SIMULATION OF CHARACTERS WITH NATURAL INTERACTIONS

A Thesis
Presented to
The Academic Faculty

by

Yuting Ye

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
College of Computing

Georgia Institute of Technology
May 2012

SIMULATION OF CHARACTERS WITH NATURAL INTERACTIONS

Approved by:

Professor C. Karen Liu, Advisor
College of Computing
Georgia Institute of Technology

Professor Lena Ting
Department of Biomedical Engineering
Emory University and
Georgia Institute of Technology

Professor Greg Turk
College of Computing
Georgia Institute of Technology

Professor Victor B. Zordan
Department of Computer Science and Engineering
University of California Riverside

Professor Henrik I. Christensen
College of Computing
Georgia Institute of Technology

Date Approved: December 5, 2012

ACKNOWLEDGEMENTS

Graduate school feels like a particularly long journey for me after navigating through three different institutes and bouncing between the east and the west coasts. I would not have been able to reach the finishing line if not for the help and support of the wonderful people I have known along the way.

I am extremely fortunate to have Karen as my advisor. She has been an inspirer, motivator, role model, and advisor. Her enthusiasm and passion of research and science never cease to encourage and amaze me. She was always there to provide the needed push that keeps me on the right track; she has also repeatedly shown potentials in me that I don't know I possess . I would not be nearly as successful without her guidance.

I am also thankful to have Dr. David Brogan as my advisor in my formative years at graduate school. He introduced me to physics-based character animation when I started. And he reflected on his graduate years at Gatech in many of our conversations so I felt connected to Tech even before I had the chance to come.

I am grateful to my thesis committee members: Dr. Greg Turk, Dr. Henrik Christensen, Dr. Lena Ting, and Dr. Victor Zordan. Lena opens up the door of computational neuromotor control to me and provides the technical foundation that eventually led to the work in Chapter 3 of this thesis. Henrik broadens the scope of my work with valuable knowledge and experience from robotics control. Victor has been a supporter and mentor and always goes out of his way to encourage me. Greg provides advice not only in research but also in various aspects of being a successful researcher. I am also grateful to my qualifier committee members, Dr. Irfan Essa and Dr. Charlie Kemp. They made me a much stronger and solid researcher at

the beginning stage. I would also like to thank Dr. Jarek Rossignac. The weekly group meeting will be much less fun and inspiring without his pointed questions and comments. I learned from him to be curious and ask fundamentals questions. I want to also thank Dr. David Luebke and Dr. Stacy Marsella, who had been my mentor and guided me through my stay at UVA and ISI, respectively.

I would have lost my sanity during school transfers and SIGGRAPH deadlines without the company and comfort from my precious friends, around or apart. I am very happy to call some of them now doctor-to-be, doctors and professors. My most sincere gratitude goes to Chengdu Huang, Liqian Luo, Rui Wang, Xiaohua Jiang, Lingjia Tang, Lin Gu, Tenghui Zhu, Lei Fang, Xiangyu Jin, Yaogang Lian, Xiaodong Zhang, Wei Jiang, Meng Zhu, Bo Sun, Kangyu Zhang, Xuefeng Song, Appu Goundan, May-Chen Kuo, Hermann Chong, Mei Si, Brian Whited, Howard Zhou, Huamin Wang, Matt Flagg, Sangmin Oh, Kihwan Kim, Chris Wojtan and Olga Symonova, Michael Su, Mingxuan Sun, Ping Wang, Tina Zhuo, Mark Luffel, Jason Williams, Kai Ni and Haining Lu, Karthik Raveendran, Jie Tan and Yuting Gu, Topraj Gurung, and Sehoon Ha. I am sorry if I neglect to mention your name!

In particular, Dr. Sumit Jain has been an invaluable friend and colleague. We have shared many ups and downs, good and bad memories together. There have been countless moments when he saved my sanity with his patience. He showed me that I can always turn to him for any kind of help and advice in any time. I would not have survived graduate school without Sumit alongside.

Finally and most importantly, thanks to Mom, Dad, and Wei for their unconditional love and support and confidence in me, whatever I do, wherever I am.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS OR ABBREVIATIONS

COM	center of mass.
DDP	differential dynamic programming.
DOF	degree of freedom.
EMG	electromyography.
ICA	independent component analysis.
IK	inverse kinematics.
IO	inverse optimization.
LQR	linear quadratic regulator.
mocap	motion capture.
NMF	nonnegative matrix factorization.
PCA	Principal Component Analysis.
QP	quadratic programming.
ROM	range of motion.
SQP	sequential quadratic programming.
$\mathbf{E}$	eigenvector.
$\mathbf{F}$	a set of contact forces.
$\mathbf{J}$	the Jacobian matrix, a linear map from generalized coordinates to Cartesian coordinates.
$\bar{\mathbf{Q}}$	reference motion.
$\mathbf{q}$	joint configuration.
$\dot{\mathbf{q}}$	joint velocity.
$\ddot{\mathbf{q}}$	joint acceleration.
$\bar{\mathbf{q}}$	a reference joint configuration.
$\mathbf{U}$	a sequence of torques.
$\mathbf{u}$	a joint torque vector.
$\bar{\mathbf{u}}$	a reference torque vector.

SUMMARY

The goal of this thesis is to synthesize believable motions of a character interacting with its surroundings and manipulating objects through physical contacts and forces. Human-like autonomous avatars are in increasing demand in areas such as entertainment, education, and health care. Yet modeling the basic human motor skills of locomotion and manipulation remains a long-standing challenge in animation research. The seemingly simple tasks of navigating an uneven terrain or grasping cups of different shapes involve planning with complex kinematic and physical constraints as well as adaptation to unexpected perturbations. Moreover, natural movements exhibit unique personal characteristics that are complex to model. Although motion capture technologies allow virtual actors to use recorded human motions in many applications, the recorded motions are not directly applicable to tasks involving interactions for two reasons. First, the acquired data cannot be easily adapted to new environments or different tasks goals. Second, acquisition of accurate data is still a challenge for fine scale object manipulations. In this work, we utilize data to create natural looking animations, and mitigate data deficiency with physics-based simulations and numerical optimizations.

We develop algorithms based on a single reference motion for three types of control problems. The first problem focuses on motions without contact constraints. We use joint torque patterns identified from the captured motion to simulate responses and recovery of the same style under unexpected pushes. The second problem focuses on locomotion with foot contacts. We use contact forces to control an abstract dynamic model of the center of mass, which sufficiently describes the locomotion task in the

input motion. Simulation of the abstract model under unexpected pushes or anticipated changes of the environment results in responses consistent with both the laws of physics and the style of the input. The third problem focuses on fine scale object manipulation tasks, in which accurate finger motions and contact information are not available. We propose a sampling method to discover contact relations between the hand and the object from only the gross motion of the wrists and the object. We then use the abundant contact constraints to synthesize detailed finger motions. The algorithm creates finger motions of various styles for a diverse set of object shapes and tasks, including ones that are not present at capture time.

The three algorithms together control an autonomous character with dexterous hands to interact naturally with a virtual world. Our methods are general and robust across character structures and motion contents when testing on a wide variety of motion capture sequences and environments. The work in this thesis brings closer the motor skills of a virtual character to its human counterpart. It provides computational tools for the analysis of human biomechanics, and can potentially inspire the design of novel control algorithms for humanoid robots.

CHAPTER I

INTRODUCTION

Interactive virtual characters play an increasingly important role in entertainment and education, often times performing in live action shots with human actors and collaborating with human players in video games or a training session. The close interaction between virtual characters and human raises the bar for computer graphics research to generate realistic characters that appear and perform intelligently in unforeseeable situations. Advanced rendering techniques synthesize natural appearance by simulating the multi-layered scattering of photons under the skin [33], adapting to lighting conditions in interactive rates [34]. The realism in appearance in turn requires the avatar to possess comparable motor skills to a human so as to stimulate positive and empathic emotional responses [88, 102, 103]. A believable virtual human is expected to not only navigate around obstacles, pick up objects, and interact with other virtual humans autonomously, but also respond to unexpected perturbations, adapt to changes in the environment, and most importantly, perform in a manner that feels natural to humans. While we have the uncanny ability to detect the slightest discrepancy in synthetic motions, our understanding of natural movements and its underlying generative mechanism is no match to our sophistication in perception and performance. Synthesis of the most basic human motor skills of locomotion and manipulation remains a grand challenge in animation research and human motor control.

The most principled way of generating motion is through physics simulation. Natural phenomena such as wavy ocean [22], splashy water [113], fire and smoke [39], or passive motions such as a poked [38], smashed [130], or melted [19] Stanford bunny

model can be simulated faithfully and efficiently. Compared to these dynamic systems with tens of thousands of degrees of freedom, simulation of the human body may seem trivial. While simulation of a ragdoll is effortless, controlling a character to perform in a meaningful way faces many challenges presented by both the mechanical system and the biological system.

Characteristics of the mechanical system presents three hurdles to the design of a robust control strategy for the most basic motor skills: posture control, balance and locomotion control, and object manipulations. First, the relation between control force and the resultant motion is governed by a coupled and nonlinear dynamic system that in general has no analytical solution. Computing the required forces to track a pose precisely is nontrivial. Second, as a consequence of the momentum conservation law, internal forces *per se* cannot directly change the position of the center of mass (COM) to perform locomotion and maintain balance. This is also known as the under-actuation problem. Locomotion can be realized only through exerting contact forces and utilizing frictions from the environment. However, contacts and frictions introduce nonlinear kinematic constraints and discontinuities to the dynamic system, further complicating the problem of controller design on a third count. Grasping and manipulation tasks are more difficult in this regard because contacts and forces are vital components in the interaction between the hand and the object. Finally, a robust controller often needs to consider the accumulated performance in a long term under nonlinear dynamics and kinematics, under-actuation, and discontinuity.

Moreover, the laws of physics alone is far from sufficient to describe human motion. Our movements are the results of a complex process that requires coordinations of the brain, the central nervous system, and the musculoskeletal system to accomplish intended goals under biomechanical and physical constraints [94]. It is the neuromechanical system rather than physics that characterizes human movements. Limb proportions, muscle strength, emotional states, personal preferences based on past

experience and culture influences, *etc.* all contribute to a person's unique movement style [96]. Some of these factors are tasks-specific while others are persistent across activities for an individual. The complexity of human motion is daunting and fascinating at the same time that it intrigues researchers from many scientific disciplines such as robotics and biomechanics in addition to computer animation.

Biomechanicists study how the human body solves these difficult problems through examination of the biomechanical structure and analysis of both kinematic data and muscle activities of human performing motor tasks. Some common beliefs are that human motions are "lazy" and energy efficient, and exhibit recognizable coordination patterns. These observations lead to simple passive models such as the inverted pendulum model and its variations as the underlying model for walking and balancing, and categorization of balance recovery strategies into the ankle strategy and the hip strategy [54]. Insights from biomechanics shed light on the controller design in robotics and character control. Theses models have been successfully applied to develop locomotion controllers, in which forces are *actively* applied to mimic the motion of a *passive* structure of certain physics properties. Similarly, grasp taxonomy based on the analysis of the anatomical model of the hand, such as the precision grip and power grip [91], has been the primary means of grasp control in robot hands. Although these high level principles are successful as general guidelines for reproducing human motor skills, they do not direct low level details of a motion. On one hand, the low level control implementation is still highly specialized. Controller design remains a time and resource consuming task that rely on experience and trial-and-error. On the other, while the models are meant to explain the common denominator of human motions, they have very limited power in describing the styles and variations of natural movements. Expressing preferences within this control framework is not only difficult, but also interfere with the control goals in most cases.

A more systematic control framework is variational optimization. It naturally reflects the insight that human motions, like other natural passive motions, are optimal in one way or another, within constraints imposed by the physical world and our biomechanical structure. Energy minimization and jerk minimization are the mostly commonly used objective term to describe human motion. Casting control as an optimization also has the advantage that any high level preference or control goals can be specified easily as a performance metric. The optimization process automatically figure out the best way to achieve the goals without lower level user guidance. Trajectory optimization approaches have been applied in character animation for authoring both highly dynamic [129, 79, 101] and low energy [78] motions from minimum user input, as well as for *offline* adaptation of existing motions to different physical constraints and control goals by maintain the optimality metric [99, 111]. However, the generic objective terms similarly suffer from the problem that detail variations are not captured or explained within the model. Although better biomechanical models [78] or more accurate modeling of the dynamics system with signal dependent noise [46, 116] can be used to mitigate some of the problems, they are usually computationally expensive.

An affordable and readily available source of natural and stylistic motions is the kinematic data from motion capture thanks to the advance in modern motion capture (mocap) technology. Motions can be treated as a form of data, just as image and text, which has intrinsic structures that capture characteristics of natural movements. Statistical models can be built to describe the manifold of "naturalness", and be used to interpolate and extrapolate realistic motions from relatively sparse examples. However, these models are difficult to be used in conjugate with physics-based simulation or incorporated in controller design. The major reason is that they cannot distinguish the influence of environmental and physical constraints from individual preference of styles in the training data. Therefore, they cannot be adapted to different physical

constraints or unexpected events. Some preliminary attempts exist to empirically trade off the factor of physical environment in the training motion, but the results are far from ideal. Style-content separation is an interesting open research question in learning human motion styles.

In light of the accumulated experiences in capturing characteristic motions in controller design from different disciplines and schools of approaches, this work attempts to marry the benefits from biomechanical principles, optimization and motion capture. We additionally have an ambition to provide a systematic scheme for intuitive controller design so that believable motions can be easily directed and widely applied.

1.1 Approach

Our approach builds on the believe that characteristic motions should co-exist rather than conflict with robust control. The common approach of trading off robustness with naturalness in controller design is a consequence of trying to achieve both goals in the control space directly. We argue that rich details and variation exist in motions under the same constrained environment suggest that styles "live" in a subspace tangential to constraints. Based on this insight, we derive control algorithms for three motor tasks under different physical constraints.

We choose to use a motion capture sequence as input to the control algorithm for two reasons. First, as motion capture technology becoming more commonplace, motion gradually becomes an intuitive interface for authoring and directing contents. Supplying one single motion sequence to express the high level intention of motion is arguably more natural and preferable than specifying rules or constraints for a task. In addition, this interface can be integrated seamlessly with traditional manually authored animations because they are simply kinematic data of a skeletal model. We will show in this work that even partial motion data can be a useful source of input to synthesize motions and control when the task is highly constrained. Second, since

we do not yet have a descriptive model for modeling styles in a motion, it is a sensible choice to use the motion as a whole.

In the following chapters, we will describe three algorithms that synthesize interactions with rich details and styles. We exploit the contact constraints involved in an interaction, and explore the subspace within physics constraint and kinematics constraints defined by the contacts to express characteristics in natural human motions.

Chapter 3 addresses the problem of posture responses to perturbations in the absence of contact constraints, i.e. the upper body motion in locomotion. We derive controllers that capture the unique patterns in the reference motions, and retain the pattern at the presence of disturbances. Without external influence from the environment, the motion can be completely described by the internal joint torques. While posture control is relatively easy without concerning balance, the question is how should the character behave when perturbed such that it is compliant to changes yet able to recover in a way that seems natural to the unperturbed motion.

We propose to identify the actuation space of joint torques in the input motion, then turn a control problem into a passive simulation by constraining the torques in its rank space at all times. We hypothesize that the torque actuation space characterizes the motion and remains unchanged with and without perturbations. The idea is inspired by the *muscle synergy* theory in neural control [117], which states that muscle activations are synchronized, and only a small number of synchronized groups are utilized in familiar tasks. Nonnegative matrix factorization (NMF) has been used by neuroscientists to analyze electromyography (EMG) data because EMG signals are positive numbers. In our work, we use Principal Component Analysis (PCA) to identify the null space of joint torques, and transform the control problem into passive simulation in the null space. We choose PCA instead of NMF because joint torques are bidirectional. Results show that our algorithm can successfully synthesized convincing

responses to unexpected perturbations for different locomotion activities without case-by-case authoring effort.

In Chapter 4, we addresses the balance issue in locomotion at the presence of perturbations or changes in the environment. Biped locomotion balance is usually modeled by an inverted pendulum located in the COM. Control torques are applied to mimic this passive and reactive behavior. Although the simplicity of this model leads to many successful applications, it usually behave as a stiff system to ensure robustness, thus creating unnatural motions. In this work, we also propose a similar abstract model of the COM. Our work depart from the pendulum model in that we explicitly utilize ground contact forces as control signals to direct the COM motion. Optimization is used to compute the desired amount of ground reaction forces to maintain the COM in a balance state. Following the insights in biomechanics of walking, we additionally regulate the angular momentum around the COM for better balance behavior.

The use of an abstract model benefits the optimization problem thanks to its simplicity. As a result, we can formulate more advanced optimization problem such as enforcing terminal constraints and adjustments of timing to provide more flexibility in motion synthesis. Moreover, we can apply optimal feedback controllers from the optimization result in online simulation, and adapt the input motion to different environments, in anticipation of future changes. The full body motion is reconstructed from the reference motion by respecting the dynamic states of the COM model. As a result, we obtain response and balanced motions that are dynamically consistent with the environment, and still retain the gross styles in the reference. Because the COM constraint and angular momentum affect the entire body, the synthesized motion exhibits coordinations across joints.

The major drawback of our abstract model is its dependency on the contact information from the reference motion. Change of contacts is difficult because the

dynamics is discontinuous at the point of change, and prohibits the use of continues optimization. The problem is more pronounced in object manipulations because real-world interactions between the hands and the objects are very complex and with frequent changes. Therefore, robotic manipulators focus on stable grasps that ensure force closure. Research in grasping and manipulation largely concerns about stability analysis, how to precisely achieve a stable grasp, and how to maintain or regain stability when disturbed. However, humans are never precise in handling everyday objects. We employ many reactive strategies based on the object's motion to just "get the job done". A lot of interesting phenomena such as slipping or finger gaiting emerge as a result. In computer animation, we have not been able to recreate manipulations with rich details systematically.

Data acquisition is also challenging for manipulation when the fine scale finger movements and gross scale body movements need to be captured at the same time. Optical motion capture devices suffer from occlusions induced by interactions between the hand and the object. The use of inertia sensors, on the other hand, negatively affect the performance of the tasks because of the cumbersome attachment. While it's still possible to capture close range small scale object manipulation tasks, data of more practical and common scenarios that require coordination between the full body and the hands, such as cooking, fetching a book from a bookshelf, or carrying a box from one place to another, are not available.

In Chapter 5, we address the problem of synthesizing realistic hand-object manipulation motions with rich details. We observe that the intricate finger motions are largely constrained by the spatial relations between the wrist and the object, which we can obtain easily from modern mocap settings. Although the contact constraints are discrete, we can use sampling techniques to discover feasible solutions. The different detailed and characteristic manipulation strategies can be sufficiently encoded into the change of contact points. By intelligently confining the contact points within

physical constraints, we can efficiently discover a variety of hand motions without any prior knowledge of stability or grasp taxonomy.

1.2 Contributions

The work in this dissertation has several contributions to human character animation.

1. **A framework for synthesis of stylistic postural responses and recovery.** We propose a generic controller that carries the motion style of a captured sequence to dynamics responses and recovery in an interactive application. The framework is generic and robust to apply on different motion styles and character proportions. It can be seamlessly integrated with any existing kinematics based controller without modifications on either party. It is also be applied to a variety of realistic scenarios such as dodging obstacles, stepping on a banana peel, or trying not to spill coffee while walking.

2. **A simplified dynamic model for balance control in locomotion.** We propose a simple model that abstracts the fundamental dynamics of an underactuated structure. It is the first model that casts balance control problem as an active system exerting contact forces to propel the COM. Since the biomechanical structure is abstract out, the model can be applied to a large class of mechanical structures and motion contents.

3. **Application of advanced optimal control algorithm on human character animation.** Owe to the simplicity of the abstract model, we are able to leverage advanced optimal control algorithm to develop robust controllers for human motion. Our controller can flexibly adjust the duration and final goals of the motion online, and uses unilateral contact forces to achieve the goals. The flexibility and generality of our formulation allows the synthesized motion to deviate significantly from the input while still maintain plausibility.

4. **A framework for synthesizing intricate finger movements in hand-object manipulations.** We propose a generic sampling framework to synthesize complex hand manipulations with rich details and large varieties. This is the first method capable of synthesizing physically plausible hand motions with natural finger gaits and contact changes with no prior knowledge of the task or grasp taxonomy. Our method is not limited to generate stable manipulation or tasks motions. It is also flexible to adapt the hand motion to different object properties.

CHAPTER II

RELATED WORK

In this chapter, we review related research in the area of human motion control and synthesis, including hand manipulations. In Section 2.1, we review a few common approaches to synthesizing responsive human motions. We then review current research in hand manipulations in Section 2.2. Complex interactions with discrete contact events such as those in grasping are usually solved with sampling methods. In Section 2.3, we briefly review recent animation research that uses sampling to derive controllers with frequent contacts. Finally, in Section 2.4, we review two research results in biomechanics on human motor controls, which greatly inspire the work presented in this thesis.

2.1 Human motion synthesis

Kinematics-based approaches and physics-based approaches are two major means to create character animations. Hybrid approaches that combine the two methods utilize the benefits of both. On a parallel track, optimization is a useful tool that all approaches explore to develop sophisticated algorithms. To effectively deal with high-dimensional human motions, various dimensional reduction and model reduction techniques are proposed. In the following sections, we will give a brief introduction of these approaches in animation research.

2.1.1 Kinematics approaches

Kinematics control has been the common practice in industry for character animation because it provides intuitive and precise control over both the contents and the styles of the motion. However, manually designed motions are highly specialized and

difficult to change in response to run time input. With motion capture techniques, motion data become easier to gather in large quantities, thus boost the development of data-driven algorithms for motion editing. Kovar *et al.* [68] and Arikan and Forsyth [12] concatenate and interpolate different motion clips where they are most similar to switch between actions according to user commands. Responsiveness of the character depends on the availability of motions in the database. Many researchers parameterize the database to explore a continuous space for interpolation [67, 105, 47, 145], and fill in data where much needed [100]. With limited motion clips, reinforcement learning can improve responsiveness by selecting the best motions for a task in the long run [120, 84, 74]. Lee and Popović [73] further used inverse reinforcement learning to control an intelligent character that can adapt high level user intents to different environment settings.

While a few walking cycles suffice simple navigation tasks, highly dynamic motions such as push response and balance recovery require much larger number of special purpose motion clips. With a large database, Arikan [13] and Yin *et al.* [144] use external forces as a heuristic to select the appropriate motion, then apply small deformations to generate realistic responses.

This thesis proposes algorithms that enhance a single motion sequence with a wide range of dynamics responses, therefore removing the dependency on a motion database. Nonetheless, our methods can enhance the capability of the motion database when applied to more motions.

2.1.2 Physics-based simulations

Compared to kinematically controlled characters, physically simulated characters are dynamically consistent with the physical world and are responsive to the environment. Earlier work by Hodgins *et al.* [52, 131] demonstrates that complex human movement and maneuvers, such as running, jumping, diving, and tumbling, can be physically

simulated in a virtual environment. Designing a robust controller, however, requires immense manual efforts and expertise. Researchers have to study every gory detail of an activity and develop specific algorithms for every stage of execution, which takes in a large set of carefully chosen parameters. While the results are very compelling, adaptation of these special-purpose controllers to different structures or activities is very difficult.

A few research aims to adapt controllers to novel situations. Hodgins *et al.* [51] adapted the same controller to characters of different proportions, and Faloutsos *et al.* [36] composed controllers of different goals for more complex tasks. Others proposed automatic algorithms to ease the design of control parameters, although balance is largely omitted. Neff and Fiume [93] applied antagonistic controllers to generate more compliant motions; Allen *et al.* derived analytical solutions for PD controllers that can control the timing [9] and target states [10] of the motion.

As researchers gain better understanding of biped locomotion over the years, general balance controllers have been developed. Yin *et al.* [143] proposed a simple yet robust biped balance control strategy, which has been used as the baseline to develop controllers of more advanced skills [142, 26] and more natural motions [126, 127]. Last year, we have seen general biped controllers that give rise to natural and compliant walking motions [133, 87, 27]. The progress made in biped balance control in animation has been amazing. Yet we have not seen general algorithms beyond walking or intuitive control of styles beyond a few set points.

2.1.3 Hybrid approaches

A promising research direction for natural and responsive animation is to combine the advantages of both kinematics control and physics-based simulation. For example, tracking the upper body motion from a captured sequence can generate expressive results [146, 141, 9].

Static balance when tracking motion capture data can be achieved by carefully maintaining contacts [5] and minimizing angular momentum [81]. In most cases, however, balance requires special treatment because the kinematics data impose additional constraints on top of all challenges of a balance controller. For dynamic balance, many researchers proposed separate balance strategies in conjunction with upper body tracking [147, 143, 121, 76], or map the kinematics data to an inverted pendulum to develop balance strategies [136, 71]. Alternatively, Sok *et al.* [107] modified the captured motion offline for better balance behavior during tracking. Linear quadratic regular (LQR) is another effective algorithm to faithfully track a full body motion. It linearizes the dynamic system around the reference motion to derive a feedback controller that closely tracks the reference data. Da Silva *et al.* [29] applied LQR on a simplified model to balance various bipedal locomotion. Muico *et al.* [89] modified LQR to take into account the nonlinear dynamics of the current moment, and adapt the controller to environment contacts. Their characters closely micmic the captured motions even when the environment is altered. However, LQR-based tracking controllers are robust only to small deviations from the reference motion. Larger variations may require switching among references.

With a few captured responsive motions, myself and Liu [140] used a low-dimensional statistical model to model the perturbation dynamics and applied simulations as guidance to navigate in the motion space. Similarly, Lee *et al.* [75] built a motion field from captured motions by interpolating motion dynamics using nearest neighboring states, then propelled the character in this velocity field.

Researchers also proposed to switch between dynamics simulation and motion capture data whenever necessary [104, 83, 92]. In particular, Zordan *et al.* [148] proposed a framework that uses minimal simulation interval after the impact and relies on the captured motions only when perturbations are not presented.

2.1.4 Optimization-based approaches

Optimization has been a useful tool for both kinematics-based methods and physics-based methods. Kinematics-based methods use the equation of motion as constraints to generate optimal trajectories that are consistent with the physical world. Users can encode high-level goals as constraints or directly specify poses and contacts constraints to direct the motion [129, 37, 101]. Optimization is also used offline to adapt a motion sequence to characters of drastically different structures [43], to various altered goals [99, 111], or to a different task with the same style [78]. Incorporating fundamental principles of animal movements in the objectives can synthesize convincing motions beyond human characters. Wu and Popović [132] generate realistic bird flight motions comparable to video footage. Wampler and Popović [125] developed an optimization framework to generate energy efficient locomotion for imaginary characters.

Similarly, physics-based simulations use short horizon optimization to adjust static balance in real time by minimizing momentum [81], and to achieve multiple tasks simultaneously [5, 3, 59]. In addition, de Lasa and colleagues [31, 32] also develop new optimization algorithms to execute multiple tasks in straight priority so that important goals will always be achieved when feasible. Long horizon optimization, on the other hand, has been useful for tracking locomotion because the control strategy anticipates future events and respects the final goal. In particular, LQR has been proven effective in response to external forces and environmental perturbations while following the reference motion [29, 89]. Da Silva *et al.* [30] further showed a remarkable result that optimal tracking controller can be easily combined to generate optimal controllers for new goals.

2.1.5 Dimensional reduction and model reduction

Principal component analysis (PCA) has been frequently used to reduce the dimensionality of motion data for applications in computer graphics, robotics, and computer

vision due to its simplicity and effectiveness. In computer animation, PCA is typically used to expedite the computations for motion blending, recognition, or modeling [17, 60, 101, 15, 20].

Simplified representations of human body have been proposed to reduce the complexity of the human dynamic system in simulation or optimization of character motion [99]. These abstract models, however, are typically designed for specific types of activities, such as a spring-mass model for running and hopping [16], or an inverted pendulum for standing [136] or walking [7, 70, 109]. The abstract dynamics model proposed in this work makes no assumption about the character structure or activity, therefore is generic across a wide range of motions.

2.2 Hand manipulations

Many researchers have proposed different approaches to synthesizing detailed hand motion in computer animation. Hand motion can be directly captured from the real world and played back in the virtual world [82]. However, when the motion involves object manipulation, occlusion and imprecision become major issues. Previous work has applied kinematic approaches to create grasping motion [14, 55, 65] or manipulation of musical instrument [64, 35]. These methods add great details to the character animation, but the resulting motions usually lack physical realism and variability. Our method also applies inverse kinematics to generate joint motions for the hand. However, the contact points used to constrain the hand poses are computed in consideration of motion diversity and physical realism.

Physical simulation is another promising approach to synthesizing hand animation [97, 69, 110, 122, 6]. Previous methods developed grasp controllers using recorded hand motion [97, 69] and contact forces [69] . Because the motion is physically simulated, one can apply the same controller to different dynamic situations or objects.

Our method has different focus but it is straightforward to integrate our search algorithm for contact points to enhance existing dynamic controllers. Physically plausible hand motion can also be generated by optimization-based approaches. Liu [77] formulated a layered optimization that solves for contact forces, contact positions, joint torques, and hand motion. We also formulate a convex optimization to compute contact forces, but the contact points are computed by a very different approach. Instead of formulating an nonconvex continuous optimization which may or may not converge at single solution highly sensitive to a particular objective function and initial values, we apply a randomized approach to produce a set of solutions, all of which are physically plausible but visually diverse.

Detailed manipulation exploiting contact mode switching or finger relocation has been extensively studied in robotics. These sophisticated strategies adjust hand poses and contact positions in concert to achieve a larger scale of manipulation. Common strategies, such as controlled sliding and rolling contact [118, 18, 25, 23], or finger gait [53, 45, 134] can largely improve the capability of robotic manipulators. We draw many insights from the robotics literature, but our method is fundamentally different in that we do not use the prior knowledge to synthesize each specific manipulation strategy. Rather, we employ a generic randomized algorithm to discover those strategies efficiently and automatically.

Creating a natural scene with rich and close interaction between humans and the environment has been a challenging research problem. Many existing approaches combine the motion capture data with motion adaptation techniques to synthesize interaction between the character and the objects in the environment. Gleicher [43] used kinematic constraints to fix the character's hands on the manipulated objects. Yamane *et al.* [137] proposed a global path planner to synthesize the object's trajectory while maintaining the kinematic constraints and naturalness of motion capture data. Ho *et al.* [50] introduced a mesh representation to maintain implicit spatial

relationship of the scene during motion editing. Jain and Liu [57] coupled full-body mocap data with manipulated objects via physical simulation. Through the dynamic coupling, human motion adaptation can be driven by the edited motion of the object. In this work, we aim to create interactive scenes in much greater details than what previous methods produced. The hand motion must be dynamically and kinematically consistent with the objects and the full-body motion, while exhibits the level of complexity and diversity of real human hands during manipulation.

