)
Joint+Tendon (stac)	0.0095	(0.0164)	(0.0009)
Joint+Tendon+Marker (stac)	(0.0023)	(0.0313)	(0.0009)

Table 2.2: Humanoid: cross sensor prediction on novel data (MAE).

Sensor Set	Marker(m)	Joint Pot(volt)	Tendon (m)
Joint+Tendon(manual)	0.0836	0.0000	0.9707
Joint+Tendon(stac)	0.0114	(0.0012)	(0.0028)
Joint+Tendon (stac-$\sigma_p = 0$)	0.0199	(0.0000)	(0.0035)

San (Tbl. 2.2), we performed a very coarse manual calibration for joint/tendon potentiometer gains and biases. We did not calibrate the markers. Comparing to manual calibration, STAC reduced marker error 7.3-fold when using joint/tendon sensors to infer joint angles.

The last row in Tbl. 2.2 reports the performance when the parameters are trained with the classical transparent joint-angle approach (see Sec. 2.4.2) assuming noise free joint angle sensor ($\sigma_p = 0$). The performance is worse than STAC.

2.8 Discussion

We proposed an efficient approach "STAC", that jointly estimates sensor parameters as well as joint angles from multiple redundant sensors. Contrary to previous approaches, STAC can handle complex biologically inspired configurations in which the mapping from sensors to joint angles is one to many (i.e. not a function). This allows STAC to handle a much wider range of sensors than classical methods, like linear length sensors linking multiple joints. With the aid of multiple markers, our approach converges with little or no initialization. The algorithm was evaluated on complex 38-joint humanoid robot as well as a 24-joint tendon-driven hand – with very good results.

2.9 Acknowledgment

Support for this work was provided by NSF grants IIS-INT2-0808767. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

The text of Chapter 2, in full, is submitted Humanoid 2013 as in “Tingfan Wu, Yuval Tassa, Javier Movellan, and Emanuel Todorov. STAC: Simultaneous tracking and calibration. In *submission to Humanoids 2013*”. The dissertation author was the primary investigator and author of this paper.

Chapter 3

Semi-parametric Gaussian Process for Robot System Identification

Abstract: One reason why control of biomimetic robots is so difficult is the fact that we do not have sufficiently accurate mathematical models of their system dynamics. Recent non-parametric machine learning approaches to system identification have shown good promise, outperforming parameterized mathematical models when applied to complex robot system identification problems. Unfortunately, non-parametric methods perform poorly when applied to regions of the state space that are not densely covered by the training dataset. This problem becomes particularly critical as the state space grows. Parametric methods use the available data very efficiently but, on the flip side, they only provide crude approximations to the actual system dynamics. In practice the systematic deviations between the parametric mathematical model and its physical realization results in control laws that do not take advantage of the compliance and complex dynamics of the robot. Here we present an approach to robot system identification, named Semi-Parametric Gaussian Processes (SGP), that elegantly combines the advantages of parametric and non-parametric approaches. Computer simulations and a physical implementation of an underactuated robot system identification problem show very promising results. We also demonstrate the applicability of SGP to ar-

ticulated tree-structured robots of arbitrary complexity. In all experiments, SGP significantly outperformed previous parametric and non-parametric approaches as well as previous methods for combining the two approaches.

3.1 Introduction

System identification has been one of the major challenges for progress in the control of biologically inspired robots. Obtaining a good model is crucial for finding control laws that can take advantage of the compliance and complexity of these robots. For example, when using computed torque control with an inaccurate model, the un-modeled dynamics are treated as noise. The results of this are stiff control laws to deal with any un-modeled deviations from the system dynamics. More sophisticated approaches, like differential dynamic programming or iterative LQR [85] can take advantage of the dynamic properties of complex robots but these approaches require accurate models of the robot dynamics. In practice, accurate dynamical models are critical to develop control laws that are compliant, energy efficient and safe.

There are two major approaches for system identification: parametric and non-parametric. Parametric approaches rely on parameterized Newtonian physics models of the robot's dynamics. The advantage of these models is that they capture a great deal of prior knowledge that does not need to be learned from data. For example, we know that robots are subject to gravitational forces, viscous forces and joint constraints. It is indeed wasteful to have to go through a laborious data-gathering and machine learning process to discover these well known constraints. The disadvantage of parametric models is that they are only crude idealizations of the actual system dynamics. For traditional industrial robots these un-modeled dynamics are often ignorable. However, for modern biologically inspired robots these errors result in significant control inefficiencies.

Non-parametric machine learning approaches avoid the model under-specification problem by directly learning from training sample data. Two popular non-parametric approaches include Locally Weighted Projection Regression

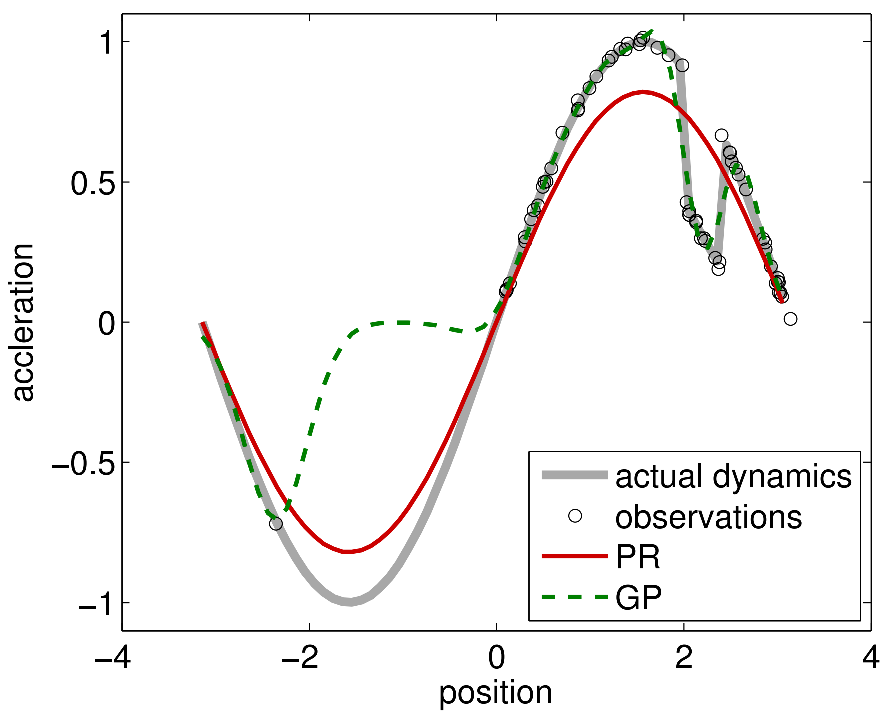


Figure 3.1: Comparing parametric (PR) model and non-parametric model, Gaussian process (GP) on learning acceleration of a pendulum.

(LWPR) [70, 86] and Gaussian Process (GP) [56] and neural networks [73]. The major advantage of these models is the ability to fit virtually any dynamics as long as they are locally smooth. The predictions are then based on interpolation between similar (or nearby) instances in the training data. However, since non-parametric models rely on local neighborhood training data to make predictions, they do not generalize well to unexplored state regions with little or no training data. Covering the entire state space becomes exponentially harder as the complexity and number of degrees of freedom in the robot increases. Thus it appears quite desirable to combine the benefits of parametric and non-parametric approaches. However, doing so in an efficient way is not trivial. A reasonable approach would be to first fit a parametric model and then fit a non-parametric model to the errors made by the parametric model. However such an approach is not ideal: it sequentially optimizes two models, rather than jointly optimizing them. Here we propose an approach that allows joint optimal inference with parametric and non-parametric models that efficiently uses the available data. The approach is formalized in the framework of what we call Semi-parametric Gaussian Process (SGP), a type of Gaussian process designed for optimal inference with combinations of parametric and non-parametric models.

3.1.1 Toy Problem

Here we illustrate the properties of parametric and non-parametric approaches, with a simple toy problem. We collect data from a pendulum that happens to have a significant friction irregularity when it reaches angles between 2 and 2.5 radians. We use a parametric model derived from the classic viscous-free equations of motion

$$\ddot{\theta} = w \sin(\theta),$$

where w is an unknown parameter. Note this model encodes a great deal of information about the forces and constraints operating on the system, however it does not know about the fact that there are some significant friction forces that happen to be a function of θ. To test this model we collected a sample of data and found the value of w that best fit this data. We call this approach Parametric Regression (PR). Fig.3.1 shows that PR successfully captured the general trend of the data. It was also able to make reasonable predictions around state regions that had very little training data. However, as expected, PR could not capture the irregularities due to friction in the interval between 2 and 2.5 radians. We also tried a standard non-parametric model (GP) shown to perform well in the robotics system identification literature. GP did a remarkably good job at capturing the effects of state dependent friction. However it did not generalize well in regions that had little or no training data.

Below we first review related work in semi-parametric and non-parametric approaches for system identification. Next, we present our formalization of the problem of optimally combining parametric and non-parametric models, in what we call a Semi-parametric Gaussian Process. After deriving the optimal inference (system identification) equations we study its performance on the toy problem presented above and in actual physical system identification problems, including a simple reaction wheel and a 3-link robotic arm.

3.2 Related Work

Semi-parametric regression has been a popular class of methods when partial knowledge is available [68]. The idea of combining Gaussian Process Regression with a global linear model was first explored in [9] where they use a GP to model the residual from a polynomial regression.

Gaussian Process has been used widely for system identification in robotics. However, most work used GP as a pure non-parametric model [16, 19, 20, 56, 67], which did not utilize the prior knowledge of the system and applied GP to learn the function directly. Such blackbox approaches are convenient but the lack of global model[1] unavoidably leads to poor generalization performance in unseen space and thus requires large amount of training data to cover the operational space.

Recent work [36] on modeling the dynamics of a blimp suggested using GP to learn the residual of the parametric Newtonian differential model. Their combined model achieved significant performance improvement over pure Newtonian model, which further demonstrated the benefit of semi-parametric model. However, since their model was not linear-in-parameter. They could not benefit from simultaneous optimization of parametric and non-parametric model. Instead, the non-parametric model is applied after parametric identification which may result in suboptimal model as shown in Sec. 3.4.1.

[1]In fact, GP probably should be classified as global model. However, due to the typical use of local basis function (eg. radial basis function), the impact of a data point is restricted to its neighborhood.

3.3 Semi-parametric Gaussian Process

We define an SGP as a collection of stochastic processes indexed by $u \in \mathcal{R}^p$ and subject to the following constraints

$$W = \mu + R \tag{3.1}$$

$$X(u) = f(u)W + Z(u), \quad \text{for all } u \in \mathcal{R}^p \tag{3.2}$$

$$Y(u) = g(u)X(u) + V(u), \quad \text{for all } u \in \mathcal{R}^p \tag{3.3}$$

where

- $\mu \in \mathcal{R}^{n_w}$ is the prior mean of W
- R is an n_w-dimensional Gaussian random vector with zero mean and variance matrix $\sigma_w \in \mathcal{R}^{n_w \times n_w}$.
- $X(u), Y(u)$ take values in $\mathcal{R}^{n_x}$ and $\mathcal{R}^{n_y}$ respectively.
- $f : \mathcal{R}^p \to \mathcal{R}^{n_x \times n_w}$ is a matrix function of a vector.
- Z is a n_x-dimensional Gaussian process with zero mean and with covariance structure specified below.
- $g : \mathcal{R}^p \to \mathcal{R}^{n_y \times n_x}$ is a matrix function.
- V is a n_y dimensional Gaussian process with zero mean and with covariance structure specified below

We developed the SGPs specifically for system identification in robotics problems. u represents the current state: joint angles, angular velocities and either control signals (in forward models) or accelerations (in inverse dynamics models). $f(u)$ represents the prediction made by the parametric model. It is either the predicted acceleration (in a forward model) or the torque that produced the observed acceleration (in an inverse model), and W represents the unknown model parameters. Here we take advantage of the fact that the inverse dynamics models of any articulated robots are linear-in-parameter [28, 35]. The prior knowledge about W is modeled by a prior mean μ and a prior covariance matrix σ_w. Z represents structural deviations from the parametric model. Z is independent of R and its structure is captured via a matrix function $\kappa : \mathcal{R}^{p\times p} \to \mathcal{R}^{n_x\times n_x}$ that controls the covariance structure of Z,

$$\mathbb{C}(Z(u), Z(\tilde{u})) = \kappa(u, \tilde{u}), \text{ for } u, \tilde{u}$$

A popular choice is the squared exponential kernel

$$\kappa(u, \tilde{u})_{ij} = \delta(i, j)\sigma_z \exp(-(u - \tilde{u})^T P^{-1}(u - \tilde{u})/2). \tag{3.4}$$

where $\sigma_z > 0$ and $P = \mathbf{diag}(\ell_1^2, \ell_2^2, \ldots, \ell_p^2)$ are model parameters. $X(u)$ represents the fused prediction made by the parametric part of the model $f(u)W$ and the non-parametric part of the model $Z(u)$. $Y(u)$ is a vector of sensory observations. It is assumed to be a known function of the state $X(u)$ plus additive white sensor noise $V(u)$. Thus the $V(u)$ vectors are independent of Z, R and of each other, i.e., for all $u, \tilde{u} \in \mathcal{R}^{n_u}$, the covariances are

$$\mathbb{C}(V(u), V(\tilde{u})) = \delta(u, \tilde{u})\ \sigma_v \tag{3.5}$$

$$\mathbb{C}(Z(u), V(\tilde{u})) = 0 \tag{3.6}$$

$$\mathbb{C}(Z(u), R) = 0. \tag{3.7}$$

3.3.1 Learning/Inference

Given a data set $\mathcal{D} = \{(u^{[1]}, y^{[1]}), \cdots, (u^{[s]}, y^{[s]})\}$ of input-output pairs, our goal is to make inferences about the value of $X(q)$ for arbitrary query points $q \in \mathcal{R}^p$. This can be accomplished using the standard equations for conditional Gaussian distributions. The expected value of $X(q)$ given the available training data is

$$\mathbb{E}[X(q) \mid \mathcal{D}] = \mathbb{E}[X(q)] + k\,(y - \mathbb{E}[Y]) \tag{3.8}$$

where

$$\mathbb{E}[X(q)] = f(q)\mu \tag{3.9}$$

$$\mathbb{E}[Y(u^{[i]})] = g(u)f(u^{[i]})\mu \tag{3.10}$$

$$Y = \begin{pmatrix} Y(u^{[1]}) \\ \vdots \\ Y(u^{[s]}) \end{pmatrix}, \qquad y = \begin{pmatrix} y^{[1]} \\ \vdots \\ y^{[s]} \end{pmatrix} \tag{3.11}$$

Next,

$$k = \sigma_{qd}\sigma_{dd}^{-1} \tag{3.12}$$

where

$$\sigma_{dd} = \begin{pmatrix} (\sigma_{dd})_{11} & (\sigma_{dd})_{12} & \cdots & (\sigma_{dd})_{1s} \\ (\sigma_{dd})_{21} & (\sigma_{dd})_{22} & \cdots & (\sigma_{dd})_{2s} \\ \vdots & \vdots & \vdots & \vdots \\ (\sigma_{dd})_{s1} & (\sigma_{dd})_{s2} & \cdots & (\sigma_{dd})_{ss} \end{pmatrix} \tag{3.13}$$

with

$$\begin{aligned} (\sigma_{dd})_{ij} &= \mathbb{C}[Y(u^{[i]}), Y(u^{[j]})] \\ &= g(u)\mathbb{C}[X(u^{[i]}), X(u^{[j]})]g(u)^T + \mathbb{C}[V(u^{[i]}), V(u^{[j]})] \end{aligned} \tag{3.14}$$

and

$$\mathbb{C}[X(u^{[i]}), X(u^{[j]})] = f(u^{[i]})\sigma_w f(u^{[j]})^T + \kappa(u^{[i]}, u^{[j]}) \tag{3.15}$$

σ_{qd} be defined as follows

$$\sigma_{qd} = \begin{pmatrix} (\sigma_{qd})_1 & (\sigma_{qd})_2 & \cdots & (\sigma_{qd})_s \end{pmatrix} \tag{3.16}$$

with

$$\begin{aligned} (\sigma_{qd})_j &= \mathbb{C}[X(q), Y(u^{[j]})] && (3.17) \\ &= \mathbb{C}[X(q), g(u^{[j]})X(u^{[j]}) + V(u^{[j]})] \\ &= \mathbb{C}[X(q), X(u^{[j]})]g(u^{[j]})^T && (3.18) \\ &= \Big(f(q)\sigma_w f(u^{[j]})^T + \kappa(q, u^{[j]})\Big)g(u^{[j]})^T && (3.19) \end{aligned}$$

The uncertainty about the model's prediction (its variance matrix) is given by the following formula

$$\mathbb{V}[X(q) \mid \mathcal{D}] = \sigma_{qq} - k\sigma_{dd}k^T \tag{3.20}$$

These equations provide the optimal way to combine the prior parametric knowledge about the system, the non-parametric knowledge of structural deviations from the parametric model, and the available training data.

Since we assume the parametric model is just a rough approximation to the actual robot dynamics we are not particularly interested on making inferences about its parameters. Instead we want to use our knowledge and uncertainty about the parameters as a component to make better predictions. Thus we marginalize over the posterior distribution of W given the training data. Note however that our beliefs about W, both the posterior mean and posterior variance have an effect in the inference process. Automatically, as we get more data we may become more certain about the value of W, thus, changing our inferences as we marginalize over W. This becomes particularly important in regions with sparse training data.

3.3.2 Relationship to Gaussian Processes(GP)

A standard Gaussian process is typically defined as a collection of unidimensional Gaussian random variables $X(u)$ with zero mean and a covariance structure defined by a kernel function, i.e.,

$$\mathbb{C}(X(u), X(\tilde{u})) = \kappa(u, \tilde{u}) \tag{3.21}$$

This can be seen as a special type of SGP in which the parameter W is visible and known to be zero ($\mathbb{E}[W] = 0$) with no uncertainty ($\mathbb{V}[W] = 0$), and X is also directly visible, i.e., $Y(u) = X(u)$. In the SGP notation this corresponds to $g(u) = I_{n_y}$, the n_y-dimensional identity matrix and $\mathbb{V}[V(u)] = 0$ for all $u \in \mathcal{I}$.

3.3.3 Relationship to Parametric Regression(PR)

In a standard parametric regression model X is a unidimensional variable, $n_x = 1$. The goal is to predict X as a linear combination of feature variables, i.e.,

$$X(u) = f(u)w \tag{3.22}$$

and there is additive i.i.d. Gaussian noise $V(u)$ to the observations, i.e.,

$$Y(u) = X(u) + V(u), \tag{3.23}$$

where $\mathbb{V}[V(u), V(\tilde{u})] = \delta(u, \tilde{u})\sigma_v$. The parameter w is typically chosen to minimize the squared error using a dataset of $f(u), Y(u)$ values,

$$\hat{w} = \operatorname{argmin} \sum_i \|Y(u^{[i]}) - f(u^{[i]})w\|^2 \tag{3.24}$$

The solution takes the following form

$$\hat{w} = (\sum_i f(u^{[i]})^T f(u^{[i]}) + \sigma_v \mathbf{I})^{-1} (\sum_i f(u^{[i]})^T y_i). \tag{3.25}$$

Once $\hat{w}$ is found, X can be estimated for new query points, i.e.,

$$\hat{X}(q) = f(q)\hat{w} \tag{3.26}$$

This can be seen as a special type of SGP in which the term Z is a constant process, $\mathbb{E}[Z] = 0$, $\mathbb{V}[Z] = 0$, and both σ_w and the observation function $g(u)$ are identity matrices.

3.3.4 Hyper Parameter Selection

As in GPs, the hyper-parameters of an SGP (σ_w, σ_v and the parameters in kernel function κ) can be set using cross-validation methods or maximum likelihood estimation. For the popular squared exponential kernel, the kernel parameters include σ_z and $\ell_i, i = 1, 2, \ldots, p$. For problem with larger input dimension p, the space of hyper-parameters can be too large for grid search in cross-validation. Therefore, we take the maximum likelihood approach to find an (locally) optimal set of hyper-parameters maximizing the likelihood of the training data,

$$\begin{aligned}
&\log p(Y|X, f(\cdot), g(\cdot)) \\
&= -\frac{1}{2} \left(g(u)f(u)\mathbb{E}[W] - Y(u)\right)^T \left(\sigma_{qd}\sigma_{dd}^{-1}\sigma_{qd}^T + \sigma_v\right)^{-1} \\
&(g(u)f(u)\mathbb{E}[W] - Y(u)) - \frac{1}{2} \log\det(\sigma_{qd}\sigma_{dd}^{-1}\sigma_{qd}^T + \sigma_v) \\
&- \frac{n}{2} \log 2\pi.
\end{aligned}$$

In practice, we put uninformative zero-mean prior over W, $\mu = \mathbf{0}$, which simplifies the likelihood to

$$\begin{aligned}
&\log p(Y|X, f(\cdot), g(\cdot)) \\
&= -\frac{1}{2} Y(u)^T (\sigma_{qd}\sigma_{dd}^{-1}\sigma_{qd}^T + \sigma_v)^{-1} Y(u) \\
&- \frac{1}{2}\log\det(\sigma_{qd}\sigma_{dd}^{-1}\sigma_{qd}^T + \sigma_v) - \frac{n}{2}\log 2\pi.
\end{aligned}$$

Then, we initialize the parameter with some heuristic guess:

- σ_v: 1/100 of sample variance of $y^{[i]}$'s.
- σ_z: sample variance of $y^{[i]}$'s.
- ℓ_i: sample variance of $u^{[i]}$'s.
- σ_w: some large number denoting the uninformative prior over W.

Finally, we optimize the likelihood using conjugate gradient ascent as in [63] to find a local maximum.

3.4 Experiments

3.4.1 Toy Problem

A popular and reasonable way to combine parametric and non-parametric approaches uses a two-step approach: first the parametric model is fit to the available data and then a non-parametric model is fit to the errors made by the parametric model. Fig. 3.2 shows that SGPs combine parametric and non-parametric prior knowledge in a smarter way. The curve shows the result of the two-step approach (PRGP) and the SGP approach on the simple pendulum task. Both approaches perform well in the region densely populated with training data. However, in the region with sparse training data SGP achieves better prediction than PRGP. The reason is that SGP jointly fits the global parameter W and Gaussian process Z at the same time, thus allowing them to interact and complement each other: The non-parametric part of the SGP "explains away" the irregular

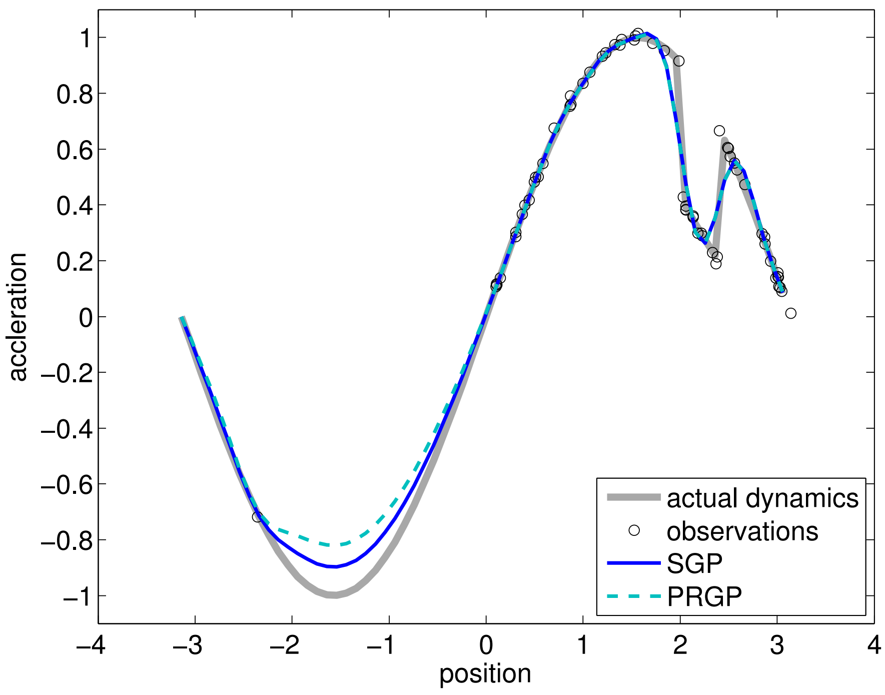
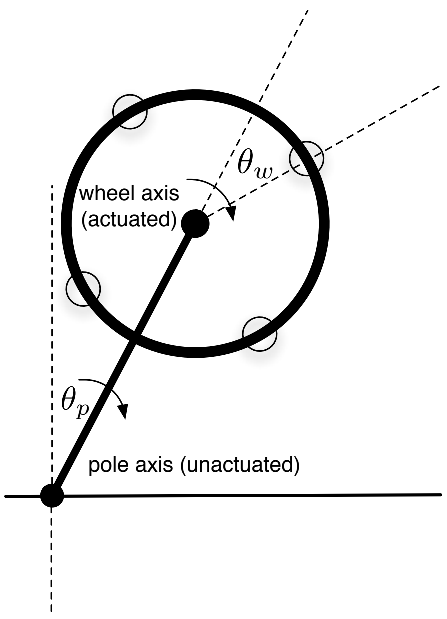


Figure 3.2: Comparing the two-stage method (PRGP) model and SGP on learning acceleration of a pendulum. Note the predictions from SGP are closer to the desired dynamics in the region with sparse data (position $[-2, 0]$).



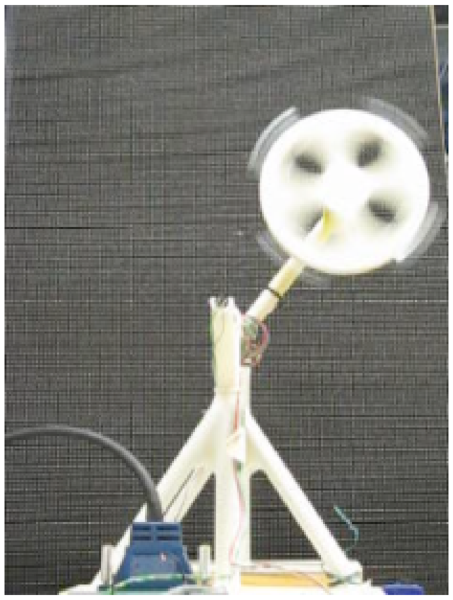
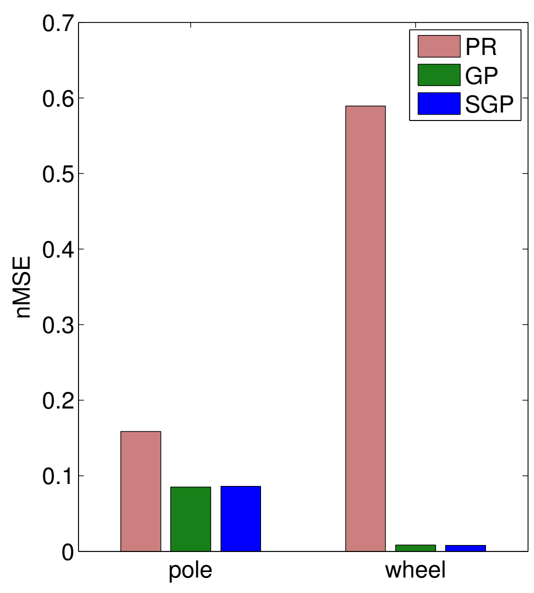


Figure 3.3: The reaction wheel and our physical implementation.

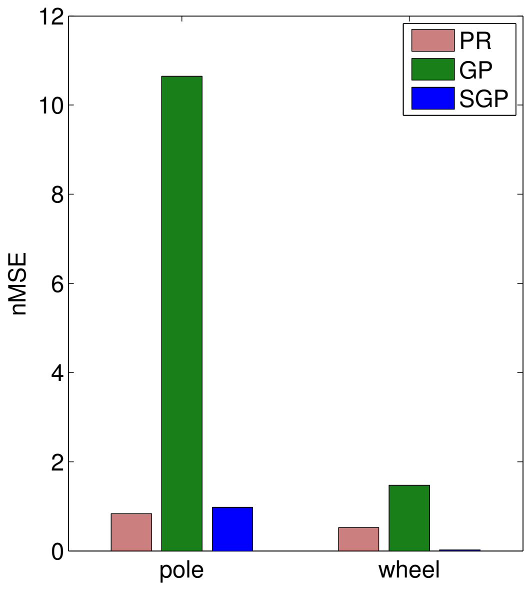
bump, and allows for the parametric model parameter W to be less affected by it. The result is a more accurate global model. In the two-phase approach, the non-parametric model (GP) cannot affect the predictions made by the parametric model, resulting in worse predictions around the regions with sparse training data.