2.3 Sampling-based motion control

Sampling is a generic and effective way of solving complex control problems that often involve discrete events such as contacts. It has been applied to control the final states of rigid body simulations [123, 21] and character animation [80, 107, 124, 95, 106]. For methods involving actively controlled systems, determining a proper sampling space is crucial. For example, Liu *et al.* [80] sampled the desired joint angles around the nominal trajectory, Sok *et al.* [107] sampled the initial joint configuration of the character, and van de Panne and Fiume [124] sampled the weights of a neural network for the sensors and actuators of a controlled system. Our method generates random samples in the domain of contact positions on the objects. This choice of sampling space has the advantages of providing important constraints to hand poses, and at the same time, being highly constrained by the state of contact forces. The former simplifies the process of creating final hand motion and the latter greatly reduces the number of required samples. A few applications in computer animation adopt randomized path planning algorithms from robotics literature [24, 137]. Our problem is different in that we demand a set of paths with large diversity. The existing path planning methods, such as Rapidly-exploring Random Trees [72] and Probabilistic Roadmap [63], are not designed for exploration of all possible paths in that regard.

2.4 Biomechanics principles

Animation and robotics researcher constantly consult neuroscientists for principles in human motor control to help create more natural and robust motions. The work in this thesis is greatly inspired by two research results in biomechanics research, namely muscle synergies and angular momentum regulation in walking.

2.4.1 Muscle synergies

Biomechanics researchers have applied dimension reduction techniques to the muscle activation data measured from behavioral experiments [119, 8]. Ting [114] suggested that a limited set of muscle synergies, defined as low-dimensional modules formed by muscles activated in synchrony, are used to control the center of mass after postural perturbations. Researchers apply nonnegative matrix factorization (NMF) to the measured EMG data and discover activation patterns during postal perturbation responses. Our method is inspired by Ting's work in that we formulate the dynamic equations in the space of muscle synergies, rather than the space of joint configurations. Because we work with the aggregated joint torques instead of real muscle activations, we apply PCA instead of NMF to identify the null space of torque actuation which do not play an important role in the reference motion.

2.4.2 Angular momentum regulation

Many human motor skills require control of whole body linear and angular momentum to achieve task-level goals while maintaining balance. Several researchers in computer graphics have demonstrated that aggregate body momentum can be a compact representation for editing ballistic motion [79, 4] or locomotion [66]. Regulating linear and angular momenta have also been investigated for balance control. A rich body of research in robotics demonstrated the positive effect of minimizing angular momentum on walking and stepping [98, 62, 44, 49]. Macchietto *et al.* [81] showed that simultaneously controlling the center of mass and the center of pressure via changes of

momenta resulted in much more robust and fluid motion for standing balance against perturbations. Our method confirms the importance of momentum control in human motion both in terms of maintaining balance and producing natural movements. Rather than tracking the momentum trajectory from the input motion, we apply a zero-angular momentum strategy from biomechanics and robotics literature. As a result, our method produces robust control for high dynamic motions without any assumptions about the momentum patterns.

CHAPTER III

POSTURAL RESPONSES CONTROL IN TORQUE ACTUATION SPACE

This chapter describes an algorithm for controlling the postural responses of a character to small-scale perturbations that affect mainly her upper body motion (Figure 1) [138]. The control strategy is derived from a single reference kinematic motion, either motion captured or hand-animated, to retain the unique movement style in both responses and recovery. This algorithm is robust to external forces in arbitrary directions on different body parts at any moment in time, and generic to work on a wide range of motion styles and activities. Its simplicity allow for seamless integration with any technique that produces balanced lower body motion in the presence of large perturbations.

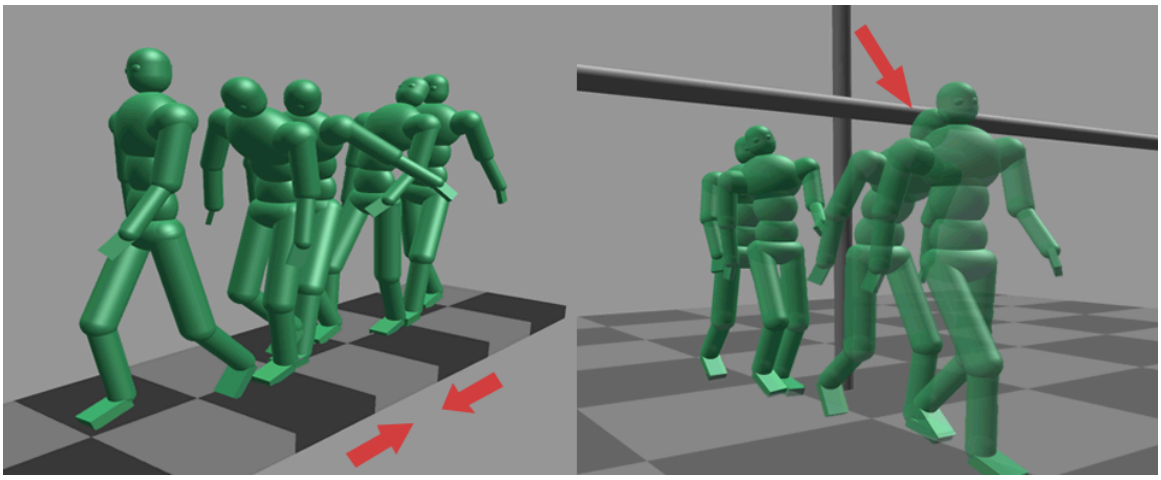


Figure 1: A captured backward walk adapted to a moving platform (left) and a new environment with obstacles (right).

3.1 Motivations

Our approach is motivated by the observation that less-controlled joint degrees of freedom (DOFs) are usually more compliant when perturbed. If we are able to identify those compliant DOFs, we can apply a hybrid method that only considers dynamics in the compliant DOFs and kinematically controls the rest of the character. Instead of determining these DOFs by heuristics and hand-tuning their physical parameters, we use Principal Component Analysis (PCA) to define a new set of coordinates, ranked by the level of joint actuations in the input motion. Our method provides a more principled way to identify the less actuated coordinates (corresponding to eigenvalues close to zero) specific to each input motion sequence. We denote those DOFs as near-unactuated coordinates.

To synthesize the input motion under perturbations, we enforce the dynamic equations of motion only in the near-unactuated coordinates while kinematically maintaining the original joint trajectories. Because the near-unactuated coordinates use very little internal torques in the input motion, enforcing the dynamic equations with zero internal actuation does not visually modify the input motion when there is no external perturbation. When the character is perturbed, however, the near-unactuated coordinates will compliantly react to the external force while the actuated coordinates will attempt to maintain the input joint positions. Because the lower body motion is typically less compliant and the internal joint torques cannot be obtained without accurate measure of contact forces, our technique only considers the dynamics of the upper body motion.

Enforcing dynamic constraints in the near-unactuated coordinates leads to two main advantages. First, the responsive motion varies due to different activities, styles, and individuals. This is because each motion and perturbation results in a unique response based on the specific joint torque usage in the input motion. Second, our formulation bypasses the problem of active body control. The generalized coordinates

in our parameterization are not aligned with the mechanical joint space, but rather aligned with a more meaningful *actuation space* derived from the input motion. By choosing the appropriate coordinates to enforce the equations of motion, our approach can create physically responsive motion based on kinematic pose control without explicitly computing the joint actuations. In practice, our technique can be adapted transparently to any kinematically controlled framework without the aid of a forward simulator or additional motion data.

We demonstrate the simplicity and robustness of our approach by showing a wide range of input motions with arbitrary perturbations. Our results show that realistic recovery motion emerges as a consequence of the interaction of the kinematic and the dynamic control. For example, the character sticks out her arms to recover from a push. We believe this behavior is due to the fact that the objective function must pull the joints back to the original trajectories without using any internal torques in the near-unactuated coordinates.

3.2 Overview

Our algorithm first computes the near-unactuated coordinates from an input motion offline, then simulates the motion by imposing dynamic constraints in the near-unactuated coordinates. The entire algorithm can be described in three simple steps.

1. Given an input motion sequence $\bar{\mathbf{Q}}$, solve the inverse dynamics problem to obtain the internal joint torques $\mathbf{U}$ on the upper body.

2. Apply PCA on the covariance matrix of $\mathbf{U}$ to obtain a set of eigenvectors $\mathbf{E}$. Define the near-unactuated coordinates $\hat{\mathbf{E}}$ as a subset of $\mathbf{E}$ with the k smallest corresponding eigenvalues.

3. Formulate a constrained optimization problem at each frame to solve for a pose that satisfies the equations of motion in $\hat{\mathbf{E}}$, while maintaining the reference

motion $\bar{\mathbf{Q}}$.

3.3 Preprocessing

We represent the character's skeleton as a transformation hierarchy of 18 body nodes with 24 DOFs on the upper body, and 12 DOFs on the lower body, denoted as $\mathbf{q}_u$ and $\mathbf{q}_l$ respectively. The global translation and orientation are represented by six DOFs at the root of the hierarchy, denoted as $\mathbf{q}_r$.

In the preprocessing step, we identify the near-unactuated coordinates from a manually selected portion of the input motion, usually a cycle of a periodic motion such as a walking cycle.

We first solve for joint torques using the equation of motion.

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{u} + \sum \mathbf{J}^T \mathbf{f}, \tag{1}$$

where $\mathbf{M}$ is the inertia matrix, $\mathbf{C}$ is the Coriolis-Centrifugal force, $\mathbf{J}$ is the Jacobian matrix that projects external force $\mathbf{f}$ to the generalized coordinates, and $\mathbf{u}$ is the generalized torque vector. Since only the lower body experiences forces from the ground in the input motion, and the $\mathbf{J}$ does not span the upper body DOFs, we can easily solve for $\mathbf{u}^t$ on the upper body for any frame t from the input motion alone. Joint velocity $\dot{\mathbf{q}}$ and acceleration $\ddot{\mathbf{q}}$ are approximated from the reference motion using finite difference.

$$\dot{\mathbf{q}}^t = \frac{\mathbf{q}^t - \mathbf{q}^{t-1}}{\Delta t}, \tag{2}$$

$$\ddot{\mathbf{q}}^t = \frac{\mathbf{q}^{t+1} - 2\mathbf{q}^t + \mathbf{q}^{t-1}}{\Delta t^2}, \tag{3}$$

where Δt is the time step.

The generalized joint torque space is then formed by concatenating all torques in the selected input motion cycle.

$$\mathbf{U} = [\mathbf{u}^1, \mathbf{u}^2, \cdots, \mathbf{u}^{\mathrm{T}}] \in \Re^{24 \times \mathrm{T}}, \tag{4}$$

where T is the total number of frame in the motion.

The torque space can be decomposed into a set of orthonormal basis $\mathbf{E}$ using eigen decomposition.

$$\mathbf{U}\mathbf{U}^T = \mathbf{E} \begin{bmatrix} e_0 & & & \\ & e_1 & & \\ & & \ddots & \\ & & & e_{24} \end{bmatrix} \mathbf{E}^T \tag{5}$$

Columns in $\mathbf{E}$ are eigenvectors ranked from the largest corresponding eigenvalue e_0 to the smallest e_{24}. We divide $\mathbf{E}$ into two sets, $\mathbf{E} = [\check{\mathbf{E}}\ \hat{\mathbf{E}}]$: $\check{\mathbf{E}}$ contains the first $24-k$ eigenvectors and $\hat{\mathbf{E}}$ contains the rest of the eigenvectors with k smallest corresponding eigenvalues. We then define $\hat{\mathbf{E}}$ as the set of near-unactuated coordinates. In our implementation, k is empirically set to 10 for all the examples except for the Tai-Chi motion.

3.4 Simulation with constraints

We discretize time domain into intervals of $\Delta t = 1/60$s as in the input motion. At each time step, we solve for upper body joint angles $\mathbf{q}_u$ of the next interval $t+1$ by formulating a constrained optimization. We use dynamic constraints $\mathbf{C}_D$ to ensure that the near-unactuated coordinates have zero internal actuation at all times.

$$\mathbf{C}_D \equiv \hat{\mathbf{E}}^T \mathbf{u}^t(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{f}) = \mathbf{0}. \tag{6}$$

The joint torque vector $\mathbf{u}$ is computed via Equation (1), thus expressed as a function of $\mathbf{q}$, $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$ and $\mathbf{f}$. If there is no perturbation ($\mathbf{f} = \mathbf{0}$), the original motion is close to satisfying $\mathbf{C}_D$. When a perturbation occurs ($\mathbf{f} \neq \mathbf{0}$), however, the character must adjust her motion to maintain $\mathbf{C}_D = \mathbf{0}$.

We use a spring-like objective to track the input motion $\bar{\mathbf{Q}} = (\bar{\mathbf{q}}^1, \bar{\mathbf{q}}^2 \cdots, \bar{\mathbf{q}}^{\mathrm{T}})$ and

a damping objective to model the dissipation in the dynamic system:

$$\mathbf{G}_p = \mathbf{q}_u^{t+1} - \bar{\mathbf{q}}_u^{t+1} \tag{7}$$

$$\mathbf{G}_v = \frac{\mathbf{q}_u^{t+1} - \mathbf{q}_u^{t}}{\Delta t} \tag{8}$$

When human is perturbed unexpectedly, there is typically a delay between the perturbation and muscle activation due to the latency in sensory feedback [85, 40]. The delay on arm movement due to sensory feedback usually ranges from 150-250 ms. We incorporate this delay by minimizing the torque change for 200 ms after the perturbation in the highly actuated coordinates $\check{\mathbf{E}}$.

$$\mathbf{G}_u = \frac{\check{\mathbf{E}}^T(\mathbf{u}^t - \mathbf{u}^{t-1})}{\Delta t} \tag{9}$$

In summary, we formulate the following optimization at each time step to solve for upper body motion:

$$\underset{\mathbf{q}_u^{t+1}}{\operatorname{argmin}} \quad \|\mathbf{G}_p\|^2_{\mathbf{W}_1} + \|\mathbf{G}_v\|^2_{\mathbf{W}_2} + \|\mathbf{G}_u\|^2_{\mathbf{W}_3} \tag{10}$$

$$\text{subject to} \quad \mathbf{C}_D = \mathbf{0}. \tag{11}$$

$\mathbf{W}_1$, $\mathbf{W}_2$, and $\mathbf{W}_3$ are diagonal weight matrices. In all our experiments, we use the same set of parameters. $\mathbf{W}_1$ is set to $200{\cdot}\mathbf{1}$. For $\mathbf{W}_2$, the first $3{\times}3$ block corresponding to the DOFs of the spine is set to $30 \cdot \mathbf{1}_{3\times3}$, and the remaining diagonal elements are set of 10. $\mathbf{W}_3$ is set to $\frac{1}{30} \cdot \mathbf{1}$ at the time of perturbation, and then smoothly decreased to zero in 200 ms. These values only reflect the relative importance of the objectives. They do not depend on the input motion or the skeletal model.

At the absence of perturbations, the input motion $\bar{\mathbf{Q}}$ is the minimizer to this optimization problem by construction. Therefore, we can reproduce the reference trajectories precisely. At the presence of perturbations, the minimizer is no longer zero due to the dynamic constraints $\mathbf{C}_D$. Consequently, the optimal motion has to deviation from the reference according to the perturbations, resulting in responsive

behaviors. After the perturbations and $\mathbf{G}_u$ are removed, the character starts recovery to the reference. Since $\mathbf{q}$ is already different from $\bar{\mathbf{q}}$, the two objective terms $\mathbf{G}_p$ and $\mathbf{G}_v$ together determine an intermediate recovery pose as the optimal motion. Changing the ratio of their weights $\mathbf{W}_1$ and $\mathbf{W}_2$ can result in motions that under-shoot or over-shoot the reference. This optimal recovery motion, however, is not always achievable due to the dynamic constraints. The interplay of the objective and dynamic constraints results in interesting recovery behaviors.

3.5 Lower body posture

Although our method focuses on the upper body response, we formulate a simple formula for the root and lower body motion when the character is perturbed. Since our method does not model the ground contact and friction forces, the impact of the perturbation on the root can simply be modeled as an impulse, proportional to the external force $\mathbf{f}$.

$$\dot{\mathbf{q}}_r^{t+1} = \dot{\mathbf{q}}_r^t + \frac{\Delta t}{m}\mathbf{J}_r^T\mathbf{f} \tag{12}$$

where m is the total mass of the character and $\mathbf{J}_r$ consists of the columns of $\mathbf{J}$ corresponding to $\mathbf{q}_r$. If the root movement causes footskating or penetration of the ground, we apply a simple inverse kinematics method on the lower body to fix the foot contacts.

3.6 Results

We applied our method to a variety of cyclic motions with different styles performed by different subjects. Most motions are robust to perturbations with 10 dynamic constraints except for the Tai-chi motion where 5 constraints are used. Our results reveal that dynamic constraints in the near-unactuated coordinates produce compliant responses to unexpected perturbations and coordinated recovery motions customized to the input motion.

3.6.1 Eigenvector analysis.

To demonstrate the importance of the joint actuation space, we conducted several experiments of a normal walk with different choices of coordinates in which dynamic constraints are enforced. We first simulated the same input motion with different numbers of dynamic constraints in the near-unactuated coordinates. The character appears more responsive as the number of dynamic constraints increases. However, the character is not able to completely recover from a perturbation when there are more than 12 dynamic constraints (Figure 2(d)). When the number of dynamic constraints increases to 16, the character simply fails to track the input motion (Figure 2(b)). The second experiment simulated the motion with a single dynamic constraint in the coordinate corresponding to the largest eigenvalue. The result shows that the character is not able to maintain the original motion without actuation in the most important coordinate (Figure 2(a)).

Currently, the number of dynamic constraints are chosen empirically by conducting a few experiments using our algorithm. To better understand the distribution of eigenvalues, we plot them for a few reference motions across individuals and activities. Figure 3 shows the distribution of eigenvalues for each motion, and Figure 4 shows the accumualted distributions of eigenvalues. The graphs indicate that only a very small number of eigenvalues (≤ 5) are dominant in the energy spetrum in all the motions. However, in our simulation, the space spanned by those relatively smaller eigenvalues are still essential in producing the reference motion and to recover from perturbations. Although we can observe several gaps in the distribution of eigenvalues, we are not able to use them as a guidance for choosing the unactuated space. A more thorough analysis of the characteristics of the eigenvalue distribution in the future can provide more insights on the underlying mechanisms of our method.

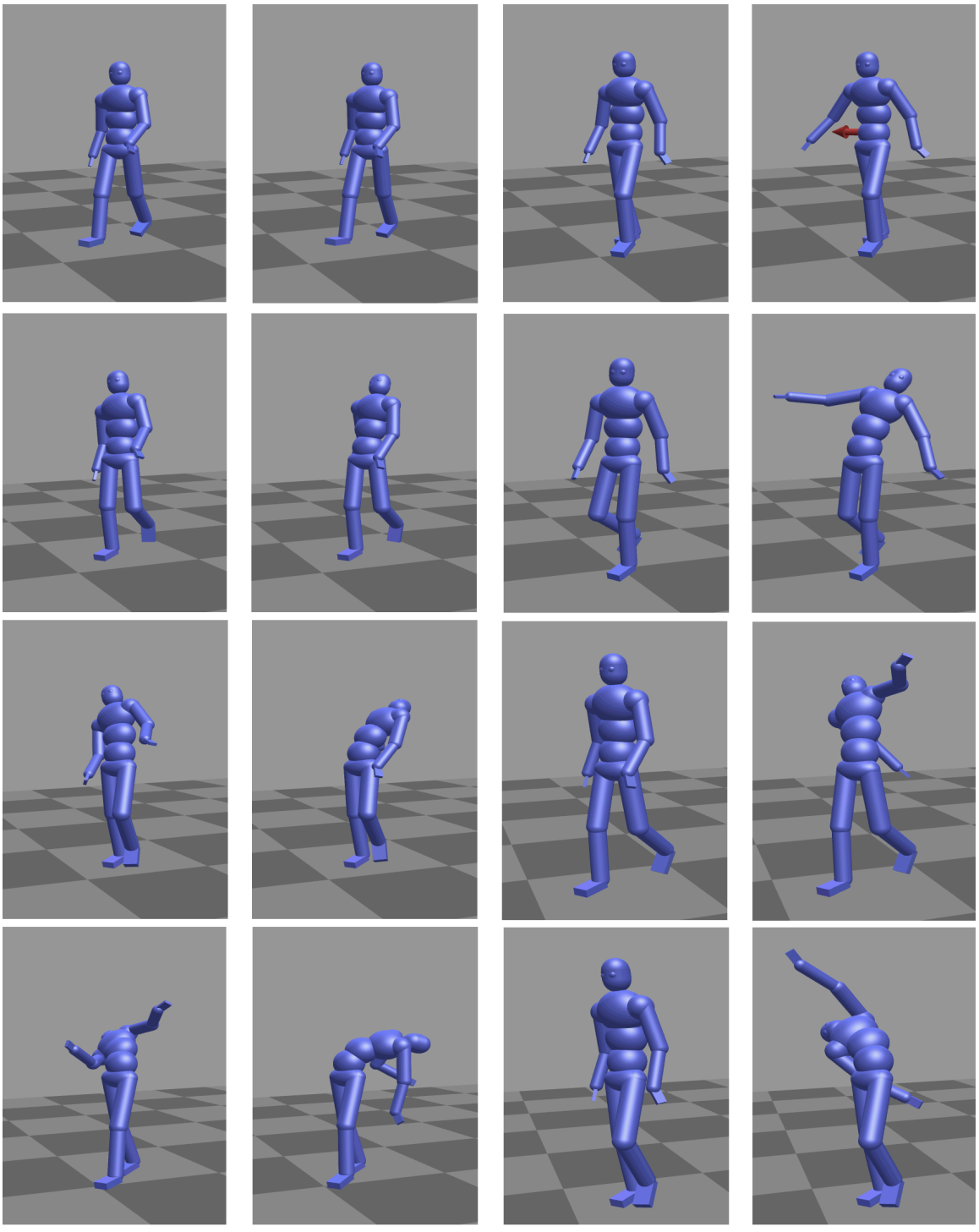


Figure 2: (a) The character fails to track the walking motion with a single dynamic constraint in the most important coordinate. (b) The character fails with 16 dynamic constraints. (c) The character tracks the the walking motion with 12 dynamic constraints. (d) The character fails to recover from a mild push to the right arm with 12 dynamic constraints.

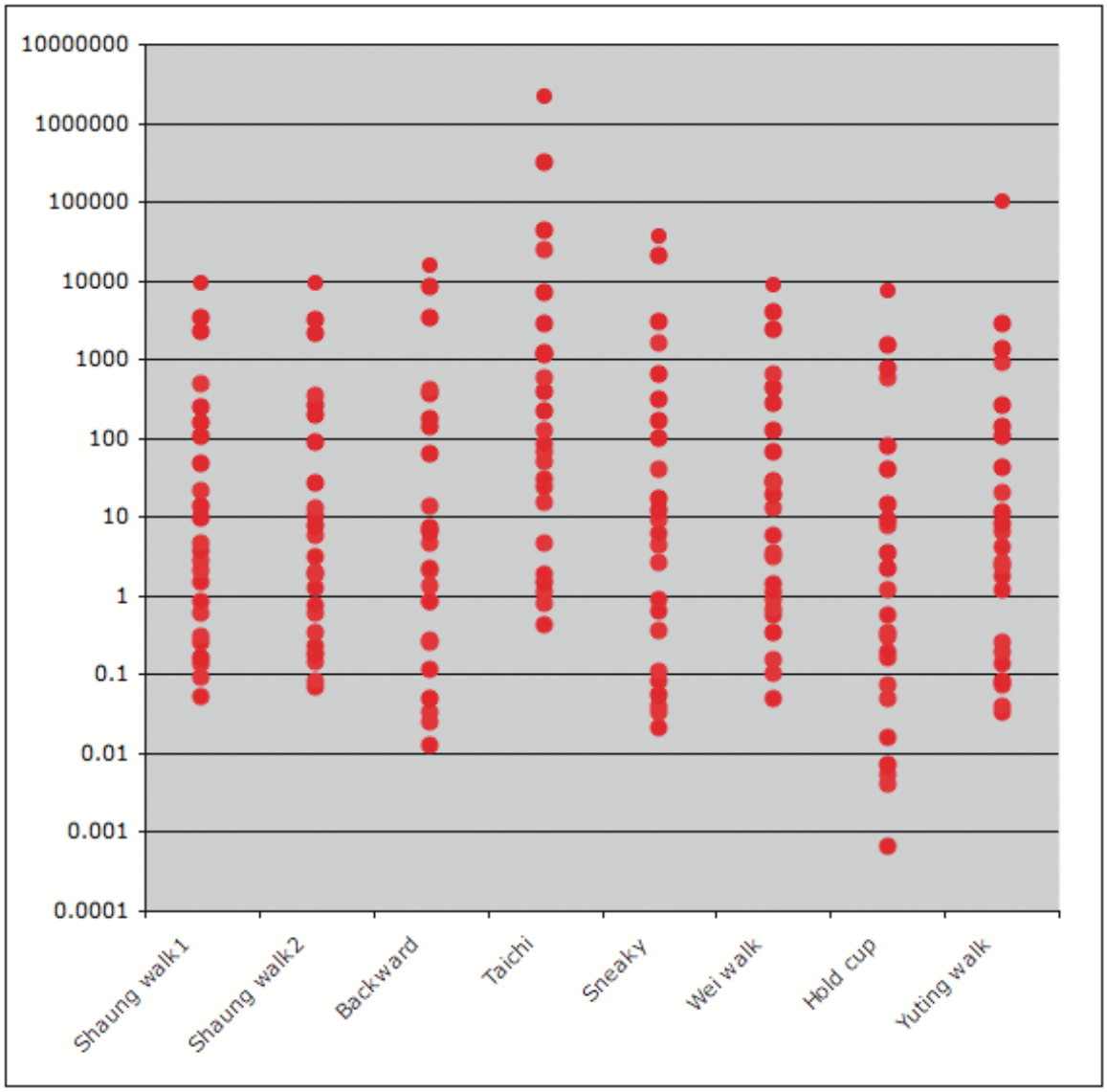
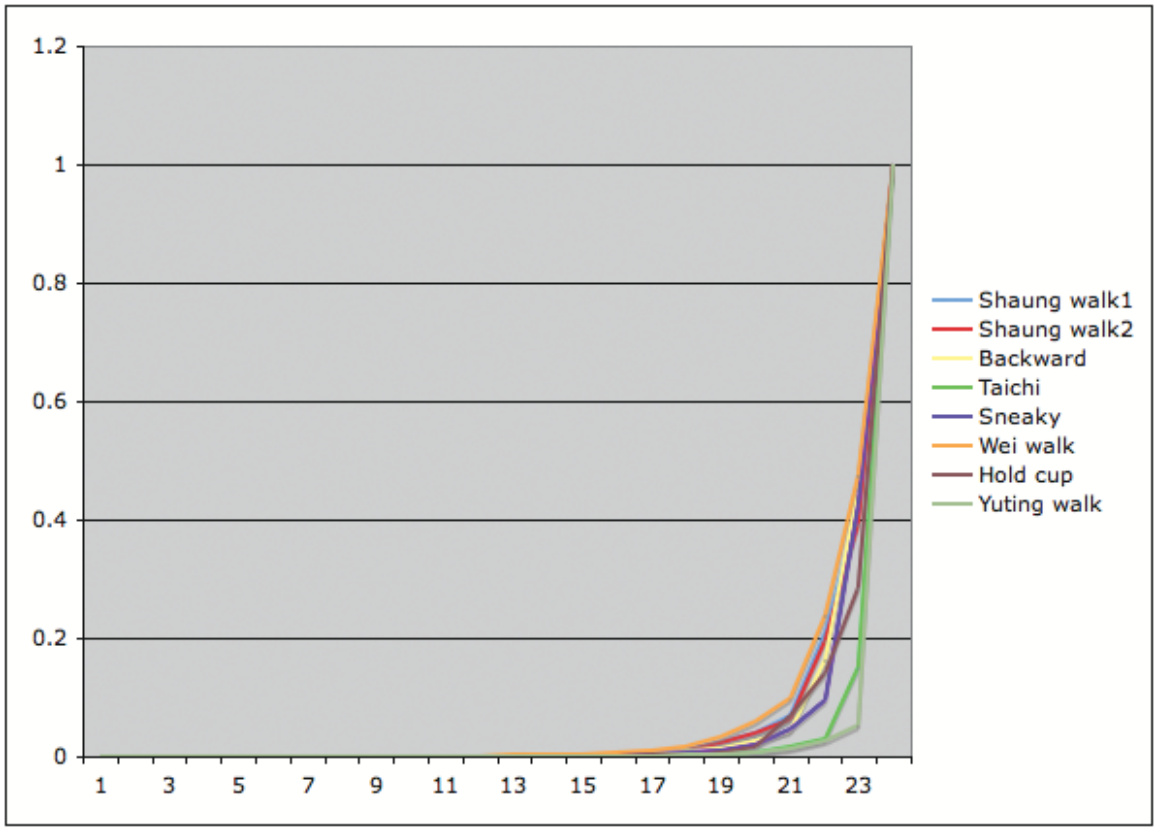
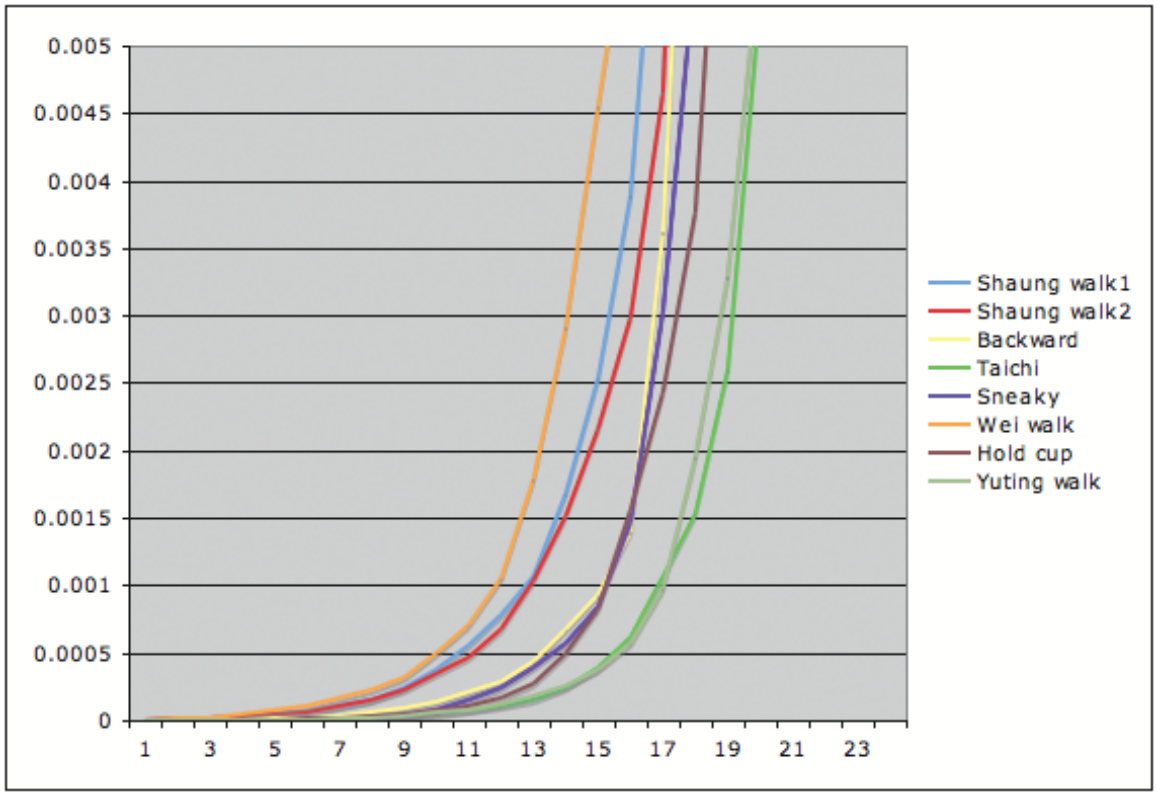


Figure 3: Eigenvalue distributions of several motions across individuals and activities.



(a) Energy distribution of all 24 eigenvalues.



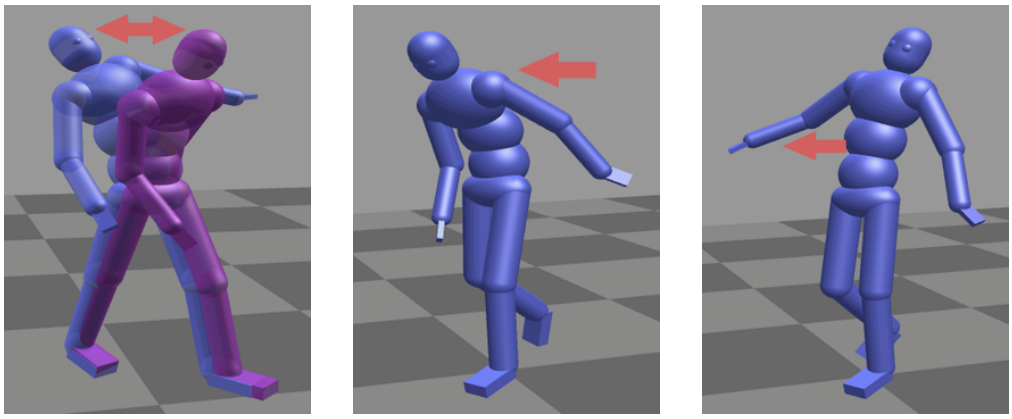
(b) Energy distribution of the smaller eigenvalues.

Figure 4: Distribution of energies of the eigenvalues in log scale.

3.6.2 Response to perturbations and recovery.

Our first experiment applied the same external force on different body parts of a 1.7m, 80kg male character during different phases of a normal walking sequence. The results show that the same push incurs a larger response during the single support than the double support. Moreover, the character exhibits more stability when the push is applied on the same side of the supporting leg. When pushed on the arms, the character reacts more compliantly than when pushed on the head or shoulder (Figure 5). The second experiment tested the effect of different external forces directions. The character has a harder time recovering from a backward push than a forward one, indicating that his torso actuation is asymmetric along the sagittal direction. In addition to producing highly coordinated reactions, our method also preserves individual styles. We demonstrated that the large-scale arm movement of a female character (1.5m, 40kg) is preserved in her reactive motions. We scaled the magnitude of the external forces proportionally to the female subject's weight.

Our method also allows the user to interact with the character by perturbing the root movement. To illustrate this, we simulated the reaction of the character stepping on a fast moving platform. As the root accelerates abruptly, the character's upper body reacts passively and gradually recovers to the original motion pattern.

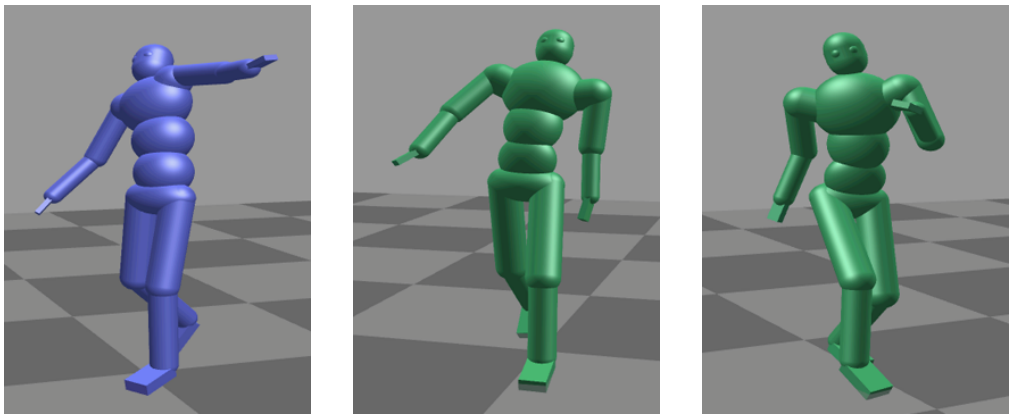


(a) Head (b) Left shoulder (c) Right arm

Figure 5: Perturbations (indicated by red arrows) on different body parts

3.6.3 Styles.

The coordinates in the actuation space encode muscle usage and coordination specific to the input motion. As a result, each motion sequence reacts to the unexpected perturbations with a unique *style*. We applied the same set of external forces to normal walk, backward walk, and sneaky walk performed by the same male character (Figure 6). In comparison to other motions, the normal walk exhibits higher coordination among the upper body as it counteracts the disturbance using the torso and both arms simultaneously. The backward walking motion exhibits higher stability against a forward push but responses compliantly to a backward push. In the sneaky walk, the character maintains a more stable posture with the center of mass position lower than other motions. The results show that the same amount of force induces smaller responses on a sneaky walk.

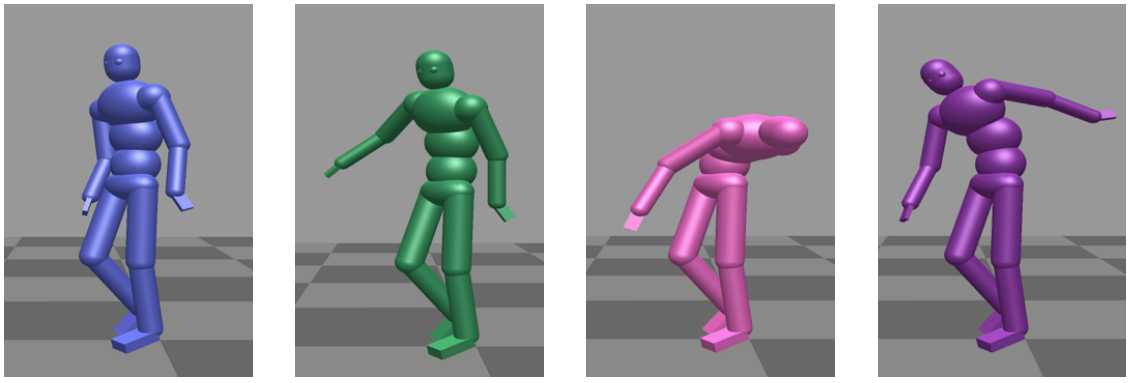


(a) Forward walk (b) Backward walk (c) Sneaky walk

Figure 6: The character recovers from a backward push while performing different activities.

To compare the actuation among styles of individuals, we extracted the near unactuated coordinates of one individual performing a normal walk, and applied them to simulate another individual's normal walk under perturbations (Figure 7). The results show that plausible reactive motions can be generated only when the two individuals have similar weight and height. We also conducted similar experiments

for different action styles. The actuation of a sneaky walk reproduces a normal walk faithfully without disturbances, but generates unrealistic response when perturbed.



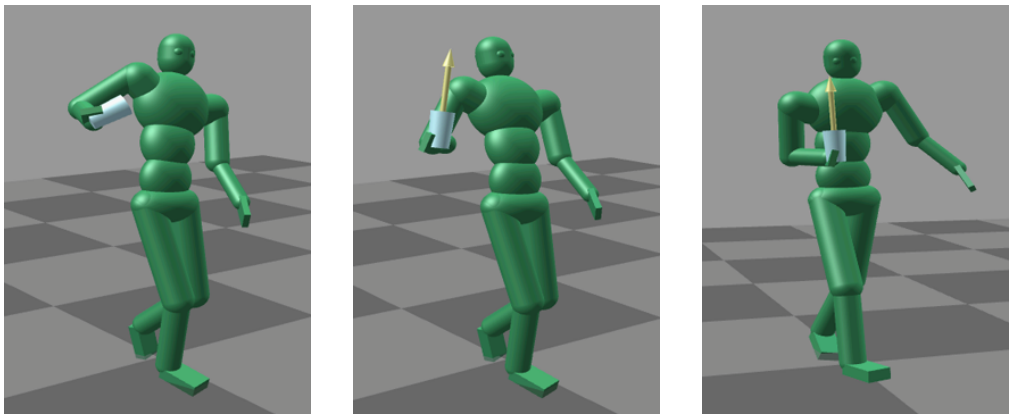
(a) (b) (c) (d)

Figure 7: Different actuation spaces applied on the same walking motion under a forward push. (a) Reference response. (b) Using actuations from another individual of similar build performing a similar walk, the character responses to the push and recovers. (c) Using actuations from another individual of considerably different build, the character cannot recover from the push. (d) Using actuations from the same individual but performing a sneaky walk, the character produces an unrealistic response.

3.6.4 Additional objectives.

Our formulation allows the animator to include additional objectives to enforce kinematic properties of the input motion. For example, we captured a walking sequence with the subject holding a cup in his right hand. During motion synthesis, an additional objective was added to keep the cup in an upright orientation. The asymmetrical muscle usage in the left and the right arms results in many interesting behaviors. When the character is pushed on the right arm, he maintains the orientation of the cup by rotating his torso to compensate for the movement of his right arm (Figure 8(b)). In contrast, when the left arm is pushed by the same force, he stiffens his torso to reduce its movement and the impact on the right arm (Figure 8(c)).

Similarly, we added an objective that repels the character from the obstacles in the environment. If the character fails to completely avoid the obstacles, an external



(a) (b) (c)

Figure 8: The character recovers from a push while holding a cup in his right hand. (a) The character tilts the cup in response to a push on his right arm. (b) The character holds the cup upright during recovery when an objective is added to maintain the orientation of the cup (yellow arrow). (c) The character stiffens his torso in response to a push on his left arm so that the impact on the cup is reduced.

force is applied at the site of collision (Figure 1 right).

3.6.5 Static balance

Our method also works on other periodic motions such as Tai-Chi forms. Although the Tai-Chi motion requires higher overall internal torques than other locomotion sequences (only 5 near-unactuated coordinates), the highly actuated coordinates mostly lie on the frontal plane. Moreover, the torque usage of arms in the Tai-Chi motion are highly correlated. As a result, the character reacts to perturbations on the sagittal plane with both arms moving fluidly (Figure 9).

3.7 Alternative formulations

There are two alternative formulations to our problem. One variation is to subtract the mean joint torques before we perform eigen decomposition to identify the null space, which we name *mean actuation*. The other variation is to reduce the dynamic model from the standpoint of dimensionality reduction in the joint motion space. We will discuss them in more detail here.

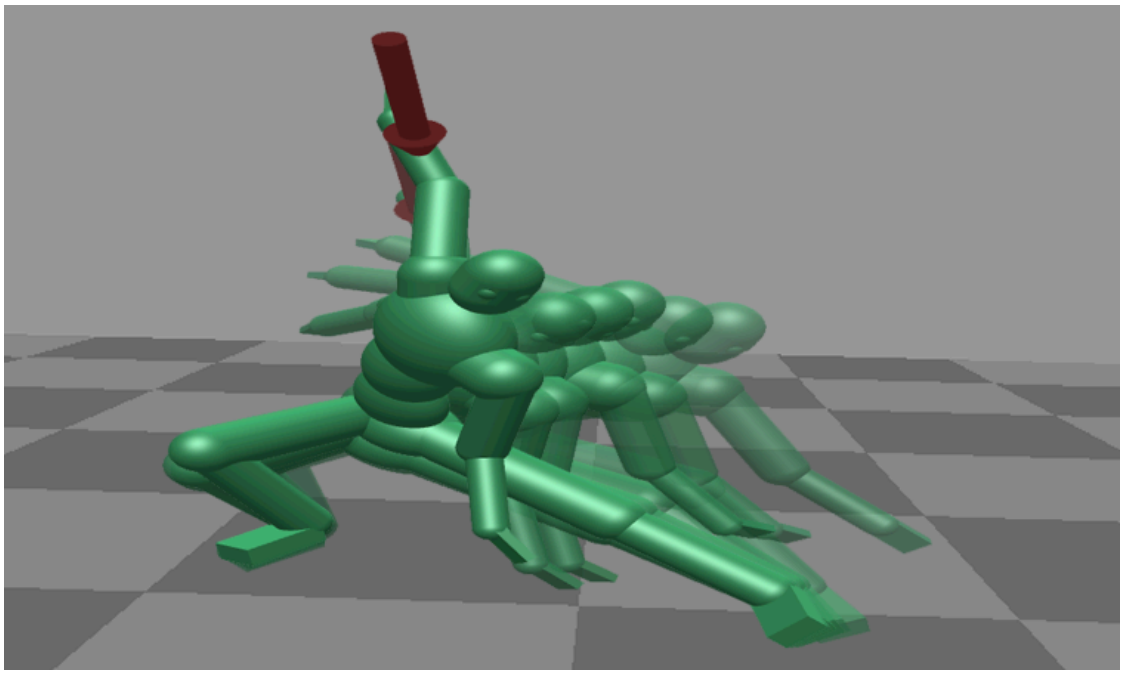


Figure 9: The character responses to a push while performing a Tai-Chi form.

3.7.1 Mean actuation

The first formulation shifts the torque space to its center by subtracting the mean value $\bar{\mathbf{u}}$ from joint torques. Performing eigen decomposition on the mean subtracted torques will result in a new set of near-unactuated coordiantes $\hat{\mathbf{E}}'$.

In online motion synthesis, the dynamic constraint in Equation (6) needs to change accoordinly to the following.

$$\mathbf{C}_D \equiv \hat{\mathbf{E}}'^T \mathbf{u}^t(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}, \mathbf{f}) = \hat{\mathbf{E}}'^T \bar{\mathbf{u}}. \tag{13}$$

This formulation essentially applies a constant mean actuation $\hat{\mathbf{E}}'^T \bar{\mathbf{u}}$ to the otherwise unactuated space. Because $\hat{\mathbf{E}}'$ and $\hat{\mathbf{E}}$ normally span different spaces, the mean actuations are not very small numbers in our motions. However, we do not observe qualitatively different behaviors between the mean actuation and the unactuated formulations. We hypothesize that as long as a subspace of torques does not adapt to perturbations, we will observe qualitatively similar coordinated responses.

3.7.2 Dimensionality reduction in motion space

The second formulation considers dimensionality reduction in the joint motion space. The system dynamics is accordingly projected to the low dimensional motion space [1].

We can decompose the motion space into a rank space $\check{\mathbf{B}}^{24\times(24-l)}$ and a null space $\hat{\mathbf{B}}^{24\times l}$ by applying eigen decomposition to the covariance of $\bar{\mathbf{Q}}$ as we did in Equation (5). The joint configuration $\bar{\mathbf{q}}^t$ can be completely represented by a low dimensional motion vector $\mathbf{s}$ such that

$$\bar{\mathbf{q}}^t = \check{\mathbf{B}}\mathbf{s}^t, t = 1, 2, \cdots, T. \tag{14}$$

Taking time derivatives on both sides gives rise to the mapping of velocities. That is, velocities should lie in the same motion subspace.

$$\dot{\mathbf{q}} = \check{\mathbf{B}}\dot{\mathbf{s}}. \tag{15}$$

We denote the generalized torques that correspond to $\mathbf{s}$ as τ. From Equation (15) and the principle of virtual work, we can derive the mapping between $\mathbf{u}$ and τ as

$$\tau = \check{\mathbf{B}}^T\mathbf{u}. \tag{16}$$

The form of Equation (16) seems to suggest that $\check{\mathbf{B}}$ and $\check{\mathbf{E}}$ may correspond to the same space. Intuitively, $\check{\mathbf{B}}$ and $\check{\mathbf{E}}$ are both reduced torque spaces that can sufficiently generate the motion $\mathbf{q}$. While $\check{\mathbf{E}}$ is constructed to be the smallest space, however, it is not obvious whether $\check{\mathbf{B}}$ is also the smallest. In other words, we question whether the motion space and the torque space are of the same dimension [2]. Our intuition is that the motion space is possible to be larger than the torque space. For example, the torque space of a passive system is an empty space, but the corresponding motion space can be of full rank. The rationale behind is that as torques span a larger space,

[1] Professor Emo Todorov pointed out this approach in a discussion.
[2] Jie Tan pointed out the difference in dimensions during a discussion.

the motion should be more structured and span a smaller space. Therefore, we can deduce that $\check{\mathbf{E}}$ is a subspace of $\check{\mathbf{B}}$, and thus $\hat{\mathbf{B}}$ is a subspace of $\hat{\mathbf{E}}$. We can also prove $\hat{\mathbf{B}}^T\mathbf{u} = \mathbf{0}$.

This result suggests that we can also use joint motions to identify an unactuated subspace. In theory, this subspace is probably lower dimensional than the one identified from torques, thus may result in less compliant motions when perturbed. In practice, it is difficult to tell from data the true dimensions of $\hat{\mathbf{E}}$ and $\hat{\mathbf{B}}$ due to the presence of noise. In fact, previous work of dimensionality reduction in walking motion discover a much larger null space than what we are using in this work. This may be explained by the fact that $\hat{\mathbf{E}}$ is computed through inverse dynamics, thus suffered from the exaggerated noise in accelerations computed from finite difference, and the modeling error in the skeleton. $\hat{\mathbf{B}}$ seems like a better choice in practice because it is less sensitive to noise and does not depend on the estimated mass distribution of the skeleton.

3.8 Discussions

Our approach uses inverse dynamics methods and principal component analysis, both of which are known to be sensitive to input noise. Fortunately, our method does not directly apply the computed torques to simulate motion but only uses them to derive the actuation space of the input activity. We tested the robustness of our method against data noise by randomly selecting different cycles from the input motion. The results show that sporadic noise in the motion has negligible effect as long as the input motion contains sufficient clean data.

Independent component analysis (ICA) is also a useful tool in discovering features from data. We do not use it in this work because we are interested in identifying the space as a whole rather than understanding the individual source of variations. In the future, we would like to explore a sensible parameterizations of the actuation

space using ICA or other learning methods to associate the synergies with functional capabilities.

We have found that our technique works better when the input motion has relatively low velocity. When tested on a high-speed turning motion, our method produces visually stiff responses. We believe the major cause of failure is that the eigenvalues computed from the turning motion are significantly higher, indicating most of coordinates are highly actuated. Enforcing the same number of dynamic constraints ($k = 10$) resulted in overly passive motion that fails to track the original motion. However, using fewer dynamic constraints produces overly stiff reactive motion to perturbations.

The technique in this chapter focuses only on the upper body response and is not suitable for large perturbations that incur the loss of balance or changes of high-level behaviors. We anticipate that the technique can be applied to the whole body motion if we can accurately measure the ground contact forces. One possibility is to estimate the ground contact forces from motion capture data using the method described by Liu *et al.* [78]. Another promising future direction is to combine our technique with sophisticated balance controllers that determine the lower body and root movements.

In the next chapter, we will describe an algorithm to derive robust full body balance controllers from motion capture sequences. The controllers can adapt the sequences to various dynamics environments.

CHAPTER IV

LOCOMOTION CONTROL WITH OPTIMAL FEEDBACK

This chapter describes a technique that adapts a motion capture sequence to virtual environments with large-scale physical perturbations in real-time [139]. We propose an abstract dynamic model to describe the dynamics of any given input motion, and automatically derive an optimal feedback controller that adjust the motion of the abstract model as well as the completion time of the motion on-the-fly in anticipation of changes in the long-term goal (Figure 10) and at the presence of unexpected pushes. An online optimization interactively reconstruct the full body motion that retain the styles of the input and respect the motion of the abstract model. We applied our algorithm to a wide range of motions including different styles of walking, running, and squatting. The resulting controllers are robust to large perturbations and changes in the environment.

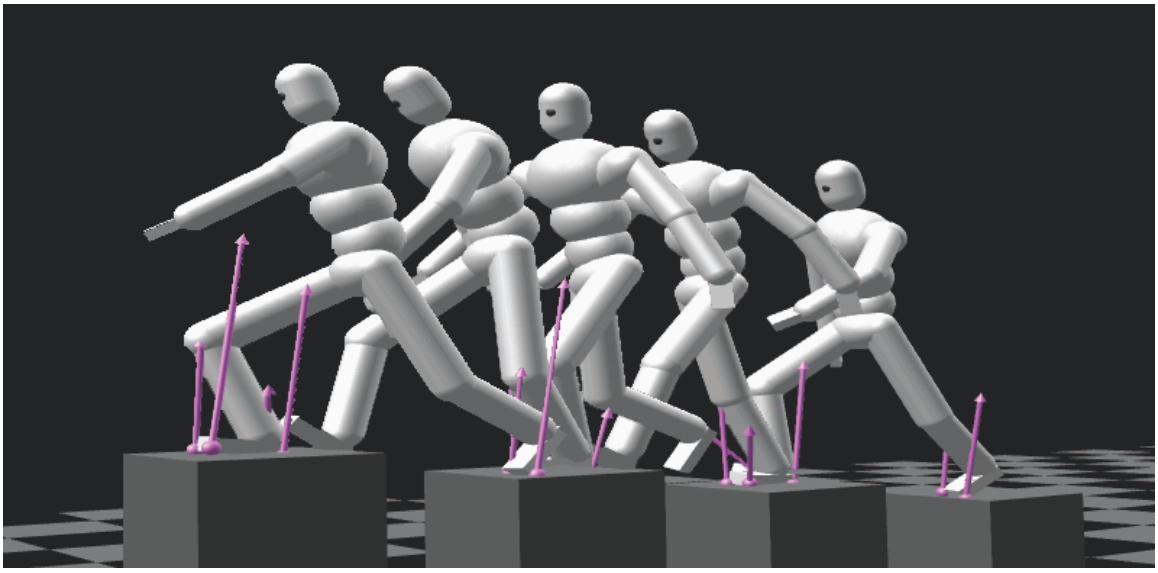


Figure 10: The character adjusts a large step to walk up a staircase of $0.1m$ height.

4.1 Motivations

Within the computer animation community, the primary technique for real-time adaptation of a motion sequence to physical interactions is motion tracking via feedback control algorithms. Strict motion tracking, however, is often an undesirable behavior when the performing character experiences a large perturbation, or is acting in a new environment. People anticipate potential future interactions, and may also re-plan their movements as the environmental situation dictates. In addition, human movement is governed by biomechanical and physical principles that strongly influence the shape and trajectory of the actions taken. In contrast, the goal of current feedback control algorithms for motion tracking is strictly adherence to the reference motion itself. Consequently, they often produce unnatural looking results when they need to recover from strong deviations from the original motion.

In this chapter, we describe a new approach to design feedback controllers robust to perturbations in the virtual world. We seek to design a controller that allows for online re-planning of long-term goals and incorporates an accurate nonlinear dynamic system with high-level balance strategy. Our formulation of motion tracking as an optimal control problem provides two key advantages over previous tracking controllers. First, our controller respects the final goal state and is flexible to adjust the completion time. Its ability to modify the final goal state and completion time produces strategies with anticipatory and replanning behavior. Essentially, the controlled character can "see" the change of the environment ahead of time and adjust the control forces properly in advance. Second, incorporation of a nonlinear dynamic system provides more accurate estimates of the control outcome, and a high-level balance strategy would ensure robust behaviors. As a result, our controller can perform well with large feedback errors. The combination of these improvements enables our control algorithm to generate realistic and robust adaptations from a reference motion to widely varying conditions.

While optimal control theory offers useful tools for solving feedback controllers for a variety of problems including those with final constraints and flexible completion time, our particular problem poses unique challenges. Our feedback controller requires the solution of a two-point boundary optimal trajectory problem, which is known to be very difficult for large nonlinear dynamic systems. The nonlinearity and complexity of human motion makes a full-body formulation impractical. A practical alternative is to formulate a linear quadratic regulator (LQR): a linearized dynamic system with quadratic objective functions. However, this simplification cannot handle higher-order objectives such as angular momentum regularization, which is an important biomechanical principle shown to be essential for balance. Moreover, the linear approximation of dynamics fails rapidly for large state errors. To deal with these issues, we designed an abstract dynamic system that expresses fundamental aspects of human motion, especially the relation between contact forces and angular momentum, and is still manageable by existing trajectory optimization techniques such as differential dynamic programming (DDP).

The abstract dynamic model takes global motion, including center of mass position and linear and angular momentum, as state and contact forces as control. A control policy of this model addresses one of the most fundamental problem in human motion: the relation between the under-actuated degrees of freedom and the contact forces. With no assumptions of the underlying kinematics structure, our abstract model is generic enough to represent any motions that utilize contacts.

Our method greatly enhances the capability of one single motion capture sequence under different dynamical conditions. Results show that our controller performs robustly for different types of motion, including a normal walk, a big stride walk, a wander with random turns, a squat exercise, a run, and a hop. Our controller consistently produces natural responses to dynamical and environmental perturbations.

4.2 Overview

The input to our algorithm is a reference motion sequence $\bar{\mathbf{Q}}$, and the output is a real-time motion controller that tracks the input motion and allows for both passive responses to perturbations and active re-planning of goals. As illustrated in Figure 11, our algorithm consists of an offline optimization and an online simulation. The offline optimization solves an optimal feedback controller for the abstract model. The online simulation then uses the controller to simulate motions for the abstract model, and synthesizes full-body motions that are dynamically consistent with the abstract model and kinematically consistent with the reference.

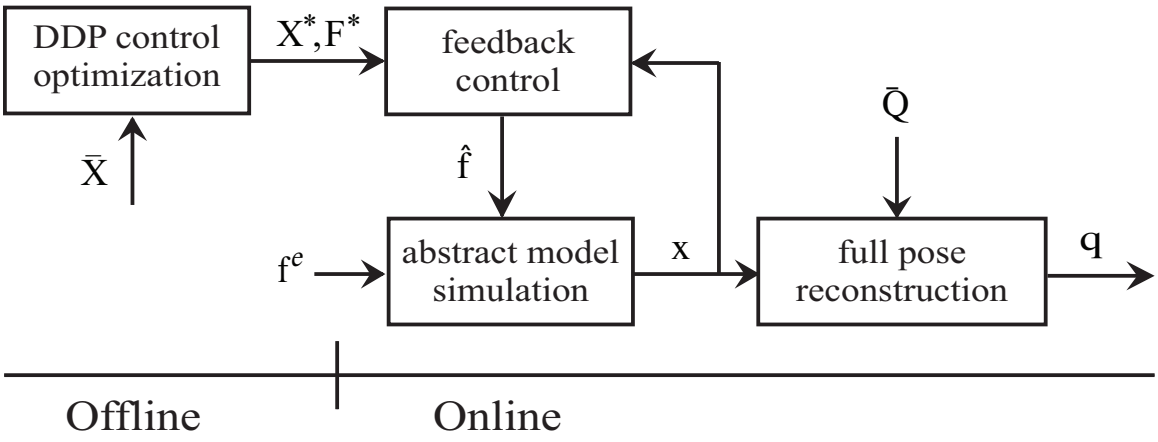


Figure 11: Algorithm overview.

In the offline optimization, we use DDP to solve an optimal control problem for the abstract model. This control problem uses minimum angular momentum and control forces to reproduce the reference trajectory of the abstract model $\bar{\mathbf{X}}$ computed from $\bar{\mathbf{Q}}$. The solution is a trajectory of control forces $\mathbf{F}^*$ and the corresponding motion trajectory $\mathbf{X}^*$. In addition, we derive a feedback controller that optimizes the same problem in neighboring trajectories of $\mathbf{X}^*$ and allows for changes in motion timing and the final state.

During online simulation, users can apply external forces $\mathbf{f}^e$ to the simulator, or change the environment on-the-fly. The feedback controller will automatically adjust

the control forces $\hat{\mathbf{f}}$ and timing to account for any deviations in states, changes of the final state, and differences in contact positions from the reference motion. The resulting motion $\mathbf{x}$ will robustly respond to the user supplied stimuli while maintaining the desired course of action. Finally, we use an optimization to reconstruct the full body motion $\mathbf{q}$ in every frame by matching the abstract model motion $\mathbf{x}$ with minimum deviation to $\bar{\mathbf{Q}}$.

4.3 Abstract model

The abstract model is showing in figure 4.3. Its state variable $\mathbf{x}$ is defined as the global motion of the character and the control variable is defined as the contact forces $\mathbf{F}$. The global translational motion can be described by the position of the center of mass (COM), $\mathbf{C}$, and linear momentum $\mathbf{P}$ and angular momentum $\mathbf{L}$. We also want to represent the global orientation, but it cannot be directly computed from COM, so we approximate its effect with the integral of angular momentum Φ:

$$\Phi(t) = \Phi(t_0) + \int_{t_0}^{t} \mathbf{L}. \tag{17}$$

$\Phi(t_0)$ is simply set to zero.