3.4.2 Simulated Reaction Wheel

We compared SGP with PR and GP on a simulated reaction wheel problem. A reaction wheel is a pendulum with a flywheel attached to the end of its pole (see Figure 3.3). It has two parallel axes: the pole axis and the wheel axis. The pole axis is un-actuated while the wheel axis has an electric motor. Both axes are equipped with sensors which measure the angles θ_p and θ_w. The angular velocities $\dot{\theta}_p, \dot{\theta}_w$ are obtained via finite differences. The state space of the system is given by



(a) pole down



(b) pole up

Figure 3.4: System identification error for **simulated** reaction wheel in pole-down and pole-up testing conditions.

$x = [\theta_p, \theta_w, \dot{\theta}_p, \dot{\theta}_w]^T$. In this case the Newtonian equation of motion is as follows

$$\frac{d}{dt}\begin{bmatrix}\theta_p \\ \theta_w \\ \dot{\theta}_p \\ \dot{\theta}_w\end{bmatrix} = \begin{bmatrix}0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ w_1 & w_2 & w_3 & w_4 \\ w_5 & w_6 & w_7 & w_8\end{bmatrix}\begin{bmatrix}\sin\theta_p \\ \theta_w \\ \dot{\theta}_p \\ \dot{\theta}_w\end{bmatrix} + \begin{bmatrix}0 \\ 0 \\ w_9 \\ w_{10}\end{bmatrix} m, \tag{3.27}$$

where $w \in \mathcal{R}^{10}$ is the model parameter and $m \in \mathcal{R}$ is the control (voltage to the motor driver) applied to the flywheel. Note the dynamics are a linear function of the parameters and therefore we can apply the SGP approach to model the forward dynamics. The task was to predict the acceleration of both axes, $y = [\frac{d}{dt}\dot{\theta}_p, \frac{d}{dt}\dot{\theta}_w]$, given the current state x and control signal m. The performance criterion was normalized mean square error(nMSE). The normalization was with respect to the variance of the ground truth of target y. The goal of this experiment was to evaluate how well PR, GP and SGP work when the parametric model provides only a "rough" approximation to the real system dynamics. To do so, we obtained training data from the full Newtonian model but fitted a parametric model with no viscous friction. Thus the training data came from a model with 10 parameters but the parametric model only had 6 parameters,i.e., $W \in \mathcal{R}^6$. The feature matrix of the parametric model looked as follows:

$$f(x, m) = \begin{bmatrix} \sin(\theta_p) & \theta_w & m & 0 & 0 & 0 \\ 0 & 0 & 0 & \sin(\theta_p) & \theta_w & m \end{bmatrix} \tag{3.28}$$

We used a squared exponential kernel with independent length-scale hyper-parameter in each dimension. Since the sensors directly observe the angles of the axes, we have the $g(x)$ be an identity matrix. All the hyper-parameters were optimized by maximum marginal likelihood.

The training traces were obtained by controlling the reaction wheel with a mix of sinusoids signal of random frequencies for 1000 time steps at 100Hz. Due to the fact that this is an under-actuated problem, the distribution of θ_p in the training data was heavily biased toward the pole-down position. For testing, two different groups of data were collected:

- **pole-down:** traces collected in the same manner as training data.
- **pole-up:** traces collected while the pole was balanced by an LQR controller at the upward position. Note since there was noise in the simulated dynamics, the reaction wheel randomly wondered around the balancing point rather than being static.

The purpose of the two test sets was to simulate queries in regions of dense and sparse training data. The simulation was repeated 5 times and the average was reported. Figure 3.4 shows the performance of parametric regression (PR), Gaussian Processes (GP) and Semi-parametric Gaussian Processes (SGP). Note in the pole down position, which has a high density of training data, GP and SGP outperformed PR. This shows that in this region better predictions could be made by interpolation between local training data. However in the pole up position, where there were sparse training data, the parametric model did much better and the non-parametric model (GP) did quite poorly. SGP learned to rely on the parametric model in this case for the pole acceleration, and on training data for the fly-wheel acceleration thus outperforming the GP and the PR approaches.

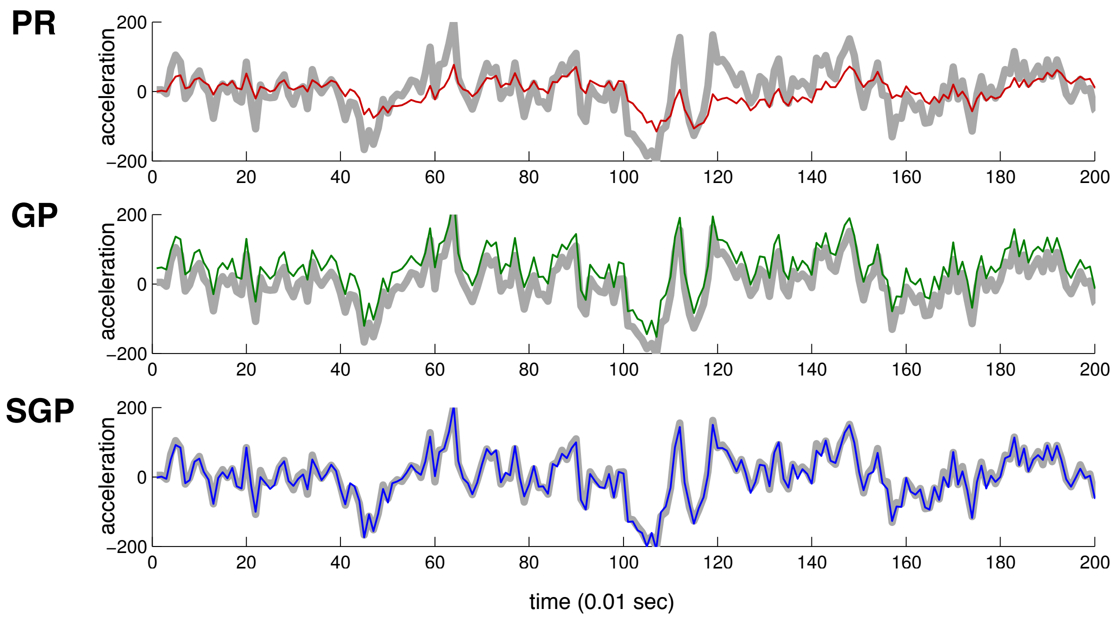


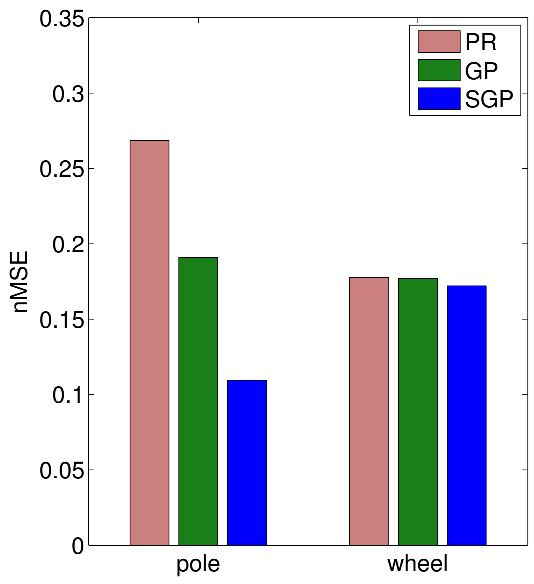
Figure 3.5: The predicted (color-narrowband) versus actual (gray-wideband) accelerations of the wheel axis in the pole-up condition of **simulated** reaction wheel.

In some cases, SGP performed surprisingly better than both GP and PR. For example, for the wheel axis in the pole-up condition, SGP achieved almost perfect prediction while PR and GP performed much worse (Fig. 3.4(b)). We plot the traces for this specific condition in Fig. 3.5. Note the high accuracy of the SGP method when compared to he other two. The predicted acceleration of the other two method is either too small in magnitude (PR) or biased (GP).

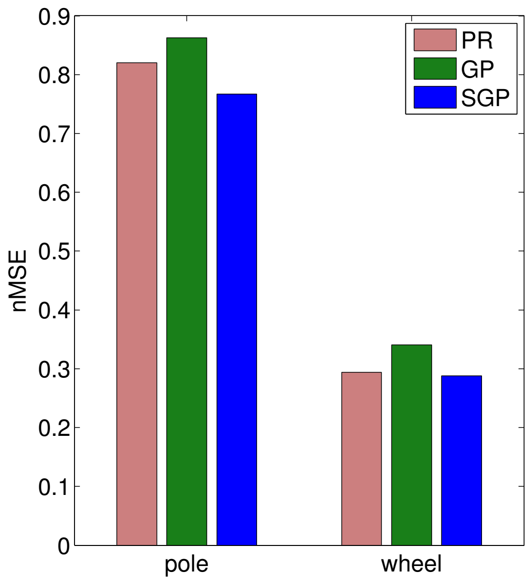
3.4.3 Physical Reaction Wheel

The data collection procedure was the same as in the simulated reaction wheel problem. Except that for the parametric model we used the full 10 parameter Newtonian model in (3.27). The hope was that the non-parametric model could capture the dynamics not modeled by the best known Newtonian model for our problem.

Figure 3.6 plots the performance the three methods. Overall, the trend resembles what we observed in the simulation. In the pole-down testing condition where there were dense training data, SGP and GP outperformed PR. In the pole-



(a) pole down



(b) pole up

Figure 3.6: System identification error for **physical** reaction wheel in pole-down and pole-up testing conditions.

up condition with sparse training data PR and SGP beat GP. In all cases SGP was the leading method.

Figure 3.7 shows some traces of pole-axis acceleration predicted by various models. First we examine the pole-down condition (left column): parametric model, PR, predicted very smooth curve but missed those small spikes. Some of these spikes might be state dependent noise that could be captured by the Gaussian Process in GP and SGP model. Indeed, both model captured some of the spikes. But overall, SGP predicted the data better.

As for the pole-up condition (right column), since the state space was quite distant from the training data collected in the pole-down condition, the non-parametric component contributes little to the final prediction. In this particular case, GP predicted basically a flat-line (the zero-mean prior of GP), which happened to be a good guess for pole-up balancing data. The prediction of SGP was dominated by its parametric model as the prediction followed PR closely.

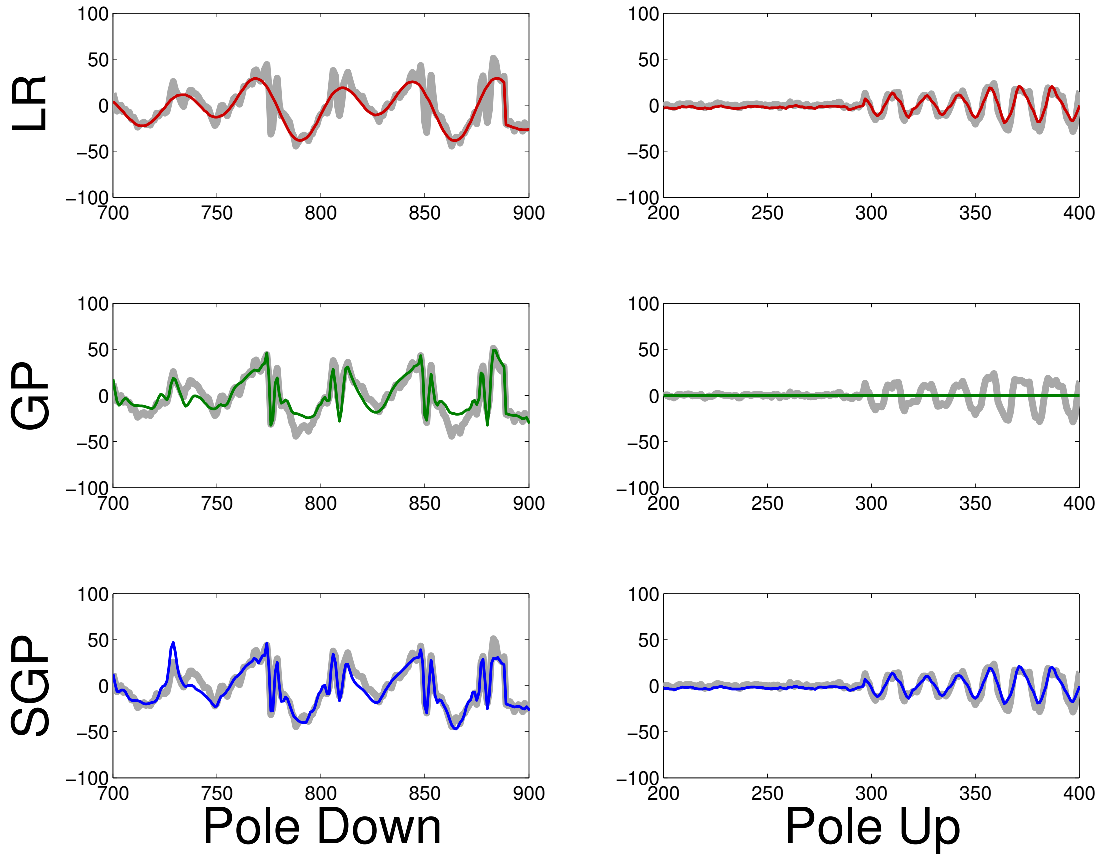


Figure 3.7: The predicted (color-narrowband) versus actual (gray-wideband) accelerations of the pole axis of **physical** reaction wheel. We only plotted a small fraction of the total data to reveal the detail in closeup.

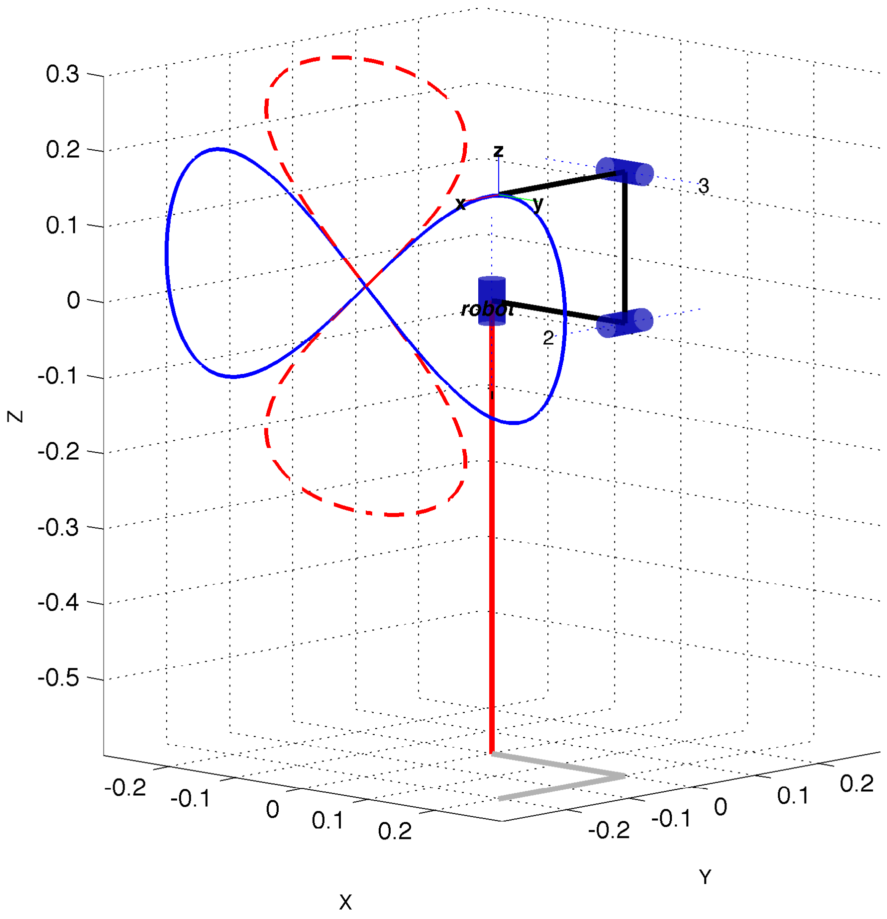


Figure 3.8: The simulated 3-link robot used in the experiment. The figure-eight horizontal (blue-solid) and vertical (red-dash) trajectories are also plotted.

3.4.4 Multi-Link Robot Arm

Next we experimented on learning the inverse dynamics model of a multi-link robot arm as a demonstration of the applicability of SGP to arbitrary tree-linked robot structure. The configuration of our simulated 3-link arm is shown in Fig. 3.8. The goal was to learn the dynamic model on the horizontal (blue-solid) figure-eight curve and then tested on vertical (red-dashed) one. In this way, we could compare how different algorithms fitted the training data and generalized to unseen curve. Although we experimented with a 3-link arm for simplicity, SGP scales to as many degree-of-freedom open-chain robot as GP scales.

For data collection, we drove the robot along these two figure-eight curves with a simple proportional controller. Meanwhile, we recorded the trace $\{\theta_i, \dot{\theta}_i, \ddot{\theta}_i, \tau_i\}_{i=1:t}$, where $\theta_i, \dot{\theta}_i, \ddot{\theta}_i, \tau_i \in \mathbb{R}^3$. In the process, artificial state dependent noise was injected by increasing the friction when joint 3 (the last joint that connects the end-effector) approached its joint limits. Such noise in real-life may be caused by cables between the links.

An inverse dynamics model predicts the required joint torque τ given the current joint angle θ, joint angular velocity $\dot{\theta}$, and the desired angular acceleration $\ddot{\theta}$. The inverse dynamics of an articulated open-chain tree structured is the following:

$$\tau = -M(\theta)\ddot{\theta} - C(\theta, \dot{\theta})\dot{\theta} + \tau^{\nu} + \tau^{c} + \tau^{g}, \tag{3.29}$$

where M is the inertia matrix, C is Coriolis and centripetal forces matrix, $\tau^{v} = \dot{\theta}\eta_{\nu}$ is the viscosity, $\tau^{c} = \mathbf{sign}[\dot{\theta}]\eta_c$ is the Coulomb friction and τ^{g} is the gravitational force.

To apply the regression algorithms, (3.29) can be written into in linear-in-parameter form [1, 28, 35],

$$\tau = f(\theta, \dot{\theta}, \ddot{\theta})w, \tag{3.30}$$

where $w \in \mathbb{R}^3$ is the dynamics parameters to be identified, including mass, center-of-mass, inertia tensor for each link and viscosity/Coulomb parameters (η_{ν}, η_c) for each joint. Fig. 3.9 compares the prediction error of the three algorithms. In the left panel, all three model performed very well for the first two joints when tested on the data in the neighborhood of training data. However, for joint 3, due to the additional un-modeled noise, LR failed badly and GP also performed worse than SGP. Further inspection suggested that GP did not fit the discontinuities caused by friction very well. The optimal length-scale parameter l_i selected by maximum likelihood was too large to fit the discontinuities. In comparison, the parametric model in SGP had absorbed these discontinuities which resulted in a better fit. Similar discontinuity fitting problem was also discovered in [15]. On the right panel, PR generalized to new vertical trajectory quite well, closely followed

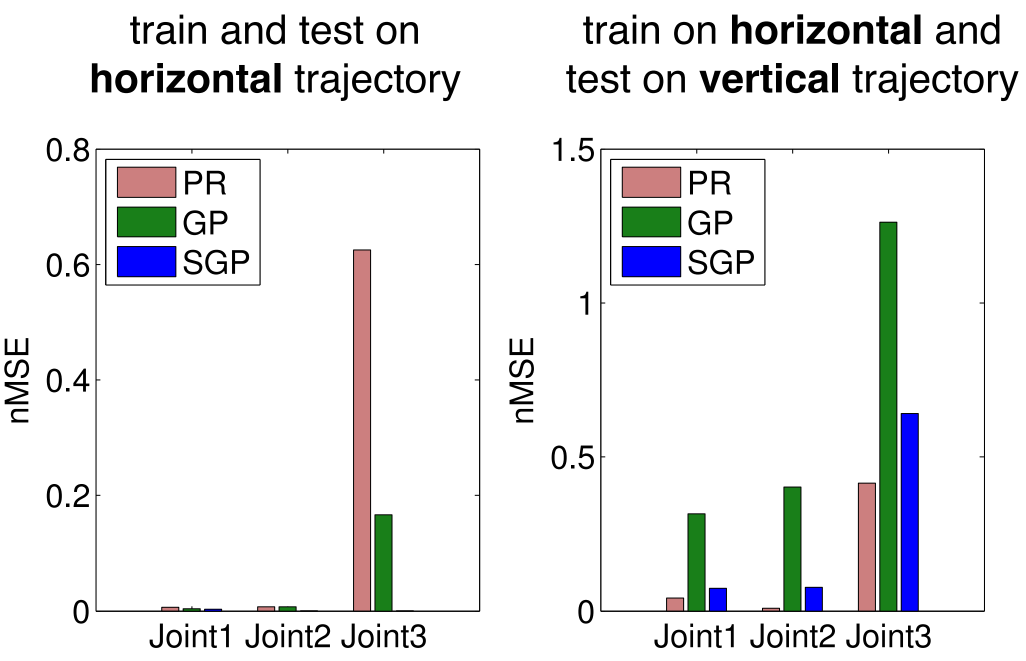


Figure 3.9: System identification error for simulated 3-link arm. Models trained using horizontal figure-eight trajectory are tested on (left-panel) another similar horizontal figure-eight trajectory (right-panel) the vertical figure-eight trajectory.

by SGP. GP failed badly in this case as the testing data were quite distant from the training data.

3.5 Discussion

In all the experiments, we found that SGP performs comparable or better than the standard parametric (PR) and non-parametric (GP) approaches.

In the real world experiment, the non-parametric model (GP) performed substantially better than the parametric model (PR) when dense training data was available. This suggests that there were state dependent dynamics that the parametric Newtonian model could not pick up. SGP performed best in data rich and data poor conditions but the difference was not as large in the real world experiment as in the computer simulation.

3.5.1 Knowing that You Don't Know

Knowing how confident a model should be of its own predictions is very important for designing optimal control policies [20]. One major problem for parametric models is that they tend to be overly-confident when making predictions about regions with sparse training data. In addition to making better overall predictions, SGP also appeared to had a better sense of its own uncertainty. Figure 3.10 plots the predictions made by the different models (means and confidence intervals) for the angular acceleration of the pole in the simulated reaction wheel experiment. For better visualization, we show the model predictions for different values of the pole's angle θ_p with the other 4 input variables $(\theta_w, \dot{\theta}_w, \dot{\theta}_p, m)$ set to zero. After training with data clustered around $\theta_p = \pi$, the pole-down position, all the models become more confident about the trained region. Although PR and SGP make similar predictions in the region distant from the training data, PR is overly-confident about these predictions. This could prove dangerous when making long term planning. Interestingly GP is also overconfident but for a different reason: it produces confidence intervals that do not cover the correct values. In this case the problem is due to the fact that the expected values of the posterior

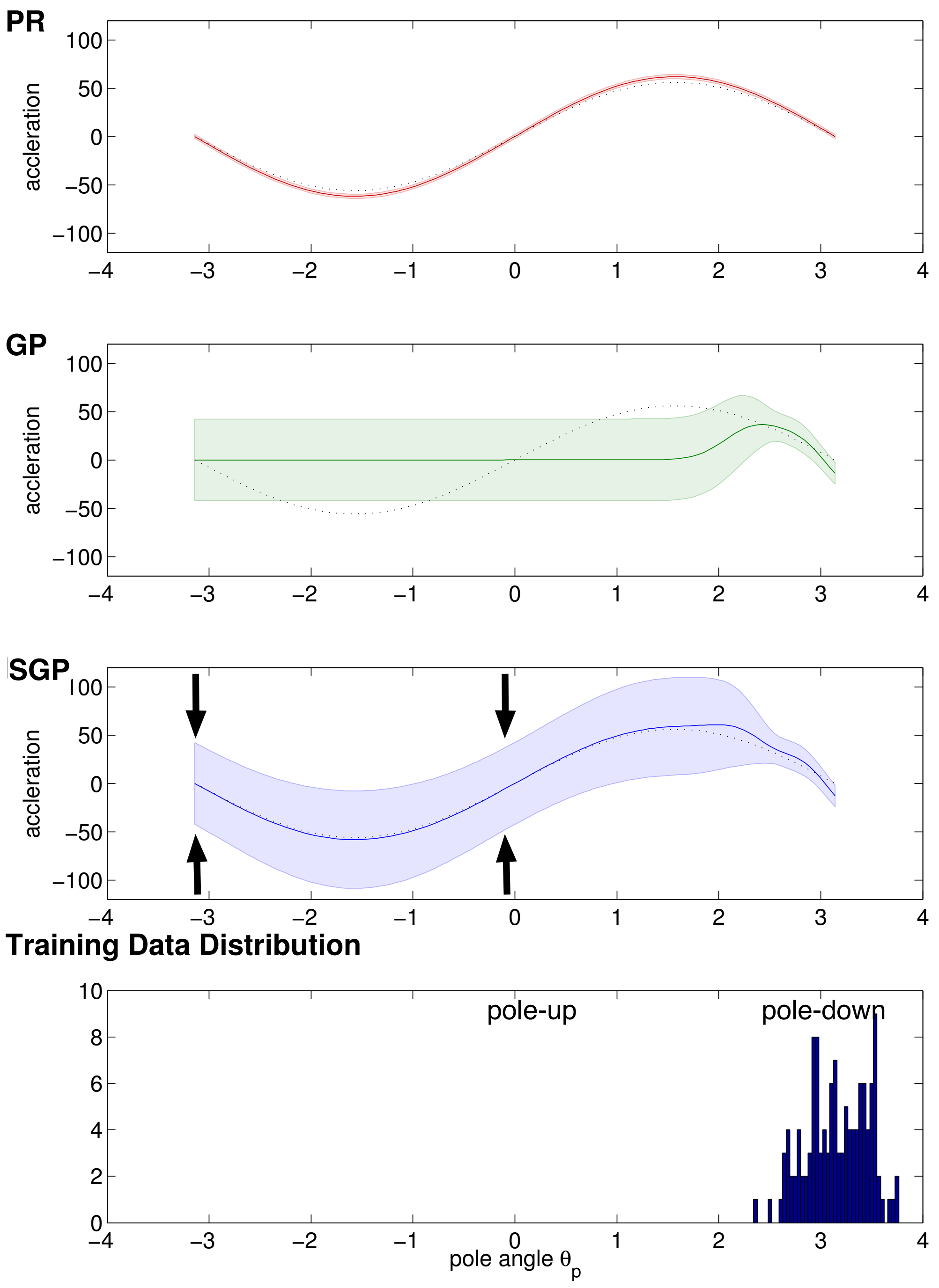


Figure 3.10: The prediction and confidence interval given by 3 different models after small amount of training. The dotted blackline is the desired dynamics to be fit.

distribution are too far off from the true values. SGP shows an interesting non-uniform shrinkage of the confidence interval as pointed out by the arrows. These points are close to the region of training data in the space spanned by the basis induced by the parametric model f.

3.6 Conclusions

We presented a principled approach, named Semi-Parametric Gaussian Processes, to combine parametric and non-parametric models in robot dynamics system identification problems. The method applies to articulated robots of arbitrary complexity as long as they can be expressed in tree-like structure. SGP flexibly fuses the prior knowledge available in Newtonian parametric models and the knowledge about local structure provided by non-parametric models. We showed computer simulations and physical implementation results that suggested SGPs capture the desirable properties of parametric and non-parametric models, both in terms of the accuracy of their predictions and the knowledge about their own uncertainty. One property that distinguishes SGPs from pure non-parametric approaches is the ability to make rough but useful generalizations over unexplored state regions. This may prove particularly important for solving underactuated robotics problems. In these problems it is difficult to develop a controller that explores, and thus provides training data about target regions of the state space. Without such training data it is difficult to train non-parametric models that could be used to develop effective controllers. The parametric component of SGP provides the ability to generalize to unseen regions of the state space. This knowledge while coarse, can prove useful to develop controllers that gather more data to train the fine-grain non-parametric component of the SGP.