The state variable of the abstract model is then defined as

$$\mathbf{x} = \begin{bmatrix} \mathbf{C} \\ \mathbf{P} \\ \Phi \\ \mathbf{L} \end{bmatrix}. \tag{18}$$

The dynamic equation of the abstract model can be expressed in Equation (19).

$$\dot{\mathbf{x}} = A\mathbf{x} + B(\mathbf{x}, t)\mathbf{F} + G, \tag{19}$$

where

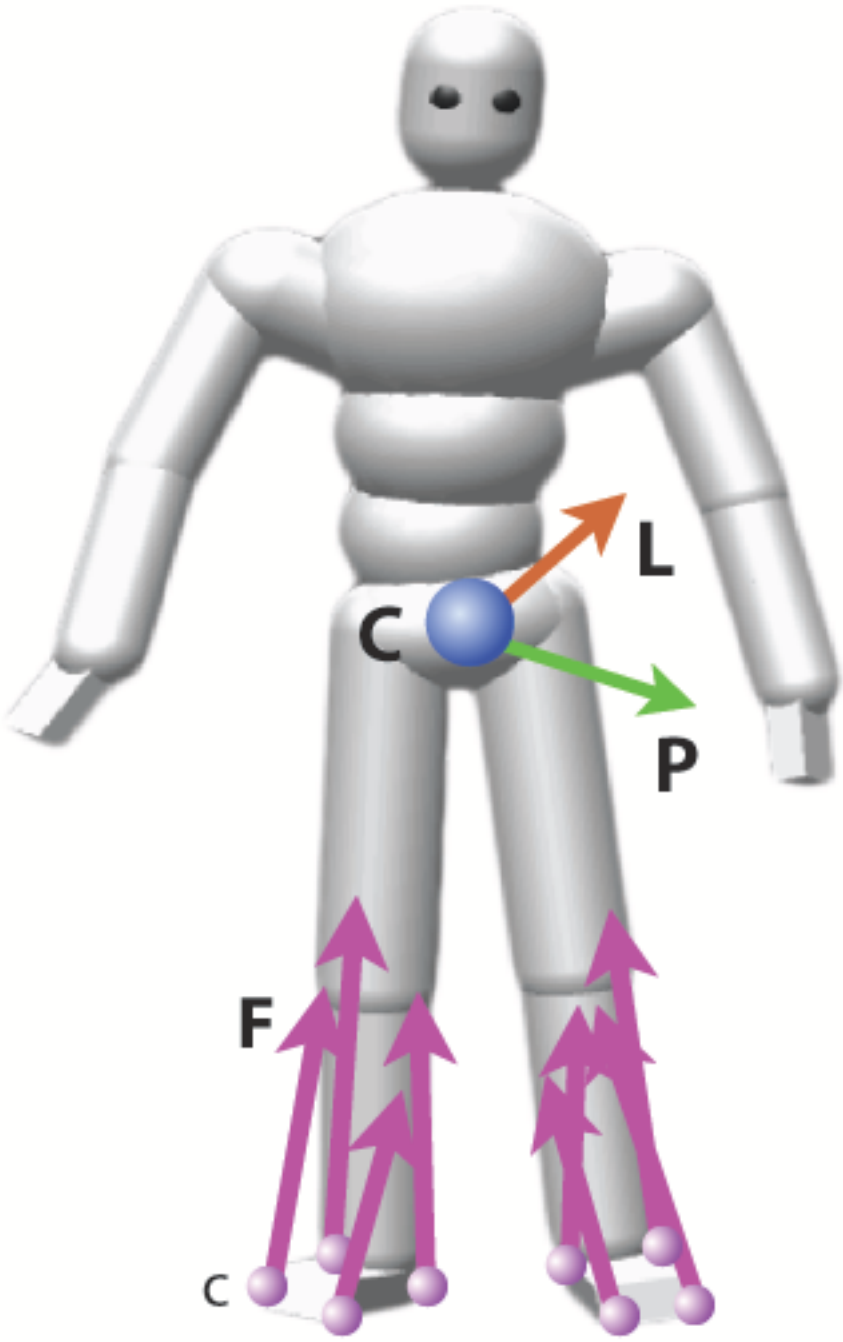


Figure 12: The entire character is reduced to an abstract dynamic model about its COM. The blue dot represents the position C, the green arrow and the orange arrow represent the linear momentum $\mathbf{P}$ and the angular momentum $\mathbf{L}$ respectively. Contact forces $\mathbf{F}$ are exerted on the contact points $\mathbf{c}$ on the feet.

$$A_{12\times12} = \begin{bmatrix} \mathbf{0} & \mathbf{I}_{3\times3}/M & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_{3\times3} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix},$$

$$B(\mathbf{x},t)_{12\times3p} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{I}_{3\times3} & \mathbf{I}_{3\times3} & \dots & \mathbf{I}_{3\times3} \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{0} \\ (c_1(t)-\mathbf{C})_\times & (c_2(t)-\mathbf{C})_\times & \dots & (c_p(t)-\mathbf{C})_\times \end{bmatrix},$$

$$G_{12\times1} = \begin{bmatrix} \mathbf{0} & M\mathbf{g} & \mathbf{0} & \mathbf{0} \end{bmatrix}^T,$$

where $\mathbf{g}$ is gravity, M is the total mass of the system, and $_\times$ denotes the skew-symmetric matrix form of a vector. The number of contacts p and their positions $c(t)$ are time-varying parameters determined by the input motion. As a result, the width of $B(\mathbf{x},t)$ depends on the number of contact points p at the time t in the input motion.

Our dynamic system is still nonlinear because of the product term of state and control. Nonetheless, without any further approximation or linearization, this abstract model significantly improves the convergence in control optimization described in Section 4.5. The problem would be otherwise impossible to solve for a full-body dynamic system. In addition, the generic representation of the abstract model enables a wide applicability of our feedback controller.

4.4 Optimal Control

Our goal is to derive a controller that is robust for a wide range of states around the reference trajectory and also reflects important properties in natural human motion such as minimum effort and regulating angular momentum to maintain balance. We

formulate an optimization problem with the desired objectives, so that we can approximate a feedback control policy for the neighboring states around the optimal solution.

Given a reference trajectory $\bar{\mathbf{X}}$ extracted from $\bar{\mathbf{Q}}$, we want to solve an optimal trajectory that respects its initial and final states. Incorporation of a final state constraint provides several benefits. First, it allows us to break down a long sequence into shorter segments and concatenate them seamlessly. Second, it explicitly enforces the motion to stay in balanced states. Third, it enables replanning of the final goal on the fly. To improve the robustness and naturalness of the control policy, we minimize angular momentum and control forces in addition to tracking the reference motion. From the optimal solution of this optimization problem, we obtain an online feedback controller for the neighboring states around the solution.

Equation (20) summarizes the optimization problem.

$$\begin{aligned} \min_{\mathbf{X},\mathbf{F}} \int_{t_0}^{t_f} (\|\mathbf{F}(t)\|^2_{\mathbf{W}_1} + \|\mathbf{x}(t)\|^2_{\mathbf{W}_2} + \|\mathbf{x}(t) - \bar{\mathbf{x}}(t)\|^2_{\mathbf{W}_3}) dt \\ \text{subject to} \quad \dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B(\mathbf{x}(t), t)\mathbf{F}(t) + G, \\ \mathbf{F}(t) \in \mathcal{K}, \\ \Psi(\mathbf{x}(t_f)) = \mathbf{x}(t_f) - \bar{\mathbf{x}}(t_f) = \mathbf{0}, \\ \mathbf{x}(t_0) = \bar{\mathbf{x}}(t_0), \end{aligned} \tag{20}$$

where t_f indicates the final time of the input motion. The contact forces are unilateral and constrained by their friction limits approximated by the friction cone $\mathcal{K}$ ([3]). $\mathbf{W}_1, \mathbf{W}_2,$ and $\mathbf{W}_3$ are diagonal weighting matrices for force minimization, angular momentum minimization, and tracking, respectively. Determining the weight is trivial for the abstract model. We will report the weights in Section 4.7.

We apply Differential Dynamic Programming ([56, 41]) to solve this fixed-time continuous optimization. The final constraint is incorporated by augmentation with

a Lagrangian multiplier μ in the objective function.

$$V = \mu^T \Psi(\mathbf{x}(t_f)) + \int_{t_0}^{t_f} (\|\mathbf{F}\|^2_{\mathbf{W}_1} + \|\mathbf{x}\|^2_{\mathbf{W}_2} + \|\mathbf{x} - \bar{\mathbf{x}}\|^2_{\mathbf{W}_3}) dt. \tag{21}$$

We initialize the controls using inverse dynamics of the reference trajectory, assuming this initial guess is close to a global solution. We follow the procedure described in Chapter 2.5 in Jacobson and Mayne [56] to solve this optimization. The midpoint method is used to numerically integrate the solution at each discrete time step. The solution $\{\mathbf{X}^*, \mathbf{F}^*, \mu^*\}$ satisfies $\Psi = \mathbf{0}$ by construction.

4.5 Optimal Feedback Control

In online simulation, we can apply the control force $\mathbf{F}^*$ to obtain exactly $\mathbf{X}^*$ with $\Psi = \mathbf{0}$. However, in the case of perturbations, such as pushes or changes in the environment, we need to adjust the control forces such that they still optimize V at the perturbed states. We use a first-order approximation to approximate the first derivatives of the perturbed states around $\{\mathbf{X}^*, \mathbf{F}^*, \mu^*\}$. Because first derivatives vanish at the solution, we get a linear feedback control policy. To produce a more robust and flexible controller, we in addition allow the final time to change, then our feedback control becomes a linear combination of small changes in $\mathbf{x}$, μ and t_f:

$$\delta\mathbf{F} = \mathbf{K}^{\mathbf{x}}\delta\mathbf{x} + \mathbf{K}^{\mu}\delta\mu + \mathbf{K}^{t}\delta t_f. \tag{22}$$

The time varying gains $\mathbf{K}^{\mathbf{x}}, \mathbf{K}^{\mu}$, and $\mathbf{K}^{t}$ can be computed from $\mathbf{X}^*$ and $\mathbf{F}^*$ (details in Appendix A).

If we can evaluate the deviation $\delta\mathbf{x}$, $\delta\mu$, and δt_f, we can use the control policy (Equation 22) to compute the deviation of control force $\delta\mathbf{F}$. The relation of $\delta\mathbf{x}$, $\delta\mu$, and δt_f is expressed in two linear equations derived from the linearization of the first-order optimality condition $V_\mu = \mathbf{0}$ and $V_{t_f} = 0$:

$$\delta V_\mu(\mathbf{X}^*, \mu^*, t_f^*) = V_{\mu\mathbf{x}}(t_c)\delta\mathbf{x} + V_{\mu\mu}(t_c)\delta\mu + V_{\mu t_f}(t_c)\delta t_f = \mathbf{0} \tag{23}$$

$$\delta V_{t_f}(\mathbf{X}^*, \mu^*, t_f^*) = V_{t_f\mathbf{x}}(t_c)\delta\mathbf{x} + V_{t_f\mu}(t_c)\delta\mu + V_{t_f t_f}(t_c)\delta t_f = 0 \tag{24}$$

4.5.1 Change of final time

In a fixed-time controller where $\delta t_f = 0$, the reference time index t_c is the same as the elapsed time index t. We can simply compute the deviation in the current state as $\delta \mathbf{x} = \mathbf{x}_t - \mathbf{x}^*_{t_c}$. However, in a free-final-time controller, t_c changes with the final time rather than incrementing along with t. At each time step, we estimate the remaining time $t_f - t_c$ based on $\delta \mathbf{x}$ and compare the new final time with t_f to get δt_f. This dependency between $\delta \mathbf{x}$ and t_c requires us to solve them simultaneously ([108]).

We first derive the relation between $\delta \mathbf{x}$ and δt_f from Equation (23) and Equation (24) as:

$$\begin{aligned} \delta t_f =& K^d \delta \mathbf{x}, \\ K^d =& \frac{V_{t_f \mathbf{x}} - V_{t_f \mu} V_{\mu\mu}^{-1} V_{\mu \mathbf{x}}}{V_{t_f \mu} V_{\mu\mu}^{-1} V_{\mu t_f} - V_{t_f t_f}}, \end{aligned} \tag{25}$$

where K^d is evaluated at t_c. We then approximate $\delta \mathbf{x}$ at $\mathbf{x}^*_{t_c}$ as:

$$\delta \mathbf{x} = (\mathbf{x}_t - \mathbf{x}^*_{t_c}) - \dot{\mathbf{x}}^*_{t_c}(t - t_c)h, \tag{26}$$

where h is the time step.

It is easy to see that $\delta t_f = (t - t_c)h$, the change in final time is the same as the change in the current reference index. Arranging terms in Equation (25) and Equation (26), we get Equation (27):

$$(t - t_c)h = \frac{K^d(t_c)}{1 + K^d(t_c)\dot{\mathbf{x}}^*_{t_c}}(\mathbf{x}_t - \mathbf{x}^*_{t_c}). \tag{27}$$

We can precompute K^d for all the time indices in the input motion offline and enumerate the entire sequence online to find a t_c that best satisfies Equation (27). Given t_c, We can compute δt_f and $\delta \mathbf{x}$, and compute $\delta \mu$ from either Equation (23) or Equation (24).

4.5.2 Change of final constraint

In addition, we derive the relation $V_\mu = \Psi$ from Equation (21). If we take derivative on both sides: $\delta V_\mu = \delta \Psi$, we can change the final constraint value by substituting

the desired change $\delta\Psi$ in Equation (23). In our case, because Ψ has no explicit dependence on time, $\delta\Psi$ is simply $\Delta\Psi$. Equation (25) and Equation (27) then become the following:

$$\delta t_f = K^d \delta \mathbf{x} + K^c \Delta\Psi, \tag{28}$$

$$(t - t_c)h = \frac{K^d(t_c)}{1 + K^d(t_c)\dot{\mathbf{x}}^*_{t_c}}(\mathbf{x}_t - \mathbf{x}^*_{t_c}) + \frac{K^c(t_c)}{1 + K^d(t_c)\dot{\mathbf{x}}^*_{t_c}}\Delta\Psi, \tag{29}$$

where

$$K^c = \frac{V_{t_f\mu}V_{\mu\mu}^{-1}}{V_{t_f\mu}V_{\mu\mu}^{-1}V_{\mu t_f} - V_{t_f t_f}}.$$

We again precompute K^c offline and specify $\Delta\Psi$ on the fly. A nonzero $\Delta\Psi$ changes the value of t_c, thus affects both the state and the final time. For example, when we change the desired final position of COM, the character will replan her motion as well as the completion time.

4.5.3 Contact force correction

Because the contact position in the dynamics system is prescribed for a fixed length, we cannot use the elapsed time to index $B(\mathbf{x}, t)$ when the final time changes during simulation. The reference index t_c is not a good candidate neither because it will cause discontinuity in contact when jumping back and forth in time. We need another time index t_d that tracks the current time of the dynamic system. Initially, t_d is the same as the elapsed time. When the final time changes, we warp the remaining time according to the current t_c, and advance t_d with a different ratio than the elapsed time. After every time step, we increment t_d by $\Delta = \frac{t_f - ht_d}{t_f - ht_c}$. If t_d and t_c are the same, $\Delta = 1$ and t_d advances at the same speed as elapsed time. When t_c jumps ahead or lags behind, Δ adjusts accordingly to catch up with t_c.

When t_c is different from t_d, the dynamic system used to compute control forces can be different from the dynamic system used for forward simulation. Direct application of the feedback control forces in simulation might cause inconsistent contact

situation. We circumvent this issue by using a method similar to Muico et. al. [89], which matches the results of control (i.e. $\dot{\mathbf{x}}$), rather than the control force itself. A simple quadratic programming (QP) solves this problem:

$$\min_{\hat{\mathbf{F}}} \|\dot{\mathbf{x}}(\mathbf{x}_t, \hat{\mathbf{F}}, t_d) - \dot{\mathbf{x}}(\mathbf{x}_t, \bar{\mathbf{F}}, t_c)\|^2$$
$$\text{subject to} \quad \hat{\mathbf{F}} \in \mathcal{K}, \tag{30}$$

where $\bar{\mathbf{F}} = \mathbf{F}^* + \delta\mathbf{F}$. Finally, we can use the solution $\hat{\mathbf{F}}^*$ as control to simulate $\mathbf{x}_{t+1}$, and then we update t and t_d to the next time step.

4.6 Pose reconstruction

The goal of pose reconstruction is to produce a full-body pose similar to the input motion sequence and consistent with the dynamics of the abstract model. At each time step, we formulate an optimization to solve for a new joint state that matches the linear and angular momentum produced from the simulation of the abstract model. We only solve for joint velocities and use explicit Euler to update the joint configurations as $\mathbf{q}_t = \mathbf{q}_{t-1} + h\dot{\mathbf{q}}_{t-1}$. The optimization is then simplified to a QP problem. The objective function tracks the joint velocity and the foot velocity in the warped reference motion. We need to specifically track the foot motion so that it is consistent with the contact positions prescribed in the abstract model. The optimization problem is defined as follows:

$$\min_{\dot{\mathbf{q}}_t} \; w_1\|\dot{\mathbf{q}}_t - g_1(\bar{\mathbf{Q}}, t_c, t_d)\|^2 + w_2\|\mathbf{J}_c(\mathbf{q}_t)\dot{\mathbf{q}}_t - g_2(\bar{\mathbf{Q}}, t_c, t_d)\|^2$$
$$\text{subject to} \quad \mathbf{J}(\mathbf{q}_t)\dot{\mathbf{q}}_t = \hat{\mathbf{x}}_t, \tag{31}$$

where $\mathbf{J}_c$ is the Jacobian for foot contacts, $\mathbf{J}$ is the Jacobian of linear and angular momentum, and $\hat{\mathbf{x}}_t$ denotes momenta from $\mathbf{x}_t$. g_1 and g_2 compute the desired joint velocities and foot velocities respectively by warping the reference motion. w_1 and w_2 are two scalar weights that balance between these two objectives. We will discuss the selection of them in Section 4.7.

4.6.1 Trajectory warping

Due to the change in final time, we need to warp the remaining trajectory in time based on the estimated remaining time (Figure 13). Function g_1 takes the reference motion $\bar{\mathbf{Q}}$, warps it according to t_c, then compute the warped velocity $\dot{\bar{\mathbf{q}}}'$ at t_d. It also tries to correct pose errors in the next time: $g_1 = \dot{\bar{\mathbf{q}}}'_{t_d} + \frac{1}{h}(\mathbf{q}_t - \bar{\mathbf{q}}_{t_d})$. Here we exclude the global translation and rotation degrees of freedom in g_1 because the global motion is determined by the abstract model. Likewise, g_2 computes the desired velocities for both the support foot and the swing foot.

4.6.2 Perturbation

When the character receives additional external forces $\mathbf{F}^e$ such as a push, the control policy does not respond immediately until the abstract state changes at the next time step. However, the perturbed state may not respect the contacts. To help maintain contacts and balance during perturbations, we allow adjustments in the contact forces to incorporate $\mathbf{F}^e$. We solve for both $\dot{\mathbf{q}}_t$ and $\mathbf{F}$ using Equation (32) when $\mathbf{F}^e$ is present and switch back to Equation (31) when $\mathbf{F}^e$ is removed.

$$\begin{aligned} \min_{\dot{\mathbf{q}}_t, \mathbf{F}} \; & w_1 \|\dot{\mathbf{q}}_t - g_3(\bar{\mathbf{Q}}, t_c, t_d, \mathbf{F}^e)\|^2 + w_2 \|\mathbf{J}_c(\mathbf{q}_t)\dot{\mathbf{q}}_t - g_2(\bar{\mathbf{Q}}, t_c, t_d)\|^2 \\ & + w_3 \|\mathbf{F} - \hat{\mathbf{F}}\|^2 \\ \text{subject to} \quad & \mathbf{J}(\mathbf{q}_t)\dot{\mathbf{q}}_t = S(\mathbf{x}_{t-1}, \mathbf{F}^e, \mathbf{F}). \end{aligned} \tag{32}$$

This optimization modifies Equation (31) on three counts. First, because the control force $\mathbf{F}$ is also a free variable, we express the desired momenta $\hat{\mathbf{x}}_t$ in terms of the simulation function S which integrates $\mathbf{F}$ and the push $\mathbf{F}^e$ from $\mathbf{x}_{t-1}$. Second, we add one additional term to match $\mathbf{F}$ to the control forces $\hat{\mathbf{F}}$ computed from the feedback controller. w_3 weights how much to change the control compared to the tracking objectives. Third, we synthesize the impact of the push on local body parts using function g_3. It imposes the generalized impulse induced by $\mathbf{F}^e$ in each joint

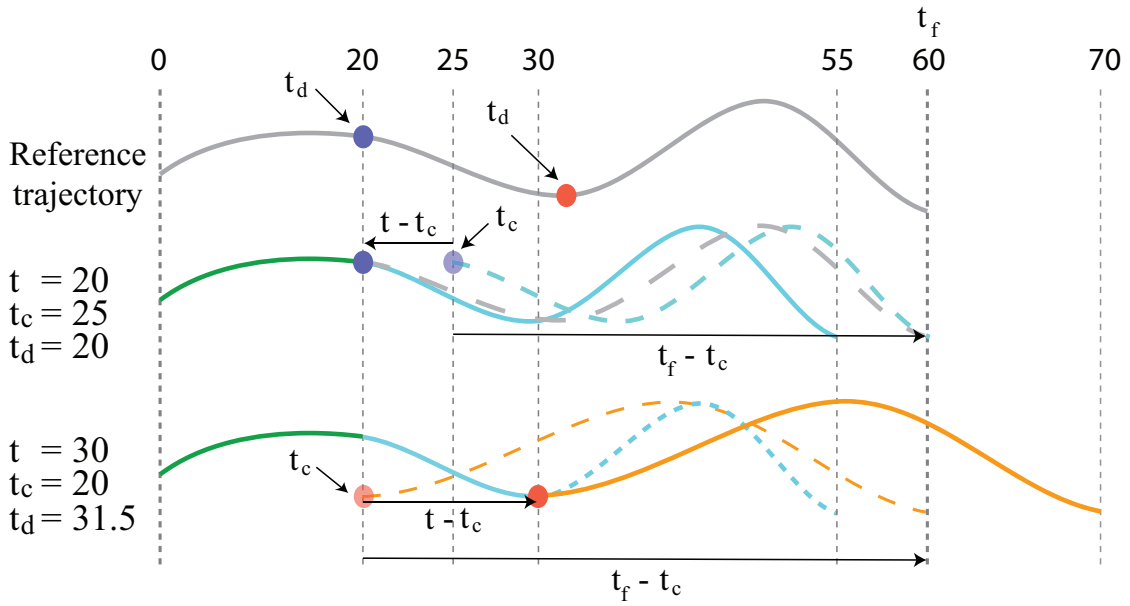


Figure 13: Our algorithm keeps track of the remaining reference trajectory and linearly warps it in time according to t_c. Initially, t and t_d both starts from zero and advance at the same rate. Second row: at frame 20, the character is pushed forward and t_c jumps forward to 25. The remaining trajectory in shortened from 40 frames to 35. Third row: 10 frames later, t reaches 30 but t_d is at about 31.5 due to warping. The character now receives a backward push that delays her for 10 frames from the reference. t_c jumps back to 20 and the remaining trajectory again is warped to 40 frames. The reference velocity is computed for the warped trajectory at t_d.

coordinate: $g_3 = \dot{\mathbf{q}}'_{t_d} - h\mathbf{M}^{-1}(\mathbf{q}_t)\mathbf{J}_e^T(\mathbf{q}_t)\mathbf{F}^e$, where $\mathbf{M}$ is the inertia matrix and $\mathbf{J}_e$ is the Jacobian of the contact point. This optimization has nonlinear constraints due to the simulation function. We solve it by formulating a sequential quadratic programming (SQP) using SNOPT [42].

4.7 Implementation details

In this section, we describe a few design choices and implementation details.

4.7.1 Concatenate Controllers

Although we can solve a motion of any length, in practice, we break down a long sequence into shorter segments, derive feedback controller for each segment, and concatenate them in simulation. A shorter sequence can be solved more easily and

efficiently in the offline optimization. In addition, to fully take advantage of our ability to change final-state on the fly, the final state of each short sequence coincides with a key event in the input motion. For example, to generate down-stair walks, we segment a normal walk at double support phase and optimize a controller for each step. During online simulation, the final COM position is lowered for each controlled segment to guide the character walk down stairs.

To create seamless transitions from controller A to controller B, we translate the reference motion and contact positions of controller B to the desired final goal of controller A. We also linearly warp the swing foot trajectory to meet the new contact points. The same procedure can also generate walks with longer or shorter steps.

Breaking down a long sequence may introduce artificial intermediate constraints and require larger forces to meet them in a short duration. Fortunately, we can remedy these problems by overlapping controllers in time and allow the control index t_c to jump across boundaries. For instance, when we overlap two consecutive controllers A and B by 20 frames, we can start to use controller B anytime during these 20 frames in online simulation. An early transition produces smoother motion by discarding the final constraint of controller A and carrying the state errors to controller B, while a later transition respects the final constraint of controller A better. Likewise, when t_c jumps beyond the range of the current controller (i.e. Equation (27) cannot be satisfied), we continue to search the optimal t_c in the neighboring controllers. For example, suppose controller B is currently in use and the perturbation causes t_c to jump to an earlier frame beyond the first frame of controller B. In this case, we use the best gain from controller A and let it take control until t_c jumps back to the range of controller B again. The overlapping period and transition timing could be adjusted for different motions.

4.7.2 Weight Objectives

Our algorithm requires tuning of only a handful of objective weights. For the offline optimization (Equation (20)), we set the weight matrix $\mathbf{W}_1$ to identity matrix, and set $\mathbf{W}_2$ and $\mathbf{W}_3$ as follows:

$$\mathbf{W}_2 = w^a \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_{3\times 3} \end{bmatrix}, \mathbf{W}_3 = w^t \begin{bmatrix} \mathbf{I}_{3\times 3} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_{3\times 3} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_{3\times 3} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}.$$

In our examples, w^a and w^t are both set to 500 for normal walk and long stepping, and they are 200 and 20 for squatting. In general, larger value of w^a produces a more robust control policy, at the expense of possible larger tracking errors. Although our experiments show that a wide range of weights produce similar results, we plan to investigate inverse optimization techniques in the future to automatically design objective functions that give rise to a given reference trajectory.

The two online optimizations (Equation (31) and Equation (32)) have only three weights in total. We use $w_1 = 1, w_2 = 5,$ and $w_3 = 0.01$ in all the examples. With w_1 and w_2 fixed, w_3 controls how much to alter the optimal control force in order to satisfy the contacts and tracking. Larger value of w_3 makes the motion more compliant to the push, but also more difficult to recover.

4.7.3 Correct Numerical Drift

Our simulation of the abstract model is physically correct up to the second-order integration error. However, matching both the COM position and momenta in the full-body pose creates an infeasible optimization problem because we solve for only the velocity and use explicit Euler to compute configuration. In other words, when a full-body state has the same linear momentum as the abstract state, it could still produce a different COM position at the next time step. We prevent the accumulation

of this numerical error by feeding back the full-body COM position to the abstract model so that the feedback control will try to correct it at every time step.

4.8 Results

We demonstrate the robustness of our algorithm by building controllers for a variety of input motions. We test the feedback control policies by applying arbitrary external forces to the character, and by altering the physical properties of the environment, such as the terrain geometry and surface friction.

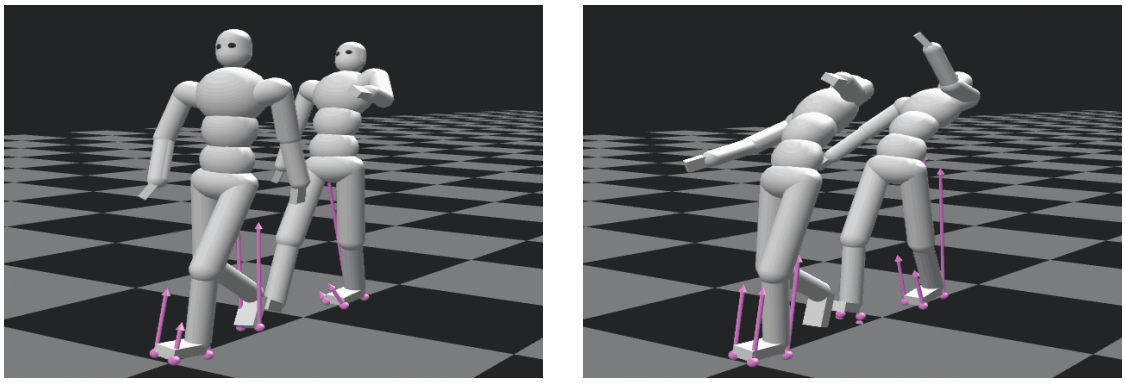
4.8.1 Performance

We test our algorithm on a 2.8GHz Intel Core 2 Duo processor. We use motions captured at 120 Hz as input and use the same frequency for simulation. The offline optimization usually converges within 10 iterations. The actual computation time depends on the length of the motion. It takes about a minute for 60 frames of animation. For online simulation, we use a character model with 42 degrees of freedom, and the simulation runs at 20 frames per second on average.

4.8.2 Change of final time

A change in completion time happens almost every time a perturbation is encountered. An obvious case is when a character receives large pushes that disrupt her motion. For example, in a normal walk, a large backward push slows down a step by 10 frames while a small forward push accelerates the step by 2-3 frames. When the character receives multiple pushes, she is able to adjust her pace repeatedly on the go. In another example when the character walks upstairs of $0.2m$ height, the final time is lengthened by 4 frames, and it is shortened by 4 frames for walking down. Similarly, it takes 4 frames longer for a $0.05m$ larger step, and 17 frames faster for a shorter step. We observe similar results of timing adjustments on other motions.

A flexible plan for completion time generates more natural and robust motion.



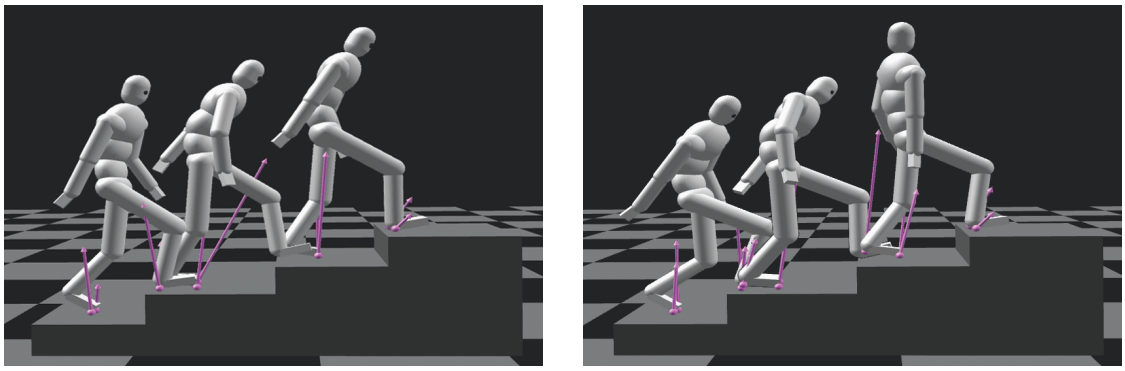
(a) With flexible final time (b) With fixed final time

Figure 14: The controller with flexible completion time produces more stable motion than the controller with fixed completion time after a large backward push.

We compare our controller to a fixed-time controller on a walking motion. In the case of small pushes, our controller always produces more stable motion with smaller contact forces. For larger pushes, the character in our motion adjusts her walking speeds to recover and is finally able to complete the step, while the fixed-final-time motion failed to recover the walk (Figure 14).

4.8.3 Change of final constraint

The ability to re-plan final constraint on-the-fly makes it easy for our controller to adapt to new environment and generate a larger variety of motions from a single reference. In the first experiment, we derive an optimal feedback controller for a normal walk on flat terrain and successfully apply it to walking on stairs with different step height ranging from $+0.3m$ to $-0.2m$. For walking upstairs, we change the final goal at the beginning of double support, and the character can raise her COM by as much as $0.3m$ during the double support phase. Walking down stairs is a more challenging task for our controller. The character has to twist her torso to reach the new contact points and to compensate for the angular momentum of the lower body. By simply changing the final state at the start of each step, the same controller can produce walking downstairs up to $0.2m$ per step.



(a) With flexible final constraint (b) With fixed final constraint

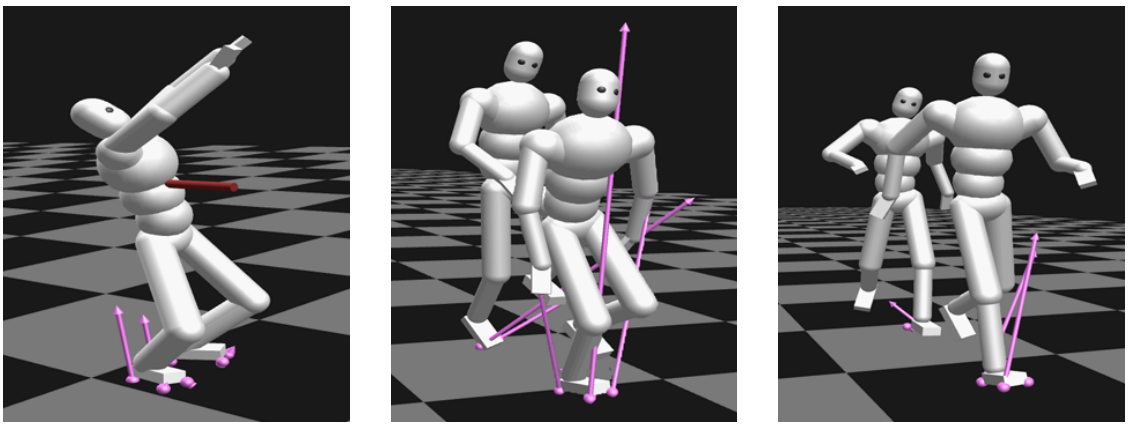
Figure 15: The controller with anticipation of changes in the final constraint produces more natural motion than the controller driven by errors in the final constraint for walking upstairs of $0.2m$.

We compare our results with a control policy that does not change the final constraint ahead of time. We first add an linear offset to the reference trajectory such that the final state of the trajectory meets the desired height. The control forces in this case are driven by the deviation between the current state and the modified reference trajectory, rather than the anticipation of the change in the final state. The character is able to walk down stairs with maximum step height of $0.1m$ and up stairs with maximum step height of $0.2m$, but the motions are visually unnatural in that the COM is always lagging behind the reference (Figure 15). Further, larger contact forces are used compared to our results.

4.8.4 Generality

Our algorithm is generic to different types of input motion. Besides a straight walking sequence, we also apply the algorithm to a long stepping (Figure 10), a squat exercise (Figure 16(a)), a hop (Figure 16(b)), and a run (Figure 16(c)). For each case, we apply random pushes to the character and observe dynamic responses and adjustments of final time. For example, when pushed backward, the long stepping takes 4 frames longer to complete and 4 frames less for a forward push. We also repeatedly push the character while she is performing a squat exercise. The character is able to balance

by continuously adjust her whole body movements and the final time.



(a) Squatting (b) Hopping (c) Running

Figure 16: Our algorithm is robust across a variety of activities.

4.8.5 Robustness

We examine the robustness of our controller by supplying pushes of different magnitudes, directions and durations. Our controller performs more robustly to pushes that do not require large change of steps. In all the examples, the character can recover from impulsive pushes (lasting less than 0.3 second) up to $200N$ in all directions. The controller is also robust against sustained pushes lasting for one second with magnitude up to $40N$.

We also compare our controller to one that tracks the reference motion without minimization of angular momentum. For reference motions with small angular momentum, both controllers perform similarly. For more dynamically challenging motions such as the long stepping, our controller exhibits better stability to small perturbations, and it can recover from extreme cases when the tracking-only controller fails. In all the tests, our controller always uses less control forces than the tracking-only controller.

4.9 Discussions

Abstraction of the dynamic model greatly facilitates solving the optimization problem, but it also ignores the biomechanical constraints of the skeleton. Consequently, the feedback control may generate kinematically impossible COM motions. Incorporating inequality constraints on the abstract states to ensure a reasonable COM position could help alleviate this problem.

The abstract dynamic model described in this chapter intelligently utilizes external contact forces to control the character's motion, thus allows for robust control algorithms. However, its dependency on prescribed contact positions largely limits the its flexibility. Because the contact points are not part of the dynamic states, we cannot model the change of contacts using controls. Consequently, we can only model static frictional contact but not sliding or rolling contacts. Moreover, our controller is not able to produce different step taking behaviors from the reference, nor can it handle motions with sporadic contacts such as sparring. In the future, we want to incorporate long-term contact planning ([128]) as a separate routine in our algorithm. With the ability to plan contacts for future events, we will be able to, for example, produce ballistic motions that prepares for a safe landing when perturbed.

In the next chapter, we will explore sampling techniques for planning contacts in complex dexterous manipulation tasks. Similar to the abstract model, the objects being manipulated completely rely on external contact forces (mostly from the hands) to propel themselves. This dynamic constraint is leveraged to generate contact points.

CHAPTER V

OBJECT MANIPULATION SYNTHESIS

Capturing human activities that involve both gross full-body motion and detailed hand manipulation of objects is challenging for standard motion capture systems. In this chapter, we introduce a new method for creating natural scenes with such human activities. The input to our approach includes full-body motion and object motion acquired simultaneously by a standard motion capture system, and our method automatically synthesizes detailed, expressive, and physically plausible hand motion (Figure 17). Instead of producing one "optimal" solution, our method presents the user a set of hand motions that exploit a wide variety of manipulation strategies and seamlessly integrate with full-body and object motion. Our results highlight complex manipulation strategies human hands employ effortlessly and unconsciously, such as static, sliding, rolling contact, as well as discrete relocation of contact points and finger gaiting.

5.1 Motivations

Full-body human motion synthesis that contains detailed hand-object manipulation is a very challenging problem in computer animation. The perception of realism not only depends on the motion on a grand scale, but also small variations in the hand movement as it interacts with its environment. As a recent study by Joerg et al. [61] shows, even very subtle desynchronization errors in hand and body motions can be detected by the human eye. As a result, the level of accuracy required for generating believable body-and-hand motion sequences raises significant challenges for existing methods of motion tracking. Existing optical motion capture systems are unsuitable

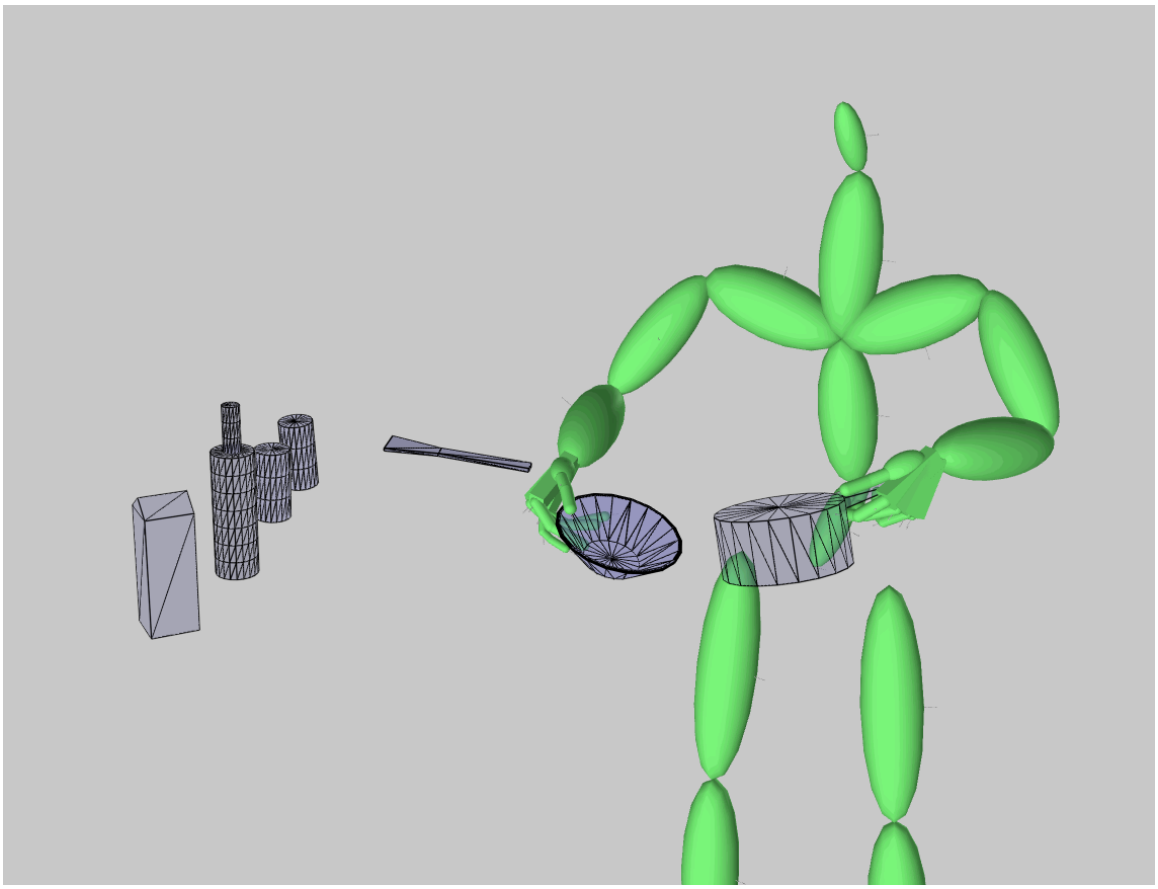


Figure 17: Use motion capture to record both the gross motion of the body and the intricate motion of the fingers in a cooking scene is very difficult due to the close interactions between the hand and the objects. Our algorithm can automatically fill in finger movements that are consistent with the scene.

at simultaneous tracking of full-body motion and detailed hand motion. The resolution and camera placement suitable for full-body tracking leads to limited precision and occlusion when focused on the hand. The most popular method in film industry for multi-resolution tracking of body and hand is to capture the full-body motion of the actors and manually animate the hand motion. This process usually takes enormous effort and highly depends on the animator's skill. Alternatively, one can capture the same scene in multiple takes and synchronize them as a post-process[82]. Performing the same scene multiple times can be difficult especially when the scene involves incidental contacts with the objects. Another option, is to use customized devices, such as data gloves, to capture hand motion separately. Unfortunately, even top of the line data gloves [1] do not provide the sufficient accuracy for hand motions with complex physical contact. Even if we can capture accurate hand motions, the exact data is often times undesirable in a production pipeline because the objects will be edited in post capturing steps.

This chapter introduces a new method for synthesizing human activities with both gross body motion and fine manipulation. Our system takes as input full-body motion and object motion simultaneously acquired by standard mocap system, and synthesizes a set of detailed, expressive, and physically plausible hand motions that seamlessly integrates with the full-body motion and object trajectory. We assume that the mocap system has sufficient resolution to capture accurate wrist motion. This assumption is reasonable for most standard mocap systems, and yet it is crucial to the success of our algorithm, because the palm and finger movement is highly constrained by the wrist configuration. Having a known wrist trajectory makes the problem more tractable. Still, given the wrist motion, there are numerous ways for the hands to achieve the desired manipulation of objects. While continuous optimization methods may like a very good fit for generating hand motions under these conditions, in practice they have not been able to recreate the level of complexity and diversity human

hand motion exhibits effortlessly. In contrast to synthetic hand motion generated by optimization techniques or rule-based procedures, human dexterous manipulation tends to utilize different contact modes, such as static, sliding, rolling contact, as well as discretely add or remove contact points. These distinctive manipulation strategies raise numerous issues for conventional continuous optimization methods because the space of contact position is highly discontinuous, and subject to nonconvex physical constraints. Further, the design of an appropriate objective function for the desired outcome remains a difficult challenge as the criteria for optimality is not obvious.

We develop a different approach for generating hand motions under constraints imposed by the full-body and object trajectory. Instead of continuous optimization over joint trajectories, we develop a discrete randomized search algorithm that explores the space of possible hand-object contact positions over time. At each time step, the algorithm stochastically chooses a set of contact points on the objects and determines whether this new set can be achieved kinematically and dynamically from the current state of the hand and the object. Because the goal of our system is to quickly generate as many complete sequences as possible while presenting a rich diversity in motion, we utilize a randomized depth first search strategy to explore the trajectory space.

The key choice we made in designing our algorithm was deciding to work in the object-contact position space instead of the joint angle space. We did so for two primary reasons. First, reconstruction of the hand pose is straightforward once the contact positions and input wrist configuration are determined. Secondly and more importantly, working in this space allows us to generate plausible sequences of contact positions very efficiently because the movement of contact points are highly constrained by the contact forces. A naive sampling approach would propose contact points randomly and reject infeasible candidates later. Unfortunately, testing feasibility involves solving dynamic equations and inverse kinematics (IK), making

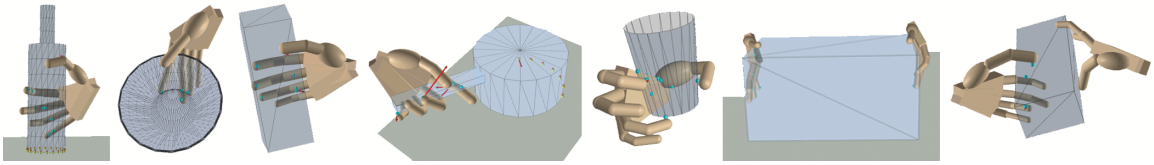


Figure 18: Our algorithm synthesizes detailed finger movements for a wide variety of objects. In the images, yellow dots are contact points between the hand and the object. Red arrows indicate contact forces applied to the hand from the object.

the importance of generating likely feasible candidates crucial for efficient operation. Using the current contact force as a precondition, we can constrain our search space to samples dynamically consistent with the current contact force. For example, if the contact force is on the boundary of the friction cone, the possible contact positions at the next frame will lie along the opposite direction of current tangential contact force. Empirically, the candidates proposed by this algorithm exhibit great variety in the motion. This is because the proposed candidates are representatives of difference contact modes. Selecting candidates from this diverse pool favors motion with frequent contact mode switching.

Our algorithm can be applied to most everyday objects and mundane manipulation scenarios to synthesize intricate finger movements without any specialized knowledge about the manipulation tasks (Figure 18). Our results demonstrate that a rich set of manipulation strategies emerge when hands frequently employ different contact modes. We show that continuous sliding and rolling contact, as well as discrete relocation of contact points, can greatly improve the believability and aesthetics of human motion. Our algorithm is also able to discover sophisticated finger gaiting strategy without any prior knowledge or assumption in the search algorithm.

5.2 Overview

The input to our system is a mocap sequence of a human performing full-body motions while physically interacting with objects in the environment. The sequence is acquired

using a mocap system calibrated for wide-range full-body motions. The resolution of the system is sufficient to capture the wrist and the object movements, but not enough for fine finger movements. Our goal is to create realistic, detailed hand motions to fill the missing gap between the full-body and the manipulated object.

Our approach is illustrated in Algorithm 1. We first search for many sequences of feasible contact point trajectories within the computation time budget (Section 5.3), then reconstruct the hand motion from each of the trajectories (Section 5.5).

We formulate the problem of searching for contact point trajectories as a randomized depth-first tree traversal. Before we introduce our search algorithm, we first define some notations as follows (Table 1). A level of the tree represents a time instance t. A node of the tree indicates a state of the manipulation problem at one time instance. A state $\mathbf{s}$ is defined by a hand pose $\mathbf{q}$, a set of contact points $\mathbf{P}$ between the hand and the object, and the corresponding contact forces $\mathbf{F}$. A hand pose $\mathbf{q}$ contains the global translation and rotation of the wrist obtained from motion capture, and the unknown joint angles of the fingers. A *contact point* $\mathbf{p}$ is defined as a pair of local coordinates on the surface of the object $\mathbf{p}.\mathbf{o}$, and on the surface of a finger $\mathbf{p}.\mathbf{h}$. A *contact force* $\mathbf{f}$ is modeled by a nonnegative scalar force f^n along the contact normal $\mathbf{n}$, and a tangential force vector $\hat{\mathbf{f}}$. The contact normal $\mathbf{n}$ is the opposite direction of the surface normal at the contact point. The corresponding Cartesian force is computed as $\mathbf{f}_C = f^n\mathbf{n} + \hat{\mathbf{f}}$.

A full state can be compactly represented as a *guiding contact configuration* $\mathbf{c}$. $\mathbf{c}$ is a small set of contact points $\mathbf{p}$, each of which, called *a guiding point*, resides on a distinctive finger. $\mathbf{c}$ contains the minimum information required to recover a full state $\mathbf{s}$. From the contact points in $\mathbf{c}$, we apply IK and collision detection to obtain $\mathbf{q}$ and $\mathbf{P}$. From $\mathbf{P}$ and the motion of the object, we can solve for the corresponding contact forces $\mathbf{F}$.

Using $\mathbf{c}$ to represent a state greatly simplifies the search process illustrated in

Table 1: Definition of symbols used in this chapter

Symbol	Definition
$\mathbf{q}$	a hand pose
$\mathbf{p.h}$	local coordinates of a point on a finger in a contact point pair
$\mathbf{p.o}$	local coordinates on the surface of the object
$\mathbf{p} = \{\mathbf{p.o}, \mathbf{p.h}\}$	a contact point pair
$\mathbf{c} = \{\mathbf{p}_i\}$	a guiding contact configuration
$\mathbf{f} = \{f^n, \hat{\mathbf{f}}\}$	contact force applied to the object
$\mathbf{n}$	normal direction for the contact force, opposite to the object surface normal
$\mathbf{f}_C = f^n\mathbf{n} + \hat{\mathbf{f}}$	contact force in Cartesian space
$\mathbf{P}$	a set of contact points between the hand and the object in one time instance
$\mathbf{F}$	a set of contact forces applied to the object in one time instance
$\mathbf{s} = \{\mathbf{q}, \mathbf{P}, \mathbf{F}\}$	a state of the manipulation problem

Algorithm 2. The core of the algorithm is to explore contact configurations that are more likely to recover a feasible state. Given a feasible state $\mathbf{s}^{(t-1)}$ at the previous level, we explore a small set of new nodes for the current level t using information from $\mathbf{P}^{(t-1)}$ and $\mathbf{F}^{(t-1)}$ in $\mathbf{s}^{(t-1)}$ (Section 5.3.1). Among the new nodes, we randomly select one of them to recover its full state $\mathbf{s}$. If $\mathbf{s}$ is kinematically and dynamically feasible (Section 5.3.2), we move on to the next level. Otherwise, we consider this node infeasible. When a feasible path is found, or when an infeasible node is encountered, we backtrack to explore more solutions. In Section 5.4, we describe a few strategies that can efficiently discover distinctive paths.

Algorithm 1: SynthesizeHandManipulation

```
𝕊 = {} ;
while isTimeLimitReached =FALSE do
    S^(0) = {} ;
    SearchContactPoints(𝕊, S^(0), 1) ;                    // Section 5.3
foreach S^(T) ∈ 𝕊 do
    q_h ← ReconstructHandMotion(S^(T)) ;                  // Section 5.5
```

Algorithm 2: SearchContactPoints ($\mathbb{S}, \mathbf{S}^{(t-1)}, t$)

```
C^(t) ← GenerateNewNodes(S^(t−1)) ;                          // Section 5.3.1
success ← FALSE ;
nTrials ← 0 ;
while nTrials < maxBranchFactor do
    c^(t) ← PickOneNode(C^(t)) ;
    isFeasible, s^(t) ← TestFeasibility(c^(t)) ;             // Section 5.3.2
    if isFeasible = TRUE then
        S^(t) ← Append(S^(t−1), s^(t)) ;
        if t=T then
            S.push(S^(T)) ;
            success ← TRUE ;
        else
            success ← success|SearchContactPoints(S, S^(t), t + 1) ;
    nTrials ← nTrials + 1 ;
return success;
```

5.3 Search for Contact Point Trajectories

This section describes Algorithm 2 in detail.

5.3.1 Generate New Nodes

This section describes how we generate the contact configurations **c** at each time instance by sampling a set of guiding points for each finger.

5.3.1.1 Initialization

To begin the search, we need to first determine when and where a finger starts to come into contact with the object. We utilize the captured motion of the wrist and the object to estimate the timing and a set of sample contact points for each finger in preprocessing. Range of motion (ROM) of a point on a finger is determined by the wrist motion and joint limits of the finger. Intersection of the ROM volume and the geometry of the object indicates surface patches on the object that the finger point can reach. We choose the belly of the distal phalanx as **p**.**h** to estimate a ROM volume, then compute interactions between the ROM and the object to

discover contact windows when the two overlap sufficiently. For the first frame of each window, we create uniform samples on the surface patches as $\mathbf{p}.\mathbf{o}$, and combine them with $\mathbf{p}.\mathbf{h}$ as the initial contact points for a finger. Guiding points are drawn from this pool during the search process when a finger initiates contact with the object. When a finger can no longer reach the object, we do not consider it in future contact configurations. We will show later in this section that the precise timing of an initial contact is not important. A finger can move around to find the most suitable contact location and apply forces at the right time.

We can further reduce the number of candidate contact points using the kinematics test described in the next subsection. The test ensures the contact point pair can be met without penetration by solving an IK problem. We thus keep only the contact points that pass the test. We decide to initiate contact points from the distal phalanx because it can sufficiently determine a finger pose during IK. However, the result of the test, as we will see later, may provide us with different finger contacts. Therefore, our grasps are not limited to the finger tips.

5.3.1.2 Recursion

In the recursive case, we generate a set of contact configurations at level t from a feasible state $\mathbf{s}^{(t-1)}$ at level $t-1$. From $\mathbf{P}^{(t-1)}$ in $\mathbf{s}^{(t-1)}$, we choose one contact point $\mathbf{p}'$ for each finger that is in contact as the seed for sampling. The chosen contact point is the furthest to the palm so that it constrains more degrees of freedom for the finger.

For each contact point $\mathbf{p}'$, we fix $\mathbf{p}.\mathbf{h}$ and move $\mathbf{p}.\mathbf{o}$ on the object surface to create new contact points. The corresponding contact force $\mathbf{f}$ for $\mathbf{p}'$ determines where $\mathbf{p}.\mathbf{o}$ should be in the next time instance according to the following constraints. When the force is greater than zero, the magnitude of the friction force determines whether the

contact point will stay or slide in the friction direction [1]. The contact point can slide only if $\|\hat{\mathbf{f}}\| = \mu f^n$, where μ is the friction coefficient. When the force is zero, the finger can release contact in the next frame. Therefore, the contact point can move in any direction and serves only as a guidance to position the finger. These conditions are listed as follows.

$$\mathbf{p}.\mathbf{o}^{(t)} = \begin{cases} \mathbf{p}.\mathbf{o}^{(t-1)}, & f^n > 0, \|\hat{\mathbf{f}}\| < \mu f^n \\ \mathbf{p}.\mathbf{o}^{(t-1)} + U(0, l_1)\mathbf{d}, & f^n > 0, \|\hat{\mathbf{f}}\| = \mu f^n \\ \mathbf{p}.\mathbf{o}^{(t-1)} + \mathbf{r}(U(0, 2\pi), U(0, l_2)), & f^n = 0 \end{cases} \tag{33}$$

In Equation 33, $U(a, b)$ denotes a random number generator with uniform distribution in $[a, b]$. $\mathbf{d}$ is the friction direction in the object coordinates, and $\mathbf{r}$ generates a random point within a circle of given radius. l_1 and l_2 control how much a finger can move on the object surface within one time step. We project the directions on the object surface to make sure the resultant $\mathbf{p}.\mathbf{o}^{(t)}$ stays on the object. In the rare case where a finger cannot reach its guiding point at $t-1$, we treat it the same as $f^n = 0$, except the $\mathbf{p}.\mathbf{o}^{(t-1)}$ is from the guiding point in $\mathbf{c}^{(t-1)}$.

Using the above sampling scheme, we create a few new contact points for each finger in the next frame. Figure 19 shows examples of contact points evolving over time on the object. The new contact points serve as guiding points in the new contact configurations $\mathbf{c}^{(t)}$. The new contact configurations lead to interesting finger gaits by allowing the finger to stick or slide on the object, and to release and re-establish contacts, while respecting the physical constraints. Note that although we do not change $\mathbf{p}.\mathbf{h}$ in the samples, it can still change over time when $\mathbf{p}'$ is different from the guiding point $\mathbf{p}$ in $\mathbf{c}^{(t-1)}$.

[1] From the object's point of view, the contact point moves in the opposite direction of friction.

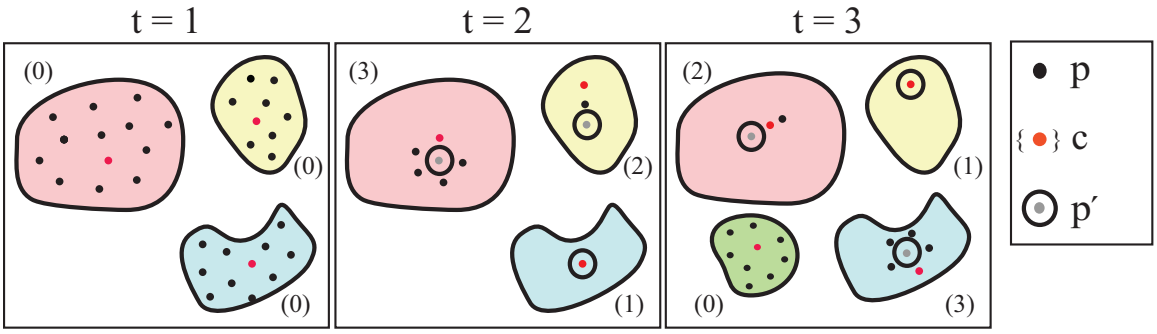


Figure 19: Contact points change over time based on the sampling strategies.

5.3.2 Test feasibility

We consider a contact configuration $\mathbf{c}$ feasible if its corresponding state $\mathbf{s}$ satisfies two feasibility conditions. $\mathbf{c}$ is kinematically feasible if we can solve for a penetration-free hand pose $\mathbf{q}$ and a set of contact points using IK and collision detection. $\mathbf{c}$ is dynamically feasible if we can solve for the corresponding contact forces $\mathbf{F}$ for $\mathbf{P}$ to generate the captured motion of the object.

5.3.2.1 Kinematics

To reconstruct a penetration-free hand pose from $\mathbf{c}^{(t)}$, we iterate between solving an IK problem and resolving hand-object penetrations. $\mathbf{c}$ fails the test if we cannot resolve penetration completely within the iteration limit, or if the fingers penetrate one another.

We use a hand model with 36 DOFs (Figure 20). The six DOFs on the wrist are given as input, and we need to solve for the other 30 DOFs. We approximate joint limits from a motion capture sequence of a ROM exercise. Since our hand model has no interdependencies among fingers, we can solve for the joint angles of each finger separately.

For each finger, we formulate a nonconvex optimization to satisfy the designated guiding contact point pair.

$$\min_{\mathbf{q}} \|f(\mathbf{q}, \mathbf{p}.\mathbf{h}) - \mathbf{p}.\mathbf{o}\|^2 \tag{34}$$

$$\text{subject to} \quad g(\mathbf{q}, \mathbf{p}.\mathbf{h})^T \mathbf{n}(\mathbf{p}.\mathbf{o}) \geq 0 \tag{35}$$

$$|\mathbf{q} - \mathbf{q}^{(t-1)}| \leq \delta\mathbf{q} \tag{36}$$

$f(\mathbf{q}, \mathbf{p}.\mathbf{h})$ in Equation 34 is the IK function that outputs the position of $\mathbf{p}.\mathbf{h}$ in the local coordinates of the object under pose $\mathbf{q}$. In Equation 35, $g(\mathbf{q}, \mathbf{p}.\mathbf{h})$ outputs the direction of the back of the phalanx that contains $\mathbf{p}.\mathbf{h}$ in the local coordinates of the object. By constraining the back of a phalanx to face the positive hemisphere of the surface normal $\mathbf{n}$ at $\mathbf{p}.\mathbf{o}$, we prevent configurations that are usually considered unnatural. Lastly, Equation 36 prevents large change across frames to favor smooth motions. $\delta\mathbf{q}$ determines how fast the fingers can move in a time step.

After solving for a finger pose, we need to resolve penetrations between the finger and the object. The contact pair with the largest penetration depth is chosen as the new target in the objective function, then we solve the IK problem again. In most cases, a few iterations of IK are sufficient to resolve penetrations completely and to satisfy the guiding point pair well, thanks to the incremental movements of both the hand and the contact points. However, there are two exceptions. One is when a contact point moves across a discrete feature of the object surface, such as an edge or a corner. The other is when a contact point moves out of reach for the finger. Both cases may result in a contact point pair different from the intended one or loss of contact as a result of penetration resolution. While a different contact pair corresponds to finger rolling, release of a contact is acceptable only when the finger applies no force in the previous frame. Finally, we obtain a penetration-free hand pose $\mathbf{q}$ from IK, and a set of contact points $\mathbf{P}$ from collision detection.

5.3.2.2 Dynamics

Given the set of contact points $\mathbf{P}$, we test whether they are dynamically feasible for the object by solving for contact forces $\mathbf{F}$ to satisfy the captured motion. We also solve for forces from the environment if contact points $\mathbf{P}_e$ are detected between the object and the environment.