3.7 Acknowledgment

Support for this work was provided by NSF grants IIS-INT2-0808767. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Chapter 3, in full, is a reprint of the material as it appears in “Tingfan Wu and Javier Movellan. Semi-parametric gaussian process for robot system identification. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 725–731. IEEE, 2012”. The dissertation author was the primary investigator and author of this paper.

Chapter 4

Modeling and Identification of Pneumatic Actuators

Abstract: Pneumatic actuators are mechanically simple and robust, have good energetic properties due to air compressibility, and are relatively cheap. Despite these advantages they are difficult to control – pressure dynamics have typical timescales on the order of 100ms, and this delay can severely cripple simplistic control approaches. The solution is to use a model-based controller with a good model of the pressure dynamics. Here we present a general parametric model of these dynamics based on both a theoretical analysis and an empirical study with a humanoid robot.

4.1 Introduction and Related Work

Pneumatic actuators are attractive for several reasons. They are naturally back-drivable, have low friction, tunable compliance and are very robust. They have a high strength-to-weight ratio – for example a typical cylinder of 5 cm diameter weighing $\sim$100 grams, running at a standard 85 psi (=590 kPa) above room pressure, produces 1160 Newtons or 260 pounds of force. Furthermore, the mechanical simplicity of pneumatics makes them inexpensive.

The central disadvantage or complication, is that they are much slower than electric motors or hydraulics, with dynamic timescales on the order of $\sim 100ms$. In order to properly control a pneumatic system, a good model of these dynamics is required. Models of such systems can in general be classified as *physical* or *parametric* models. Physical models are constructed from first principles and attempt to conform as closely as possible to the underlying physical system. Parametric models are functions with unknown constants which are found using a curve-fitting procedure. While both types of models can have good predictive properties, the design objectives are different. The physical model attempts to accurately capture all the physical properties, regardless of how important they are for prediction. The design of a parametric model, while focusing on predictive power, must also take into account secondary objectives, like ensuring good convergence and eliminating local minima in the parameter space.

Previous work has focused either on precise physical models of pneumatic systems [10] [30] [31], or on linearized parametric models [59]. In this paper, we first develop a physical model from first principles, and then use this model to guide the design of a non-linear parametric model. The work most closely related to ours is [62], where quadratic polynomials are used as a basis for the non-linear parameterization. Rather than general polynomials, we use specially crafted functions, chosen to conform to the predictions of the initial physical model. This paper is a natural continuation of our earlier work in [84].

4.2 Physical Pneumatics model

4.2.1 Ports, Valves and Chambers

A pneumatic cylinder is a device with two chambers separated by a sliding bore. The air pressure in each chamber is controlled by *valve* which can connect the chamber to one of two *ports*: the supply port connects the chamber to a compressor and the exhaust port connects the chamber to room pressure. In some setups a single valve with two output ports is connected to both chambers of a cylinder, allowing high pressure in either chamber, but not both. We chose the setup shown in Figure 4.1, where the chamber pressures can be controlled independently – to allow for the stiffness that results from high pressure on both sides. Another design

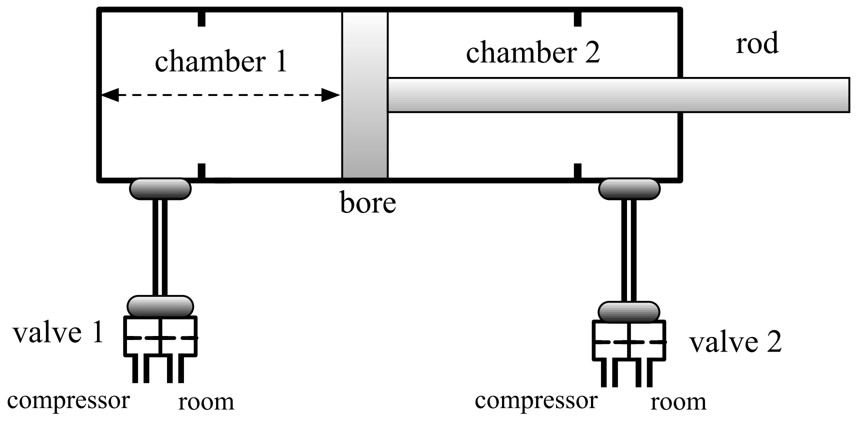


Figure 4.1: **Pneumatic cylinder.** Schematics of a cylinder with two valves, one for each chamber. Each valve has two ports, one connected to room pressure P_r and the other to the compressor P_c.

choice was to use proportional valves rather than binary valves with a Pulse Width Modulation scheme. Proportional valves offer fine-grained control of the port size, and are also less noisy than a PWM setup. The details of our particular setup are further discussed in Section 4.5, but the theoretical analysis should apply equally well to other configurations.

Port model

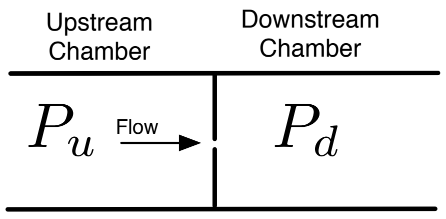


Figure 4.2: **Thin-plate port**.

The port model describes the movement of fluid that occurs when connecting two chambers (upstream and downstream) of different pressure via a small orifice (Figure 4.2). Key assumptions are that the area of the port is small, that the plate separating the chambers is thin, that the fluid is a perfect gas, that the temperatures in the two chambers are equal, and that the flow is isotropic. Under these assumptions the mass flow $\dot{m}$ is the product of the orifice area a and a function $\phi(p_u, p_d)$ of the upstream and downstream pressures:

$$\dot{m} = a \cdot \phi(p_u, p_d) \tag{4.1a}$$

$\phi(\cdot,\cdot)$ is called the *thin-plate flow function* [60].

$$\phi(p_u, p_d) = \begin{cases} z(p_u, p_d) & \text{if } p_u \geq p_d \\ -z(p_d, p_u) & \text{if } p_u < p_d \end{cases} \tag{4.1b}$$

$$z(p_u, p_d) = \begin{cases} \alpha\, p_u \sqrt{\left(\frac{p_d}{p_u}\right)^{\frac{2}{\kappa}} - \left(\frac{p_d}{p_u}\right)^{\frac{\kappa+1}{\kappa}}} & \text{for } p_u/p_d \leq \theta \\ \\ \beta p_u & \text{for } p_u/p_d > \theta \end{cases} \tag{4.1c}$$

The physical constants κ, α, β and θ are described in the Appendix. Figure 4.3 shows the air-flow $\dot{m}$ as a function of the pressure in one of the chambers while the other chamber is at room pressure, for several orifice diameters. The function is continuously differentiable. When the upstream pressure is larger than θ times the downstream pressure, the flow becomes linear in the upstream pressure and independent of the downstream pressure.

Two-port Chamber

The total flow of fluid mass into a chamber with 2 ports is the difference of the flows:

$$\dot{m}(p, a_c, a_r) = a_c \phi(P_c, p) - a_r \phi(p, P_r) \tag{4.2}$$

where a_c, a_r are the orifice areas connecting the chamber to the compressor and room respectively, and P_c, P_r are the respective constant pressures. Figure 4.4 shows this function to be monotonically decreasing, which corresponds to stable

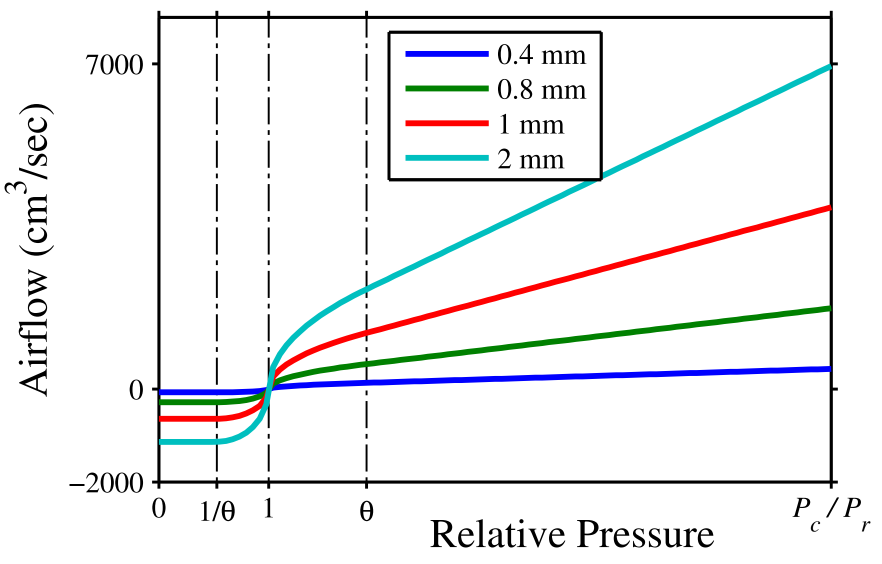


Figure 4.3: **Thin Plate Flow Function.** We plot Eq. (4.1): the air flow $\dot{m}$ as a function of the ratio p_u/p_d. The downstream pressure p_d is kept at room pressure $P_r \approx 100_{kPa}$, and p_u is varied from 0 to the compressor pressure $P_c \approx 620_{kPa} \approx 90_{psi}$. The 4 curves show the flow rate for 4 orifice diameters. The linear regime is outside the dotted vertical lines.

dynamics which converge to a steady-state pressure p_{ss} given by $a_c\phi(P_c, p_{ss}) = a_r\phi(p_{ss}, P_r)$.

Valve model

A valve is a mechanism for controlling the orifice areas of the ports. Figure 4.5 illustrates a proportional valve. For an input voltage u, a moving part called the *spool* assumes a position which is linear in u, and partially or fully obstructs the ports. As the spool moves over the port the effective area of the port will smoothly transition from a constant (very small) area when blocked, to a linearly increasing area when unblocked. The precise form of the transition depends on the relative shape of the port and the spool. We chose to model it with the function

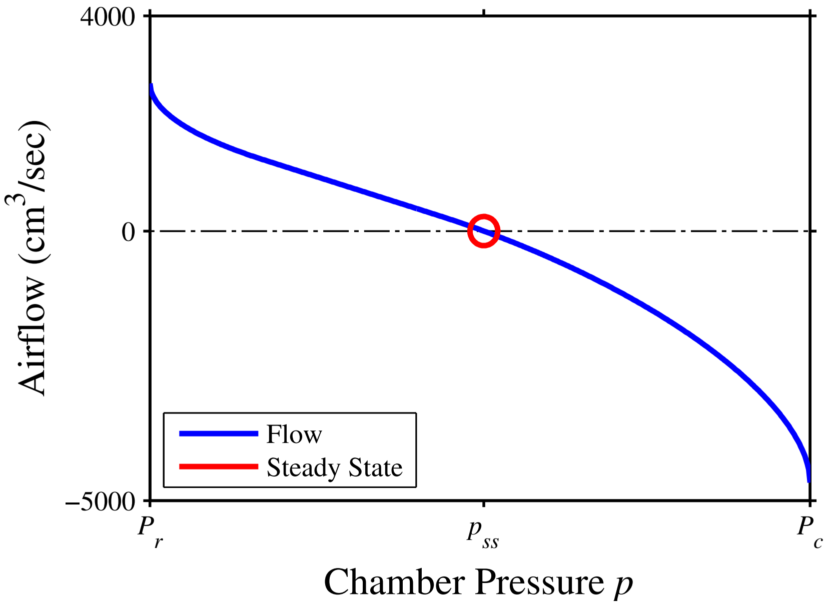


Figure 4.4: **Flow in a chamber with 2 ports.** We plot Eq. (4.2): the mass flow $\dot{m}(p, a_c, a_r)$ for orifices of 1_{mm} radius: $a_c = a_r = \pi_{mm^2}$. The red circle shows the steady-state pressure which the dynamics converge to.

$\text{smax}(x) = (\sqrt{x^2+1}+x)/2$, which is a smooth approximation to $\max(x, 0)$.

$$a_c(u) = L_c + \text{smax}((u - U_c)B - L_c) \tag{4.3a}$$

$$a_r(u) = L_r + \text{smax}((U_r - u)B - L_r) \tag{4.3b}$$

here L_c, L_r are the respective minimal areas of the compressor and room port orifices, corresponding to leakage when the ports are sealed. U_c, U_r are the voltage values at which the respective ports are sealed and B is the coefficient which translates from voltage to area. Different relative sizes of the spool and the ports will lead to different partial obstructions at mid-voltage, as shown in Figure 4.6.

Chamber Model

We can now write the pressure dynamics of a single chamber:

$$\dot{p}(p, u, \text{v}, \dot{\text{v}}) = \kappa\frac{RT}{\text{v}}\dot{m} - \kappa\frac{\dot{\text{v}}}{\text{v}}p \tag{4.4a}$$

$$\dot{m}(p, u) = a_c(u)\phi(P_c, p) - a_r(u)\phi(p, P_r) \tag{4.4b}$$

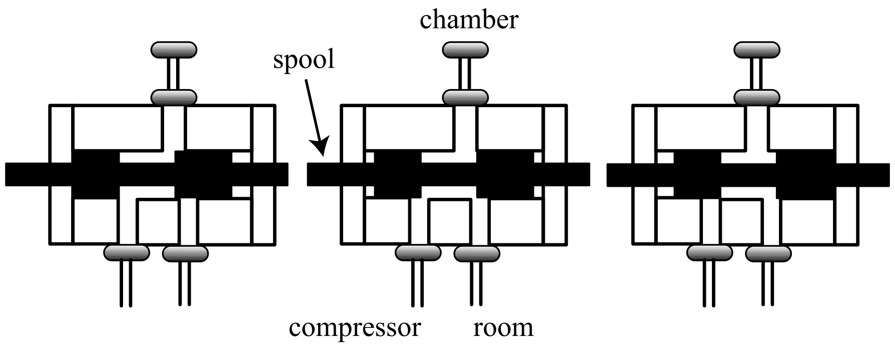


Figure 4.5: **Schematics of a proportional valve.** A movable part called the *spool* (black) moves to block or unblock the ports. Left: exhaust port open, supply port closed. Center: exhaust and supply partially open. Right: exhaust closed, supply open.

where v is the volume of the chamber, $\dot{\mathrm{v}}$ is the rate of change of that volume and κ, R and T are physical constants (see Appendix). The first term in the pressure dynamics equation (4.4a) is due to the flow from the valve, while the second one is due to compression from the piston.

Independence from the Mechanics

Given the volume v and its derivative $\dot{\mathrm{v}}$, the pressure dynamics are independent from the mechanical dynamics. Similarly, given the pressure difference between the two chambers, the force is known, and the mechanical system is independent of the pressure dynamics. Because the cylinders are rigidly attached to the limbs, v is a deterministic function of the joint angles $\mathbf{q}$. Since barometers for measuring p are cheap and accurate, as are potentiometers for measuring $\mathbf{q}$, we assume that both are indeed measured, and safely ignore the mechanics.

4.3 Parametric Model Design

The general parametric form of the pressure dynamics is

$$\dot{p} = f(p, u, \mathrm{v}, \dot{\mathrm{v}}; \mathbf{c}), \tag{4.5}$$

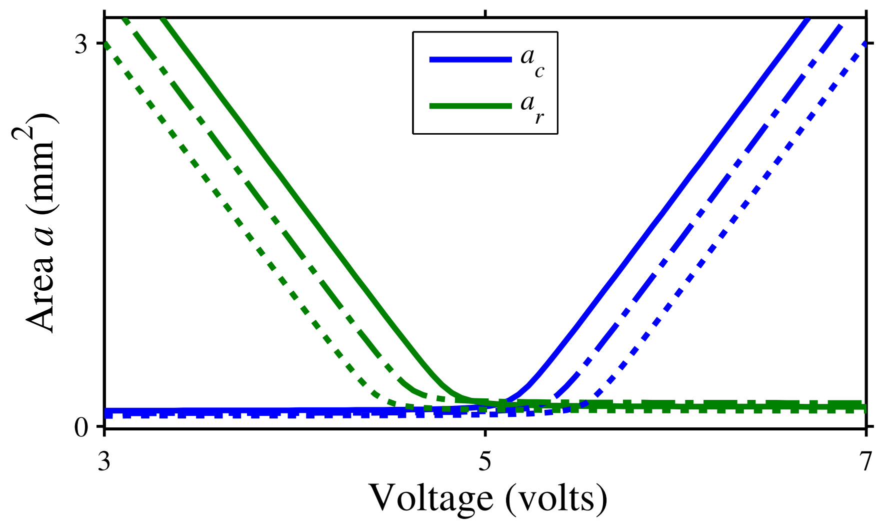


Figure 4.6: **Valve model.** We plot Eq. (4.3): Areas of the compressor port a_c and the room port a_r as a function of the voltage u applied to the valve. The different line-styles correspond to different choices of the parameters. The axes are scaled to the valves we use, see Section 4.5.

where **c** is a vector of parameters to be fit to data measured from a real pneumatic system. In principle, equations (4.1,4.3,4.4), constitute exactly such a model, but this model cannot be applied to a real system in its current form, for two reasons.

First, it contains parameters which are very difficult to measure or fit, e.g. the constants B, U_c, L_c, U_r, L_r in the valve equation (4.3). These constants depend on the precise internal geometric alignment of the spool and the ports. The three line-styles in Figure 4.6 correspond to "educated guesses" of these constants.

Second, though air dynamics are notorious for their slowness, they can also be extremely fast. Consider the vertical scale of Figures (4.3,4.4); it is $\sim 10^4$ cm^3/s. The chamber of a fully retracted cylinder can easily have a volume of ~ 0.1 cm^3, leading to a pressure change on a timescale of 10 *micro*seconds. In practice the lower bound on the timescale is determined by the valve dynamics, but these are usually quite fast, on the order of ~ 10 ms for proportional valves and much faster for switching valves. This means that in order to integrate (w.r.t time), we would need either a very small timestep or a variable-timestep integrator. This could be computationally expensive and would complicate differentiation (W.R.T state), which is often required for model-based control.

The model we want is cheap to compute and to differentiate, easy to integrate, and has parameters that can be fit reliably, using a simple procedure. The parametric form which satisfies all of these design requirements is

$$\dot{p} = \big(s(u, \mathrm{v}, \dot{\mathrm{v}}; \mathbf{c}) - p\big) \cdot r(u, \mathrm{v}, \dot{\mathrm{v}}; \mathbf{c}). \tag{4.6}$$

The function $s()$ has units of pressure and describes the steady-state pressure of the system; the function $r() > 0$ has units of inverse time and describes the total change rate in the system. The central advantage of (4.6) over (4.5) is its linearity with respect to p. Assuming fixed values for s and r over a small time-step h, we can integrate (4.6) analytically:

$$p(t+h) = s + \big(p(t) - s\big)e^{-r \cdot h} \tag{4.7}$$

This integration scheme is stable for any time-step, and is easily differentiable if $s()$ and $r()$ are differentiable.

Although we do not use the physical model (4.1,4.3,4.4) directly in our final identification scheme, we will use it to instruct our design of the parametric model (4.6) and its constituent functions $s(), r()$.

4.3.1 Steady-State Pressure

The steady-state pressure function $s()$ can be directly predicted from the physical model. Assuming a constant volume v, we eliminate the second term of (4.4a) and numerically solve for the steady-state solution $\dot{m}(p) = 0$. Figure 4.7 shows the numerical roots of (4.4a) for different values of u. The effective function $p_{ss}(u)$ shown in the figure serves two purposes:

First, it shows us the required shape of the function $s(u, \cdot, \cdot; \mathbf{c})$. It is a sigmoid with a flat kink at the origin. This flat region corresponds to the spool fully obstructing both ports. In this case the only flow is the leakage modeled by L_c, L_r in (4.3), which is independent of the precise position of the spool. We used the quadratic sigmoid

$$g(x) = x/\sqrt{x^2+1}, \tag{4.8}$$

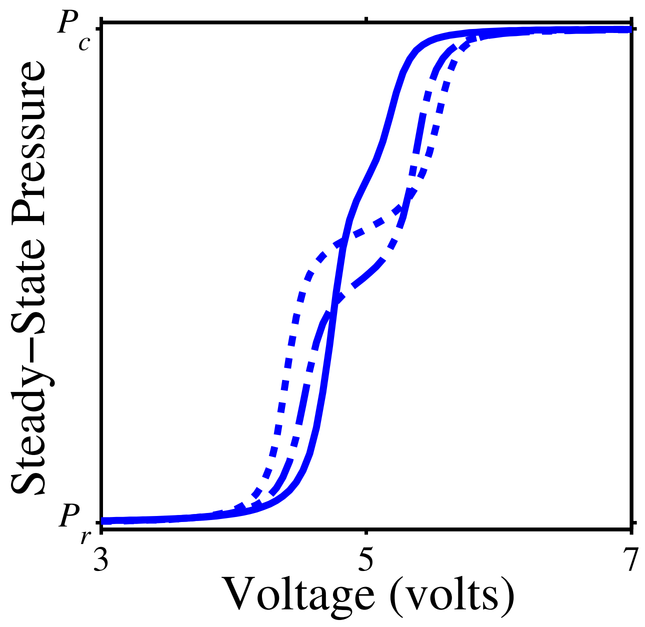


Figure 4.7: **Predicted steady-state pressure.** $p_{ss}(u)$: implicit solutions of $\dot{m} = 0$. The three line-styles correspond to the three choices of valve model constants used in Figure 4.6. See Figure 4.10 for the same curves measured on the real system.

and enabled the flat kink by adding a cubic term to the argument of the sigmoid (see below).

Second, it represents a theoretical prediction for the following simple experiment. By locking the cylinders in place we can fix the volume v; changing the command voltage u very slowly, so that the pressure is effectively at equilibrium, $\dot{m}$ vanishes and the measured pressure at the chamber should correspond to the plot in the figure.

4.3.2 Rate

The dependence of the rate function on the voltage $r(u, \cdot, \cdot; \mathbf{c})$ roughly corresponds to the total port area $a_c(u) + a_r(u)$, because $\dot{m}$ is linear in both a_c and

a_r. We therefore use the function $\sqrt{x^2+1} = \text{smax}(x) + \text{smax}(-x)$

$$k(x; a, b, c) = \sqrt{x^2 + a^2} - a + b + c \cdot x, \tag{4.9}$$

The parameters a, b, c correspond to the size of the smooth area near the origin, a vertical shift of the whole function, and a tilt of the entire function around the origin, respectively. a is related to the sealed regime of the valve, see below. The bias b is required to model the leakage terms L_c, L_r. The tilt c models asymmetries between the two ports.

4.3.3 Volume Dependence

Comparing (4.6) and (4.4a), we see that the volume v enters linearly in the denominator, and can therefore be confined to the denominator of $r()$. The volume velocity $\dot{\text{v}}$ multiplies p and linearly "drives" the ratio r/s, so it cannot be confined to one, but must appear in both $r()$ and $s()$.

4.4 The Proposed Model

Unlike the physical model where every term carries explicit units, the parametric model includes the necessary scaling and bias terms, which are found automatically by the fitting procedure. This means that sensors do not need to be calibrated independently. For example if the volume v is measured with a linear potentiometer on the piston (whose extension is linear with the volume), there is no need to explicitly convert to units of cm^3. Similarly, the output voltage of the pressure sensor need not be calibrated into physical units and can remain in sensor units.

Putting the pieces together, our model has 9 free parameters $\mathbf{c} = (c_1, c_2, ...c_9)$ per valve-chamber system which must be determined. Additionally c_b and c_s are the bias and the gain of the steady-state pressure sigmoid and are computed explicitly as $c_b = (P_c + P_r)/2$ and $c_s = (P_c - P_r)/2$. One parameter (c_γ) is chosen heuristically (see below). Written in sequential form, the model is given by (4.8),(4.9),(4.10):

$$\hat{u} = u - c_1 \tag{4.10a}$$

$$\bar{g} = g(c_2\hat{u} + c_3\hat{u}^3) \tag{4.10b}$$

$$s = c_b + c_s\bar{g} + c_4\dot{\text{v}} \tag{4.10c}$$

$$\bar{k} = k(\hat{u}, c_\gamma, c_9, c_8) \tag{4.10d}$$

$$r = \frac{c_7\bar{k} + c_5\dot{\text{v}}}{1 + c_6\text{v}} \tag{4.10e}$$

$$\dot{p}(p, u, \text{v}, \dot{\text{v}}; \mathbf{c}) = (s - p) \times r \tag{4.10f}$$

More compactly, for $\hat{u} = u - c_1$ it is

$$\dot{p} = \left(c_b + c_s\frac{c_2\hat{u} + c_3\hat{u}^3}{\sqrt{(c_2\hat{u} + c_3\hat{u}^3)^2 + 1}} + c_4\dot{\text{v}} - p\right)\times \frac{c_7\left(\sqrt{\hat{u}^2 + c_\gamma{}^2} - c_\gamma + c_9 + c_8\cdot\hat{u}\right)}{1 + c_6\text{v}} + \frac{c_5\dot{\text{v}}p}{1 + c_6\text{v}}. \tag{4.11}$$

Some of these parameters have physical interpretations that can lead to explicit constraints in the parameter space, see Table 4.1. For example the “kink factor” c_3 must be larger than -1 to maintain the monotonicity of $s(u)$.

The parameter c_γ gives the range of voltage around $u = c_1$ for which the ports are “almost” sealed, i.e. when the leakage flow is not small relative to the total flow. In terms of Eq. (4.3) this corresponds to $U_r < u < U_c$. Because $c_\gamma = 0$ creates a non-differentiable point, it is important that it remain positive, yet at $c_\gamma = 0$ the derivative $\partial k/\partial c_\gamma$ also vanishes, which leads to a bad local minimum for the fitting procedure. For these reasons we set it globally to $c_\gamma = 0.1_V$.

The constant 1 in the denominator of (4.10e) is required to collapse the multiplicative invariance of (c_5, c_6, c_7), while still allowing all 3 degrees of freedom. It can also be understood as the bias of the volume sensor, in units of volume sensor bias.

Table 4.1: Interpretation of model parameters

parameter	interpretation	comments
c_1	valve-voltage origin, $(U_c + U_r)/2$	$2 < c_1 < 8$
c_2	valve-voltage scale, B	
c_3	kink factor	$-1 < c_3$
c_4	$\partial s/\partial \dot{\mathrm{v}}$	
c_5	$\partial r/\partial \dot{\mathrm{v}}$	
c_6	volume-sensor scale	
c_7	B of Eq. (4.3)	
c_8	rate asymmetry	$-1 < c_8 < 1$
c_9	leakage, $(L_c + L_r)/2$	$0 < c_9$
c_b	pressure-sensor bias	$= (P_c + P_r)/2$
c_s	pressure-sensor scale	$= (P_c - P_r)/2$
c_γ	voltage range of leaky regime	see below

4.5 Experiment

We identified the humanoid robot shown in Figure 4.8, made by Kokoro in Japan. It has 44 pneumatically-actuated dofs; 6 of them are in the hands which were removed for the purposes of this paper, so here we are controlling only 38 dofs. Our long-term goal is to be able to make this humanoid perform various life-like movements using model based control. Before attempting such a challenging task, however, we need to identify the pneumatic system. Initial test were performed and reported in [84], where we experimented with a 2-dof arm made by the same manufacturer using the same components.

Each joint is driven by either a linear or a rotary pneumatic cylinder. The drive is direct, without any gears, belts or cables (except for a couple of joints in the humanoid). This makes the system both more compliant and more robust – indeed the robot has been hitting its joint limits at high speeds during the identification tests, without any damage. Each cylinder has two chambers fitted with solid-state pressure sensors. Each chamber is connected to a proportional valve, which can be open towards the compressor (at 90 psi) or towards the room (where the atmospheric pressure is 14.7 psi) or it can be sealed (up to small leakage). The joint angles are measured by potentiometers. The humanoid has 114 sensors and 76 controls.