The dynamics problem can be formulated as a convex conic programming. An infeasible problem implies the contact points are not feasible for the object.

$$\min_{\mathbf{F}} \sum f^n \tag{37}$$

$$\text{subject to} \quad \sum_i \mathbf{J}_i^T (f_i^n \mathbf{n}_i + \hat{\mathbf{f}}_i) = \mathbf{G} \tag{38}$$

$$\mathbf{n}_i^T \hat{\mathbf{f}}_i = 0, \forall i \tag{39}$$

$$\|\hat{\mathbf{f}}_i\| \leq \mu f_i^n, f_i^n \geq 0, \forall i \tag{40}$$

Equation (38) is the equation of motion for the object, where $\mathbf{J}$ is the Jacobian that transforms a Cartesian force to the generalized coordinates, and $\mathbf{G}$ is the generalized force of the object computed from inverse dynamics. $\mathbf{G}$ takes into account both linear and angular motions as well as gravity. To ensure every $\mathbf{f}$ is a valid contact force, we constrain the friction direction to be perpendicular to the contact normal (Equation 39), and be within Coulomb's friction cone (Equation 40). For every sliding contact in $\mathbf{P}_e$, an additional constraint $\hat{\mathbf{f}} = \mu f^n \mathbf{d}$ is applied, where $\mathbf{d}$ is the detected sliding direction. Finally, to prevent the hand from exerting excessive contact forces, we minimize the normal forces applied by the hand (Equation 37).

Complex manipulation tasks that rely on torsions are difficult to generate from the simplified point contact model. For a better approximation of contact phenomena, we apply forces on a few proxy points in the neighborhood of a contact point. If a contact point lies on a flat surface, we use two neighboring proxy points. If a

contact point lies near a sharp feature, we use three proxy points to capture the local change of geometry. Features are detected by comparing surface normals in the neighborhood. Contact forces computed at proxy points are later aggregated to the originated contact point. In this way, contact points near sharp features have a wider range of contact forces. For example, grasping on the edges of a box can generate a wider variety of motions than grasping on the faces. This phenomenon is consistent with our daily experience.

5.3.3 Expand the search tree

When a state passes both feasibility tests, we move on to search for feasible states in subsequent levels until we reach the bottom one, in which case a feasible trajectory is successfully discovered. If a state fails the feasibility tests, or after we finish with a feasible path, we backtrack to previous levels and explore new trajectories. In this way, the problem of searching for feasible contact trajectories is casted as a tree traversal problem in a straightforward depth-first manner.

5.4 Control Solution Diversities

The baseline search algorithm is inefficient because it does not exploit the spatial and temporal coherence in the solution space. A brute-force search wastes a lot of computation in similar paths. In this section, we introduce four strategies that can discover a diverse set of feasible trajectories more efficiently. They also allow users to control the styles of the solutions and the trade off between path diversity and success rate.

5.4.1 Sparse exploration

Sibling nodes often represent similar states. Skipping some sibling nodes can be an effective way to reduce computation on visually similar paths. To this end, we reduce the branching factor by making a stochastic decision whether to explore a sibling node

or backtrack to the parent node. Specifically, if a feasible trajectory is found, there is ϵ_1 chance the search algorithm will backtrack to the previous level. If a trajectory fails, the chance to backtrack is ϵ_2. We always set $\epsilon_1 > \epsilon_2$ to bias finding a feasible path first.

Similarly, we can exploit the temporal coherence in a path to reduce branching frequency. Due to the short time duration between two consecutive nodes in a path, their corresponding states are usually indistinguishable, no matter what actions they take (e.g. one slides $0.2mm$ and the other slides $0.5mm$). For a contact phenomenon, such as sliding or moving, to be visually noticeable, the same action must be taken by a few consecutive nodes. Therefore, if a finger has a static guiding contact in a frame, our algorithm will prefer the same finger remains static for a few frames. Likewise, when a finger takes a sliding action, the algorithm will let it slide or move in the same direction for a few frames. During node expansion, we use ϵ_3 to control the probability of taking a different action from the parents action. Then during backtracking, we only start new branches at nodes that take different actions. By enforcing the same action over a period of time, we limit the branching frequency and explore only paths with noticeable differences. In this way, the time complexity is exponential to the number of decisions allowed along a trajectory rather than the number of frames, which is controlled by ϵ_3.

Users can control the sparsity of exploration by choosing the appropriate ϵ_1, ϵ_2, and ϵ_3 for a problem. In all our examples, we use $\epsilon_1 = 1/2$, and $\epsilon_2 = (T - t)/(20T)$, and set ϵ_3 to be approximately $10/T$.

5.4.2 Node prioritization

In addition to allocating more computation in dissimilar paths, we can explicitly control the style in a path by prioritizing contact points at each level. For example, we can sort contact point samples by how far they are from their parents. Instead

of picking a random contact point for each finger to form a node, we can sort the contact points by the amount of movements they represent. By always choosing the most static points, we obtain a trajectory with mostly static contacts. Likewise, we can always choose the contact points with most movements to obtain a trajectory with lots of contact changes. More interesting behavior may emerge if we assign different preferences to different portions of a motion.

5.4.3 Informed backtracking

In backtracking, our goal is to explore a dissimilar path after successfully finding a solution, or to correct the current failure in an earlier time. Propagating the reason in backtracking can help us make better decisions and increase the success rate. For example, when we arrive at an infeasible node, we can utilize the cause of failure to start a new path that is more likely to succeed from an ancestor node. If a path returns with IK failure on a finger, we choose a new guiding point only for the failing finger without changing other successful ones. The new guiding point is then chosen to be the furthest point from previous failures. Likewise, if a path reports interpenetration between two fingers, we choose new guiding points that are far apart for them. In addition, if a finger fails in the first frame of its contact window, we increase the branching factor to allow more trials in finding a good initial contact point. If a path reports failure in the dynamics test, our current algorithm simply choose a new guiding point for each finger. We believe a thorough analysis of the dynamics equation could help derive a better strategy.

Another scenario is when we successfully solve for a feasible path and backtrack to explore alternatives. To choose the most different nodes in backtracking, we can record all explored nodes as we go, and select the most distinctive ones from the remaining pool. Because we reduce the branching factor and branching frequency, the storage overhead is not significant.

5.4.4 Biased force optimization

The possible branches at each level are determined by contact forces. We can explore the redundancy in contact forces to generate different grasping styles. Equation 37 reflects our preference in distributing contact froces to fingers. A uniform weighting scheme encourages even distribution of forces among contact points to favor a stable grasp. We can generate different grasping styles by changing the weighting scheme. For example, we can assign larger weights to fingers that apply no force in the previous frame. As a result, once a finger is released, it will re-establish contact again only when it arrives at an indispensable location to apply contact force. Another possible strategy is to increase the weights for a finger in proportion to how long it has been in contact. The optimal solution is then to alternate forces among fingers as if they are restless.

5.5 Reconstruct Hand Motion

Once we solve for feasible trajectories of contact points, we can reconstruct the corresponding finger motions. Although we also obtain hand poses from the kinematics test, they are usually noisy due to the randomized nature of our search algorithm. We take two post-processing steps to produce a smooth hand motion. First, we solve a spacetime constrain optimization problem to smoothen the hand poses. The resultant motion, however, may introduce new penetrations when a finger transitions between releasing and re-establishing contacts with the object, or when the hand transitions between handling two objects in the same sequence. In such cases, we resolve the additional penetrations using an iterative algorithm.

5.5.1 Smoothing

We solve for a smooth and natural hand motion that respects the previously solved contact points using a spacetime optimization.

$$\min_{\mathbf{q}_h} \sum_{t=1}^{T} E_1 + wE_2$$

$$\text{subject to} \quad f(\mathbf{q}_h^{(t)}, \mathbf{p}.\mathbf{h}^{(t)}) = \mathbf{p}.\mathbf{o}^{(t)}, \qquad t = 1, \cdots, T \tag{41}$$

E_1 and E_2 are smoothness and naturalness metrics respectively. E_2 favors a natural pose in which bending is shared among all joints on a finger. This problem is solved efficiently by starting with the hand poses from kinematics test so that constraints are met initially. Because we apply the IK constraints only to fingers that exert contact forces, fingers without constraints can move freely and possibly penetrate the object as a result.

5.5.2 Transition

From the smooth hand motion, we detect transition windows that contain penetrations, then resolve them smoothly using an iterative method. In each iteration, we first apply the kinematics test as in Section 5.3.2 to resolve all problematic frames. The results are penetration-free but noisy. We then smooth the joint angles within each transition window. During smoothing, the poses right before and after the transition window are fixed as boundary constraints. Poses in the in-between frames are bounded by a small range around their current values. The resultant motion may still penetrate the object, but the penetrations are less severe and the motion is smooth. By alternating between IK and smoothing with gradually shrinking bounds, we will converge to a smooth and penetration-free motion. This simple strategy works quite well for our problems. For more complicated scenarios, we may need to employ advanced pre-grasp planning algorithms.

Finally, we attach the resultant motions to the wrists of the character, and we complete the reconstruction of a wide-range, detailed scene with human locomotion and manipulation of objects.

5.6 Results

We apply our algorithm to a variety of hand manipulation tasks, ranging from simple lifting and turning of a box on a tabletop to a realistic cooking scene that involves objects of different shapes as well as two-hand manipulation tasks. Our algorithm automatically generates many possible hand motions with rich variation of details. In addition, the user can modify the object properties, such as geometry, material, or motion, after data acquisition process.

We use a hand model with 36 DOFs (Figure 20). The six DOFs on the wrist are given as input and the remaining 30 DOFs on the palm and fingers are synthesized by our algorithm. We solve the nonconvex IK optimization using SNOPT [42], and the convex conic programming problem using MOSEK [11]. We use Bullet [28] for collision detection.

5.6.1 Performance

We test our algorithm on a 2.8GHz Intel Core 2 Duo machine running as a single thread. The performance of our algorithm highly depends on the number of contacting fingers. With five contacting fingers, each frame takes $200ms$ on average for the kinematic test and $5-10ms$ for the dynamic test. Because our method is not designed for real-time, interactive applications, we sometimes trade off performance for more variations in results. For example, if an input sequence has many solutions, we adjust the branching frequency ϵ_1 so that the search algorithm branches less frequently and seeks for solutions with greater variations. Table 2[2] summarizes parameters and runtime of several examples.

[2] l_1 and l_2 are in millimeter (mm), mass is in kg, and time is in second.

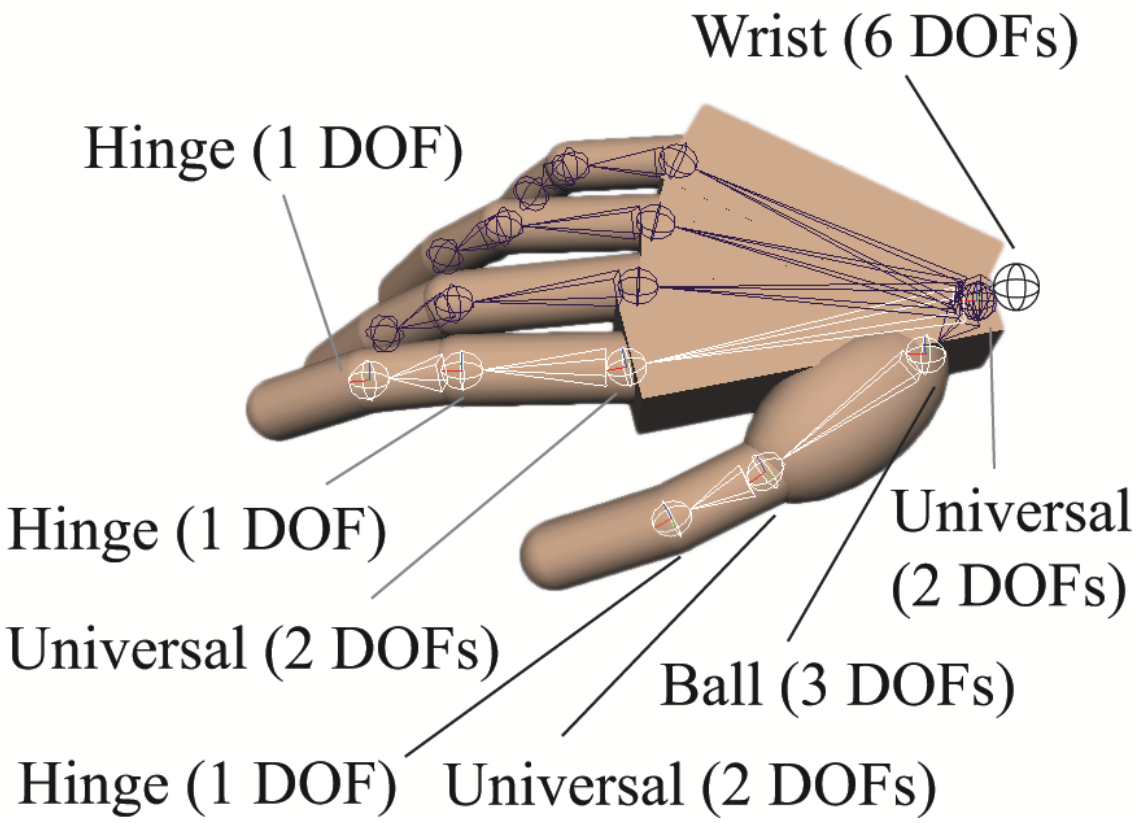


Figure 20: The hand model.

Table 2: Parameters and runtimes of several examples.

Example	l_1	l_2	ϵ_2	μ	mass	T	time	solutions
Turning a milk box	0.1	0.2	10/71	1.0	1.0	71	834	10
Pickup a milk box (1)	0.1	0.5	20/101	1.0	1.0	101	917	10
Pickup a milk box (2)	0.1	0.5	15/96	1.0	1.0	96	256	18
Pickup a milk box (3)	0.1	0.5	15/101	1.0	1.0	101	459	31
Pickup a milk box (4)	0.1	0.5	15/106	1.0	1.0	106	501	32
Pickup small bottle	0.1	0.5	30/102	1.0	0.2	102	199	20
Plate (1)	0.1	0.5	20/101	1.732	0.5	101	613	10
Plate (2)	0.1	0.5	30/221	1.732	0.5	221	1212	42
Spatula (1)	0.5	1.0	20/226	1.732	0.2	226	1323	20
Spatula (2)	0.5	1.0	20/226	1.732	0.2	226	3710	38
Pot (with Wine)	0.5	5	20/120	1.732	1.5	120	636	10
Two-hand big box (1)	0.5	2.0	10/101	1.732	1.0	101	378	10
Two-hand big box (2)	0.5	2.0	10/65	1.732	1.0	65	856	10
Two-hand small box	0.2	0.5	15/141	1.0	0.5	141	666	46
Pickup a bunny	0.1	0.5	40/231	0.5774	1.0	231	1214	30

5.6.2 A cooking scene

We test our algorithm in a realistic cooking scene where the actor moves around in a cluttered environment to fetch and manipulate kitchenwares of various shapes (Figure 21). The full body and object motions are captured using a standard motion capture setting. We segment the input sequence into short clips to improve the performance of the search algorithm.

(a) motion capture setting (b) rendering of synthesis results

Figure 21: Our algorithm synthesizes detailed hand motions for a realistic cooking scene.

Results show that the same algorithm can synthesize detailed hand motions for a variety of shapes and complex manipulation tasks without any prior knowledge or user intervention (Figure 18). In most tasks, an initial stable grasp is not sufficient for the entire motion if it remains static. Instead, our algorithm can successfully discover the appropriate contact movements to synthesize a variety of solutions for every task. Our algorithm is in fact insensitive to the initial contact location and timing. It determines the proper contact time and location from the dynamics of the object. We compare an automatically synthesized motion with one that provided with precise timing for each finger on the motion of turning a pepper bottle on the table (Figure 22). In both motions, the thumb establish and release contact at almost the same time. In the automatic motion, the index finger is primarily used to exert

rolling forces while the middle finger is used in the manually determined sequence. However, the index finger and middle finger in the two motions have qualitatively similar behaviors. The contact points on the object and on the finger both evolve to different locations as the fingers roll and move over time. The subtle, sometimes unpurposeful, movements of the fingers provide richness and realism that differentiate a human hand from a mechanical robot hand.

Objects with sharp edges such as the milk box present challenges to the dynamic test due to the discontinuity of normal direction. Grasping near an edge in a simulation could result in inconsistent forces across frames if we use the point-contact model. In reality, grasping on an edge provides a wider range of possible contact forces because the contact area captures a large range of normal directions. To reproduce such phenomena, we approximate area contact by computing forces on a few proxy points in the neighborhood of an actual contact point, capturing the local features of geometry. As a result, grasping on an edge becomes an available grasping style in our solutions.

5.6.3 Two-hand manipulations

Our algorithm can be directly applied to manipulation tasks with two hands. These tasks requires hands to coordinate and apply contact forces collectively (Figure 18). In the example of fiddling a small box with alternating hands, our algorithm accurately estimates the timing and position of finger-object contacts, simply based on the relative motion between the wrists and the object. Another example is to transport a bigger box from the table to the ground. Before lifting, the hands casually reorient the box by sliding it on the table. The contact establishment and release generated by our algorithm appear coordinated although no prior knowledge is used in our algorithm. During the transportation, the fingers slide and move on the surface of the box due to the physical constraints and relative motion between the wrist and the

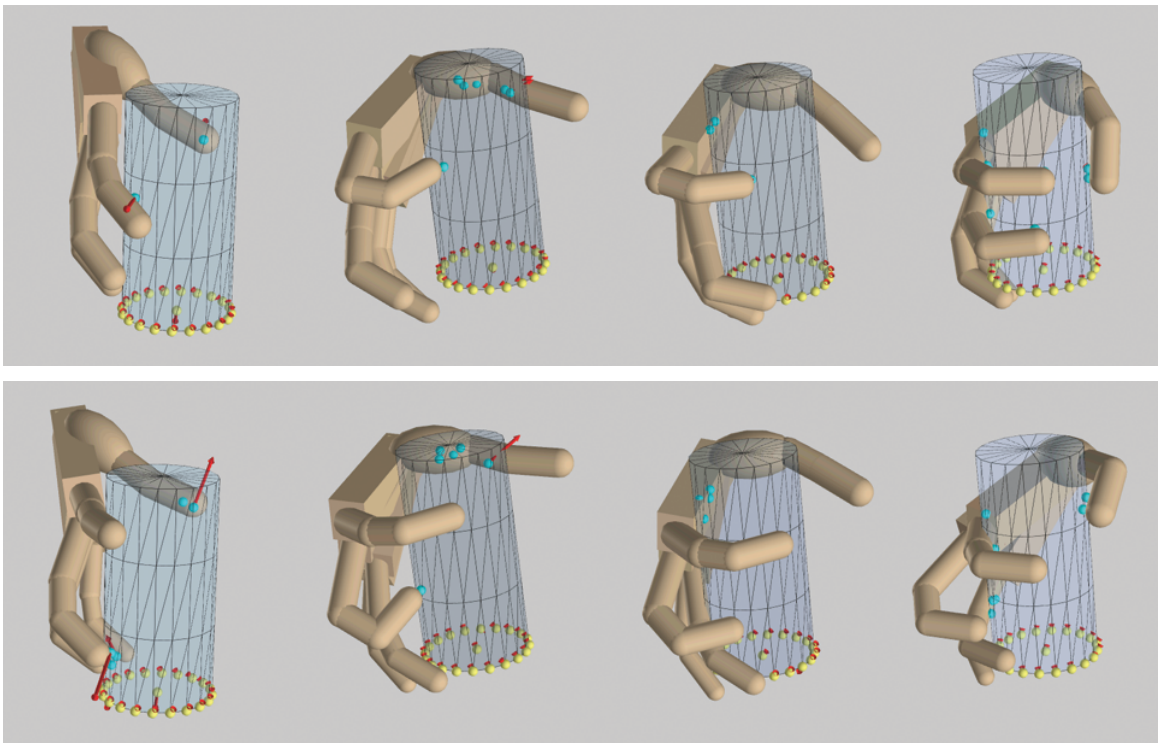


Figure 22: Top row: motion with automatic timing. Bottom row: motion with manual timing.

box.

5.6.4 Different contact strategies

Although our algorithm does not explicitly encourage the different contact strategies, the linear objective in dynamic test (Equation 37) will always prefer forces at the boundary of the constraints, i.e. sliding or no force. Therefore, we can control contact movements in a solution by utilizing the forces and playing with different weighting schemes in the objective.

The first experiment synthesizes static and sliding contact strategies respectively with different friction coefficients on a motion of sliding a box on the table. By always choosing sliding contact samples whenever possible, we synthesize finger motions that slide on the box consistently regardless of the friction coefficient. Interestingly, the sliding direction changes from the horizontal direction to vertical as the friction cone

widened. This behavior results from the force optimization as it exploits friction to minimize normal force. We also observe that the thumb and the pinky finger are almost always used to apply force because they are the most "efficient" fingers while others are used occasionally in different solutions. For example, the middle finger sometimes taps and slides on the top of the box to provide more sliding force when the friction coefficient is small. When the coefficient is large, it will only apply helping forces when positioned to the side. Similarly, we synthesize a static grasp for the same motion by always choosing static contacts in a solution. Even with a small friction coefficient, the fingers still manage to maintain the contact pairs in most solutions. When it becomes difficult for the thumb to maintain a static contact, it starts rolling as a result of collision resolution. The algorithm also decides that it will be more efficient for the middle finger to press the box onto the table once in a while in this example.

In addition to prioritizing contact strategies, we can also generate finger gaits by playing with the weighting scheme in Equation (37). In the example of rotating a paper cup in hand, the fingers has to roll and relocate contacts asynchronously to provide the necessary torques within the hand's kinematic limit. We encourage finger movements by increasing penalties for contact establishment so that free fingers will start applying forces only after they find a contact location that is more efficient than the existing ones. For the contacting fingers, rolling becomes the only feasible strategy given the dynamics of the cup and kinematic constraints of the hand. Interestingly, we observe that the choreography of fingers at capture time is encoded in the cup's varying rotation speed, which in turn leads to realistic synthetic finger motions. When we test the same algorithm on a synthetic motion with constant rotation speed, the finger motion doesn't appear natural. Similarly, we synthesize finger gaits of turning a small box in hand, and observe the same behavior of asynchronous finger relocation (Figure 23). A similar optimization scheme is also used in the cooking scene for

turning the pepper bottle before it is picked up. After the bottle is picked up, we tune down the penalty for contact establishment to reduce contact movements.

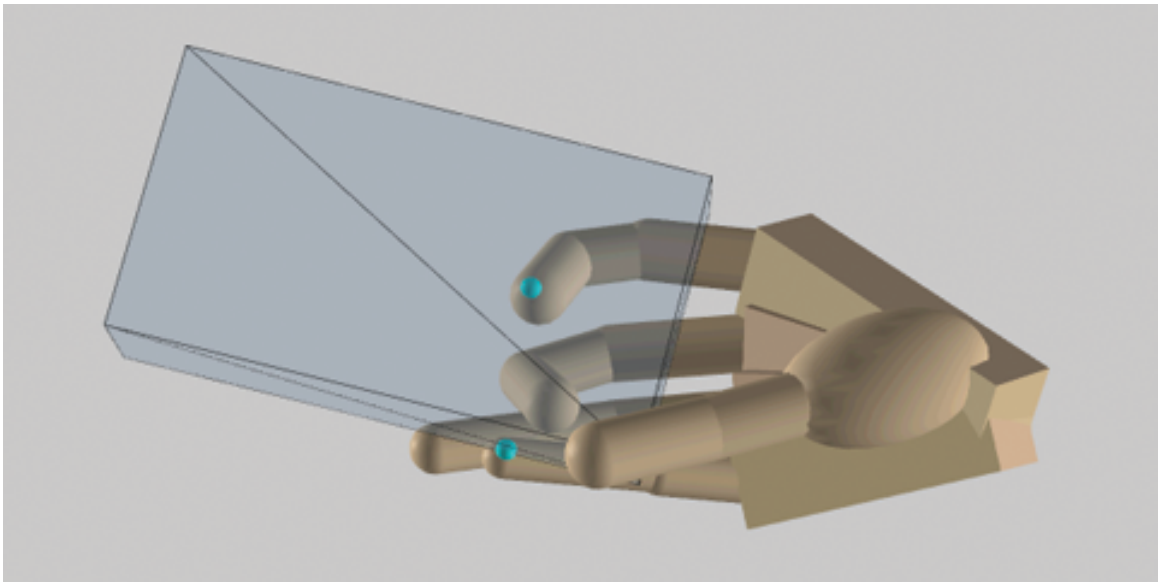


Figure 23: Fiddling with a box in hand.

5.6.5 Editing object properties

In a production pipeline, virtual props often need to be modified after capturing the actor's performance. Therefore, being able to adapt the hand motion to various object properties is highly desirable.

Our first test changes the object motion to generate different grasp styles. For example, a power grip brings the object closer to the palm while a precision grip keeps distance between the object and the palm. We conduct this experiment with the spatula motion in the cooking scene (Figure 24). The original spatula motion locates at the finger tips most of the time, resulting in a careful grasp. By moving the spatula closer to the palm, the fingers automatically curl around the spatula to form an envelop grasp, and use the palm to exert forces.

We can also use the same captured motion on objects with different shapes. We show that replacing a box with a bunny or a mug retains the quality of the original

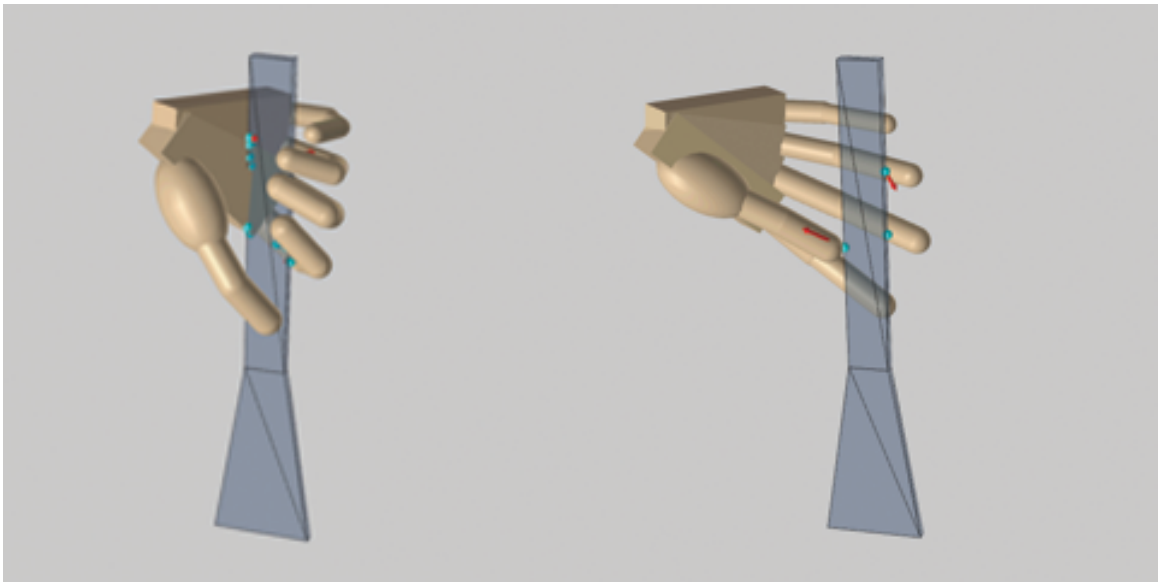


Figure 24: Different grasp styles.

motion as the fingers naturally adapt to the new object with the same wrist motion (Figure 25).

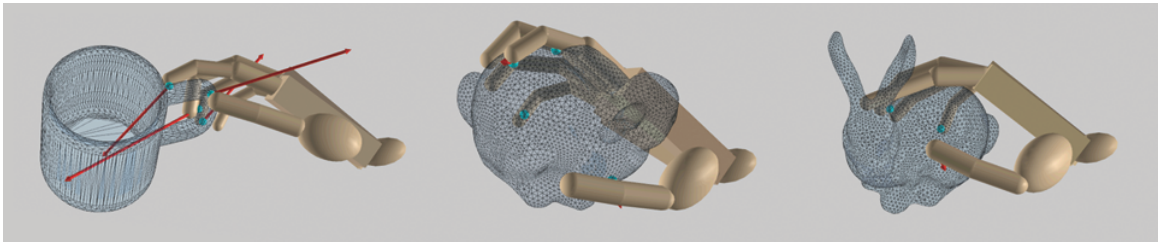


Figure 25: The hand adapts to a mug and a bunny from the same input motions.

5.6.6 Evaluation

We evaluate the quality of our results by comparing with motion capture data and video footage. We captured the hand motion of picking up a box from the table with a close-range camera setting, then use the motion of the wrist and the box to synthesize 10 solutions. By comparing the visually most similar solution to the captured data, we find that our result is qualitatively similar to the reference, although not identical (Figure 26). While the motion capture result is sometimes noisy and

contains penetrations, our results are visually and physically plausible. In addition, we can synthesize a range of different grasps for the same captured motion.

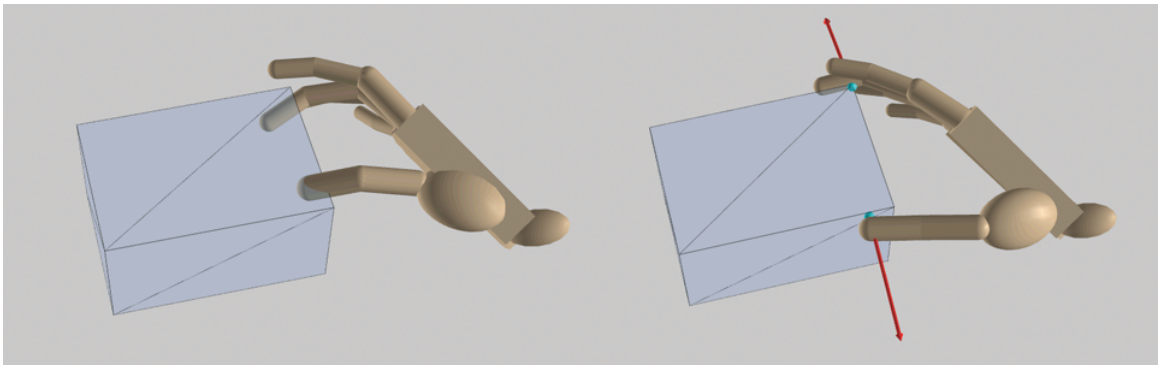


Figure 26: Left: motion capture data. Right: our synthesis result.

We also record a video of an actor fiddling with a box using one hand, and capture the wrist and object motion at the same time (Figure 23). The motion exhibits frequent contact movements and complex contact relations among the hand, the object, and the table. We apply collision detection between the hand and the object during manipulation, and resolve finger-table collision in post-processing. Although our solution is different from the actual performance, partly due to the discrepancy in hand modeling, the overall features of finger gaiting and contact sliding are present in the synthesized motion. Our result is qualitatively similar to the video footage, and appears plausible. However, this challenging example also reveal some drawbacks of our method. For example, the fingers will sometimes penetrate the table during manipulation because resolving penetration with the box and the table together is difficult. The fingers we synthesized are also further apart from each other compared to the video to prevent self-penetration.