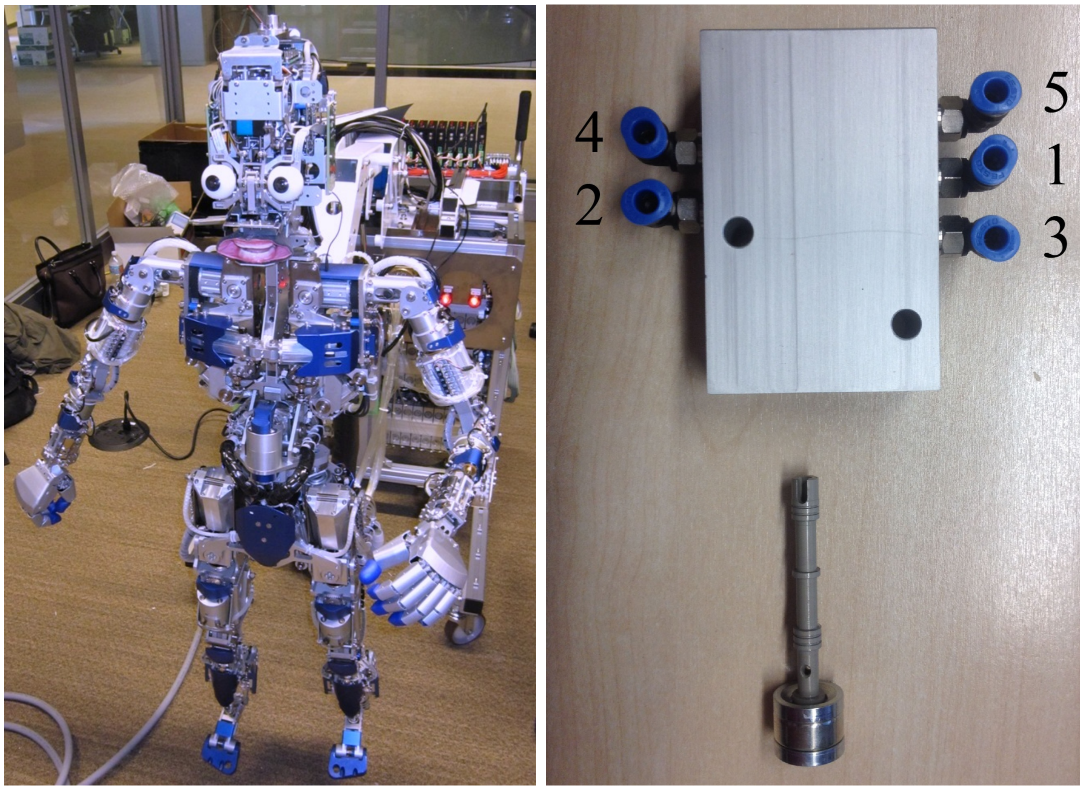


Figure 4.8: **Left: Pneumatic humanoid robot. Right: Valve and spool.** In the classic setup the compressor is connected to 1, the chambers of a single piston are connected to 4 and 2, and 5 and 3 are respective exhaust ports. We connect a single chamber to 4 with an exhaust at 5, and plug ports 2 and 3.

We use the MPYE proportional directional control valves by FESTO. The valves take a command voltage $0 < u < 10$ Volts, but we used $3 < u < 7$ Volts to avoid hitting the limits inside the valve. These valves are of the so called “5/3” type, which are usually connected to both chambers of a cylinder, and can alternatively conduct the compressor pressure either to one chamber or to the other, but not to both. We are interested in exploiting the stiffness control that is made possible by pressurizing both chambers, so chose to connect two valves to each cylinder, one to each chamber. In order to apply our model to the standard setup, the control voltage data for the single valve would simply be duplicated, and each chamber identified separately.

We use National Instruments I/O boards. The valves have 100Hz bandwidth, thus the control loop runs at 100Hz. The pressure sensors and potentiometers are analog devices that can be sampled at arbitrary rates. We are sampling all sensors at 20KHz and average every 200 samples, for a 100Hz sensorimotor loop. Such averaging is beneficial because the sensor noise is essentially white. The software system consists of the NIDAQmx drivers and a C function which reads the driver buffers, returns averaged sensor data, and sets the desired valve voltages. Data analysis was done in MATLAB.

4.5.1 Flow

We measured the steady-state flow rate as a function of voltage for the two ports in the standard setup for a single valve using a FESTO SFAB-200U flow sensor, see Figure 4.9. Since these are proportional to port area it can be seen that our valve model in (Figure 4.6) captures the principal features of the area function.

4.5.2 Steady-State Pressure

We performed the experiment described in Section 4.3.1, and measured the chamber pressure for a locked cylinder, while changing the command voltage u very slowly, to measure the steady-state pressure function $s()$. We performed this measurement for all 78 valve-chamber pairs (Figure 4.10), and noted that the resulting curves compared favorably to the solutions predicted by the physical

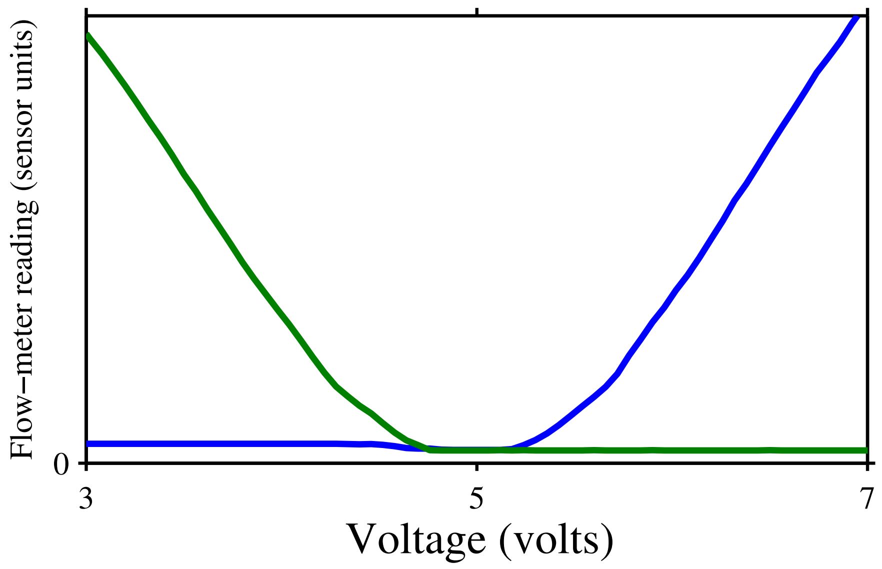


Figure 4.9: **Measured valve flow.** Compare to Figure 4.6.

model (Figure 4.7).

4.5.3 Identification

We fit the parameters $c_{1...9}$ using standard nonlinear least-squares on measured data. We recorded 100s (10^5 data points) of controlled movement during which the robot was flailing its limbs (see video attachment). These movements were controlled by a simple PID controller W.R.T the potentiometer reading, and were designed to explore the state-space as violently as possible, without hitting any joint limits so strongly that the robot would break. For each of the 76 valve-chamber systems, the data consisted of the voltage u_t, chamber pressure p_t, and joint-potentiometer reading $\mathrm{v_t}$ (which are proportional to the volume). The latter were finite-differenced to obtain joint velocities $\dot{\mathrm{v}}_t$. Rather than fit $\dot{p}$ directly, we used the integration formula (4.7) and fit the pressure at the next timestep.

$$\mathbf{c}^* = \operatorname*{argmin}_{\mathbf{c}} \sum_t (p_{t+h} - \hat{p}_{t+h}(p_t, u_t, \mathrm{v}_t, \dot{\mathrm{v}}_t; \mathbf{c}))^2$$

where $\hat{p}_{t+1}$ is the value predicted by (4.7). The values of $\mathbf{c}$ were constrained by the limits in Table 4.1. In our tests the optimization always converged to the same minimum, from different initial conditions. We have no formal guarantee of convexity, but we had never encountered bad local minima with this parameterization. It is worth noting that the particular functional form we use here is the result of a long test process, where many other functional forms were tried, which often mis-converged to bad minima. The empirically good convergence properties of our model, though difficult to quantify, are one of its strongest features.

Figure 4.11 shows an overlay of the predicted and measured change in pressure for a single valve-chamber system. Clearly the main features of the pressure derivative function are captured by the model, but a simple overlay does not properly quantify the quality of the fit.

4.5.4 Multi-step Prediction

Our end goal is to use this model for model-based predictive control. Since we do not yet have a a full kinematic and dynamic model of our robot, we can do the next-best thing, which is to test how good the prediction is for a multistep horizon. In this scenario we have a measurement of the current pressure p_t, and a planned sequence of command voltages $u_{t\dots T}$. Given a prediction of the joint angles $\mathbf{q}_{t\dots T}$ (and therefore of future chamber volumes and velocities $\mathrm{v}_{t\dots T}, \dot{\mathrm{v}}_{t\dots T}$), we can recursively integrate the pressure dynamics. Of course future joint angles depend on the chamber pressures, so this prediction would be performed simultaneously for p and $\mathbf{q}$. Here however, we can simply take the measured values of $u, \mathrm{v}, \dot{\mathrm{v}}$, while integrating p using

$$\begin{aligned}\hat{p}_0 &= p_0\\ \hat{p}_{t+h} &= \hat{p}_{t+h}(\hat{p}_t, u_t, \mathrm{v}_t, \dot{\mathrm{v}}_t; \mathbf{c}).\end{aligned}$$

Thus, the predicted pressure $\hat{p}_t$ is allowed to diverge from the measured pressure p_t. Figure 4.12 shows this divergence for the same valve-chamber system of Figure 4.11. $\hat{p}_t$ was initialized with p_t at the dotted grid lines and integrated thereafter. Surprisingly, almost no drift was detected.

Figure 4.13 shows the results of the same test performed for all 76 valve-chamber systems. Starting from 48 time points in each dataset, we integrated the pressure prediction for 2 seconds. We plot the standard deviation of the prediction error as a function of time for each valve separately, and for all $3648 = 76 \times 48$ integration sequences. Note that the vertical axis is scaled by $(P_c - P_r)/10$, so the expected drift of the predicted pressure is less than %10 of the difference between chamber and room pressure, and usually less than %4. We attribute the surprising lack of drift both to the quality of the fit and to the parameterization (4.5), which correctly captures the dynamical convergence to a set-point exhibited by the pressure.

4.6 Conclusion

In this paper we presented a 9-parameter model for the pneumatic valve-actuator system, given by (4.10). The model is based both on theoretical considerations and on empirical quality of fit. The quality of the model was demonstrated both by comparing steady-state measurements (Compare Figure 4.7 to 4.10 and Figure 4.6 to 4.9) and by comparing measurements of the dynamical model (Figures 4.11,4.12,4.13). In the near future we will use this model to do optimal control on our pneumatic humanoid robot.

4.7 Appendix

The physical constants in Eq. (4.1) are given by:

$$\alpha = C\sqrt{\frac{2M}{Z\,R\,T}\frac{\kappa}{\kappa-1}} \tag{4.12}$$

$$\beta = C\sqrt{\frac{\kappa M}{Z\,R\,T}\left(\frac{2}{\kappa+1}\right)^{\frac{\kappa+1}{\kappa-1}}} \tag{4.13}$$

$$\theta = \left(\frac{\kappa+1}{2}\right)^{\frac{\kappa}{\kappa-1}} \tag{4.14}$$

M, Z, R, T, κ, C are defined in Table 4.2.

Table 4.2: Parameters and units of the thin-plate port model.

Gas Molecular Mass	M	0.029 for air, Kg/mol
Temperature	T	$K°$
Universal Gas Constant	R	8.31 $(Pa \cdot m^3)/(mol\ K°)$
Discharge coefficient	C	0.72, dimensionless
Compressibility Factor	Z	0.99 for air, dimensionless
Specific Heat Ratio	κ	1.4 for air, dimensionless
Mass Flow	$\dot{m}$	Kg/s
Pressure	p	$Pascals$
Area	a	m^2

4.8 Acknowledgment

Support for this work was provided by NSF grants IIS-INT2-0808767. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Chapter 4, in full, is a reprint of the material as it appears in "Yuval Tassa, Tingfan Wu, Javier Movellan, and Emanuel Todorov. Modeling and identification of pneumatic actuators. In *IEEE International Conference on Mechatronics and Automation*, 2013". The dissertation author was a co-author of this paper.

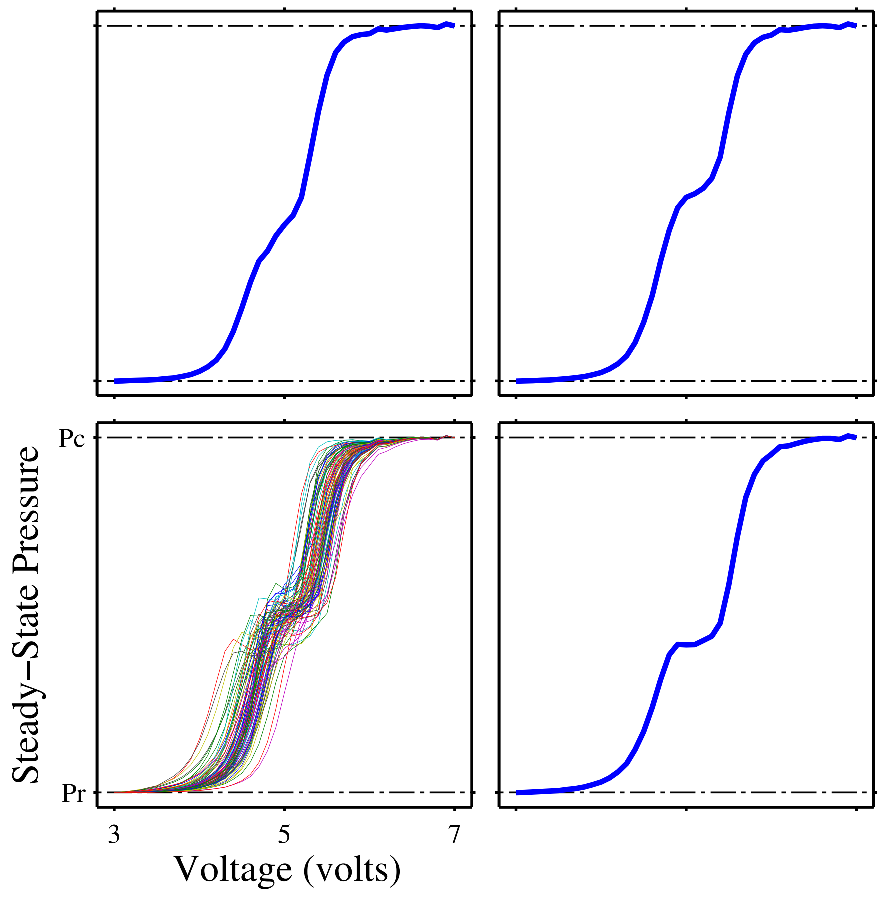


Figure 4.10: **Measured steady-state pressure.** Bottom left: overlay of steady-state pressure measurement for all 76 valves. Above and to the right: Three typical curves. Compare to Figure 4.7, which shows curves predicted by the physical model.

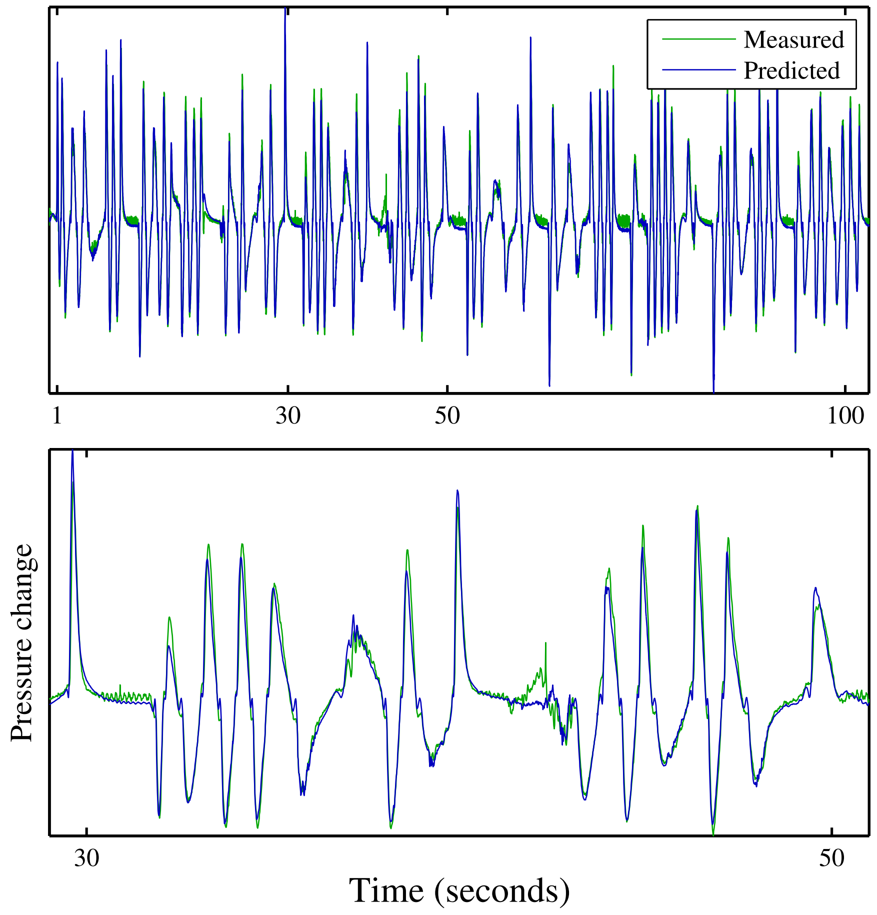


Figure 4.11: **Pressure change fit.** Shown for a a typical valve-chamber system. The light curve shows the measured pressure difference $p_{t+h} - p_t$, the dark curve shows the predicted change $\hat{p}_{t+h} - p_t$. The bottom figure shows the same data at higher temporal resolution.

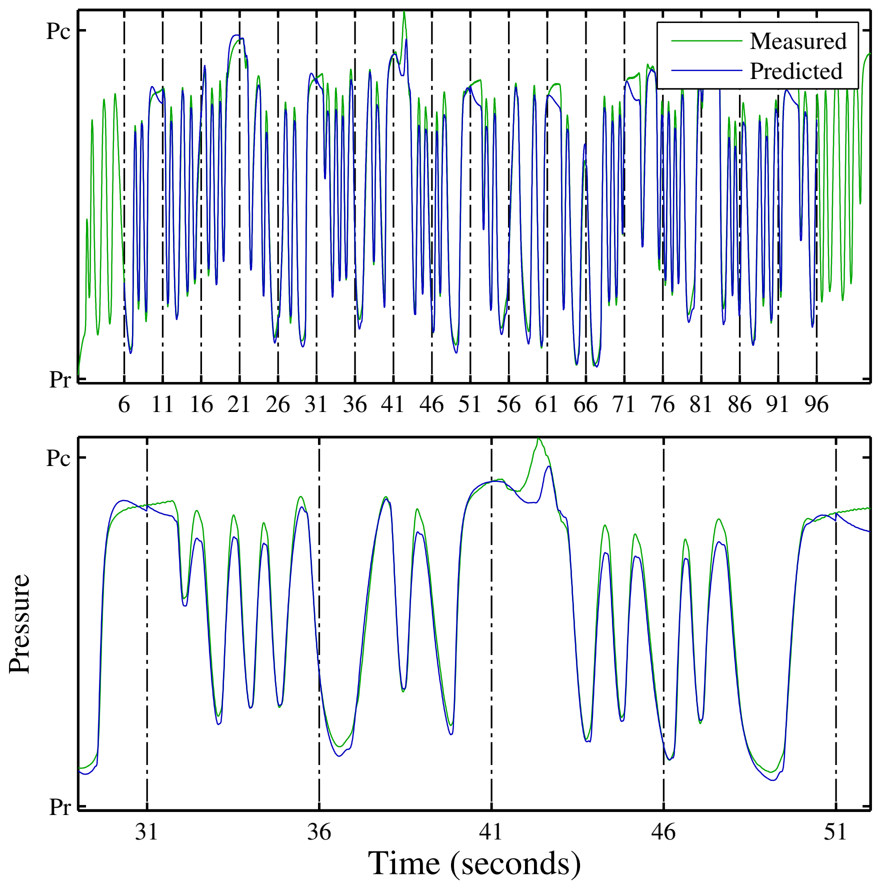


Figure 4.12: **Pressure prediction.** The light curve shows the actual pressure p_t, the dark curve shows the integrated pressure, starting at the dotted grid lines. The bottom figure shows the same data at higher temporal resolution.

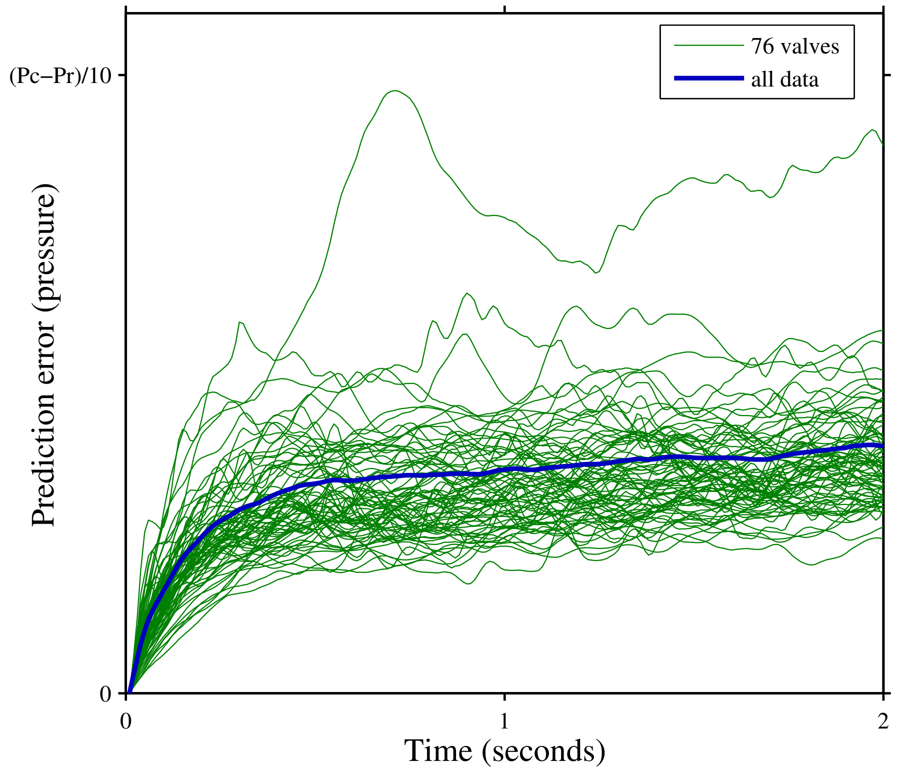


Figure 4.13: **Drift of pressure prediction.** Expected deviation of the pressure prediction from true pressure as a function of time from the last accurate measurement. Note that the vertical scale is %10 of the pressure difference $P_c - P_r$.

Part II

Learning in Humanoid Robots and Humans

Chapter 5

Learning to Make Facial Expressions

Abstract:

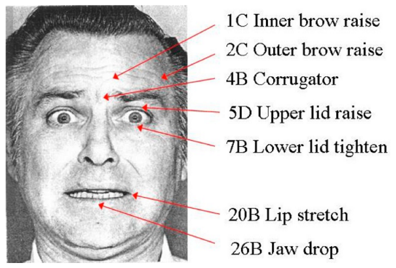
This chapter explores the process of self-guided learning of realistic facial expression production by a robotic head with 31 degrees of freedom. Facial motor parameters were learned using feedback from real-time facial expression recognition from video. The experiments show that the mapping of servos to expressions was learned in under one-hour of training time. We discuss how our work may help illuminate the computational study of how infants learn to make facial expressions.

5.1 Introduction

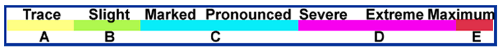
The human face is a very complex system, with more than 44 muscles whose activation can be combined in non-trivial ways to produce thousands of different facial expressions. As android heads approximate the level of complexity of the human face, scientists and engineers face a difficult control problem, not unlike the problem faced by infants: how to send messages to the different actuators so as to produce interpretable expressions.

Others have explored the possibility of robots learning to control their bodies through exploration. Olsson, Nehaniv, and Polani [58] proposed a method to learn robot body configurations using vision and touch sensory feedback during random limbs movements. The algorithm worked well on the AIBO robots. However, AIBO has only 20 degrees of freedom and is subject to well known rigid body physics. Here we utilize an android head (Hanson Robotics' Einstein Head) that has 31 degrees of freedom and non-rigid dynamics that map servo actuators to facial expressions in non trivial ways. In practice, setting up the robot expressions requires many hours of trial-and error work from people with high level of expertise. In addition as time progresses some servos may fail or work differently thus requiring constant recalibration of the expressions.

One possible way to avoid the need for costly human intervention is to develop algorithms that would allow robots to learn to make facial expressions on their own. In developmental psychology, it is believed that infants learn to control their body through systematic exploratory movements [66]. For example, they babble to learn to speak and wave their arms in what appear to be a random manner as they learn to control their body and reach for objects. This process may involve temporal contingency feedback from proprioceptive system and from the sensory system that registers the consequences of body movements on the external physical and social world [49]. Here we apply this same idea to the problem of a robot learning to make realistic facial expressions: The robot uses "expression-babbling" to progressively learn an inverse kinematics model of its own face. The model maps the relationship between proprioceptive feedback from the face and the control signals to 31 servo motors that caused that feedback. Since the Einstein robot head does not have touch and stretch sensors, we simulated the proprioceptive feedback using computer vision methods: An automatic facial expression analyzer [5] was used that estimated, frame by frame, underlying human facial muscle activation from the observed facial images produced by the android head. Once the inverse kinematics model is learned the robot can generate new control signals to produce desired facial expressions. The proposed mechanism is not unlike the body-babbling approach hypothesized by [48] as a precursor for the development of imitation in infants.



(a) The FACS Action Units (AUs)



(b) The A-E facial intensities defined in FACS.

Figure 5.1: A face can be FACS-coded into a set of numbered AUs (each number is a facial muscle group) along with letter-grades denoting intensity. Copyright©2002 Paul Ekman. Reprinted with permission.

5.2 Methods

5.2.1 Robotic Head

The robot head, “Einstein”, was developed by Hanson Robotics. The face skin is made of a material called Frubber, that deforms in a skin-like manner contributing to the realism of the robot expressions. The head is actuated by 31 servo motors, 27 of them controlling the expressions of the face and 4 controlling the neck. Figure 5.4 presents a side-by-side comparison between the location of servos in the robot head and human facial muscles. While the robot is able to simulate the actions of all major muscle groups in the face and neck, there are some important differences in the way the human muscles and the robot servo motors actuate the face. In contrast to human muscles, these servos can both pull and push loads and thus each motor can potentially simulate the action of 2 individually controlled muscle groups. Moreover in humans orbicular muscles, like the *Orbicularis oculi* and the *Orbicularis oris* produce circular contractions whereas the robot servos produce linear contractions that are coupled via circular tendons.

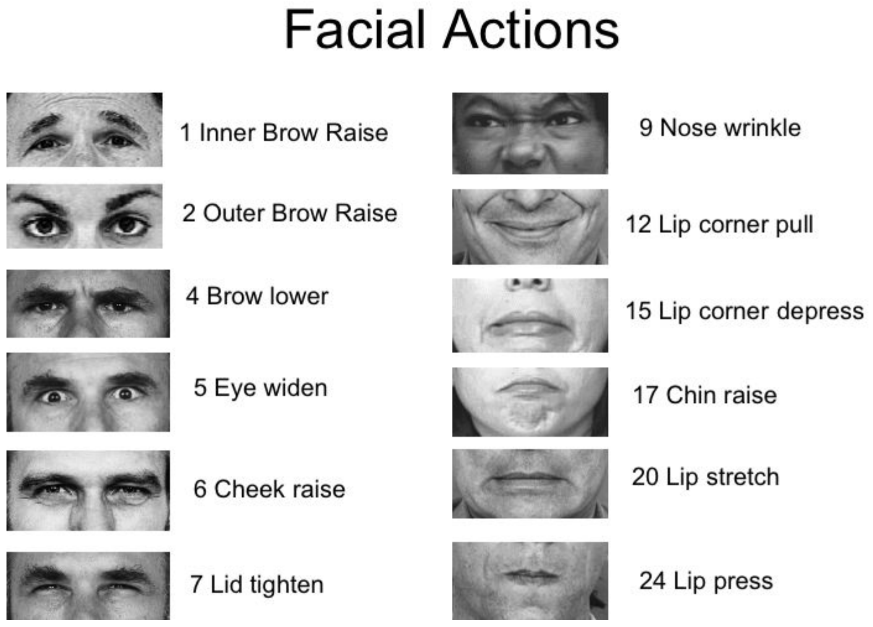


Figure 5.2: A close-up of actions units defined in FACS.

5.2.2 Facial Action Coding System

Paul Ekman [23] developed the Facial Action Coding System (FACS) as comprehensive language to code facial expressions in terms of atomic muscle movements, named facial action units (AUs). Figure 5.2 shows some major AUs. Given a face image along with the neutral face of the same person, a certified FACS coder can code the face (Figure 5.1a) in terms of active with a set of activating AUs along with their intensity measured in 5 discrete levels (Figure 5.1b) based on the appearance change on the face. These active AU can be seen as estimates of the underlying muscle activation that caused the observed expressions.

In recent years the computer vision community has made significant progress on the problem of automating FACS coding from video. Cohn's group at CMU [79] developed a system based on the use of active appearance model that tracks 65 fiducial points on the face. AUs are recognized based on the relative position of the tracked points. Our group at UCSD has been pursuing an alternative approach, called CERT (short for Computer Expression Recognition Toolbox, see Figure 5.3), to directly recognize expressions from appearance-based

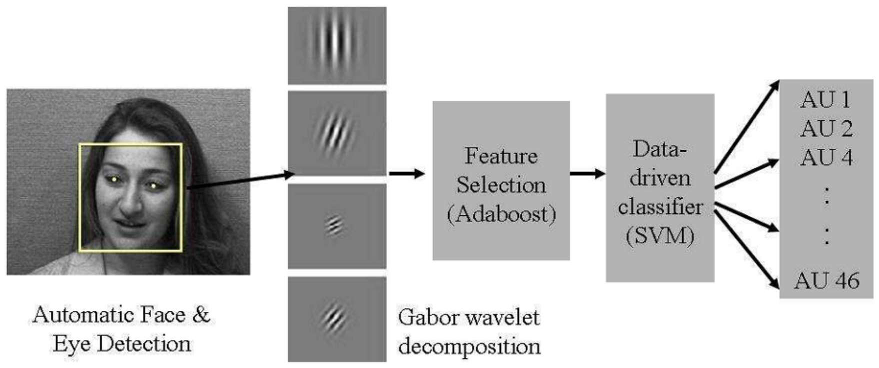


Figure 5.3: Software framework of computer expression recognition toolkit (CERT)

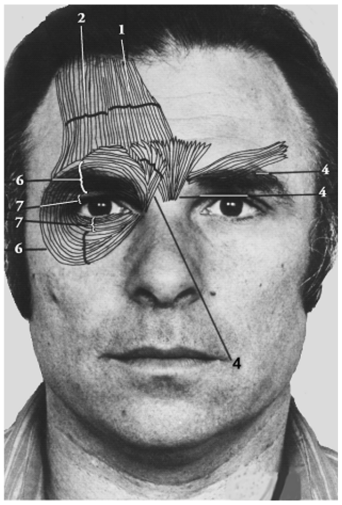
filters rather than from relative locations of fiducial points [5]. First the region of the face is automatically segmented. The obtained image patch is then passed through a bank of Gabor filters that decompose it into different spatial frequencies and orientations. Feature selection methods, like Adaboost, are used to select the more relevant filters. Finally, support vector machines (SVM) are used to classify the existence of AUs given the extracted features.

In this paper, we use CERT as a way to simulate the proprioceptive system of the human face: As the robot moved its facial servo motors CERT provided feedback about which AUs were active. AUs approximately correspond to individual face muscles, thus practically providing a proprioceptive (though visually guided) system to the robot.

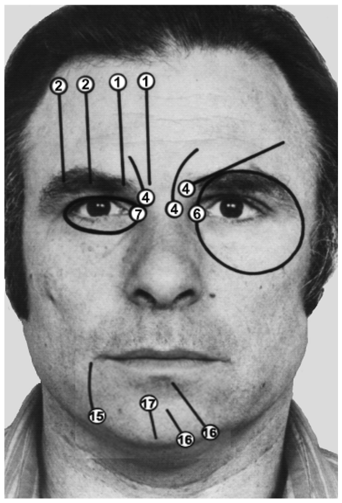
5.2.3 Learning: Random Movements and Feedback

The expression recognition software, CERT, can be seen as a non-linear function F that takes a given image I and then outputs a vector $F(I) \in \mathbb{R}^m$ of detected intensities of m AUs. Let $\mathcal{S}$ be the collection of servos used in the experiment. We denote j-th random configuration encountered during motor babbling as $\mathbf{s}_j \in \mathbb{R}^{|\mathcal{S}|}$, and the corresponding face images as $I_{\mathbf{s}_j}$. Further, let n denote the number of random movements collected.

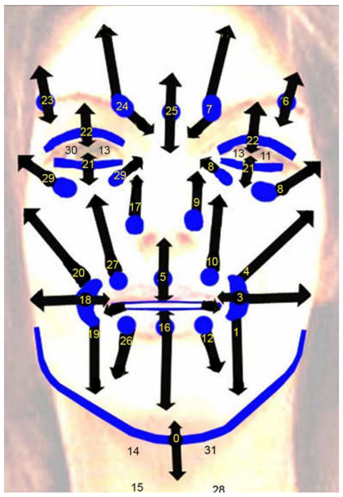
In order to produce a given expression, Einstein must learn an inverse kine-



(a) (Upper face) muscular anatomy*



(b) Facial action units associated with facial muscles*



(c) Servo layout on the robotic face

Figure 5.4: A comparison between human (a) facial muscles, (b) FACS AUs , and (c) robotic servo layouts on Einstein. Copyright©2002 Paul Ekman. Reprinted with permission.

matics model that maps desired proprioceptive signals to servo motor activations that can generate the desired proprioceptions. In this document we use a linear inverse kinematics model. For each servo i we train a linear regression model to minimize the following objective function:

$$\min_{\mathbf{c}_i, b_i} \sum_{j=1}^{n} ||(F(I_{\mathbf{s}_j})^T \mathbf{c}_i + b_i) - (\mathbf{s}_j)_i||^2, \tag{5.1}$$

where b_i is a constant bias term. Thus the problem of learning the inverse kinematics model of the face reduces to a linear least squares problem with respect to the parameters $(\mathbf{c}_i, b_i) \in \mathbb{R}^{m+1}$.

Once the model parameters are learned, we can use the model to generate new servo movements $\{\mathbf{s}'_i\}, i \in \mathcal{S}$ for any desired AU configurations $\mathbf{a}$ according to the linear mapping

$$\mathbf{s}'_i = \mathbf{a}^T \mathbf{c}_i + b_i. \tag{5.2}$$

Efficient analytical and iterative solutions exists for this problem. Thus the advantage of using linear models is that they are simple, fast, and easy to train. The obvious disadvantage is that if underlying mapping between servo actuations and expressions is not linear, the model will not work well. It was thus unclear whether the proposed approach would work in practice.

5.3 Experiments

The real-time expression recognition was done using CERT version 4.4.1 running on a Dual Core Intel Based Mac Mini. The CERT software recognizes 12 AUs (see Figure 5.5 for a list). The output of CERT is a real-valued vector for each video frame indicating the estimated intensity of each facial action. The output is roughly base-lined at zero, with outputs above zero indicating the AU was present. However, the actual baseline of neutral expression is subject dependent. Therefore, we collect the baseline for Einstein $\mathbf{a}_N$, which will be used in expression synthesis stage.

Communication with the ROBOT hardware was handled using RUBIOS2.0 a Java based open source communications API for Social Robots [40]. RUBIOS

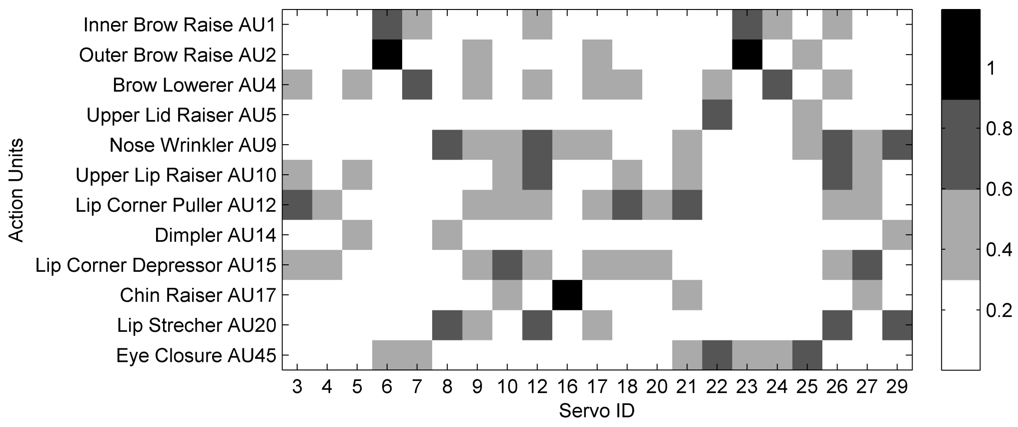


Figure 5.5: The learned connections between the AUs and servos learned in our experiment.

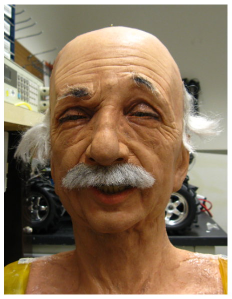


Figure 5.6: Asymmetric random facial movements.

2.0 is built on top of QuickServer, an open source Java library for multi-threaded, multi-client TCP server applications.

5.3.1 Learning

In order to collect data for learning a mapping between facial expressions and servo movements, Einstein generated a series of random servo movements (see Figure 5.6). The position of each servo was sampled uniformly and independently from the safe operation range of each servo. This phase can be seen as the "body-babbling" that allows learning a kinematic model of the face.

We excluded the servos for directing the eye gaze (servo 11, 13, 30), the jaw (servo 0), and the neck (servo 14, 15, 28, 31) since they were not related to the elementary facial muscle movements currently recognized by CERT. Two

Table 5.1: correlation coefficient how well AU input predicts servo movements.

face region	training	testing
upper	0.7868	0.7237
lower	0.5657	0.4968

additional servos, 1 and 19, were also disabled after discovering that, when random motor babbling caused pulling in opposition to servos 4 and 1, servo burnout resulted. We are currently developing a mechanism for the robot to automatically sense the energy spent by the servos and therefore to automatically avoid harmful servo configurations, possibly by adding a fatigue term that simulates the limited capacity of human facial muscles to contract for long periods of time. Such a change might also lead to more realistic learned strategies for facial expression synthesis.

We collected 500 instances of perception-production pairs. Each instance consists of the configuration of the servos and the outputs of the 12 facial action unit detectors produced by CERT. Since CERT estimates activations of individual facial muscles, here CERT could be seen as playing the role of a human proprioceptive system, informing which facial muscles are activated at every point in time. The 500 instances were then used to train the linear regression model. The results are shown in Table 5.1. We observed very good performance for expressions in the upper face region and moderate performance for the lower face. We suspect that this may be due to the facial hair on the robot (mustache) that probably reduced the accuracy of the feedback provided by CERT. However it is also possible that the underlying mapping between servos and expressions, is less linear for expressions in the lower face. We are currently investigating which of these two explanations is more consistent with the data.

Figure 5.5 displays the mapping between AU and servo control signals learned by the model. The values are normalized by the dynamic range of AU intensity and servo movements. In each row, the figure shows the set of servos related to the generation the AU, with dark shading indicating strong involvement.

For example, servo 6 and 23 plays the major roles in demonstrating AU2, while servos 9, 17 and 25 also provides minor contribution. On the other hand, each column shows which AUs predict or explain the servo movement the best. For example, the movement of servo 6 is mainly explained by AU17 (chin raise, Figure 5.2).

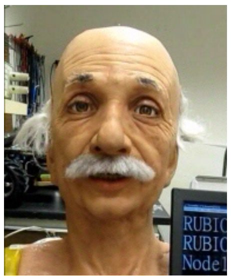
5.3.2 Action Unit Synthesis

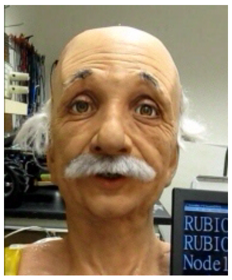
Coding of human facial action units best done in relation to a neutral face. Here we face a similar issue in that we have to account for the Einstein's neutral expression and use it as baseline to synthesize other action unit configurations. Let the baseline AU intensities of Einstein's neutral face be denoted by $\mathbf{a}_N = F(I_N)$ where I_N is the neutral expression face of Einstein when all the servos are relaxed. Then, the synthesized AU i intensities were set to $\mathbf{a}' = \mathbf{a}_N + \mathbf{e}_i$, where $\mathbf{e}_i$ is a vector of zeros with the exception that the i-th element to be one. Finally, we generated the corresponding servo movements by $\mathbf{s}'_i = \mathbf{a}'^T \mathbf{c}_i + b_i$.

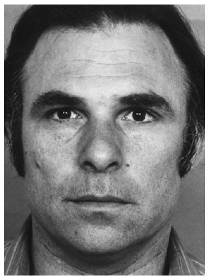
Figure 5.7 shows examples of some of the synthesized AUs. We put the neutral expression in (a) for reference. (b) is the synthesized AU1 expression (inner eyebrow raise). For comparison, we also put the neutral and AU1 expression demonstrated by a human in (c) and (d). Figure (e)-(h) gives more examples on AU2, AU4, AU5 and AU9.

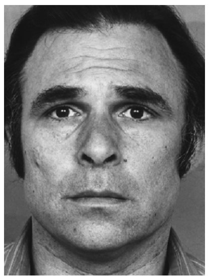
5.4 Discussion

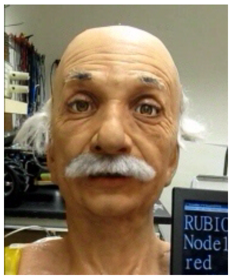
While Hanson Robotics made an effort to explicitly map each servo to individual action units in the Facial Action Coding System, we observed that the model learned to activate multiple servos to produce each AU. Subjectively the AUs learned by the model, which synthesized multiple servos, appeared more realistic than the equivalent AUs originally shipped with the robot that had been set by hand. For example, AU4 (eyebrow narrowing) is recognizable by changes in appearance that occur mainly at the midpoint between the two eyebrows. In humans the muscles that contribute to the appearance of AU4 are the left corrugator

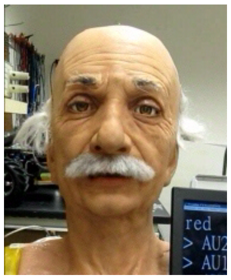

(a) Neutral

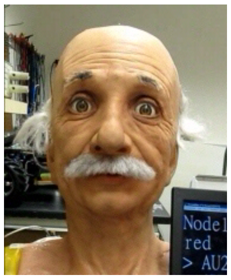

(b) AU1: Inner Brow Raise


(c) Human Neutral*


(d) Human AU1: Inner Brow Raise*


(e) AU2: Outer Brow Raise


(f) AU4: Brow Lower


(g) AU5: Eye Widen

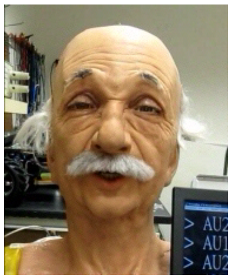

(h) AU9: Nose Wrinkle

Figure 5.7: Action units learned by Einstein

Copyright©2002 Paul Ekman. Reprinted with permission.

and right corrugator. If servos are tuned by hand, a heuristic assignment will be moving inner eyebrow servos 7 and 24. Our model learned this obvious connection clearly. However, as stated in FACS manual [24], the appearance change of AU4 "push the eye cover fold downwards and may narrow the eye aperture." Our model also learned to close the upper eyelid (servo 22) a bit to narrow the eye aperture. Similar phenomena were also found in the lower face. AU 17 "chin raise" is recognizable by the bulging around the chin region (see Figure 5.2). While the robot does not have any servos in that region of the face, the model learned to produce the appearance of bulging using 3 lip servos (servos 10,16 27).

During the experiment, one of the servos burned out due to misconfiguration. We therefore ran the experiment without that servo. We discovered that the model learned to automatically compensate for the missing servo by activating a combination of nearby servos.

Another interesting observation is that the robot learned to produce symmetric servo movements. This is likely due to the fact that the database of images of facial expressions that was used to develop the CERT software had predominately symmetric expressions.

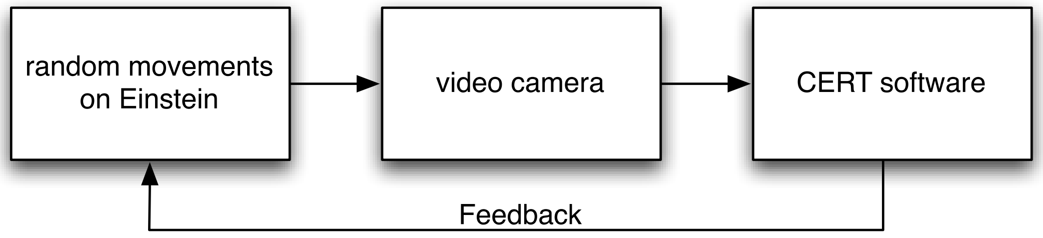
5.4.1 Developmental Implications

The primary goal of this work was to solve an engineering problem: How to approximate the appearance of human facial muscle movements with the available motors. Nevertheless this work also speaks to learning and development of facial expressions in humans. It is not fully understood how humans develop control of their facial muscles to produce the complex repertoire of facial expressions used in daily social interaction. Some aspects of facial behavior appears to be learned, and other aspects appear to be innate. For example, cross-cultural data [22] suggests that some basic expressions, such as smiles, are shared universally among all the peoples in the world, leading scientists to hypothesize that they are innate. Moreover, congenitally blind individuals show similar expressions of basic emotions in the appropriate contexts, despite never having seem them [46], and even show brow raises to emphasize speech [14].

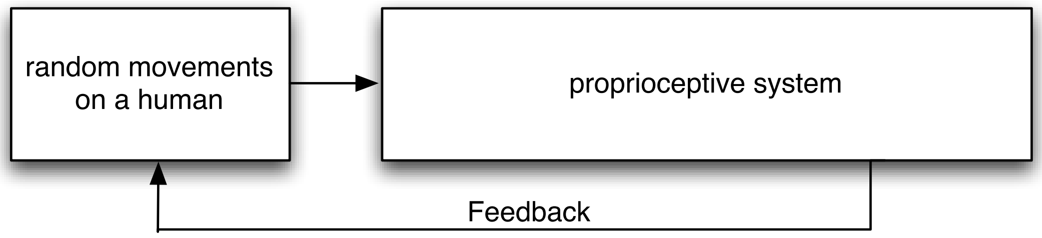
There are two distinct brain systems that control the facial muscles [65]: a sub-cortical system that is responsible for affect driven expressions and a cortical system that is responsible for voluntary expressions. During development children learn to control voluntarily their own expressions. This transition from felt to voluntary control of the face is clear to many parents when their children start producing smile with a distinctly different morphology to that of spontaneous smiles when they are posing to a camera. The mechanism proposed here would explain how cortical systems can learn to control the face in a voluntary manner. The sub-cortical system, for example, can spontaneously produce expressions of emotion (e.g., smiles) that result on memorable proprioceptive traces. Body babbling can be used to develop an inverse model of the face and then reproduce, in a voluntary manner, the proprioceptive traces experienced during felt expressions of emotions.

Our experiment demonstrates that complex facial expressions may be learned through feedback of the type made available by CERT through the framework shown in Figure 5.8a. One possibility is that CERT was basically serving the role of a proprioceptive system (Figure 5.8b). As such the fact that CERT happens to use visual input is incidental. Similar feedback to that produced by CERT could have been obtained using proprioceptive sensors rather than visual sensors. Another possibility is that people can actually encode the expressions observed by others in a manner that mimics the function of CERT (Figure 5.8c). There is empirical evidence that during social interaction people tend to mimic the facial expressions of their interlocutors [21], which implies that humans have the capability to visually encode facial expressions and map them onto their own muscle movements. This behavior could effectively serve as a mirror that would provide information about the effects of one's own muscle movements onto the external appearance of facial expressions. Blind children appear to have problems masking expressions of negative emotions [27], indicating that seeing others may be important for gaining voluntary control of facial expressions .

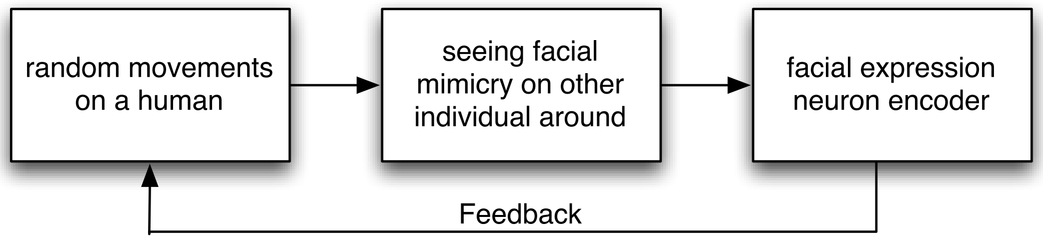
We are currently experimenting with an active learning mechanism to allow the robot to actively choose muscle movements, "facial babbling," so as to



(a) Einstein



(b) human I


random movements on a human
seeing facial mimicry on other individual around
facial expression neuron encoder
Feedback

(c) human II

Figure 5.8: The proposed framework of learning to demonstrate facial expression on Einstein(a) and human(b)(c).

optimize learning efficiency. Instead of making random movements, the brain may move the face in more efficient ways to quickly reduce the uncertainly of the internal expression-to-muscle model. Such active exploration may employ information maximization similar to models of human exploratory behavior in eye-movements [13].

Note that while the current system learned atomic expressions of emotions, as defined in FACS, holistic expressions of emotions such as expressions of happiness, sadness, anger, surprise, and disgust are, in principle, combination of individual action units. We are currently investigating whether the expressions learned this way are also currently investigating the mechanisms for learning holistic expressions of emotion,

5.5 Acknowledgments

Support for this work was provided by NSF grants SBE-0542013 and NSF IIS INT2-Large 0808767. The Einstein robot heads were purchased with a DURIP grant. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Chapter 5, in full, is a reprint of the material as it appears in “Tingfan Wu, Nicholas J Butko, Paul Ruvulo, Marian S Bartlett, and Javier R Movellan. Learning to make facial expressions. In *IEEE 8th International Conference on Development and Learning, 2009 (ICDL 2009)*, pages 1–6. IEEE, 2009”. The dissertation author was the primary investigator and author of this paper.

Chapter 6

DNA for Optimal Control

Abstract: We introduce Diffusion Network Adaptation (DNA), a framework for finding approximate solutions to continuous time, continuous state, continuous action optimal control problems. We present two reinforcement learning algorithms developed under this framework, one model based and the other model free. We test the algorithms in computer simulations and in a complex pneumatic humanoid robot that had to learn how to kick a ball. The algorithms are mathematically elegant, easy to use, and achieve state of the art performance. The DNA framework provides interesting links to recent reinforcement learning algorithms and helps explain why these algorithms work well in conditions that violate the assumptions under which they were originally developed.

6.1 Introduction

A wide range of models in physics [29], finance [8], psychology [52, 64], robotics [12, 38], computer animation [42], computational neuroscience [81], and artificial neural networks [54] are controlled diffusion networks of the following form [33, 51, 57]

$$dX_t = \bar{\mu}(t, X_t, U_t)dt + \sigma(X_t)dB_t \tag{6.1}$$

where X_t is an n dimensional vector representing the state of the network, $\bar{\mu}(\cdot)$, called the drift, represents the deterministic part of the system dynamics, U_t is a control signal that can be used to modulate the system dynamics, B_t is a Brownian noise process, and $\sigma(X_t)$, called the dispersion matrix, determines how the Brownian noise affects the system dynamics. In robotics X_t typically represents joint angles and angular velocities, U_t represents torques, and $\sigma(X_t)$ represents uncertainty in the robot dynamics. In finance X_t may represent the value of a portfolio, U_t the investments made on different assets, and $\sigma(X_t)$ the market volatility. In psychology the states X_t may represent activation of internal representations and the control signals represent external stimuli. In recurrent neural networks the states X_t represent some potentials, and the control signals represent afferent input currents.

The term "diffusion network" refers to the fact that these networks can be seen as a set of interconnected nodes in which probability diffuses according to the standard Fokker-Plank equations of fluid dynamics [33, 51, 57]. The machine learning literature contains an unsupervised learning algorithm for training diffusion networks to generate probability distributions of continuous paths [50, 53–55]. Here we show that this algorithm provides a powerful framework for solving continuous optimal control problems. We refer to this framework as DNA (Diffusion Network Adaptation). Here we present two reinforcement learning algorithms developed under this framework, the first model based and the second model free. The algorithms are mathematically elegant, easy to use, and achieve state of the art performance. The DNA framework provides interesting links to recent reinforcement learning algorithms and helps explain why these algorithms work well in conditions that violate the assumptions under which they were originally developed.

6.2 Statement of The Problem

Our goal is to control a diffusion network using parameterized policies of the form $U_t = f(t, \theta, X_t)$, where $\theta \in \mathcal{R}^p$ is the policy's parameter vector. For convenience we will refer to θ as the policy. For each policy θ we let X^θ represent the corresponding diffusion process. Since the control signal U_t is a function of X^θ

and θ, we can redefine the system dynamics as follows

$$dX_t^\theta = \mu(t, \theta, X_t^\theta)dt + \sigma(X_t^\theta)dB_t \tag{6.2}$$

where

$$\mu(t, \theta, x_t) = \bar{\mu}(x_t, f(t, \theta, x_t)), \text{ for } x \in \mathcal{R}^n \tag{6.3}$$

This representation links nicely with the artificial neural network literature, where the parameter θ typically represents the synaptic weights between point neurons [54]. For generality we let the initial state probability density $p(x_0 \mid \theta)$ be part of the policy.