Our improved algorithm is more efficient in discovering variability in the solution space compared to the baseline. We visualize the search tree for both algorithms on a motion sequence of 230 frames after exploring 3000 nodes. While the baseline algorithm spends most computation within a narrow space (Figure 27), our algorithm

covers a significantly larger space and provides more variations in the solutions (Figure 28). Figure 29 shows some of the solutions we discover within the first 50 solutions.

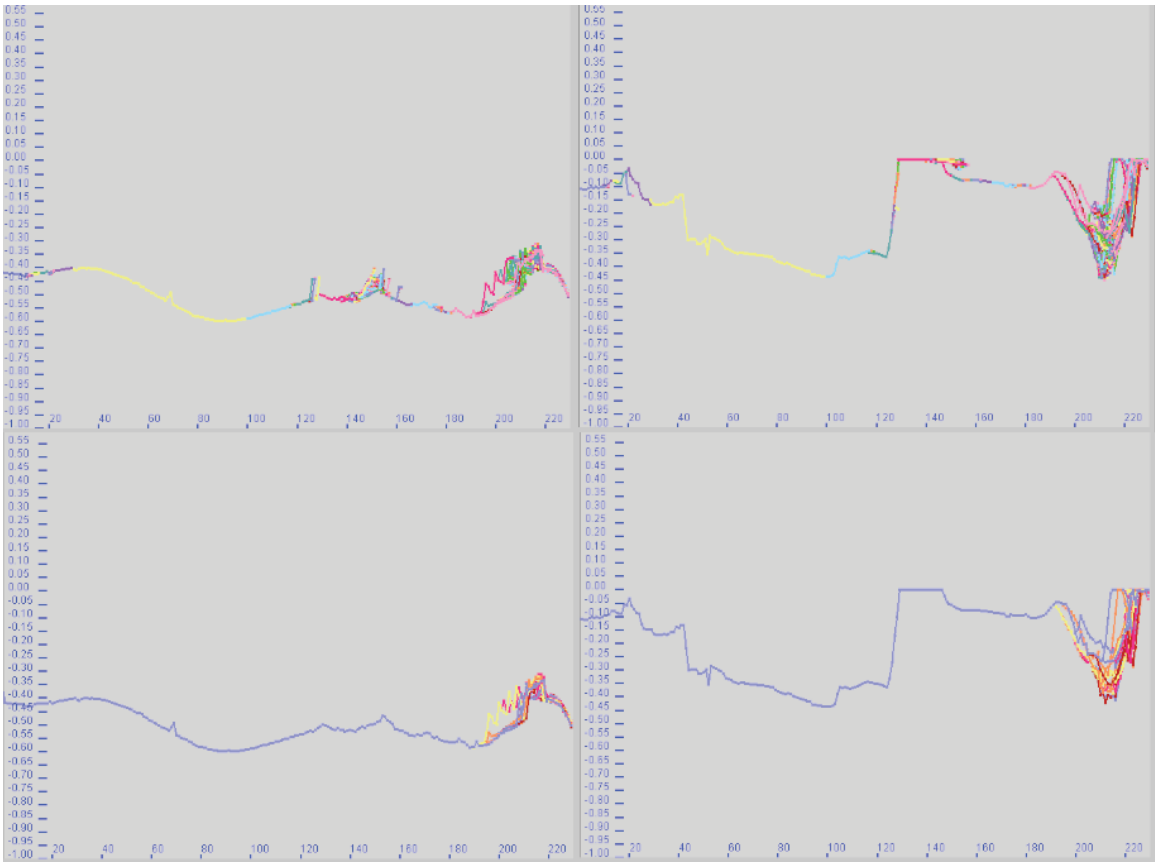


Figure 27: Search tree for a joint with 3000 nodes using the baseline algorithm. A node has up to 3 branches and the search branches out every 5 frames. The search clusters around a small portion of the motion space. Top left: z-axis of the MCP joint of the index finger; top right: z-axis of the PIP joint of the ring finger; bottom left: successful trajectories of the index finger; bottom right: successful trajectories of the ring finger.

5.7 Discussions

Although we design the algorithm to be generic, its capability is confined by the hand model being used. First, a rigid hand cannot model deformation at the site of contact and the conformation of the palm to the object. The rigid models also present difficulties to collision resolution. For instance, a tight grasp or a fist would be hard to model with rigid palm and fingers. Another example is simultaneous

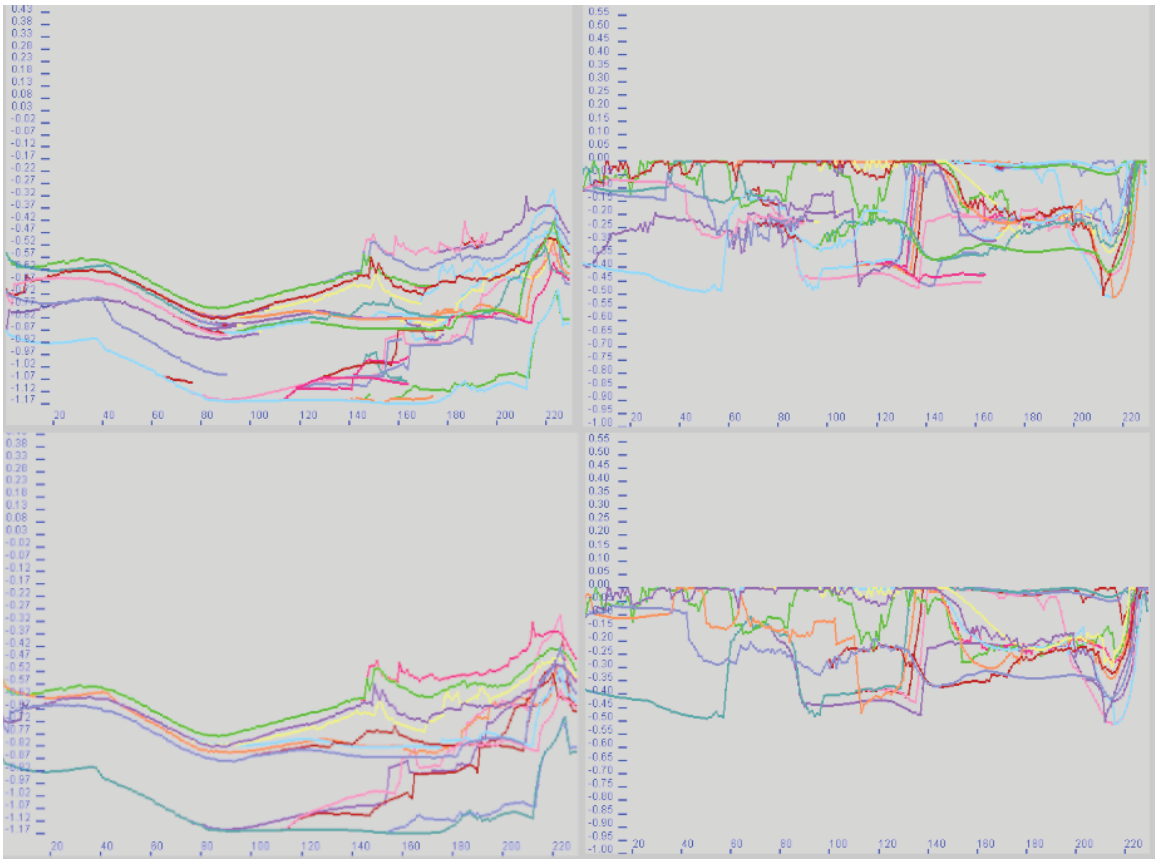


Figure 28: Search tree for a joint with 3000 nodes. Top left: z-axis of the MCP joint of the index finger; top right: z-axis of the PIP joint of the ring finger; bottom left: successful trajectories of the index finger; bottom right: successful trajectories of the ring finger.

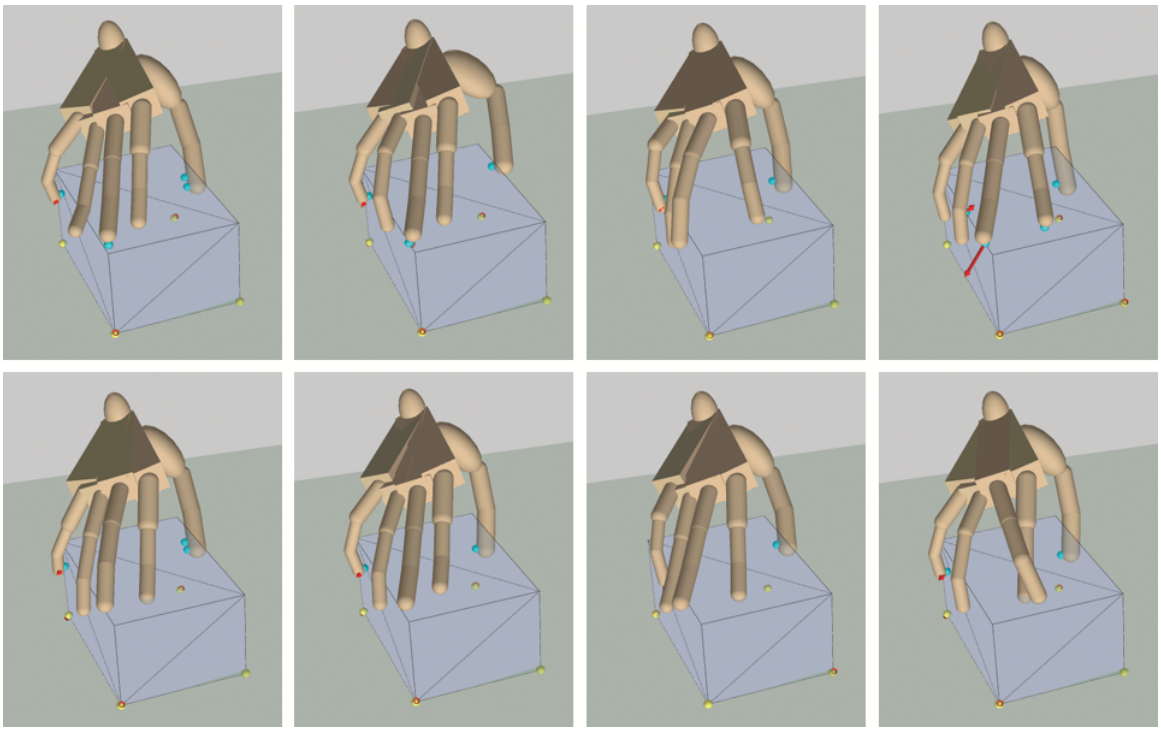


Figure 29: Our algorithm explores a variety of solutions.

contact with the object and the environment such as a finger going between a box and the table before picking up the box. Second, interdependencies among fingers are not modeled currently. As a consequence, the fingers appear to be independent and sometimes unnatural in some results. Incorporating a more accurate hand model with anatomically correct structure [135, 112] and deformable skins [58] is a fruitful future direction to pursue.

The bottleneck of our current framework is the kinematic test which resolves penetration by a few iterations of nonconvex optimization. While the method is straightforward, it does not work well for extreme cases such as contacting with a sharp corner or with small objects such as a pen or a piece of paper. A more robust and efficient collision resolution routine can greatly improve the performance and capability of our method.

Our search algorithm currently has limited planning capability because it only takes into account the current frame. Consequently, it cannot automatically adjust

the contact strategies to suit any input motions. We can improve the performance in challenging cases by planning with short horizon or receding horizon (between decision points), and incorporate a short term cost function and grasp quality metrics [86] to decide whether to terminate a branch earlier.

Our algorithm is a useful complement to existing motion capture techniques, and we can benefit from future advance in data capture to gather high quality input data of the wrist and the objects. Currently, our method occasionally suffers from noisy input data due to occlusion.

CHAPTER VI

CONCLUSION AND FUTURE WORK

In this thesis, we have presented algorithms for controlling a full body character balancing and responding to dynamic events in the environment, as well as performing everyday object manipulations in a natural manner. By exploring style variations in the subspace of kinematics and physics constraints, we incorporate the previously conflicting goals of achieving robustness and naturalness into the same control framework. As a result, our algorithms allow for more general controllers that can adapt to various constrained environments and be flexible about their action plans. Our algorithms can be generalized to a wide variety of motion contents and character structures, allowing for more intuitive controller design.

We first presented in Chapter 3 a control algorithm that synthesizes stylistic postural responses to small-scale perturbations. By enforcing the dynamic constraints in the actuation space, the virtual character responds to arbitrary unexpected perturbations in a style consistent to the input motion.

The main assumption of our approach is that only a small set of coordinated muscle groups are activated for performing rhythmic motions. Biomechanics researchers have also hypothesized that postural responses under perturbations can be activated by a few muscle synergies [117]. Our results suggest that the same muscle synergies used for the input motion can also produce reasonable recovery motions from small perturbations, thereby lending support to the hypothesis of muscle synergies as building blocks for constructing motor output patterns.

Our experiments also reveal distinctive motion features among individuals and activities. For example, we observe that the distribution of eigenvalues accurately

categories the energy level of a motion, e.g. the Tai-Chi performance is significantly more energetic than normal walking, and walking in similar styles retains similar energy levels. When the same individual performs different actions of similar energy levels, the distribution of eigenvalues are also similar. Another interesting observation is that the coordinations of responses reveal features of the actuation space. Although the eigen-basis is subject to an arbitrary rotation, the space they span should be unique. Our observation of unique coordination patterns in perturbation responses seems to support the existence of a characteristic actuation space. A natural extension of our work would be to conduct functional analysis in the actuation space to identify and understand the motor functions encoded in the space and their interactions. It will provide more insights about the building blocks of a motion and answer the question of how to choose the un-actuated coordinates. Currently, it is not clear whether the human body switch to different muscle synergies in the presence of sustained or large-scale perturbations to maintain balance. From our experiments, some coordinates are not relevant in reproducing the reference motion, but are essential for recovery from small perturbations. A functional analysis of such coordinations may provide us with more insights of the recovery process.

To address the balance problem at the presence of large disturbances, we introduced in Chapter 4 a novel technique to control and synthesize real-time character motions under physical perturbations and changes in the environment. We designed an optimal feedback controller that allows for online re-planning of final goals and completion time. The abstract dynamic model incorporates an accurate dynamic model of the COM and high-level balance strategies such as angular momentum regulation. Our results show that varying the timing and final goal of the motion is critical for producing robust and realistic results. We are encouraged to see that such a simple dynamic model is able to capture the high level dynamic features of a variety of activities, including walking, long stepping, squatting, running, hopping, and

wandering with random turns and stops. The generic form of the abstract model will be useful to generate motions that utilize environment contacts as the primary source of external actuation. We believe the same method will also perform well in sports or gymnastic motions such as climbing, swing, and skiing *etc.* , as well as quadruped motions.

The use of a simple abstract dynamic model also enables the application of advanced optimal control theories, thus allow us to synthesize motions that deviate significantly from the reference, and enrich the set of possible perturbations. However, the current full-body pose reconstruction algorithm still heavily depends on the reference. A promising future direction is to extend our algorithm in Chapter 3 to full body motion using the contact force information solved from the abstract dynamic model. Once we obtain the actuation/un-actuation space for the full body, we can use them to simulate the full body motion under the momenta constraint provided by the abstract model. In this way, we can allow for even larger deviations from the reference. Such a framework can intelligently extract strong models from a single reference motion, and it has minimum dependencies on the reference trajectory during synthesis.

The simplicity and generality of the abstract model makes it ideal for building powerful super controllers from a library of motions. Recent research has shown promising results in compositing controllers of different behaviors optimally in a simulation. Da Silva *et al.* [30] has shown that a class of controllers can be optimally composed to enlarge the capability of each individual controller. Muico *et al.* [90] further applied this theory to build a robust super controller from a handful of tracking controllers, each of which is built from an example motion. However, the application of this theory is limited by the base controllers in two counts. First, solving base controllers for all kinds of human activities is an open problem in itself. Second,

the capability of the super controller heavily depends on the quality of the base controllers. We believe our abstract model is a fruitful direction in tackling these two issues because it is applicable to a large variety of activities, and generalized well from one single example. Compositing controllers for the abstract model will greatly reduce the complexity of the algorithm and require only a small number of inputs.

The abstract model, however, doesn't address the issue of discontinuous contact constraints that happen especially frequently in hand-object manipulation tasks. Chapter 5 described a sampling methods that explore solutions to manipulation problems within the discrete contact space. The success of our algorithm suggests that a major portion of the high level manipulation goals and planning happens in the wrist level, and reflects in its result, i.e. motions of the objects. Therefore, utilizing motions of the wrist and the objects, we are able to discover a variety of realistic solutions using a brute force greedy search algorithm. Our algorithm is also a useful complement to existing search algorithms such as Rapidly-Exploring Random Tree (RRT) and probabilistic roadmap. Most current search algorithms focus on efficient explorations in the spatial domain as most of their applications are navigation tasks. For example, for most mobile platforms, the dynamic constraints that transitions the spatial position in time are not the determining factor for a navigation task. However, in our problem of detailed manipulations, the discrete contact dynamics plays a dominant role in determining the feasibility of a solution. We have demonstrated that a carefully chosen sampling scheme in transition works effectively with simple uniform samples in space. Further investigation into the performance of state-of-the-art searching algorithms with ours in discovering finger gaits for manipulation will be helpful in better understand the properties of the problem and our algorithm.

The detailed hand motions we discover can serve as input to many applications. For instance, data-driven methods have shown to be useful in controlling full body motions, but their usage is very limited in hand manipulations because of the difficulty

in data acquisition. Using our method, we can automatically generate a variety of hand motions with different styles, well covering the motion space. With the resultant motions, we will then be able to apply current state-of-the-art tracking controllers to synthesize manipulation strategies under perturbations.

A boarder implication of our algorithms is a unified view of the balance problem in locomotion and hand manipulation problem. These two problems are usually solved in isolation as two separate research topics with specific domain knowledge. In our work, we show that they can be both treated as an un-actuated models being actuated or controlled by exerting forces from contact points. Motions of the un-actuated model and contact point-force pairs are the two sides of a coin. If we have information of one of them, we can solve for the other. In Chapter 4, we solve the motion of the COM from information of foot contacts. And in Chapter 5, we solve the finger-object contacts from motion of the objects. Furthermore, by treating contact point-force pairs as control signals, we can apply advance optimization techniques to develop powerful controllers of the un-actuated model (e.g. COM or rigid body objects). There are, however, two remaining pieces of work to complete the loop. First, we would like to apply our contact search algorithm in locomotion to solve for foot contacts that best balance the COM. Second, we would like to develop feedback controllers in the same way as in Chapter 4 for hand manipulations using motions we solve in Chapter 5 as input.

6.1 Applications

Animation. Although research in character animation has rapidly advanced in recent years, the impact has not been able to reach far into the movie and gaming industry. While the industry adapts relatively quickly to automatic algorithms in synthesizing natural phenomena or special effects such as ocean or explosion, character motions are exclusively created by hand. Animators are reluctant to give up

their control over every gory details of a character's motion. Even motion capture is deemed unsuitable as the final output because the data is not flexible to change at will and adapt to cartoon or super hero physics. The primary use of motion capture is to provide reference or ground truth for animators.

One barrier to use automatic algorithms for synthesizing character motion is the control interface. Most algorithms are designed by and for programmers. Knowledge of the low level implementation details or problem formulation is required to understand the input to the algorithm. Nontechnical users will probably take more effort to use the system than creating the motions by hand. In addition, assumptions and domain knowledge incorporated in the system design limit the applications of specific algorithms in a production setting, when the practical problems do not meet the requirement. Only algorithms with minimum problem specific assumptions and intuitive "control knobs" will be suited in an industrial setting.

We strive to meet these two goals in the design of our algorithms. For example, our methods do not depend on a particular character model or system dynamics, removing the burden of per-case tuning and allowing for altering the properties of the physical world. An important feature of our algorithm is its hybrid nature that combines kinematics control with physics-based simulation. As a result, it provides better directability compared to pure physics-based methods, and it automatically ensures realism by enforcing the most essential physical properties in a motion. More importantly, a hybrid methods provides control to decide what and how much to "cheat" physics for controllability in kinematics. We believe our algorithms serve as a convenient prototype for the industry to start adapting and utilizing animation research results in production.

There are a few features we can already think of to make our algorithms more usable. For example, the abstract model can be used as a guidance to help animators design physically plausible motions (real-world or cartoon physics). We can run our

algorithm on motions animated manually to compute a COM trajectory that respects the laws of physics. We can then present the result and the corresponding modification to the full body motion as suggestions of which part in the motion can be made more physical. For synthesizing hand motions with our algorithm, we already provide control of the overall style (more steady or clumsy), and allow for modulations of various properties of the hand and the objects. In practice, artists often demand direct forward kinematics and inverse kinematics control. We would like to include these features in the future such that the user can directly specify angles for a particular joint or set a contact point at a particular coordinates. Further, an intuitive user interface for visualizing and selecting desired motion from a large motion dataset can be immensely useful. Inspired by the Many-Worlds-Browsing technique [123], one possible future direction is to create an interactive interface that allows the user to browse and adjust parts of the scenes with ease. To provide seamless interaction experience, we can solve many solutions in parallel thanks to the sampling nature of our algorithm.

Robotics. Traditional robotics controllers had been primarily focusing on robust and accurate control with no regard to the motion quality, partly due to limitations of the control hardwares and lack of feedback sensors. Therefore, animation control algorithms that generate high quality motions in a simulated world is not suitable for controlling robots in the real world. Robotics and animations were considered two separate fields. However, with recent advances in material sciences and manufacturing technology, robotic actuators are becoming considerably lighter and faster, which enables the execution of flexible and precise motions. Sensors have also been improved considerably in terms of accuracy and feedback modals (i.e. visual, acoustic, haptics *etc.*). The physics between real and simulated worlds are becoming more and more consistent. In addition, the increasing popularity of personal assistant robots requires

compliant and responsive controllers that are versatile in an unstructured environment and safe to interactive with. The demand in robotics control is becoming more aligned to research goals in character animation. We can now start to share algorithms and platforms between humanoid robots and virtual characters. The work presented in this thesis brings in natural interactions to physics-based simulation, therefore has high potential to also improve controllers for humanoid robots that interact and corporate with humans.

Biomechanics. Several aspects of our algorithms are inspired by insights from biomechanics research. Our software systems, in turn, can serve as a computational platform for testing hypothesis on human biomechanics or neuromotor control. Due to the limited computation tools, most traditional computational neuroscience research focuses on the analysis of a very detailed and specific aspect of a motion from observed data. A powerful computation platform that allows for prediction and synthesis of detailed and full scale motions from hypothesis will be immensely helpful to advance the field. A fruitful direction to extend our work for this purpose is to work with a detailed anatomical model. It would be interesting to see whether our methods on articulated rigid body can also be applied with success to a more realistic biomechanical model. As a first step, comparing the results of our method with Torres and Ting's work [117] on different ambulation tasks may provide insights on the appropriate abstraction to study human motions. Our methods of identifying eigen-torques and controlling COM using contact forces are also useful for analyzing the acquisition and adaption of motor patterns on health individuals as well as on patients with mobile disability because both the torques and COM trajectories captures distinctive features of motions. The results can then help shred light on the design of prosthetics devices or bionic devices, as well as rehabilitation procedures. In general, opportunities of marrying neuroscience with computational power are numerous,

existing and promising.

6.2 Future work

Motion Retargeting. Motion retargeting has been an important problem for computer animation for two major reasons. First, the shape and proportion of a character often change gradually as the story progresses in an animation sequence. It is therefore important to make sure motions can be applied consistently to the character throughout the story. Second, as an increasingly prevailing method for animation, motion capture almost always requires retargeting motions from an actor to a virtual character of different shape and build or even different species. Although the motion contents can be of many forms, such as full body motion, hand-object manipulation, facial expressions, or a combination of them, the key questions in retargeting are common. To retarget a motion, we need to define what content can transfer directly and what needs to be adapted to different models. To adapt motions between different models, we need to define the correspondence or a mapping between the source and the target. Existing methods usually establish kinematic correspondence between end effectors, and solve IK problems as a mean of retargeting [43, 48]. The correspondence plays a key role in the success of the algorithm, usually requiring prior knowledge of the models and making strong assumptions about the motion context. A specific character model, or the same character model with a different type of motion requires unique treatments to ensure a meaningful mapping between the source and target. Therefore, retargeting is currently solved with considerable manual effort and domain knowledge, when automatic methods come into picture only in a late stage of the pipeline. A general and automatic approach of establishing correspondence will greatly facilitate the retargeting process.

Although the work in this thesis doesn't focus on the retargeting problem, our goal is to develop general solutions that work for a wide range of models and motions. An

important insight we gain is that optimization is a promising generic representation of motions, which doesn't depend on the model or the motion content. The kinematic or dynamic features of the system can be specified as constraints, and the high level intention of the motion can be specified as an objective function. Low level execution of the motion is a direct consequence of achieving the goal optimally within the constraint space. Therefore, if we can describe the intention of the motion in model-independent terms, such as moving the COM from one location to another using certain velocity, we can retarget this motion to different characters by solving an optimization with the same objective but different constraints. This is the approach we have been taken in this thesis, and by concurrent work of feature-based motion synthesis [3, 59, 32]. In practice, it is not always possible to specify a motion by high level model-independent objectives. A fruitful next step is to retarget model-specific objectives to different character models, which should be considerably more straightforward and intuitive for automatic algorithm design.

Inverse Optimization. Although optimization appears to be a powerful representation of motion, in practice, it is difficult to formulate a solvable optimization problem that yields meaningful results. Often times, the system dynamics are not known a priori, and it's difficult to describe a motion by a few high level features. Instead, we are given one or a few motions for a particular model as examples of desirable output without much knowledge of the underlying generation mechanism. The process of identifying the optimization problem from the example outputs is an inverse optimization (IO) problem.

Inverse optimization is closely related to a type of reinforcement learning that learns the value function. It has been applied in robotics to allow robots mimic movements from human demonstrations and trial-and-error. When the cost function is defined as a linear combination of features, the relative weights can be recovered by

exploring convexity of the problem. Abbeel and Ng *et al.* [2] applied the method in aircraft auto-piloting by learning weights from experienced pilots. Lee and Popović [73] improved on the method to a more accurate result for deterministic systems, and successfully applied the method in navigation tasks. When the cost function takes a generic form, but the problem is affine in control, Todorov [115] explore the linearity in control, and use sample points to approximate the cost function in the neighborhood of the examples. However, their algorithm has been applied only to simple mechanical models. Expanding the capability of inverse problems to various types of human motions is a promising future direction that will benefit many scientific fields.

Abstract inertial model. Solving an IO problem usually involves many evaluations of the optimization problem. Being able to solve the optimization problem efficiently is key to the success of an IO problem. Unfortunately, most problems in character animation suffer from nonlinearity and high dimensionality issues. The sources of these difficulties can come from the model kinematics and dynamics, as well as the motion features. However, it is commonly believed that the high dimensionality is an artifact of model and motion representation, and the intrinsic dimensionality of natural human motion is low. Many dimensionality reduction techniques have since been explored to reduce the complexity of motion representation, but most of them pay the price of a narrower range of representable motion while still suffering from the nonlinearity issue. On the other hand, we believe an abstract model is a better approach in reducing both dimensionality and nonlinearity without sacrificing generality. Our unified view of motion as the global DOFs being controlled by contact forces is especially useful in formulating solvable forward and inverse optimization problems. We have already shown that an abstract model of the COM can easily incorporate the otherwise nonlinear features such as the linear and angular momenta;

and a naïve uniform sampling method is sufficient to handle the discrete contact dynamics under this framework.

The major drawback of our current abstract model is the lack of inertial effects. Utilizing the change of body shape as a mean of angular momentum control is an important feature for articulated or deformable models. Modeling the inertia in our abstract model requires an additional six DOFs, but the fundamental framework does not need to change. We believe the exploration of inertia space will enable more interesting control strategies, and it will be a fruitful enhancement to our abstract model and contact control framework. Such a model can facilitate the development of forward and inverse optimization techniques in character animation.

Deformable and anatomically correct human model. An abstract model *per se*, even with inertial dynamics, does not yet represent a meaningful character motion. Mapping from the abstract model to a full scale model is an important final step to make the abstract model framework useful. Since the abstract model captures only the most essential aspects, quality of the final result depends heavily on the capability of the full scale model. While the traditional articulated rigid body model appears to be a perfect balance between fidelity and complexity, it becomes increasing important to model the deformation of flesh and skin, as well as the anatomical details that cannot be represent by articulated rigids. We can carry the abstraction idea further by employing a level of detail approach. For example, we can layer the anatomical model on top of the articulated rigid body model, then attach deformable skins on the anatomical structure. Mappings between each consecutive level can be improved on individually without affecting the other layers. Jain and Liu [58] has presented some preliminary work in combining a deformable skin with articulated rigid bodies. We believe this is a promising direction to pursue in order to achieve fully detailed motions with affordable computation.

APPENDIX A

FEEDBACK GAINS

We provide a compact description of our implementation on computing feedback gains in Equation (22). Please refer to Jacobson and Mayne [56] for complete derivations of implicit final time problems (Chapter 2.3.5) and final constraint problems with inequality constraints in control (Chapter 2.5).

We denote the dynamic function (Equation (19)) as f, and use subscripts to represent partial derivatives. In our problem, the Hamiltonian is defined as $H = L + V_{\mathbf{x}}^T f$, with the objective L defined inside the integral in Equation (20). The feedback gains are computed as follows:

$$\begin{aligned}
\mathbf{K}^{\mathbf{x}} &= - H_{\mathbf{FF}}^{-1} Z (H_{\mathbf{F}} + f_{\mathbf{F}}^T V_{\mathbf{xx}}), \\
\mathbf{K}^{\mu} &= - H_{\mathbf{FF}}^{-1} Z f_{\mathbf{F}}^T V_{\mathbf{x}\mu}, \\
\mathbf{K}^{t} &= - H_{\mathbf{FF}}^{-1} Z f_{\mathbf{F}}^T V_{\mathbf{x}t_f},
\end{aligned}$$

where

$$Z = \mathbf{I} - g_{\mathbf{F}}^T (g_{\mathbf{F}} H_{\mathbf{FF}}^{-1} g_{\mathbf{F}}^T)^{-1} g_{\mathbf{F}} H_{\mathbf{FF}}^{-1}.$$

$\mathcal{K} = \{\mathbf{F} | g(\mathbf{F}, \Lambda) \geq \mathbf{0}\}$ approximates a static friction cone using a linear combination of basis vectors in the columns of Λ. $g = \Lambda^T \mathbf{F}$ is the unilateral constraint of contact forces in which the projection of $\mathbf{F}$ on Λ has to be positive. The gradient $g_{\mathbf{F}}$, therefore, is simply Λ^T.