6.2.1 Mathematical Preliminaries

The mathematics of discrete time processes do not always apply to continuous diffusion processes, thus care must be taken to use the appropriate tools from stochastic calculus. In particular, the sample paths generated by diffusion networks are continuous, non differentiable, and have finite quadratic variation [33, 51, 57]. More importantly these paths do not have the standard Lebesgue probability densities used for discrete time processes. Instead when working with diffusion we define a reference process Y that induces a reference probability measure (the substitute for the Lebesgue measure used in discrete time systems). We then formulate path densities with respect to this reference. Here we use the zero-drift version of X^θ as the reference process

$$dY_t = \sigma(Y_t)dB_t \tag{6.4}$$

We let $p_y(y_0)$ represent the initial state distribution of the reference process. Let $L(\theta, x)$ be the relative density of the n-dimensional continuous path x in the interval $[0, T]$. Thus $L(\theta, x)$ is the Radon-Nykodim derivative [33, 51, 57] of the probability measure induced by the process X^θ with respect to the probability measure induced by the reference process Y. In practice this means that expected values of functions of paths from X^θ can be computed using expected values of functions of reference paths Y, i.e.

$$\mathrm{E}\Big[h(X^\theta)\Big] = \mathrm{E}\Big[L(\theta, Y)h(Y)\Big] \tag{6.5}$$

A key theorem from stochastic calculus, called Girsanov's theorem [33, 51, 57] tells us that

$$L(\theta, x) = \frac{p(x_0 \mid \theta)}{p_y(x_0)} \exp\Big\{ \int_0^T \mu'(\theta, x_t)k(x_t)dx_t - \frac{1}{2}\int_0^T \mu'(\theta, x_t)k(x_t)\mu(\theta, x_t)dt \Big\} \tag{6.6}$$

where

$$k(x_t) = (\sigma(x_t)\sigma(x_t)')^{-1} \tag{6.7}$$

6.3 Unsupervised Learning of Path Distributions (Imitation Learning)

In unsupervised learning problems a system is exposed to a probability distribution of real valued vectors and adjusts its policy to approximate the observed distribution. In the robotics literature, this is sometimes known as imitation learning. I our case a diffusion network is exposed to a probability distribution of continuous n-dimensional paths, and the goal is to find a policy that approximates the observed path distribution as best as possible. Here we will utilize the approach first derived in [50, 53–55]. We let Z represent the observed processes whose distribution we want to approximate. We let the value of policy θ decrease with the KL divergence between the distribution of observed paths Z and the distribution of X^θ. We also let the value increase a term $\log q(\theta)$ that favors a priori some policies over others

$$v(\theta) = -\mathrm{E}\Big[\log \frac{\Lambda(Z)}{L(\theta, Z)}\Big] + \log q(\theta) \tag{6.8}$$

where Λ is the Radon-Nykodim derivative of the observed process Z with respect to the reference process Y and

$$\mathrm{E}\Big[\log \frac{\Lambda(Z)}{L(\theta, Z)}\Big] \tag{6.9}$$

is the continuous path version of the KL divergence between the observed process Z and the diffusion network process X^θ. Taking gradients

$$\nabla_\theta v(\theta) = \mathrm{E}\Big[F(\theta, Z)\Big] + \nabla_\theta \log q(\theta) \tag{6.10}$$

where $F(\theta, x)$ is the continuous time version of the Fisher score with respect to θ, i.e,

$$\begin{aligned} F(\theta, x) &= \nabla_\theta \log L(\theta, x) && (6.11)\\ &= \nabla_\theta \log p(x_0 \mid \theta) + \int_0^T \nabla_\theta \mu(t, \theta, x_t) k(x_t)\Big(dx_t - \mu(\theta, x_t)dt\Big) && (6.12) \end{aligned}$$

6.3.1 Application to Robotics

For Newtonian problems, including articulated robots, the drift function takes the following form

$$\mu(t, \theta, x_t) = a(x_t) + b(x_t) f(t, \theta, x_t) \tag{6.13}$$

where x_t is a vector with the joint angles and joint angle velocities, $b(x_t)$ is the inverse of the robot's inertial matrix, and $a(x_t)$ contains terms due to friction, gravitational, inertial, and Coriolis forces. Thus in this case

$$\nabla_\theta \mu(t, \theta, x_t) = \Big(\nabla_\theta f(t, \theta, x_t)\Big) b'(x_t) \tag{6.14}$$

and

$$F(\theta, x) = \nabla_\theta \log L(\theta, x) = \int_0^T \Big(\nabla_\theta f_t(\theta, x_t)\Big) b'(x_t) k(x_t) \Big(dx_t - \mu(\theta, x_t)dt\Big) \tag{6.15}$$

Let the prior term be of the form $-\theta' q \theta/2$, where q is a positive definite matrix. Let $\phi(x_t)$ be an $n \times p$ matrix of features of the state x_t. For policies of the form $u_t = \phi(x_t)\theta$, the Fisher score takes the following form

$$F(\theta, x) = \int_0^T \phi'(x_t) b'(x_t) k(x_t) \Big(dx_t - a(x_t)dt - b(x_t)\phi(x_t)\theta dt\Big) \tag{6.16}$$

Thus

$$\begin{aligned}\nabla_\theta v(\theta) &= \mathrm{E}\Big[\int_0^T \phi'(Z_t)b'(Z_t)k(Z_t)\Big(dZ_t - a(Z_t)dt - b(x_t)\phi(Z_t)\theta dt\Big)\Big] - q\theta \\ &= \mathrm{E}\Big[\int_0^T \phi'(Z_t)b'(Z_t)k(Z_t)\Big(dZ_t - a(Z_t)dt\Big)\Big] \\ &\quad - (q + \mathrm{E}\Big[\int_0^T \phi'(Z_t)b'(Z_t)k(Z_t)b(Z_t)\phi(Z_t)dt\Big])\theta \end{aligned} \tag{6.17}$$

Setting the gradient equal to zero and solving gives us an analytical solution for the optimal policy

$$\theta^* = \Big(q + \mathrm{E}\Big[\int_0^T \phi'(Z_t)b(Z_t)'k(Z_t)b(Z_t)\phi(Z_t)dt\Big]\Big)^{-1} \tag{6.18}$$

$$\mathrm{E}\Big[\int_0^T \phi'(Z_t)b'(Z_t)k(Z_t)\Big(dZ_t - a(Z_t)dt\Big)\Big] \tag{6.19}$$

6.4 Optimal Control/Reinforcement Learning

In optimal control and reinforcement learning problems our goal is to find a policy that maximizes an expected reward. Let $R(\theta, x)$ be a non-negative scalar function that represent the reward accumulated by the n-dimensional continuous path x in the interval $[0, T]$. Let $v(\theta)$ be the value achieved by policy θ, i.e., the expected reward achieved by process X^θ

$$v(\theta) = \mathrm{E}\Big[R(\theta, X^\theta)\Big] \tag{6.20}$$

One popular way to solve this problems is to take the gradient of the value function with respect to θ and use it as the basis for a gradient-based optimization of v. This is known in the literature as policy gradient. Here we adopt the iterative EM approach first proposed in [18] for discrete time problems: Instead of optimizing v directly, we optimize an auxiliary function Q. Improvements in Q are guaranteed to result on improvements of v. For completeness here we briefly present the continuous time version of the approach. Let κ be a fixed policy. We want to find a new policy θ^* with better value than κ. Let θ be a candidate policy. Note

$$\begin{aligned}
\log \frac{v(\theta)}{v(\kappa)} &= \log \frac{\mathrm{E}\Big[R(\theta, X^\theta)\Big]}{\mathrm{E}\Big[R(\kappa, X^\kappa)\Big]} \\
&= \log \frac{\mathrm{E}\Big[R(\theta, X^\kappa) L(\theta, X^\kappa)/L(\kappa, X^\kappa)\Big]}{\mathrm{E}\Big[R(\kappa, X^\kappa)\Big]} \\
&= \log \mathrm{E}\Big[\frac{R(\theta, X^\kappa)}{\mathrm{E}\Big[R(\kappa, X^\kappa)\Big]} \frac{L(\theta, X^\kappa)}{L(\kappa, X^\kappa)}\Big] \qquad (6.21)
\end{aligned}$$

We decompose the reward $R(\theta, x)$ as the product of two non-negative scalar functions, $q(\theta)$ and $q(\theta)$. The first one represents the a priori desirability of policy θ and the second represents the desirability of a path x. For example, $q(\theta)$ can be used to favor policies that utilize less energy, and $g(x)$ could be used to favor paths that achieve a specific goal

$$R(\theta, x) = q(\theta) g(x) \qquad (6.22)$$

then

$$\begin{aligned}
\log \mathrm{E}\Big[\frac{R(\theta, X^\kappa)}{\mathrm{E}\Big[R(\kappa, X^\kappa)\Big]} \frac{L(\theta, X^\kappa)}{L(\kappa, X^\kappa)}\Big] &= \log \mathrm{E}\Big[\frac{q(\theta) g(X^\kappa)}{\mathrm{E}\Big[q(\kappa) g(X^\kappa)\Big]} \frac{L(\theta, X^\kappa)}{L(\kappa, X^\kappa)}\Big] \\
&= \log \mathrm{E}\Big[\frac{g(X^\kappa)}{\mathrm{E}\Big[g(X^\kappa)\Big]} \frac{q(\theta) L(\theta, X^\kappa)}{q(\kappa) L(\kappa, X^\kappa)}\Big]
\end{aligned} \qquad (6.23)$$

Using Jensen's inequality

$$\begin{aligned}
\log \mathrm{E}\Big[\frac{g(X^\kappa)}{\mathrm{E}\Big[g(X^\kappa)\Big]} \frac{q(\theta) L(\theta, X^\kappa)}{q(\kappa) L(\kappa, X^\kappa)}\Big] &\geq \mathrm{E}\Big[\frac{g(X^\kappa)}{\mathrm{E}\Big[g(X^\kappa)\Big]} \log \frac{q(\theta) L(\theta, X^\kappa)}{q(\kappa) L(\kappa, X^\kappa)}\Big] \\
&= Q(\kappa, \theta) - Q(\kappa, \kappa) \qquad (6.24)
\end{aligned}$$

where for any two policies κ,θ, the Q function is defined as follows

$$Q(\kappa, \theta) = \mathrm{E}\Big[\frac{g(X^\kappa)}{\mathrm{E}\Big[g(X^\kappa)\Big]} \log \Big(L(\theta, X^\kappa)\Big)\Big] + \log q(\theta) \qquad (6.25)$$

Thus

$$\log v(\theta) - \log v(\kappa) \geq Q(\kappa, \theta) - Q(\kappa, \kappa) \tag{6.26}$$

It follows that if we find a value of θ such that

$$Q(\kappa, \theta) > Q(\kappa, \kappa) \tag{6.27}$$

then

$$v(\theta) > v(\kappa) \tag{6.28}$$

Thus rather than optimizing $v(\theta)$ we can iteratively optimize $Q(\kappa, \theta)$. We start with a initial policy κ and find a new policy θ for which $Q(\kappa, \theta) > Q(\kappa, \kappa)$. We then let κ be the new policy and optimize again $Q(\kappa, \theta)$ with respect to θ. We iterate this procedure until convergence. The approach guarantees that after each iteration we will obtain a policy with better value than the previous one. Thus, we guaranteed convergence to a local maximum of the value function. Here we will optimize $Q(\kappa, \theta)$ with respect to θ by finding analytical solutions that set the gradient of Q with respect to θ equal to zero.

6.5 Model Based DNA

Taking the gradient of $Q(\kappa, \theta)$ with respect to θ

$$\nabla_\theta Q(\kappa, \theta) = \nabla_\theta \log q(\theta) + \mathrm{E}\Big[\frac{g(X^\kappa)}{\mathrm{E}\Big[g(X^\kappa)\Big]} F(\theta, X^\kappa)\Big] \tag{6.29}$$

Note

$$\mathrm{E}\Big[\frac{g(X^\kappa)}{\mathrm{E}\Big[g(X^\kappa)\Big]} F(\theta, X^\kappa)\Big] = \mathrm{E}\Big[\frac{g(S)}{\mathrm{E}\Big[g(S)\Big]} W(S) F(\theta, S)\Big] = \mathrm{E}\Big[F(\theta, S)\Big] \tag{6.30}$$

Where S is a process such that the Radon-Nykodim derivative of X^κ with respect to S is as follows

$$W(x) = \frac{\mathrm{E}\Big[g(X^\kappa)\Big]}{g(X^\kappa)} \tag{6.31}$$

One way to get unbiased estimates of expected values with respect to S is to sample from X^κ and to weight the samples times $g(X^\kappa)$. Note

$$\nabla_\theta Q(\kappa, \theta) = \mathrm{E}\Big[F(\theta, S)\Big] + \nabla_\theta \log q(\theta) \tag{6.32}$$

is the equation for solving the unsupervised learning problem we studied before. In other words by maximizing $Q(\kappa, \theta)$ with respect to θ we are implicitly trying to approximate a distribution similar to that produced by X^κ but with each path x weighted by its value $g(x)$.

6.5.1 Application to Robotics

As in the unsupervised learning case, for policies linear on the parameters we can find the optimal value of θ analytically

$$\begin{aligned}\theta^* = \Big(q + \mathrm{E}\Big[\int_0^T \phi'(S_t)b(S_t)k(S_t)b(S_t)\phi(S_t)dt\Big]\Big)^{-1} \\ \mathrm{E}\Big[\int_0^T \phi'(S_t)b'(S_t)k(S_t)\Big(dS_t - a(S_t)dt\Big)\Big]\end{aligned} \tag{6.33}$$

Once we found θ^* we let $\kappa = \theta^*$ and iterate again until convergence.

6.6 Model Free DNA

In the previous algorithm the Fisher score requires knowledge of the system dynamics. For example, in (6.33) we need to know the a, b functions. This includes the robot's inertial matrix, gravitational forces, friction forces etc. In some cases system identification methods may be used to develop an approximate model. However this approach becomes difficult when contact forces are involved, and unravels when the robot dynamics include interaction with people. Here we show that a model free version of the DNA algorithm can be developed in a straightforward manner. To do so we expand the network state vector X to include the state of the original diffusion network process O that we wish to control (e.g., a robot) and an auxiliary hidden random process H that determines the policy used to control O, i.e. $X_t = (O_t, H_t)'$. We let μ_o, σ_o represent the drift and dispersion of the original diffusion network, p_o its initial state distribution, and B_o the Brownian motion that drives the uncertainty of the original network. Thus

$$dO_t^\theta = \mu_o(t, H_t^\theta, O_t^\theta)dt + \sigma_o(O_t^\theta)dB_{o,t} \tag{6.34}$$

Note that H_t^θ represents what we used to call θ, i.e, H_t^θ determines the policy used at time t to control the original network O. Under this formulation we let the policy that controls O change with time. The dynamics of the policy H_t are determined by its initial state distribution p_h, its own drift function μ_h and dispersion processes σ_h. We let the policy dynamics and its initial state distribution p_h be a function of a set of hyperparameters θ, i.e.,

$$dH_t^\theta = \mu_h(t, \theta, H_t^\theta)dt + \sigma_h(H_t^\theta)dB_{h,t} \tag{6.35}$$

with initial distribution $p_h(h_0 \mid \theta)$. The combined system $X = (O, H)$ is governed by the standard diffusion network equations, and thus we can apply to it the DNA framework

$$dX_t^\theta = \mu(t, \theta, X_t^\theta)dt + \sigma(X_t)dB_t \tag{6.36}$$

with

$$X_t^\theta = \begin{pmatrix} O_t^\theta \\ H_t^\theta \end{pmatrix}, \quad \mu(t, \theta, x) = \begin{pmatrix} \mu_o(t, h_t, o_t) \\ \mu_h(t, \theta, h_t) \end{pmatrix}, \quad B_t = \begin{pmatrix} B_{o,t} \\ B_{h,t} \end{pmatrix} \tag{6.37}$$

$$\sigma(x_t) = \begin{pmatrix} \sigma_o(o_t) & 0 \\ 0 & \sigma_h(h_t) \end{pmatrix}, \quad p(o_0, h_0 \mid \theta) = p_o(o_0)p_h(h_0 \mid \theta) \tag{6.38}$$

where $x = (o, h)$ is an arbitrary path. Note now the path x combines a path o of the original network states (e.g., the robot angles and angular velocities) and a path h of policy parameters. The Fisher score with respect to θ takes the usual form

$$\begin{aligned} F(\theta, x) &= \nabla_\theta \log L(\theta, x) && (6.39) \\ &= \nabla_\theta \log p(x_0 \mid \theta) + \int_0^T \nabla_\theta \mu(t, \theta, x_t) k(x_t) \Big(dx_t - \mu(\theta, x_t)dt \Big) && (6.40) \end{aligned}$$

Most importantly, since $\nabla_\theta \mu_o(t, h_t, o_t) = 0$ the Fisher score no longer depends on the dynamics of the original network

$$F(\theta, x) = \nabla_\theta \log p(x \mid \theta) + \int_0^T \nabla_\theta \mu_h(t, \theta, h_t)(\sigma_h'(h_t)\sigma_h(h_t))^{-1}\Big(dh_t - \mu_h(t, \theta, h_t)dt\Big) \tag{6.41}$$

Note the policy dynamics μ_h, σ_h are chosen by us, and the dynamics of the original network are no longer needed to computer the Fisher score. Below we show an example.

Example: We choose the policy parameter h_0 from a Gaussian distribution with mean vector θ_1 and inverse covariance matrix θ_2. We then let that policy evolve in a manner that is not dependent on θ, e.g

$$dH_t = \alpha_t dt + \beta_t(H_0 - H_t)dt + \sigma_h(H_t)dB_{h,t} \tag{6.42}$$

where α_t, β_t are fixed functions of time. In this case the Fisher score with respect to θ_1 is as follows

$$F_1(\theta, x) = \nabla_{\theta_1} \log L(\theta, x) = \nabla_{\theta_1} \log p(h_0 \mid \theta) = \theta_2(h_0 - \theta_1) \tag{6.43}$$

and the Fisher score with respect to θ_2 takes the following form

$$F_2(\theta, x) = \nabla_{\theta_2} \log L(\theta, x) = \frac{1}{2}\theta_2^{-1} - \frac{1}{2}(h_0 - \theta_1)(h_0 - \theta_1)^T \tag{6.44}$$

Thus the gradient of the $Q(\kappa, \theta)$ function with respect to θ_1 looks as follows

$$\nabla_{\theta_1} Q(\kappa, \theta) = \mathrm{E}\Big[\frac{g(X^\kappa)}{\mathrm{E}\Big[g(X^\kappa)\Big]} F_1(\theta, X^\kappa)\Big] \tag{6.45}$$

$$= \theta_2 \mathrm{E}\Big[\frac{g(X^\kappa)}{\mathrm{E}\Big[g(X^\kappa)\Big]}\Big(H_0^\kappa - \theta_1\Big)\Big] \tag{6.46}$$

Setting the gradient to zero and solving for θ_1 we get

$$\theta_1^* = \frac{\mathrm{E}\Big[g(X^\kappa)H_0^\kappa\Big]}{\mathrm{E}\Big[g(X^\kappa)\Big]} \tag{6.47}$$

The gradient of the $Q(\kappa, \theta)$ function with respect to θ_2 looks as follows

$$\nabla_{\theta_1} Q(\kappa, \theta) = \mathrm{E}\Big[\frac{g(X^\kappa)}{\mathrm{E}\Big[g(X^\kappa)\Big]} F_2(\theta, X^\kappa)\Big] \tag{6.48}$$

$$= \frac{1}{2}\theta_2^{-1} - \frac{1}{2}\mathrm{E}\Big[\frac{g(X^\kappa)}{\mathrm{E}\Big[g(X^\kappa)\Big]}(H_0^\kappa - \theta_1)(H_0^\kappa - \theta_1)^T\Big] \tag{6.49}$$

Setting the gradient to zero and solving for θ_1 we get

$$(\theta_2^*)^{-1} = \frac{\mathrm{E}\Big[g(X^\kappa)(H_0^\kappa - \theta_1^*)(H_0^\kappa - \theta_1^*)^T\Big]}{\mathrm{E}\Big[g(X^\kappa)\Big]} \tag{6.50}$$

At the end of the iteration we let $\kappa = \theta^*$ and find a new θ^* using equations (6.47) and (6.50). We keep iterating until convergence is achieved.

6.7 Prior Work

The original work on unsupervised diffusion network adaptation was first presented in [55] and later extended in [50, 53, 54]. The discrete time version of the EM approach to reinforcement learning was first presented in [18]. A similar approach was adopted by the POWER algorithm [37], one of the most successful reinforcement learning approaches to robot control to date. The DNA framework is also related to path integral (PI) approaches to reinforcement learning. The PI framework and the DNA framework are designed for continuous time systems and make full use of the powerful machinery of stochastic calculus. However PI does not adopt the EM approach and instead uses the continuous time HJB optimality equation. In doing so it makes assumptions about the structure of the system dynamics that are not needed in DNA. The PI framework was greatly refined in [12] for applications to model free robot control. This resulted on the PI2 algorithm, which is considered comparable in performance to the POWER algorithm. In recent years researchers have made a number of changes to PI2 that, while not theoretically motivated, have been shown to improve performance. This changes include the use of non-Brownian motion on the policy parameters [77], and the application of blind optimization methods, like Covariance Matrix Adaptation [75]. These variations can be theoretically grounded within the DNA framework.

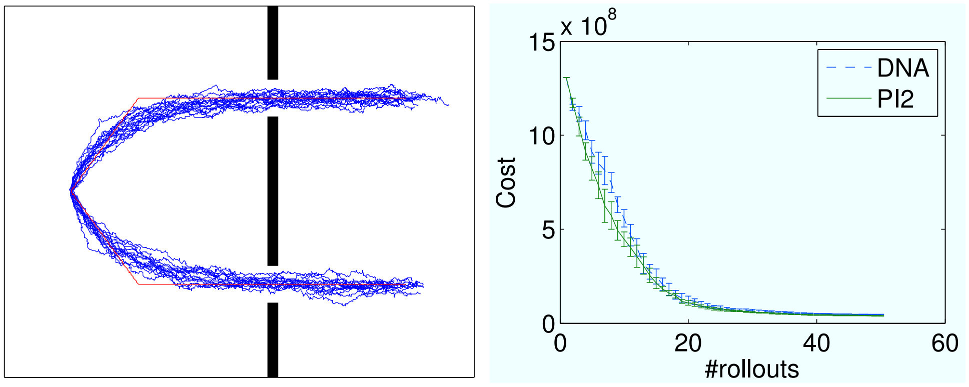


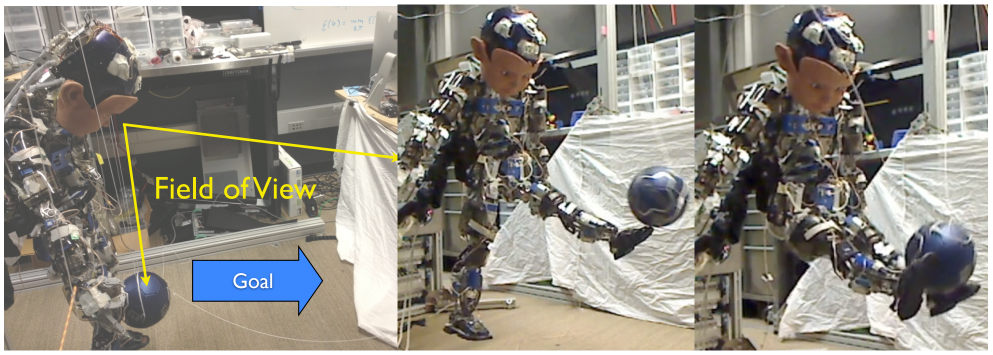
Figure 6.1: Left: Double Slit Diffusion Task. Right: Learning Curves for the Via Point Task

6.8 Simulations and Experiment with Humanoid Robot

Figure 6.1 Left shows an illustration of imitation learning. The task was to control a point mass to move between to chambers connected by two slits. The controller observed two paths, shown in red in the figure, that take the point mass from the left side chamber to the right side chamber. The features for the control policy were radial basis functions. The actions are force vectors that can vary as a function of time, position, and velocity. The figure shows the observed path distribution (red) and the distribution of paths learned by the robot in one single trial (blue).

We then compared the model-free DNA algorithm to the PI2 algorithm [12] on a standard reinforcement learning problem. The task was to control a point mass on a 2D space to move it in 1 second from an initial location to a target location via a intermediate location, while minimizing the total control cost. The control policy was parameterized using dynamic motor primitives [12]. Figure 6.1 Right compares the convergence speed of DNA and PI2. We did not observe significant difference in performance between the two algorithms despite the fact that DNA is much simpler to implement than PI2.

We then conducted a real world experiment on a robot named Diego-San,



(a) soccer experiment setup (b) one leg strategy (c) two leg strategy

Figure 6.2: Humanoid robot soccer kicking experiment

one of the most complex humanoids built to date. It has 94 degrees of freedom that control 38 body joints (each joint has a pair of agnostic and antagonistic pneumatic actuators) and 18 facial muscles. The robot was designed to simulate the control properties of a human infant body. In general it is very difficult to control using standard robot control approaches due to the compliance, and elasticity provided by the pneumatic actuators. The task was to kick a ball as high as possible with all the actuators on the legs. Diego-San's trunk was fixed to a harness to simulate a similar study performed with human infants. The reward signal was the height of the ball, provided in real time by the cameras in Diego-Sans' eye sockets. The control policy was parameterized using dynamic motor primitives [12]. Figure 6.2(a) shows the field of view. In one condition we positioned the ball close to the left foot and in the other condition we positioned the ball equidistant from his feet. Diego-San had to learn which of the actuators had an effect on the ball height, and to synchronize them in a manner that maximized the impact of the kick. When the ball was positioned close to the left foot it took Diego-San 60 trials, a total of 3 minutes, to learn to kick the ball efficiently. He did so by swinging his left leg and keeping his right leg stationary(Fig. 6.2(b)). When the ball was positioned equidistant between the legs, Diego-San learned in about the same time a strategy that we had not anticipated and that turned out to better than kicking: he held the ball with his feet and moved it upwards (Fig. 6.2(c)).

6.9 Acknowledgment

Support for this work was provided by NSF grants IIS-INT2-0808767. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Chapter 6, in full, is submitted as “Tingfan Wu and Javier Movellan. DNA: Dynamic network adaptation for optimal control. In *submission to Neural Information Processing Systems 2013*”. The dissertation author was the primary investigator and author of this paper.

Chapter 7

Collecting A Developmental Dataset of Reaching Behaviors: First Steps.

Abstract: At birth infants are faced with the difficult task of learning to control their bodies to interact with the physical and social world around them. From a motor control point of view the problem infants face is monumental, due to the high number of degrees of freedom, high compliance, and low-repeatability provided by the human musculoskeletal system. In fact the complexity of the control tasks that infants solve so effortlessly far outstrips the abilities of the most sophisticated approaches to motor control and artificial intelligence. Observing how infants develop such motor skills as they interact with objects and caregivers may provide insights for new approaches to robotics. One obstacle for progress in this area is the lack of datasets that simultaneously capture the motion of multiple limbs of infants and caregivers across the developmental process. Here we present our first steps towards the collection of one such datasets, focused on the development of reaching behaviors. We describe the technical and logistic problems we faced so far and the solutions we found. We also show preliminary analyses that illustrate how the collected data suggests new approaches to motor control in robotics, and to theories of motor development in infants.

7.1 Introduction

The sensory-motor systems of the brain generate movements that are compliant and non-repeatable, yet remarkably well-adapted to an unstructured, uncertain and non-stationary world. Contrary to this, the fields of robotics and motor control has for the most part focused on simplifying the control problem by using stiff, highly geared actuators, emphasizing repeatability over compliance, and avoiding unstructured conditions. This approach worked well for industrial applications and revolutionized the assembly line. However, in order to develop robotic technology that could transform daily life, it is important to focus on robots that approximate the control properties of the human body. The problem is that due to the complexity, compliance, non-repeatability, and temporal dynamics of the human body, most of the control schemes used in current practice become inapplicable.

Infants face the very difficult task of learning to control their bodies to interact efficiently with the physical and social world around them. From a motor control point of view this problem is formidable, due to the high number of degrees of freedom, high compliance, and low-repeatability provided by the human musculoskeletal system. The solution to this problem eludes the most sophisticated approaches to robotics and artificial intelligence. Yet infants solve this problem seamlessly within a few years of life. Understanding how this is done and reproducing this process in robots may have profound scientific and technological consequences. One obstacle for progress in this line of work is the lack of datasets that jointly capture the joint development of infant and caregiver body motions at high temporal and spatial resolution. Such datasets may provide the key to reverse engineer human motor development and to synthesize it in new human-mimetic control algorithms for robots. Another obstacle for progress is the lack of robot systems that approximate the complexity, compliance and control dynamics of the human body. Based on these motivating ideas, we have been pursuing a project whose goal is to gain a computational understanding of how infants learn to control their bodies. One component of this project focuses on the development of a sophisticated humanoid robot named Diego San (see Fig.7.1), that approxi-

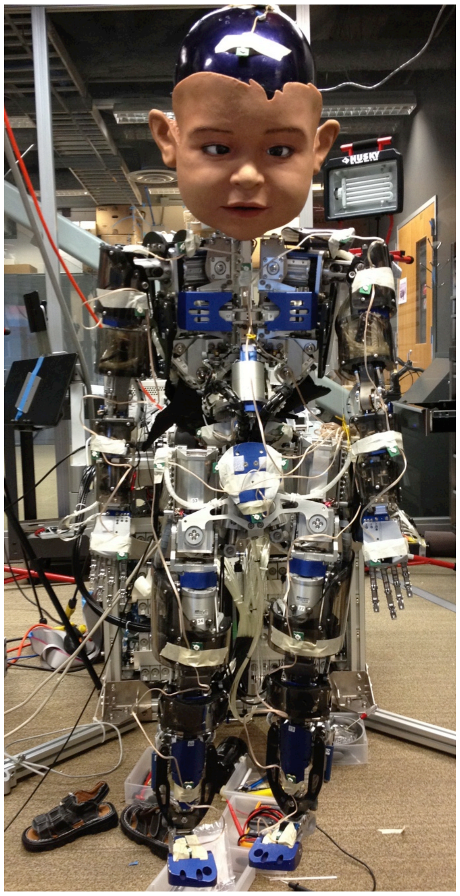


Figure 7.1: Diego: a humanoid robot that approximates human body complexity and dynamics.

mates the complexity, compliance and control dynamics of the human body. The other component, which is the focus of this paper, focuses on the collection of a motion capture dataset to understand the development of reaching behaviors in the context of physical and social interactions with the world. While the use of motion capture has recently become popular in the motor development literature, a unique aspect of our work is the attempt to simultaneously capture the motion of the entire body (arms, legs, trunk, head) in both the infant and caregiver. In this paper, we present our initial steps towards capturing such data. We describe the technological and logistic problems we are facing and the solutions we are finding to these problems. We also describe our preliminary steps analyzing the data obtained so far.

7.2 Setup and Data Collection Procedure

Although motion capture technology is now quite mature, most modern systems are optimized for single adults in standing postures with low levels of occlusions. However, our experiment focuses on infants in a natural lying posture interacting with their caregivers. Based on pilot work with several motion capture systems, we decided to use the PhaseSpace Impulse, because of the fact that it uses active LED markers. In contrast to systems using passive markers (such as the Vicon system), active LED markers have individual digital signatures so that the system can easily tell the identity of each marker at each point in time. This is particularly important in setups, like ours, with a large number of occlusions. Another alternative would have been to use magnetic based motion capture. Unfortunately due to the magnetic characteristics of our experimental room this was not an option we could use. A disadvantage of active markers is that they need to be powered via cables attached to a small wireless driver (see Figure 7.2). Our final system utilized 10 infrared cameras. The cameras were installed around the perimeter of a 3.3m$\times$3.3m sound-attenuated playroom for recording. Additionally, four fixed video cameras are used to supplement the infrared system. This includes a small headband camera worn by the mother to record the infant's expressions during the sessions.

7.2.1 Mocap Suit Design

As there are no commercially available motion capture suits for infants we had to develop our own. This apparently mundane task turned out to be surprisingly difficult. While we have made significant progress, every capture session teaches us new lessons on how to improve the design. Originally we had 4 design criteria:

- **Scalability** the system needed to allow for the addition and removal of LED markers depending on task demands.
- **Low profile** the wiring and placement of the suit markers should not interfere with infant movement, or distract the infant during sessions. Additionally, the markers could not touch the infant's skin.
- **Durability** the longitudinal design involved multiple observation sessions a week with each infant being seen over the course of 8-10 weeks. The suit needed to be robust enough to functionally endure these sessions.
- **Redundancy** capturing the motion of an infant lying in the prone position with his or her mother leaning over the infant presented significant line of sight problems that are not typically encountered in traditional motion capture on adults. Thus, the suit had to be designed with a high degree of marker redundancy to optimize the amount of time each part of the infant's body was able to be tracked by the PhaseSpace system.

Each system of markers connects to an individual wireless LED driver, which supplies power for the markers and sends data from the markers to the server. Initially, the 24-gauge ribbon cable that connects the LED driver was spliced and an array of parallel LEDs was connected to each port in the LED driver. The markers were then attached using Velcro adhesive strips connected to an infant onesie. The procedure was to dress the infant in the onesie, attach the

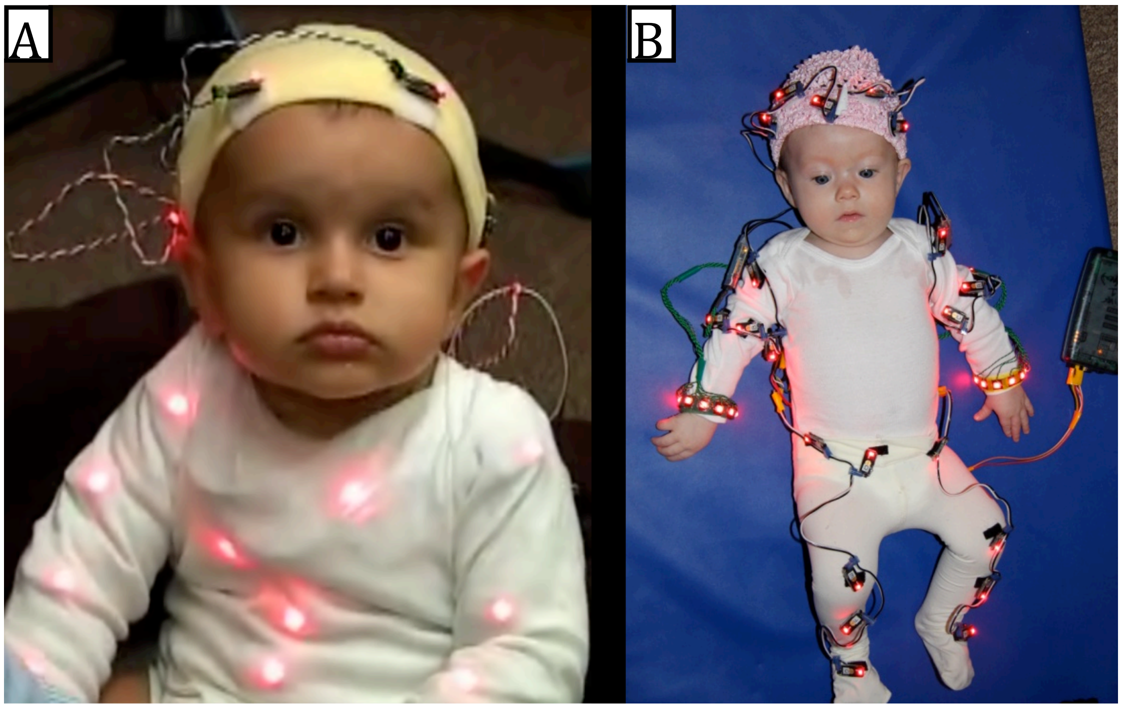


Figure 7.2: Versions of the infant suit. A) An early suit prototype, note the diffusion of light from the LEDs on the infant's arms and body created by the external onesie. B) The current suit design, note the absence of the external onesie, and the additional markers

markers, and then place another onesie over the markers. (See Fig. 7.2A). After several pilot sessions with using this configuration, we found that the additional onesie caused diffusion in the LEDs, causing inconsistent tracking of markers.

A modified suit was designed (see Fig.7.2B), this time using series connections from the LED driver with connectors that could be crimped onto the wire directly. The series connections both reduced the amount of wire and made for a more logical arrangement of the LED strings. Each LED string could accommodate up to 7 markers per string. Five LED strings were constructed, two for the arms, two legs, and a head/body string. Given earlier tracking problems caused by the outermost onesie (see Fig.7.2A), we piloted a session with only the inner onesie and the now external wiring. Surprisingly, the pilot infant did not appear to notice the external wiring or the LED markers. We are currently using this configuration for all data acquisition. To date, no infants have spent a noticeable amount of time gazing at or touching the markers or other components of the suit.

7.2.2 Experimental Procedure

On each motion capture session infant and caregiver are brought into the motion capture playroom and the infant is placed on his/her back with caregiver facing the child (see Fig. 7.3). Caregiver is instructed to interact with infant in the following conditions:

1. Face to face without any toys for 2 minutes.
2. Caregiver offers a series of toys for 10 minutes with the goal of initiating reaching.
3. Unstructured infant play with mobile for 1 minute.

The session ends with the examiner presenting toys to the infant in predetermined series of positions.

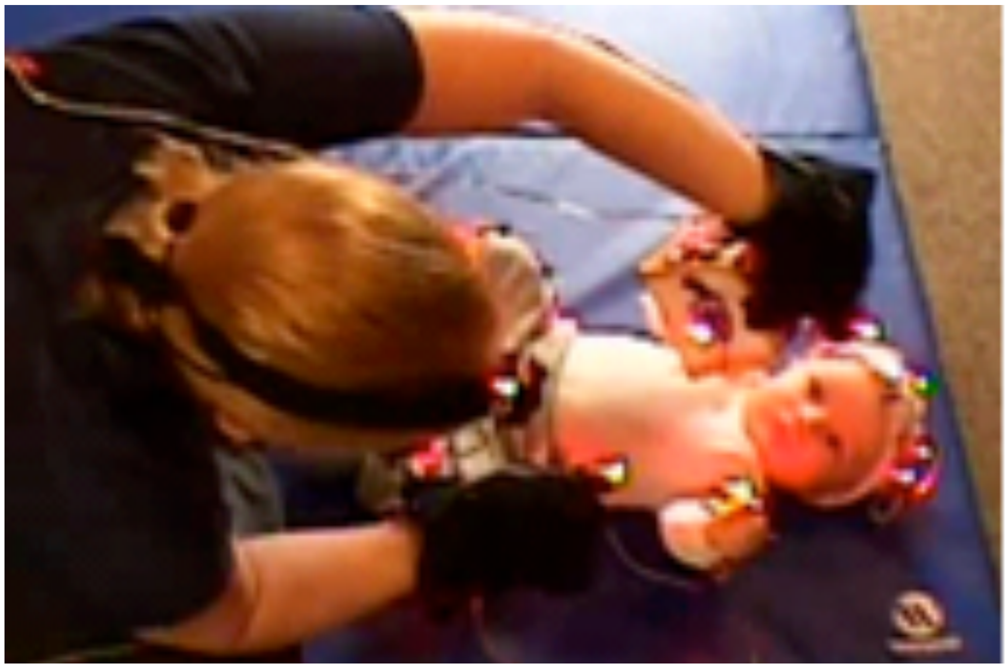


Figure 7.3: An example of our experiment scenario. The infant is looking at the toy shown by the mother.

7.3 Preliminary Results

7.3.1 The Nature of the Developmental Process

The classic robot control literature typically measures the difficulty of a control problem in terms of the number of degrees of freedom and joint compliance. A robot with a large number of compliant joints, results on a highly dimensional problem with tightly coupled non-linear dynamics. Such problems are very difficult to solve with current approaches. A classical solution is to reduce the effective number of degrees of freedom by temporarily or permanently stiffening some joints, so that the control problem decouples into a small set of independent equations. Perhaps influenced by classic robotics theory, the literature in developmental psychology has adopted a similar point of view. For example, it has been proposed that infants' early attempts to reach engage a low number of degrees of freedom. From this point of view development proceeds from engagement of less to engagement of more degrees of freedom.

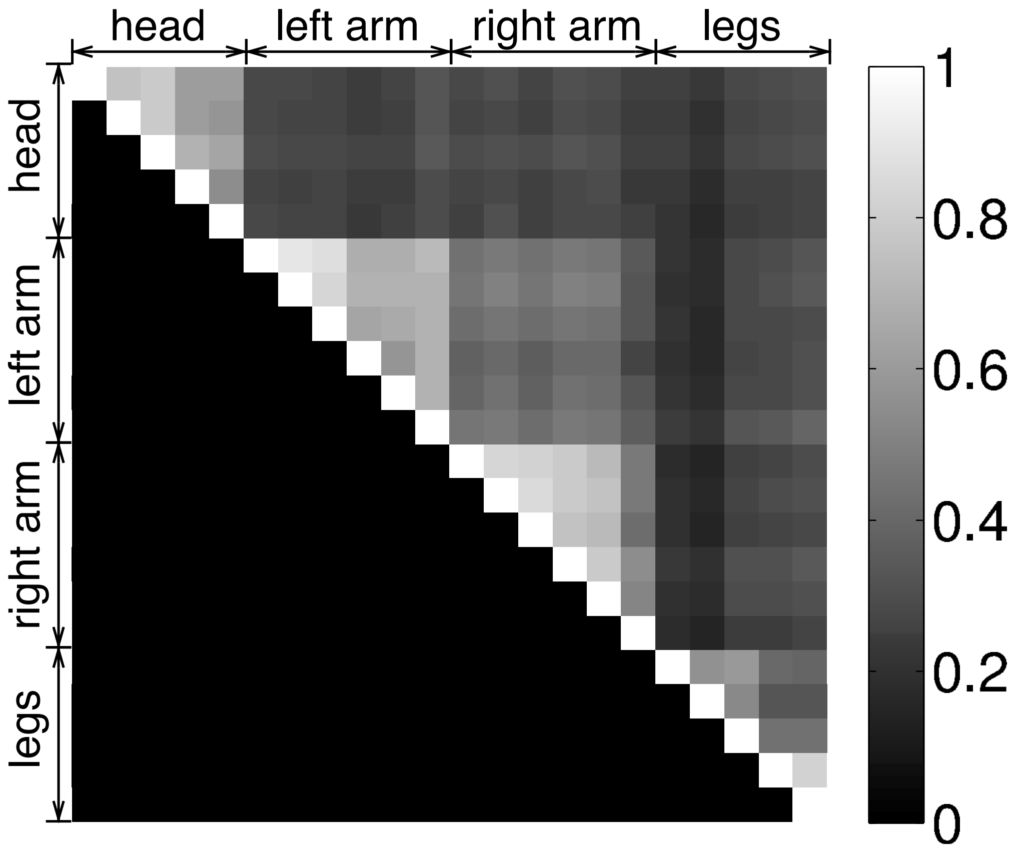


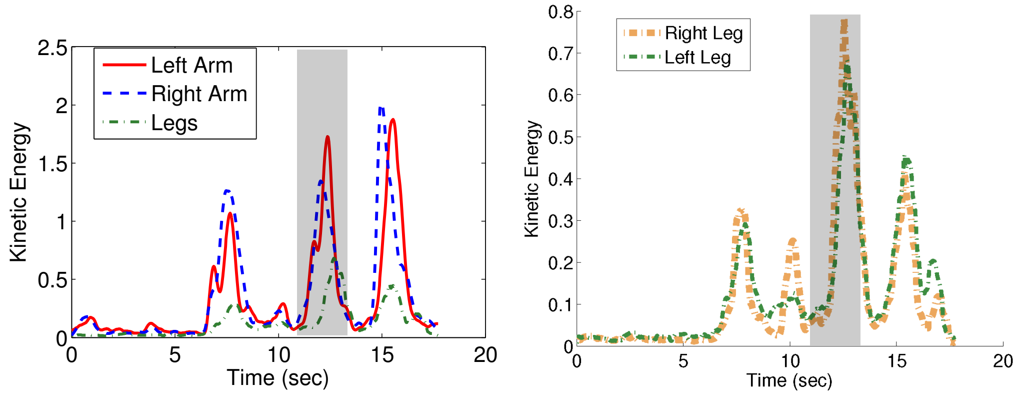
Figure 7.4: correlation of movement magnitude between body regions by marker

Unfortunately much of the behavioral evidence supporting this less-to-more view of motor development has focused on laboratory experiments that capture the motion of a single infant arm as it reaches towards an object in very restricted conditions [6, 7].

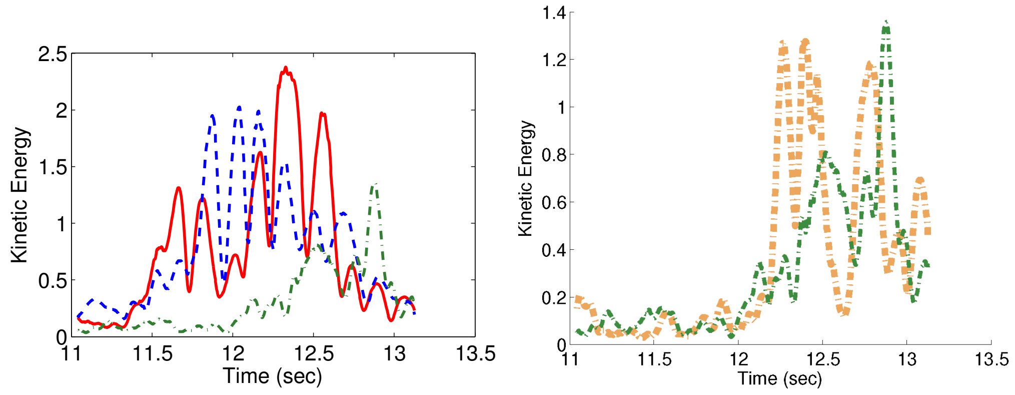
Thus our dataset will offer a unique opportunity to test current theories of motor control by analyzing how the different parts of the body, not just an individual arm, coordinate throughout development. To begin exploring this issue we performed some preliminary analysis on the data obtained so far, focusing on how hands and legs move while mom offers baby a toy for him/her to reach. The analysis was performed on a single motion capture session of an 18 week old infant. First, the instantaneous motion energy of each marker was estimated by squaring the displacement between adjacent frames. A displacement was counted only when a marker is visible in both frames and the displacement is reasonable (speed < 24m/s) so as to exclude missing markers and motion marker jitter. The displacement values were normalized per marker across the entire session. Second, the temporal correlations between the average group motion energy for marker located on two segments of the infant's body were computed over the entire session. If two limbs were moving at the same time regardless of the direction of movement, there will be a high correlation between the corresponding markers (Fig.7.4).

Not surprisingly, high correlations were observed for markers located within each of the following groups: head, left arm, right arm, left leg and right leg. Interestingly, high correlations were also observed between markers on different limbs (e.g., 0.4 Pearson correlation coefficient between the left and right arm). Basically it appears that infants were simultaneously using all the limbs (the two arms and legs). To explore the temporal unfolding of inter-limb correlation, we plotted the motion energy over time as the infant attempted to reach for an object (see Figure 7.5(a)).

Note that the two arms and legs move in synchrony bursts that lasts approximately 4 seconds. Looking closer at the second burst (see Fig.7.5(b)), we see a fine grain motion energy plot. Figure 7.6 shows the actual movement of all tracked groups in the horizontal plane.



(a) a 20s segment (smoothing window = 1s)



(b) zoom-in the second episode (shaded) in (a) (smoothing window = .05s)

Figure 7.5: Kinetic energy for left arm, right arm, and legs during a 20 second segment

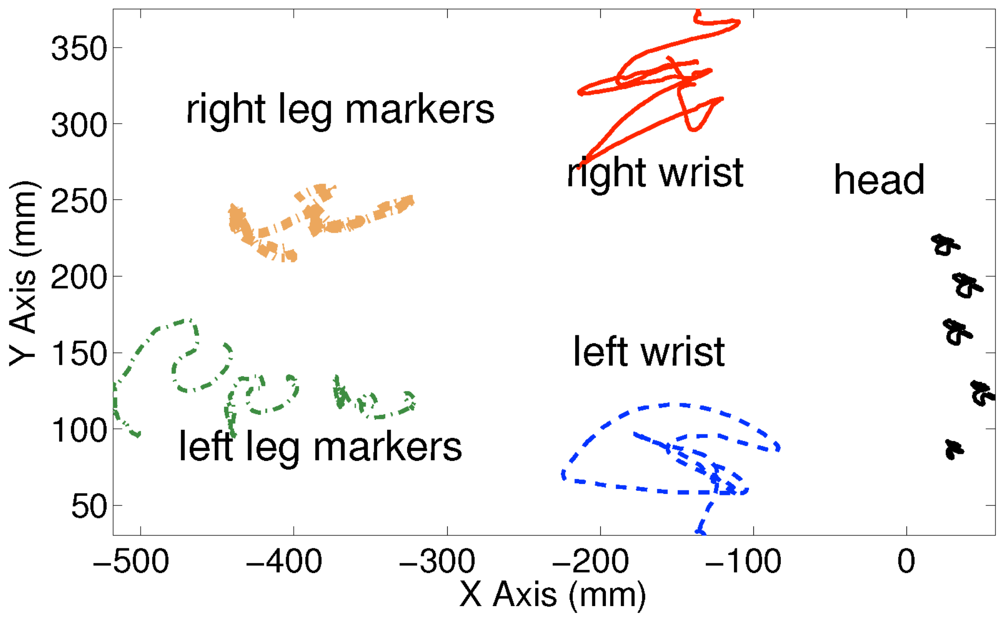


Figure 7.6: Marker trajectories (projected on to the X-Y plane) of the movement episode shown in Fig. 7.5

Thus the evidence, while preliminary, does not appear to be consistent with a less-to-more view of motor development. If anything early in development children appear to use many more joints than necessary. Rather than reaching with a single arm, they appear to be simultaneously reaching with both their arms and limbs.

7.3.2 Analyzing Mother Infant Contingency Structures

An important component of the dataset we are collecting is the fact that we simultaneously motion capture infant and caregiver data. Our use of motion capture technology allows the analysis of contingencies between behavior of mother and infant at a very high spatial and temporal resolution with minimal human coding. The issue of which actions of a particular partner within the context of a dyadic interaction engender predictable (contingent) responses of the other partner has long been a focus of developmental science [26, 34]. Most work in this area follows two basic steps: first use coarse-grained hand-coded variables to describe the dyadic interaction (e.g. coding infant gaze as being to one of several candidate loci of attention); second, use techniques such as cross-correlation to identify the temporal properties of influence between mother and infant [26]. Here we address this issue by examining to what extent the 3D position of a marker on the infant can be predicted by the 3D position of a marker on mother and vice-versa. Specifically, we are interested in the following questions:

1. Movement of which parts of mother's body are most predictive of the orientation of the infant's head?

2. What is the temporal delay for each possible location on the mother's body that is maximally predictive of the orientation of the infant's head?

In this section we present our initial steps in developing computational strategies to address these questions. We used windowed Canonical Correlation Analysis [78] of the 3D position of one marker on the mother and the 3D position of the marker at the center of the infant's forehead.

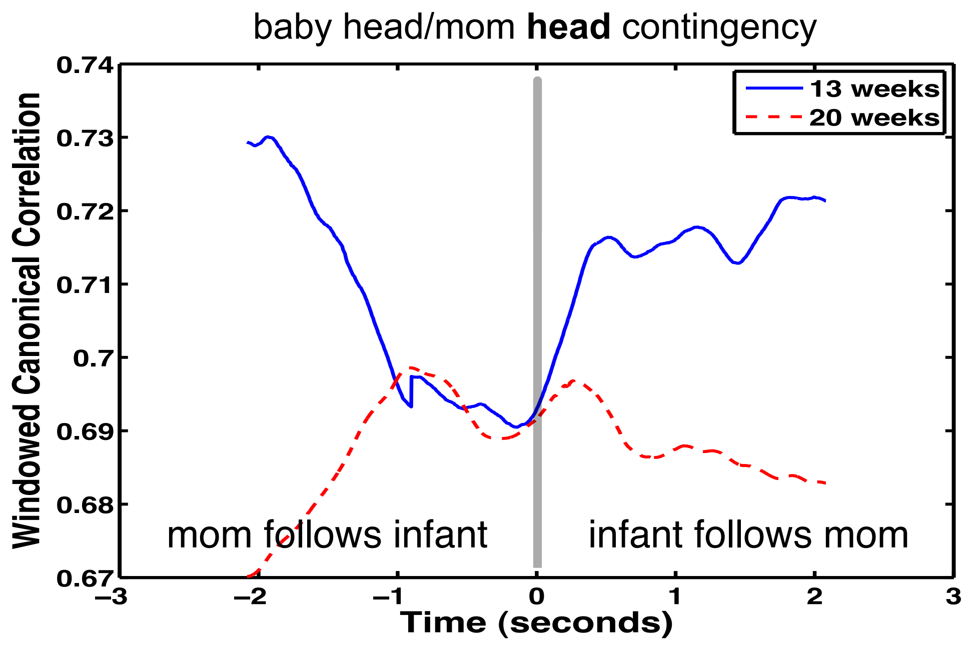
Within the context of time series analysis, Canonical Correlation Analysis is a technique for projecting each of two multivariate time series to create two univariate time series that are maximally correlated. In the context of motion capture analysis we can view this as computing two directions of motion: one for marker 1 and one for marker 2 such that the motion of each of these markers projected onto the computed directions is maximally correlated. For example, Canonical Correlation might project the movement of mother's hand in the direction transverse to the infant's body and the infant's head motion direction along the same direction, indicating that motion of mother's hand across the infant's body is predictive of shaking of the infant's head. For each experiment one time series was composed of 3D positions of the marker on the center of the infant's forehead and the second time series was the 3D position of a particular marker on the mother. We used two marker locations on mother to investigate which was most informative of the infant's head orientation. Specifically, one marker was on mother's right hand and the second marker we investigated was at the center of mother's forehead.

We computed the canonical correlation as a function of time by using a running temporal window. We utilized two different window lengths: one short window (2.08 seconds) that gives us fine temporal resolution but a somewhat noisy estimate of the local canonical correlation, and a longer window (16.67 seconds) that gives us a more stable characterization of the canonical correlation that is useful for characterizing coordination over entire sessions. For each session analyzed and for each time window length, the canonical correlation was computed using a sliding window with 50% temporal overlap between adjacent windows (e.g. for the 16.67 second window the first temporal window would be 0s to 16.67s and the second would be 8.34s to 25s, followed by 16.67s to 33.33s).

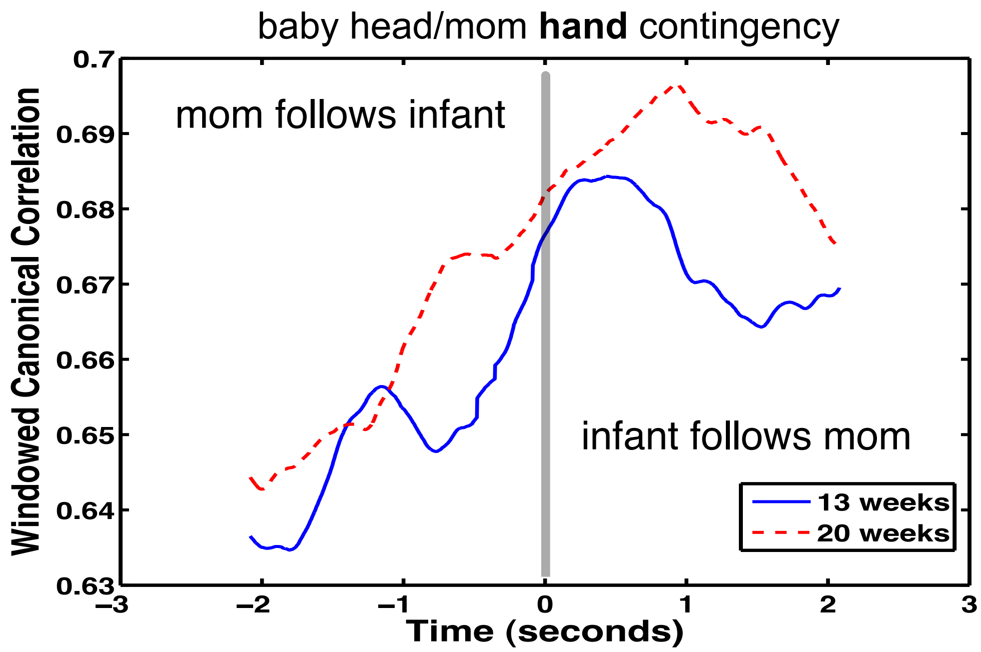
We examined one particular mother-infant dyad (Subject 10) at two time periods, 7 weeks apart. The Infant was 13 weeks old at the first session and 20 weeks old at the second. For each session we computed windowed canonical correlations between the two markers on mother and the marker at the center of the infant's forehead. The plots in Fig.7.7 show the mean canonical correlation values for each of the sessions as a function of delay. Peaks in the negative region indicate periods in which the movement of the mother's marker was predicted by a preceding movement of the infant's head, i.e., mom was following the infant. Peaks in the positive region indicate periods in which the infant is following mom. Figure 7.7 (a) suggests an approximately equal number of episodes in which mom's head motion follows infant's head motion and episodes in which infant head motion follows mom head motion. The time delay in these contingencies decreases with development. Figure 7.7 (b) shows that for the most part infant's head follows mom's hand movement, rather than mom's hand movement following infant's head. Again the temporal delay of this contingency decreases with development.

In addition to analyzing the mean canonical correlation over the entire session, we also investigated whether short-time Canonical Correlation Analysis (window length of 2.08 seconds) could be used to to spotlight meaningful episodes of mother-infant interaction. The 3 regions with highest canonical correlation were as follows:

1. Subject 10 at 13 weeks: Mother struck a butterfly mobile that she was holding above the infant with her right hand and approximately 1 second later the infant shook its head. This segment had a high canonical correlation with a



(a)



(b)

Figure 7.7: Windowed Canonical Correlation for two sessions between infant head movement and mother head movement (top) and infant head movement and mother right hand movement (bottom).

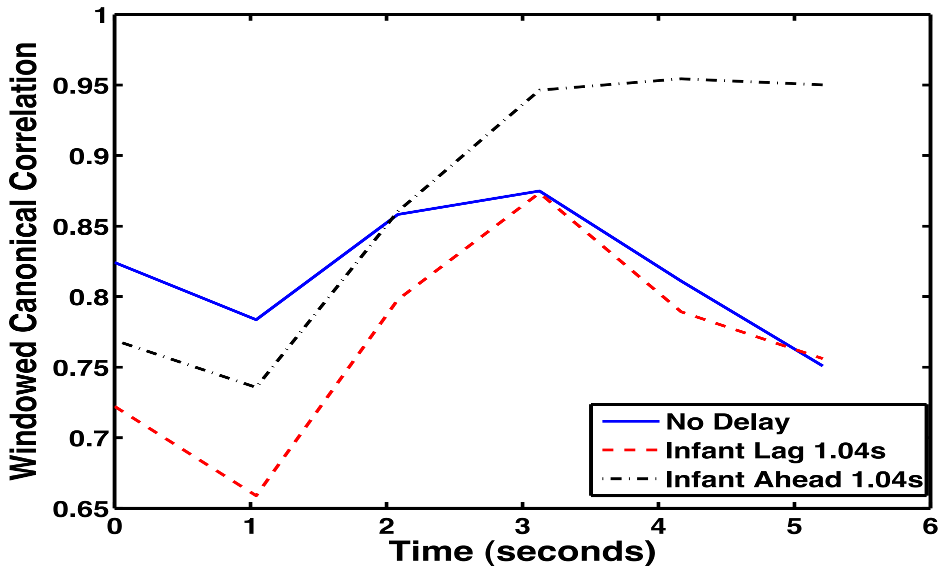


Figure 7.8: Windowed Canonical Correlation for a segment where the infant shakes head and subsequently moves a toy that mother is holding (which causes her hand to move). This event happens at time = 4 seconds on the graph. The event causes the canonical correlation for Infant Ahead 1.04s to spike since the head motion precedes the motion of the marker on mother's right hand.

positive delay of 1.08 seconds whereas the canonical correlation for 0 delay and a delay of -1.08 seconds was lower.

2. Subject 10 at 13 weeks: While the infant grasped a toy that mother holding simultaneously, the infant shook his head and then moved the object with his hand. The movement of the infant's hand caused mother's hand to move as well. This episode is an example of a contingency between mother motion and infant motion that was entirely created by infant motor movements. The Windowed Canonical Correlations are given in Fig.7.8.

3. Subject 10 at 20 weeks: Mother tickled infant with her right hand, 1 second later the infant shook his head

Interestingly, most of the highest coupling events were not instances of the infant tracking the movement of mother's hand, but rather were composed of dynamic interactive events that incorporated both play with objects as well as social interaction.

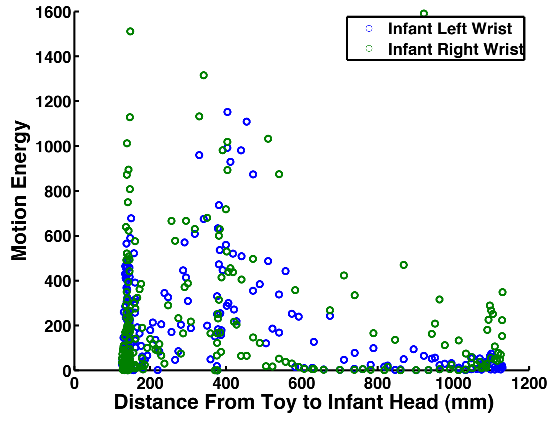
While the present analysis is promising there are several extensions that we will pursue in the future:

- **Multiple Markers** Canonical correlation between multiple points on infant and multiple points on mother may give interesting contingency patterns that go beyond the single body part to single body part analyses presented here.

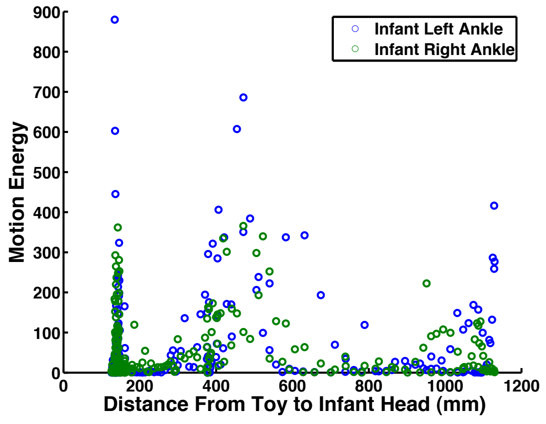
- **Canonical Correlation with Infant Kinematics** Instead of using the location of the center marker on the infant's head, it may be more informative to look at quantities derived from the markers such as infant head orientation or joint angles of various limbs of the infant.

7.3.3 Analyzing Infant Motion Energy to Head-Toy Distance

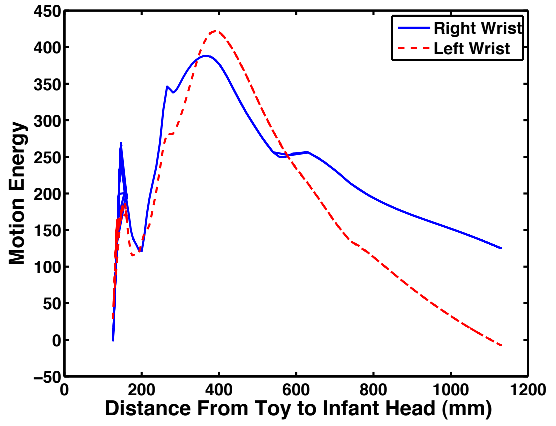
Infant's limb movement appears to be episodic. In particular, there are fast-moving high motion energy (HME) episodes and slow-drifting low motion energy



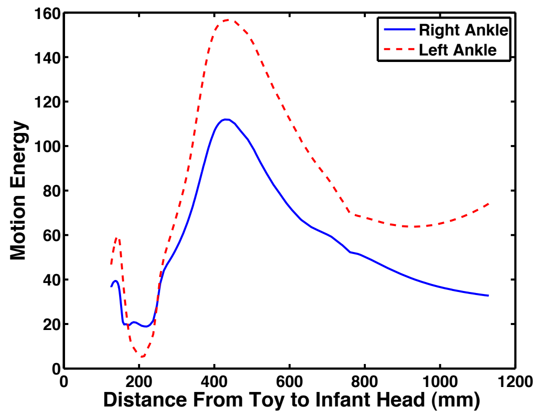
(a) arms (marker at wrist)



(b) legs (marker at ankle)



(c) arms-smoothed (marker at wrist)



(d) legs-smoothed (marker at ankle)

Figure 7.9: motion energy of arms (a) and legs (b) with respect to toy-head distance; smoothed motion energy of arms (c) and legs(d)

(LME) episodes. After watching the video carefully, we speculate that a transition from HME to LME is typically triggered by hand contact with the toy; a transition from LME to HME is typically triggered by the toy moving out of baby's reach.

To further analyze this speculation, we plot limb motion energy against toy-infant distance. Same as in the previous section, motion energy is calculated using sliding windows. Whether the toy is within reaching distance or not is measured by the distance between the markers on the toy and the markers on the infant's head. We do not label the touch events as they are hard to judge from the video. Figure 7.9(a)(b) plots the raw data. Each point represents the average motion energy and toy-infant distance calculated from a sliding window at certain time. Figure 7.9 plots the smoothed curves of the raw data for better visualization.

For both arms and legs, there is a high energy peak at toy-head distance around 40cm which trigger the most of fast movements.

7.4 Discussion and Conclusion

The human musculoskeletal system is a sophisticated machine designed to support movements that are compliant, versatile, and adapted to the uncertain nature of the world. The price paid for this compliance and versatility is the fact that simple control approaches like the ones used in contemporary robotics, do not work. From an engineering point of view, infants face a formidable motor control problem when learning to control their own bodies. While controlling robots with the complexity and dynamics of the human body is currently beyond our most sophisticated algorithms and powerful computers, infants learn to do so seamlessly in less than two years. Uncovering how this is done will have profound scientific and technological consequences. A critical limitation of the literature on motor development is the fact that it is carved into isolated research niches e.g., reaching, facial expressions, crawling, walking, shared attention. Each of these niches typically studies one part of the body as it performs a single task, e.g., hands and arms reaching for an object. While this research strategy is reasonable, it may result in a distorted perspective of how infants really learn to control their bodies. Here we presented our first steps to address the current limitations in the developmental literature.

The preliminary data obtained so far is already presenting a different perspective about the development of motor control. For example, current work on the development of reaching emphasizes the fact that infants reach with very little movement in the elbow's joint when compared to adults. This is interpreted as evidence of a "less-to-more" developmental trajectory [6, 7]. This point of view seems plausible because we tend to assume that moving less degrees of freedom is easier than moving more degrees of freedom. Our experience with the compliant humanoid robot Diego San, being developed as part of this project is that this is not necessarily the case, i.e., due to the high compliance of the joints, getting Diego San to move one degree of freedom at a time is quite difficult. Interestingly our motion capture data suggests that during early reaching episodes infants not only move their arms and hands but also engage their legs, head and face. If anything the data suggests a developmental trajectory that progresses from engaging more degrees of freedom (reaching with two arms and two legs), to engaging less degrees of freedom (reaching with one arm). We are also finding that the physical and social contexts of motor development are tightly coupled. The result is that behavioral categories that are natural from an adult perspective may be artificial from an infant's perspective. For example, when a caregiver is present, making facial expressions, vocalizing, or moving the legs, may be as effective to make contact with an interesting object, as reaching with the arms and hands. Thus, it could be argued that moving the legs or making facial expressions should be a legitimate part of the literature on the development of reaching. This developmental approach is quite different to the standard way we get robots to reach: simplify the control problem by using as few degrees of freedom as possible. Thus it appears that our current notion of "simple" in robotics may not match well the notion of "simple" that infants appear to work with.

The results presented here are still preliminary and thus they should be taken only as an illustration of things to come. Yet the approach and the technologies we are exploring are already contributing a different perspective on motor development. This perspective may give us clues to develop a new generation of robots that learn to control their own bodies.

7.5 Acknowledgment

Thanks to all of the families who participated.

This research is supported by the National Science Foundation under Grant No. IIS-INT2-0808767 and 0808653. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Chapter 7, in full, is a reprint of the material as it appears in "Tingfan Wu, Juan Artigas, Whitnet Mattson, Paul Ruvolo, Javier Movellan, and Daniel Messinger. Collecting a developmental dataset of reaching behaviors: First steps. In *IROS2011 Workshop on Cognitive Neuroscience Robotics*, 2011". The dissertation author was the primary investigator and author of this paper.

Bibliography

[1] C.H. An, C.G. Atkeson, and J.M. Hollerbach. Estimation of inertial parameters of rigid body links of manipulators. In *Decision and Control, 1985 24th IEEE Conference on*, volume 24, pages 990–995. IEEE, 1985.

[2] S. Aoyagi, A. Kohama, Y. Nakata, Y. Hayano, and M. Suzuki. Improvement of robot accuracy by calibrating kinematic model using a laser tracking system-compensation of non-geometric errors using neural networks and selection of optimal measuring points using genetic algorithm. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5660–5665. IEEE, 2010.

[3] Marian Bartlett, Gwen Littlewort, Jacob Whitehill, Esra Vural, Tingfan Wu, Kang Lee, Aytül Er**c**cil, Müjdat Cetin, and Javıer Movellan. Insights on spontaneous facial expressions from automatic expression measurement. *Dynamic Faces: Insights from Experiments and Computation*, 2006.

[4] Marian Bartlett, Gwen Littlewort, Tingfan Wu, and Javier Movellan. Computer expression recognition toolbox. In *8th IEEE International Conference on Automatic Face & Gesture Recognition, 2008. FG'08.*, pages 1–2. IEEE, 2008.

[5] M.S. Bartlett, G.C. Littlewort, M.G. Frank, C. Lainscsek, I. Fasel, and J.R. Movellan. Automatic recognition of facial actions in spontaneous expressions. *Journal of Multimedia*, 1(6):22–35, 2006.

[6] N.E. Berthier, R.K. Clifton, D.D. McCall, and D.J. Robin. Proximodistal structure of early reaching in human infants. *Experimental Brain Research*, 127(3):259–269, 1999.

[7] AN Bhat, HM Lee, and JC Galloway. Toy-oriented changes in early arm movements ii–joint kinematics. *Infant Behavior and Development*, 30(2):307–324, 2007.

[8] Fischer Black and Myron Scholes. The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3):637–654, 1973.

[9] BJN Blight and L. Ott. A bayesian approach to model inadequacy for polynomial regression. *Biometrika*, 62(1):79–88, 1975.

[10] J.E. Bobrow and B.W. McDonell. Modeling, identification, and control of a pneumatically actuated, force controllable robot. *IEEE Transactions on Robotics and Automation*, 14(5):732 –742, October 1998.

[11] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library.* O'Reilly Media, Incorporated, 2008.

[12] Evangelos Theodorou Jonas Buchli and Stefan Schaal. Reinforcement learning of motor skills in high dimensions: A path integral approach.

[13] N.J. Butko and J.R. Movellan. I-POMDP: An Infomax Model of Eye Movement. In *7th IEEE International Conference on Development and Learning, 2008.*, pages 139–144, 2008.

[14] Maury-Rouan C., Hirst D., and Faraco M. Eyebrow raisings and vocal pitch accent : the case of children eyebrow raisings and vocal pitch accent : the case of children blind from birth. In *International Symposium on Discourse and Prosody as a complex interface*, Université de Provence, September 2005.

[15] Kian Ming Adam Chai. *Multi-task Learning with Gaussian Processes.* PhD thesis, Institute for Adaptive and Neural Computation School of Informatics University of Edinburgh, 2010.

[16] K.M.A. Chai. Multi-task learning with gaussian processes. 2010.

[17] Richard Cook, Alan Johnston, and Cecilia Heyes. Facial self-imitation objective measurement reveals no improvement without visual feedback. *Psychological science*, 24(1):93–98, 2013.

[18] P Dayan and G. E. Hinton. Using em for reinforcement learning. *Neural Computation*, 9:271–278, 1997.

[19] M.P. Deisenroth, J. Peters, and C.E. Rasmussen. Approximate dynamic programming with gaussian processes. In *American Control Conference, 2008*, pages 4480–4485. Ieee, 2008.

[20] M.P. Deisenroth and C.E. Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the Twenty-Eighth International Conference on Machine Learning (ICML)*, 2011.

[21] U. Dimberg and M. Thunberg. Rapid facial reactions to emotional facial expressions. *Scandinavian Journal of Psychology*, 39(1):39–45, 1998.

[22] P. Ekman. The argument and evidence about universals in facial expressions of emotion. *Handbook of Social Psychophysiology*, 58:342–353, 1989.

[23] P. Ekman and W.V. Friesen. Facial Action Coding System (FACS): A technique for the measurement of facial action. *Palo Alto, CA: Consulting*, 1978.

[24] P. Ekman, WV Friesen, and JC Hager. Facial Action Coding System (FACS): Manual and Investigator's Guide. *A Human Face, Salt Lake City, UT*, 2002.

[25] E.F. Fichter. A stewart platform-based manipulator: general theory and practical construction. *The International Journal of Robotics Research*, 5(2):157–182, 1986.

[26] A. Fogel, D.S. Messinger, K.L. Dickson, and H. Hsu. Posture and gaze in early mother–infant communication: synchronization of developmental trajectories. *Developmental Science*, 2(3):325–332, 1999.

[27] D. Galati, B Sini, S Schmidt, and C Tinti. Spontaneous facial expressions in congenitally blind and sighted children aged 8-11. *Journal of Visual Impairment and Blindness*, 97(7):418–28, 2003.

[28] M. Gautier and W. Khalil. A direct determination of minimum inertial parameters of robots. In *ICRA*, pages 1682–1687. IEEE, 1988.

[29] D. T. Gillespie. *Markov Processes: an introduction for physical scientists.* Academic Press, Boston, 1992.

[30] N. Gulati and E.J. Barth. Non-linear pressure observer design for pneumatic actuators. In *Proceedings, 2005 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 783 –788, July 2005.

[31] N. Gulati and E.J. Barth. A globally stable, load-independent pressure observer for the servo control of pneumatic actuators. *IEEE/ASME Transactions on Mechatronics*, 14(3):295 –306, June 2009.

[32] J.M. Hollerbach and C.W. Wampler. The calibration index and taxonomy for robot kinematic calibration methods. *The international journal of robotics research*, 15(6):573–591, 1996.

[33] I. Karatzas and S. E. Shreve. *Brownian motion and stochastic calculus.* Sprienger-Verlag, New York, 1991.

[34] K. Kaye and A.J. Wells. Mothers' jiggling and the burst–pause pattern in neonatal feeding*. *Infant Behavior and Development*, 3:29–46, 1980.

[35] P. Khosla. Estimation of robot dynamics parameters: Theory and application. 1987.

[36] Jonathan Ko and Daniel J. Klein. Gaussian processes and reinforcement learning for identification and control of an autonomous blimp. In *in IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2007.

[37] Jens Kober and Jan Peters. Policy search for motor primitives in robotics. *Machine Learning*, 84(1-2):171–203, 2011.

[38] Jens Kober and Jan Peters. Reinforcement learning in robotics: a survey. In *Reinforcement Learning*. Springer, 2011.

[39] Vikash Kumar, Zhe Xu, and Emanuel Todorov. Fast, strong and compliant pneumatic actuation for dexterous tendon-driven hands. *ICRA 2013*, 2013.

[40] Machine Perception Laboratory. RUBIOS. `http://mplab.ucsd.edu/?page_id=392`, 2009.

[41] M. Lee, D. Kang, Y. Cho, Y. Park, and J.H. Kim. The effective kinematic calibration method of industrial manipulators using igps. In *ICCAS-SICE, 2009*, pages 5059–5062. IEEE, 2009.

[42] Sergey Levine, Jack M. Wang, Alexis Haraux, Zoran Popović, and Vladlen Koltun. Continuous character control with low-dimensional embeddings. *ACM Trans. Graph.*, 31(4):28:1–28:10, July 2012.

[43] Gwen Littlewort, Jacob Whitehill, TingFan Wu, Nicholas Butko, Paul Ruvolo, Javier Movellan, and Marian Bartlett. The motion in emotion – a CERT based approach to the fera emotion challenge. In *IEEE International Conference on Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011*, pages 897–902. IEEE, 2011.

[44] Gwen Littlewort, Jacob Whitehill, Tingfan Wu, Ian Fasel, Mark Frank, Javier Movellan, and Marian Bartlett. The computer expression recognition toolbox (CERT). In *IEEE International Conference on Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011*, pages 298–305. IEEE, 2011.

[45] Fei Long, Tingfan Wu, Javier R Movellan, Marian S Bartlett, and Gwen Littlewort. Learning spatiotemporal features by using independent component analysis with application to facial expression recognition. *Neurocomputing*, 2012.

[46] D. Matsumoto and B. Willingham. Spontaneous facial expressions of emotion of congenitally and noncongenitally blind individuals. *Journal of Personality and Social Psychology*, 96(1):1–10, 2009.

[47] D. Mellinger and V. Kumar. Minimum snap trajectory generation and control for quadrotors. In *2011 IEEE International Conference on Robotics and Automation (ICRA),*, pages 2520–2525. IEEE, 2011.

[48] Andrew N. Meltzoff and M. Keith Moore. Explaining facial imitation: a theoretical model. *Early Development and Parenting*, 6:179–192, 1997.

[49] D.S. Messinger, M.H. Mahoor, S. Cadavid, S.M. Chow, and J.F. Cohn. Early Interactive Emotional Development. In *7th IEEE International Conference on Development and Learning, 2008.*, pages 232–237, 2008.

[50] P. Mineiro, J. Movellan, and R.J. Williams. Learning path distributions using nonequilibrium diffusion networks. *Advances in neural information processing systems*, pages 598–604, 1998.

[51] J. R. Movellan. Tutorial on stochastic differential equations. *MPLab Tutorials. http://mplab.ucsd.edu*, 2006.

[52] J. R. Movellan and J. L. McClelland. The Morton-Massaro law of information integration: Implications for models of perception. *Psychological Review*, 108(1):113–148, 2001.

[53] J. R. Movellan and P. Mineiro. A diffusion network approach to visual speech recognition. In *Proceedings of the International Conference on Audio Visual Speech Processing*, pages 92–97. 1999.

[54] J. R. Movellan, P. Mineiro, and R. J. Williams. A Monte-Carlo EM approach for partially observable diffusion processes: Theory and applications to neural networks. *Neural Computation*, 14(7):1507–1544, 2002.

[55] J.R. Movellan. A local algorithm to learn trajectories with stochastic neural networks. In J. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in neural information processing systems*. Morgan Kaufman, San Mateo, 1994.

[56] D. Nguyen-Tuong and J. Peters. Local gaussian process regression for real-time model-based robot control. In *International Conference on Intelligent Robots and Systems*, pages 380–385. IEEE, 2008.

[57] B. Oksendal. *Stochastic Differential Equations: An introduction with applications. Fifth Edition.* Springer Verlag, Berlin, 1998.

[58] L. Olsson, C. Nehaniv, and D. Polani. From unknown sensors and actuators to actions grounded in sensorimotor perceptions. *Connection Science*, 18(2):121–144, 2006.

[59] S. R. Pandian, F. Takemura, Y. Hayakawa, and S. Kawamura. Pressure observer-controller design for pneumatic cylinder actuators. *IEEE/ASME Transactions on Mechatronics*, 7(4):490 –499, December 2002.

[60] R. H. Perry, D. W. Green, and J. O. Maloney. *Perry's chemical engineers' handbook*, volume 7. McGraw-Hill New York, 1984.

[61] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, 2009.

[62] Zhihong Rao and G.M. Bone. Nonlinear modeling and control of servo pneumatic actuators. *IEEE Transactions on Control Systems Technology*, 16(3):562 –569, May 2008.

[63] C.E. Rasmussen and H. Nickisch. Gaussian processes for machine learning (gpml) toolbox. *The Journal of Machine Learning Research*, 9999:3011–3015, 2010.

[64] R. Ratcliff. A theory of memory retrieval. *Psychological Review*, 85:59–108, 1978.

[65] W.E. Rinn. The neurophysiology of facial expression: A review of the neurological and psychological mechanisms for producing facial expressions. *Psychological Bulletin*, 95:52–77, 1984.

[66] P. Rochat. Self-perception and action in infancy. *Experimental Brain Research*, 123(1):102–109, 1998.

[67] Axel Rottmann and Wolfram Burgard. Learning non-stationary system dynamics online using gaussian processes. In *Pattern Recognition*, volume 6376 of *Lecture Notes in Computer Science*, pages 192–201. Springer Berlin / Heidelberg, 2010.

[68] D. Ruppert, M.P. Wand, and R.J. Carroll. *Semiparametric regression*, volume 12. Cambridge Univ Pr, 2003.

[69] Paul Ruvolo, Tingfan Wu, and Javier R Movellan. Control by gradient collocation: Applications to optimal obstacle avoidance and minimum torque control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1173–1179. IEEE, 2012.

[70] Stefan Schaal, Christopher G. Atkeson, and Sethu Vijayakumar. Real-time robot learning with locally weighted statistical learning. In *ICRA*, pages 288–293. IEEE Press, 2000.

[71] Bruno Siciliano and Oussama Khatib. *Springer handbook of robotics*. Springer, 2008.

[72] Karan Sikka, Tingfan Wu, Josh Susskind, and Marian Bartlett. Exploring bag of words architectures in the facial expression domain. In *Computer Vision–ECCV 2012. Workshops and Demonstrations*, pages 250–259. Springer, 2012.

[73] J. Sjoberg, H. Hjalmarsson, and L. Ljung. Neural networks in system identification. 1994.

[74] O. Sorkine. Least-squares rigid motion using svd. *Technical notes*, 2009.

[75] Freek Stulp and Olivier Sigaud. Path Integral Policy Improvement with Covariance Matrix Adaptation. *International Conference on Machine Learning ICML*, 2012.

[76] Yuval Tassa, Tingfan Wu, Javier Movellan, and Emanuel Todorov. Modeling and identification of pneumatic actuators. In *IEEE International Conference on Mechatronics and Automation*, 2013.

[77] R Thomas and Martin Felder. State-Dependent Exploration for Policy Gradient Methods. pages 234–249, 2008.

[78] B. Thompson. Canonical correlation analysis. 2000.

[79] Y. Tian, T. Kanade, and J.F. Cohn. Recognizing Action Units for Facial Expression Analysis. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, pages 97–115, 2001.

[80] E. Todorov. Mujoco: A physics engine for model-based control (under review), 2011a. *URL http://www. cs. washington. edu/homes/todorov/papers/MuJoCo. pdf.*

[81] E. Todorov. Stochastic optimal control and estimation methods adapted to the noise characteristics of the sensorimotor system. *Neural Computation*, pages 1084–1108, 2005.

[82] E. Todorov. Probabilistic inference of multijoint movements, skeletal parameters and marker attachments from diverse motion capture data. *IEEE Transactions on Biomedical Engineering*, 54(11):1927–1939, 2007.

[83] E. Todorov, C. Hu, A. Simpkins, and J. Movellan. Identification and control of a pneumatic robot. In *2010 3rd IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob),*, pages 373–380. IEEE, 2010.

[84] E. Todorov, C. Hu, A. Simpkins, and J. Movellan. Identification and control of a pneumatic robot. In *Biomedical Robotics and Biomechatronics (BioRob), 2010 3rd IEEE RAS and EMBS International Conference on*, page 373–380, 2010.

[85] E. Todorov and W. Li. A generalized iterative lqg method for locally-optimal feedback control of constrained nonlinear stochastic systems. In *American Control Conference, 2005*, pages 300–306. IEEE, 2005.

[86] S. Vijayakumar, A. D'souza, and S. Schaal. Incremental online learning in high dimensions. *Neural Computation*, 17(12):2602–2634, 2005.

[87] Yiwen Wang, Tingfan Wu, Garrick Orchard, Piotr Dudek, Michele Rucci, and Bertram E Shi. Hebbian learning of visually directed reaching by a robot arm. In *Biomedical Circuits and Systems Conference, 2009. BioCAS 2009. IEEE*, pages 205–208. IEEE, 2009.

[88] Jacob Whitehill, Paul Ruvolo, Tingfan Wu, Jacob Bergsma, and Javier Movellan. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. *Advances in Neural Information Processing Systems*, 22(2035-2043):7–13, 2009.

[89] Tingfan Wu, Juan Artigas, Whitnet Mattson, Paul Ruvolo, Javier Movellan, and Daniel Messinger. Collecting a developmental dataset of reaching behaviors: First steps. In *IROS2011 Workshop on Cognitive Neuroscience Robotics*, 2011.

[90] Tingfan Wu, Marian S Bartlett, and Javier R Movellan. Facial expression recognition using gabor motion energy filters. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2010*, pages 42–47. IEEE, 2010.

[91] Tingfan Wu, Nicholas J Butko, Paul Ruvolo, Jacob Whitehill, Marian S Bartlett, and Javier R Movellan. Action unit recognition transfer across datasets. In *IEEE International Conference on Automatic Face & Gesture Recognition and Workshops (FG 2011), 2011*, pages 889–896. IEEE, 2011.

[92] Tingfan Wu, Nicholas J Butko, Paul Ruvolo, Jacob Whitehill, Marian S Bartlett, and Javier R Movellan. Multilayer architectures for facial action unit recognition. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics.*, 42(4):1027–1038, 2012.

[93] Tingfan Wu, Nicholas J Butko, Paul Ruvulo, Marian S Bartlett, and Javier R Movellan. Learning to make facial expressions. In *IEEE 8th International Conference on Development and Learning, 2009 (ICDL 2009)*, pages 1–6. IEEE, 2009.

[94] Tingfan Wu, Chih-Jen Lin, and Ruby C Weng. Probability estimates for multi-class classification by pairwise coupling. *The Journal of Machine Learning Research*, 5:975–1005, 2004.

[95] Tingfan Wu and Javier Movellan. DNA: Dynamic network adaptation for optimal control. In *submission to Neural Information Processing Systems 2013*.

[96] Tingfan Wu and Javier Movellan. Semi-parametric gaussian process for robot system identification. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 725–731. IEEE, 2012.

[97] Tingfan Wu, Yuval Tassa, Javier Movellan, and Emanuel Todorov. STAC: Simultaneous tracking and calibration. In *submission to Humanoids 2013*.