Once we compute the coefficients $\mathbf{K}^{\mathbf{x}}, \mathbf{K}^{\mu}$, and $\mathbf{K}^{t}$, we can use Equation (22) to compute the feedback force $\delta\mathbf{F}$. In practice, taking a full step of $\delta\mathbf{F}$ may violate the unilateral constraint. We instead search for a maximum step length $\lambda \in (0, 1)$ to keep $\mathbf{F}$ within the feasible region such that $\mathbf{F} = \mathbf{F}^* + \lambda\delta\mathbf{F} \in \mathcal{K}$.

Derivatives of V are computed by integration of the following ordinary differential equations backward from t_f to t_0 at $\mathbf{X}^*$ and $\mathbf{F}^*$. These expressions are simplified for our problem thanks to the linearization of the friction cone.

$$
\begin{aligned}
\dot{V}_{\mathbf{x}} =& - H_{\mathbf{x}}, \\
\dot{V}_{\mathbf{x}\mu} =& - (f_{\mathbf{x}} + f_{\mathbf{F}} Z^T \mathbf{K}^{\mathbf{x}})^T V_{\mathbf{x}\mu}, \\
\dot{V}_{\mathbf{x}t_f} =& - (f_{\mathbf{x}} + f_{\mathbf{F}} Z^T \mathbf{K}^{\mathbf{x}})^T V_{\mathbf{x}t_f}, \\
\dot{V}_{\mu t_f} =& V_{\mathbf{x}\mu}^T f_{\mathbf{F}} Z^T H_{\mathbf{FF}}^{-1} Z f_{\mathbf{F}}^T V_{\mathbf{x}t_f}, \\
\dot{V}_{\mathbf{xx}} =& - H_{\mathbf{xx}} - f_{\mathbf{x}}^T V_{\mathbf{xx}} - V_{\mathbf{xx}} f_{\mathbf{x}} \\
& + (H_{\mathbf{Fx}} + f_{\mathbf{F}}^T V_{\mathbf{xx}})^T Z^T H_{\mathbf{FF}}^{-1} Z (H_{\mathbf{Fx}} + f_{\mathbf{F}}^T V_{\mathbf{xx}}), \\
\dot{V}_{\mu\mu} =& V_{\mathbf{x}\mu}^T f_{\mathbf{F}} Z^T H_{\mathbf{FF}}^{-1} Z f_{\mathbf{F}}^T V_{\mathbf{x}\mu}, \\
\dot{V}_{t_f t_f} =& V_{\mathbf{x}t_f}^T f_{\mathbf{F}} Z^T H_{\mathbf{FF}}^{-1} Z f_{\mathbf{F}}^T V_{\mathbf{x}t_f}.
\end{aligned}
$$

Boundary conditions at t_f are the following:

$$
\begin{aligned}
&V_{\mathbf{x}} = \Psi_{\mathbf{x}}^T \mu, \qquad && V_{\mathbf{x}\mu} = \Psi_{\mathbf{x}}^T, \qquad && V_{\mathbf{x}t_f} = H_{\mathbf{x}} + V_{\mathbf{xx}} f, \qquad && V_{\mu t_f} = \Psi_{\mathbf{x}} f, \\
&V_{\mathbf{xx}} = \mathbf{0}, && V_{\mu\mu} = \mathbf{0}, && V_{t_f t_f} = H_{\mathbf{x}}^T f + f^T V_{\mathbf{xx}} f.
\end{aligned}
$$

REFERENCES

[1] *Cyberglove Systems, http://www.cyberglovesystems.com/.*

[2] Abbeel, P. and Ng, A., "Apprenticeship learning via inverse reinforcement learning," in *21st International Conference on Machine Learning*, 2004.

[3] Abe, Y., da Silva, M., and Popović, J., "Multiobjective control with frictional contacts," in *Eurographics/SIGGRAPH Symposium on Computer Animation*, pp. 249–258, 2007.

[4] Abe, Y., Liu, C. K., and Popović, Z., "Momentum-based parameterization of dynamic character motion," in *Eurographics/SIGGRAPH Symposium on Computer Animation*, pp. 173–182, 2004.

[5] Abe, Y. and Popović, J., "Interactive animation of dynamic manipulatin," in *Eurographics/SIGGRAPH Symposium on Computer Animation*, pp. 195–204, 2006.

[6] Albrecht, I., Haber, J., and Seidel, H.-P., "Construction and animation of anatomically based human hand models," in *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 98–109, July 2003.

[7] Alexander, R., "Simple models of human movement," *Applied Mechanics Reviews*, vol. 48, no. 8, 1995.

[8] Alexandrov, A., Frolov, A., Horak, F., Carlson-Kuhta, P., and Park, S., "Feedback equilibrium control during human standing," *Biological Cybernetics*, vol. 93, no. 5, pp. 309–322, 2005.

[9] Allen, B., Chu, D., Shapiro, A., and Faloutsos, P., "On the beat!: timing and tension for dynamic characters," in *ACM SIGGRAPH/Eurographics symposium on Computer animation*, pp. 239–247, 2007.

[10] Allen, B., Neff, M., and Faloutsos, P., "Analytic proportional-derivative control for precise and compliant motion," *International Conference on Robotics and Automation (ICRA)*, May 2011.

[11] Andersen, E. D., Jensen, B., Jensen, J., and Sandvik, R., "Mosek version 6," Tech. Rep. TR-2009-3, Ulf Worsøe, October 2009.

[12] Arikan, O. and Forsyth, D. A., "Interactive motion generation from examples," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 483–490, 2002.

[13] Arikan, O., Forsyth, D. A., and O'Brien, J. F., "Pushing people around," in *ACM SIGGRAPH/Eurographics symposium on Computer animation*, pp. 59–66, 2005.

[14] Aydin, Y. and Nakajima, M., "Database guided computer animation of human grasping using forward and inverse kinematics," *Computers and Graphics*, vol. 23, no. 1, pp. 145–154, 1999.

[15] Barbic, J., Safonova, A., Pan, J.-Y., Faloutsos, C., Hodgins, J. K., and Pollard, N. S., "Segmenting motion capture data into distinct behaviors," in *Graphics Interface*, vol. 62, pp. 185–194, May 2004.

[16] Blickhan, R., "The spring-mass model for running and hopping," *Journal of Biomechanics*, vol. 22, no. 11-12, 1989.

[17] Brand, M. and Hertzmann, A., "Style machines," in *SIGGRAPH*, pp. 183–192, 2000.

[18] Cai, C. and Roth, B., "On the spatial motion of a rigid body with point contact," in *Proc. IEEE Int. Con. Robotics and Automation*, pp. 686–695, 1987.

[19] Carlson, M., Mucha, P. J., Van Horn, III, R. B., and Turk, G., "Melting and flowing," in *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '02, (New York, NY, USA), pp. 167–174, ACM, 2002.

[20] Chai, J. and Hodgins, J. K., "Constraint-based motion optimization using a statistical dynamic model," in *SIGGRAPH*, p. 8, 2007.

[21] Chenney, S. and Forsyth, D. A., "Sampling plausible solutions to multibody constraint problems," in *SISGGRAPH*, pp. 219–228, Aug. 2000.

[22] Chentanez, N. and Müller, M., "Real-time eulerian water simulation using a restricted tall cell grid," in *ACM SIGGRAPH 2011 papers*, SIGGRAPH '11, (New York, NY, USA), pp. 82:1–82:10, ACM, 2011.

[23] Cherif, M. and Gupta, K. K., "Planning quasi-static fingertip manipulations for reconfiguring objects," *IEEE Trans. on Robotics and Automation*, vol. 15, no. 5, pp. 837–848, 1999.

[24] Choi, M. G., Lee, J., and Shin, S. Y., "Planning biped locomotion using motion capture data and probabilistic roadmaps," *ACM Trans. Graph.*, vol. 22, pp. 182–203, 2003.

[25] Cole, A., Hsu, P., and Sastry, S., "Dynamic control of sliding by robot hands for ragrasping," *IEEE Trans. on Robotics and Automation*, vol. 8, no. 1, pp. 42–52, 1992.

[26] Coros, S., Beaudoin, P., and van de Panne, M., "Robust task-based control policies for physics-based characters," *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, vol. 28, no. 5, p. Article 170, 2009.

[27] Coros, S., Beaudoin, P., and van de Panne, M., "Generalized biped walking control," *ACM Trans. Graph. (SIGGRAPH)*, vol. 29, no. 4, pp. 1–9, 2010.

[28] Coumans, E., "Bullet physics engine." http://bulletphysics.org, 2005.

[29] da Silva, M., Abe, Y., and Popović, J., "Interactive simulation of stylized human locomotion," in *SIGGRAPH*, pp. 1–10, 2008.

[30] da Silva, M., Durand, F., and Popović, J., "Linear bellman combination for control of character animation," *ACM Trans. Graph. (SIGGRAPH)*, vol. 28, pp. 82:1–82:10, July 2009.

[31] de Lasa, M. and Hertzmann, A., "Prioritized optimization for task-space control," in *International Conference on Intelligent Robots and Systems (IROS)*, 2009.

[32] de Lasa, M., Mordatch, I., and Hertzmann, A., "Feature-based locomotion controllers," *ACM Trans. Graph. (SIGGRAPH)*, vol. 29, no. 4, pp. 1–10, 2010.

[33] D'Eon, E. and Irving, G., "A quantized-diffusion model for rendering translucent materials," in *ACM SIGGRAPH 2011 papers*, SIGGRAPH '11, (New York, NY, USA), pp. 56:1–56:14, ACM, 2011.

[34] d'Eon, E., Luebke, D., and Enderton, E., "Efficient rendering of human skin," in *Processdings of the Eurographics Symposium on Rendering*, 2007.

[35] ElKoura, G. and Singh, K., "Handrix: Animating the human hand," in *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 110–119, July 2003.

[36] Faloutsos, P., van de Panne, M., and Terzopoulos, D., "Composable controllers for physics-based character animation," in *SIGGRAPH*, pp. 251–260, Aug. 2001.

[37] Fang, A. C. and Pollard, N. S., "Efficient synthesis of physically valid human motion," *ACM Trans. on Graphics (SIGGRAPH)*, pp. 417–426, July 2003.

[38] Faure, F., Gilles, B., Bousquet, G., and Pai, D. K., "Sparse meshless models of complex deformable solids," *ACM Trans. Graph.*, vol. 30, pp. 73:1–73:10, August 2011.

[39] Fedkiw, R., Stam, J., and Jensen, H. W., "Visual simulation of smoke," in *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '01, (New York, NY, USA), pp. 15–22, ACM, 2001.

[40] Georgopoulos, A., Kalaska, J., and Massey, J., "Spatial trajectories and reaction times of aimed movements: Effects of practice, uncertainty and change in target location," *Journal of Neurophysiology*, vol. 46, pp. 725–743, 1981.

[41] Gershwin, S. B. and Jacobson, D. H., "A discrete-time differential dynamic programming algorithm with application to optimal orbit transfer," tech. rep., Harvard University, 1968.

[42] Gill, P., Saunders, M., and Murray, W., "Snopt: An sqp algorithm for large-scale constrained optimization," Tech. Rep. NA 96-2, University of California, San Diego, 1996.

[43] Gleicher, M., "Retargeting motion to new characters," in *SIGGRAPH*, pp. 33–42, July 1998.

[44] Goswami, A. and Kallem, V., "Rate of change of angular momentum and balance maintenance of biped robots," in *Proc. IEEE Int'l Conf on Robotics and Automation*, pp. 3785–3790, IEEE, 2004.

[45] Han, L. and Trinkle, J., "Dexterous manipulation by rolling and finger gaiting," in *ICRA*, pp. 730–735, IEEE, 1998.

[46] Harris, C. M. and Wolpert, D. M., "Signal-dependent noise determines motor planning," *Nature*, vol. 394, pp. 780–784, August 1998.

[47] Heck, R. and Gleicher, M., "Parametric motion graphs," in *Proceedings of the 2007 symposium on Interactive 3D graphics and games*, I3D '07, (New York, NY, USA), pp. 129–136, ACM, 2007.

[48] Hecker, C., Raabe, B., Enslow, R. W., DeWeese, J., Maynard, J., and van Prooijen, K., "Real-time motion retargeting to highly varied user-created morphologies," in *Proceedings of ACM SIGGRAPH '08*, 2008.

[49] Herr, H. and Popović, M., "Angular momentum in human walking," in *Journal of Experimental Biology*, 2008.

[50] Ho, E. S., Komura, T., and Tai, C.-L., "Spatial relationship preserving character motion adaptation," *ACM Trans. Graph. (SIGGRAPH)*, vol. 29, no. 3, 2010.

[51] Hodgins, J. K. and Pollard, N. S., "Adapting simulated behaviors for new characters," *ACM Trans. Graph. (SIGGRAPH)*, pp. 153–162, Aug. 1997.

[52] Hodgins, J. K., Wooten, W. L., Brogan, D. C., and O'Brien, J. F., "Animating human athletics," in *SIGGRAPH*, pp. 71–78, Aug. 1995.

[53] Hong, J., Lafferriere, G., Mishra, B., and Tang, X., "Fine manipulation with multifinger hand," in *ICRA*, pp. 1568–1573, IEEE, 1990.

[54] Horak, F. B. and Nashner, L. M., "Central programming of postural movements: Adaptation to altered support surface configurations," *Journal of Neurophysiology*, vol. 55, pp. 1369–1381, 1986.

[55] Huang, Z., Boulic, R., and Thalmann, D., "A multi-sensor approach for grasping and 3-D interaction," in *Computer Graphics International '95*, June 1995.

[56] Jacobson, D. H. and Mayne, D. Q., *Differential dynamic programming*. American Elsevier Pub. Co., New York, 1970.

[57] Jain, S. and Liu, C. K., "Interactive synthesis of human-object interaction," in *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 173–182, Aug. 2009.

[58] Jain, S. and Liu, C. K., "Controlling physics-based characters using soft contacts," *ACM Trans. Graph. (SIGGRAPH Asia)*, vol. 30, pp. 163:1–163:10, Dec. 2011.

[59] Jain, S., Ye, Y., and Liu, C. K., "Optimization-based interactive motion synthesis," *ACM Trans. Graph.*, vol. 28, no. 1, pp. 1–10, 2009.

[60] Jenkins, O. C. and Matarić, M. J., "Deriving action and behavior primitives from human motion data," in *IEEE/RSJ*, pp. 2551–2556, Sept. 2002.

[61] Joerg, S., Hodgins, J., and Sullivan, C., "The perception of finger motions," *Applied Perception in Graphics and Visualization (APGV)*, 2010.

[62] Kajita, S., Kanehiro, F., Kaneko, K., Fujiwara, K., Harada, K., Yokoi, K., and Hirukawa, H., "Resolved momentum control: humanoid motion planning based on the linear and angular momentum," in *Intelligent Robots and Systems*, pp. 1644–1650, 2003.

[63] Kavraki, L., Svestka, P., Latombe, J.-C., and Overmars, M. H., "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Trans. on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[64] Kim, J., Cordier, F., and Magnenat-Thalmann, N., "Neural network-based violinistâs hand animation," in *Conference on Computer Graphics International*, pp. 37–44, June 2000.

[65] Koga, Y., Kondo, K., Kuffner, J., and Latombe, J.-C., "Planning motions with intentions," in *SIGGRAPH*, pp. 395–408, July 1994.

[66] Komura, T., Leung, H., and Kuffner, J., "Animating reactive motions for biped locomotion," in *VRST '04: Proceedings of the ACM symposium on Virtual reality software and technology*, pp. 32–40, 2004.

[67] Kovar, L. and Gleicher, M., "Automated extraction and parameterization of motions in large data sets," in *SIGGRAPH*, pp. 559–568, 2004.

[68] Kovar, L., Gleicher, M., and Pighin, F., "Motion graphs," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 473–482, 2002.

[69] Kry, P. G. and Pai, D. K., "Interaction capture and synthesis," *ACM Trans. on Graphics*, vol. 25, pp. 872–880, Aug. 2006.

[70] Kuo, A., Donelan, J., and Ruina, A., "Energetic consequences of walking like an inverted pendulum: step-to-step transitions," *Exerc Sport Sci Rev*, vol. 33, no. 2, 2005.

[71] Kwon, T. and Hodgins, J. K., "Control systems for human running using an inverted pendulum model and a reference motion capture sequence," *The ACM SIGGRAPH / Eurographics Symposium on Computer Animation (SCA 2010)*, 2010.

[72] Lavalle, S. M. and Kuffner, J. J., "Rapidly-exploring random trees: Progress and prospects," Aug. 2000.

[73] Lee, S. J. and Popović, Z., "Learning behavior styles with inverse reinforcement learning," *ACM Trans. Graph. (SIGGRAPH)*, vol. 29, pp. 122:1–122:7, July 2010.

[74] Lee, Y., Lee, S. J., and Popović, Z., "Compact character controllers," *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, vol. 28, no. 5, 2009.

[75] Lee, Y., Wampler, K., Bernstein, G., Popović, J., and Popović, Z., "Motion fields for interactive character locomotion," *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, vol. 29, pp. 138:1–138:8, December 2010.

[76] Lee, Y., Kim, S., and Lee, J., "Data-driven biped control," *ACM Trans. Graph. (SIGGRAPH)*, vol. 29, no. 4, pp. 1–8, 2010.

[77] Liu, C. K., "Dextrous manipulation from a single grasping pose," *ACM Transactions on Graphics (SIGGRAPH)*, vol. 28, Aug. 2009.

[78] Liu, C. K., Hertzmann, A., and Popović, Z., "Learning physics-based motion style with nonlinear inverse optimization," in *SIGGRAPH*, pp. 1071–1081, 2005.

[79] Liu, C. K. and Popović, Z., "Synthesis of complex dynamic character motion from simple animations," *ACM Trans. on Graphics (SIGGRAPH)*, vol. 21, pp. 408–416, July 2002.

[80] Liu, L., Yin, K., van de Panne, M., Shao, T., and Xu, W., "Sampling-based contact-rich motion control," *ACM Trans. Graph.(SIGGRAPH)*, vol. 29, no. 4, pp. 1–10, 2010.

[81] Macchietto, A., Zordan, V., and Shelton, C. R., "Momentum control for balance," *ACM Trans. Graph. (SIGGRAPH)*, vol. 28, no. 3, pp. 1–8, 2009.

[82] Majkowska, A., Zordan, V. B., and Faloutsos, P., "Automatic splicing for hand and body animations," in *Eurographics/SIGGRAPH Symposium on Computer Animation*, 2006.

[83] Mandel, M., "Versatile and interactive virtual humans: Hybrid use of data-driven and dynamics-based motion synthesis," Master's thesis, Carnegie Mellon University, 2004.

[84] McCann, J. and Pollard, N. S., "Responsive characters from motion fragments," *ACM Trans. on Graphics (SIGGRAPH)*, vol. 26, Aug. 2007.

[85] Miall, R. C., Weir, D. J., and Stein, J. F., "Visuomotor tracking with delayed visual feedback," *Neuroscience*, vol. 16, no. 3, pp. 511–520, 1985.

[86] Miller, A. and Allen, P., "Examples of 3d grasp quality computations," in *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, vol. 2, pp. 1240 –1246 vol.2, 1999.

[87] Mordatch, I., de Lasa, M., and Hertzmann, A., "Robust physics-based locomotion using low-dimensional planning," *ACM Trans. Graph. (SIGGRAPH)*, vol. 29, no. 4, pp. 1–8, 2010.

[88] Mori, M., "The uncanny valley," *Energy*, vol. 7, no. 4, pp. 33–35, 1970.

[89] Muico, U., Lee, Y., Popović, J., and Popović, Z., "Contact-aware nonlinear control of dynamic characters," in *ACM Trans. Graph. (SIGGRAPH)*, pp. 1–9, 2009.

[90] Muico, U., Popović, J., and Popović, Z., "Composite control of physically simulated characters," *ACM Trans. Graph.*, 2011.

[91] Napier, J. R., "The perhensile movements of the human hand," *Journal of Bone and Joint Surgery*, vol. 38, pp. 902–913, November 1956.

[92] NaturalMotion, *Endorphin*, 2006.

[93] Neff, M. and Fiume, E., "Modeling tension and relaxation for computer animation," in *Eurographics/SIGGRAPH Symposium on Computer Animation*, pp. 81–88, July 2002.

[94] Newell, K., "Constraints on the development of coordination," in *Motor development in children: aspects of coordination and control* (Wade, M. and Whhiting, H., eds.), pp. pp. 341–361, 1986.

[95] Ngo, J. T. and Marks, J., "Spacetime constraints revisited," in *SIGGRAPH*, vol. 27, pp. 343–350, Aug. 1993.

[96] Pataky, T., Mu, T., Bosch, K., Rosenbaum, D., and Goulermas, J., "Gait recognition: highly unique dynamic plantar pressure patterns amonest 104 individuals," *Journal of the Royal Society Interface*, 2011.

[97] Pollard, N. S. and Zordan, V. B., "Physically based grasping control from example," in *ACM SIGGRAPH/Eurographics symposium on Computer animation*, pp. 311–318, 2005.

[98] Popović, M., Hofmann, A., and Herr, H., "Zero spin angular momentum control: definition and applicability," in *IEEE/RAS International Conference on Humanoid Robots*, 2004.

[99] Popović, Z. and Witkin, A., "Physically based motion transformation," in *SIGGRAPH*, pp. 11–20, Aug. 1999.

[100] Ren, C., Zhao, L., and Safonova, A., "Human motion synthesis with optimization-based graphs," *Computer Graphics Forum*, vol. 29, no. 2, pp. 545–554, 2010.

[101] Safonova, A., Hodgins, J. K., and Pollard, N. S., "Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces," in *SIGGRAPH*, pp. 514–521, 2004.

[102] Saygin, A. P., Chaminade, T., and Ishiguro, H., "The perception of humans and robots: Uncanny hills in parietal cortex," in *Proc. of the 32nd Annual Conference of the Cognitive Science Society*, pp. 2716–2720, 2010.

[103] Saygin, A. P., Chaminade, T., Ishiguro, H., Driver, J., and Frith, C., "The thing that should not be: Predictive coding and the uncanny valley in perceiving human and humanoid robot actions," *Social Cognitive Affective Neuroscience*, 2011.

[104] Shapiro, A., Pighin, F. H., and Faloutsos, P., "Hybrid control for interactive character animation," in *Pacific Graphics*, pp. 456–461, Oct. 2003.

[105] Shin, H. J. and Oh, H. S., "Fat graphs: Constructing an interactive character with continuous controls," in *Eurographics/SIGGRAPH Symposium on Computer Animation*, July 2006.

[106] Sims, K., "Evolving virtual creatures," in *SIGGRAPH*, July 1994.

[107] Sok, K. W., Kim, M., and Lee, J., "Simulating biped behaviors from human motion data," in *SIGGRAPH*, p. 107, 2007.

[108] Speyer, J. L. and Bryson, A. E., "A neighboring optimum feedback control scheme based on estimated time-to-go with application to re-entry flight paths," in *AIAA Journal*, vol. 6 of *5*, 1968.

[109] Stephens, B. and Atkeson, C., "Modeling and control of periodic humanoid balance using the linear biped model," in *Proc. IEEE Int'l Conf. on Humanoid Robotics*, 2009.

[110] Sueda, S., Kaufman, A., and Pai, D. K., "Musculotendon simulation for hand animation," *ACM Trans. on Graphics*, vol. 27, Aug. 2008.

[111] Sulejmanpašić, A. and Popović, J., "Adaptation of performed ballistic motion," *ACM Trans. Graph.*, vol. 24, no. 1, 2005.

[112] Theodorou, E., Todorov, E., and Valero-Cuevas, F., "Neuromuscular stochastic optimal control of a tendon driven index finger model," in *American Control Conference (ACC), 2011*, pp. 348 –355, 29 2011-july 1 2011.

[113] Thürey, N., Wojtan, C., Gross, M., and Turk, G., "A multiscale approach to mesh-based surface tension flows," *ACM Trans. Graph.*, vol. 29, pp. 48:1–48:10, July 2010.

[114] Ting, L. H., "Dimensional reducation in sensorimotor systems," *Computational Neuroscience*, vol. 13, pp. 103–136, Apr. 2007.

[115] Todorov, E., "Efficient computation of optimal actions," in *Proceedings of the National Academy of Sciences (PNAS)* (McClelland, J. L., ed.), vol. 106, July 2009.

[116] Todorov, E. and Jordan, M. I., "Optimal feedback control as a theory of motor coordination," *Nature Neuroscience*, vol. 5, no. 11, pp. 1226–1235, 2002.

[117] Torres-Oviedo, G. and Ting, L. H., "Muscle synergies characterizing human postural responses," *Journal of Neurophysiology*, vol. 98, pp. 2144–2156, 2007.

[118] Tournassoud, P., Lozano-Perez, T., and Mazer, E., "Regrasping," in *Proc. IEEE Int. Con. Robotics and Automation*, pp. 1924–1928, 1987.

[119] Tresch, M. C., Cheung, V. C., and d'Avella, A., "Matrix factorization algorithms for the identification of muscle synergies: Evaluation on simulated and experimental data sets," *Journal of Neurophysiology*, vol. 95, pp. 2199–2212, 2006.

[120] Treuille, A., Lee, Y., and Popović, Z., "Near-optimal character animation with continuous control," *ACM Trans. on Graphics (SIGGRAPH)*, vol. 26, Aug. 2007.

[121] Tsai, Y.-Y., Lin, W.-C., Cheng, K. B., Lee, J., and Lee, T.-Y., "Real-time physics-based 3d biped character animation using an inverted pendulum model," *IEEE Transactions on Visualization and Computer Graphics*, vol. 99, pp. 325–337, 2009.

[122] Tsang, W., Singh, K., and Fiume, E., "Helping hand: An anatomically accurate inverse dynamics solution for unconstrained hand motion," in *Eurographics/SIGGRAPH Symposium on Computer Animation*, pp. 1–10, 2005.

[123] Twigg, C. D. and James, D. L., "Many-worlds browsing for control of multibody dynamics," *ACM Trans. Graph*, vol. 26, no. 3, 2007.

[124] van de Panne, M. and Fiume, E., "Sensor-actuator networks," in *SIGGRAPH*, vol. 27, pp. 335–342, Aug. 1993.

[125] Wampler, K. and Popović, Z., "Optimal gait and form for animal locomotion," *ACM Trans. on Graphics*, vol. 28, no. 3, pp. 1–8, 2009.

[126] Wang, J. M., Fleet, D. J., and Hertzmann, A., "Optimizing walking controllers," *ACM Trans. Graph. (Proc. SIGGRAPH Asia)*, vol. 28, no. 5, pp. 1–8, 2009.

[127] Wang, J. M., Fleet, D. J., and Hertzmann, A., "Optimizing walking controllers for uncertain inputs and environments," *ACM Trans. Graph. (SIGGRAPH)*, vol. 29, no. 4, pp. 1–8, 2010.

[128] Whitman, E. and Atkeson, C. G., "Control of a walking biped using a combination of simple policies," in *IEEE Int'l Conf. on Humanoid Robotics*, 2009.

[129] Witkin, A. and Kass, M., "Spacetime constraints," in *SIGGRAPH*, vol. 22, pp. 159–168, Aug. 1988.

[130] Wojtan, C. and Turk, G., "Fast viscoelastic behavior with thin features," *ACM Trans. Graph.*, vol. 27, pp. 47:1–47:8, August 2008.

[131] Wooten, W. L., *Simulation of Leaping, Tumbling, Landing, and Balancing Humans*. PhD thesis, Georgia Institute of Technology, 1998.

[132] Wu, J. C. and Popović, Z., "Realistic modeling of bird flight animations," *ACM Trans. Graph. (SIGGRAPH)*, vol. 22, no. 3, 2003.

[133] Wu, J.-c. and Popović, Z., "Terrain-adaptive bipedal locomotion control," *ACM Trans. Graph. (SIGGRAPH)*, vol. 29, no. 4, pp. 1–10, 2010.

[134] Xu, J., Koo, T.-K. J., and Li, Z., "Finger gaits planning for multifingered manipulation," in *IROS*, pp. 2932–2937, IEEE, 2007.

[135] Xu, Z., Todorov, E., Dellon, B., and Matsuoka, Y., "Design and analysis of an artificial finger joint for anthropomorphic robotic hands.," in *ICRA*, pp. 5096–5102, IEEE, 2011.

[136] Yamane, K. and Hodgins, J., "Simultaneous tracking and balancing of humanoid robots for imitating human motion capture data," in *Int'l Conf on Intelligent Robots and Systems (IROS)*, 2009.

[137] Yamane, K., Kuffner, J. J., and Hodgins, J. K., "Synthesizing animations of human manipulation tasks," *ACM Trans. Graph*, vol. 23, no. 3, pp. 532–539, 2004.

[138] Ye, Y. and Liu, C. K., "Animating responsive characters with dynamic constraints in near-unactuated coordinates," in *SIGGRAPH Asia*, pp. 1–5, 2008.

[139] Ye, Y. and Liu, C. K., "Optimal feedback control for character animation using an abstract model," *ACM Trans. Graph. (SIGGRAPH)*, vol. 29, no. 4, pp. 1–9, 2010.

[140] Ye, Y. and Liu, C. K., "Synthesis of responsive motion using a dynamic model," *Computer Graphics Forum (Eurographics)*, vol. 29, no. 2, 2010.

[141] Yin, K., Cline, M. B., and Pai, D. K., "Motion perturbation based on simple neuromotor control models," in *Pacific Graphics*, p. 445, 2003.

[142] Yin, K., Coros, S., Beaudoin, P., and van de Panne, M., "Continuation methods for adapting simulated skills," in *SIGGRAPH*, pp. 1–7, 2008.

[143] Yin, K., Loken, K., and van de Panne, M., "Simbicon: simple biped locomotion control," in *SIGGRAPH*, p. 105, 2007.

[144] Yin, K., Pai, D. K., and van de Panne, M., "Data-driven interactive balancing behaviors," in *Pacific Graphics*, Oct. 2005.

[145] Zhao, L. and Safonova, A., "Achieving good connectivity in motion graphs," in *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '08, pp. 127–136, 2008.

[146] Zordan, V. B. and Hodgins, J. K., "Tracking and modifying upper-body human motion data with dynamic simulation," in *Conference on Computer Animation and Simulation*, Sept. 1999.

[147] Zordan, V. B. and Hodgins, J. K., "Motion capture-driven simulations that hit and react," in *Eurographics/SIGGRAPH Symposium on Computer Animation*, pp. 89–96, 2002.

[148] Zordan, V. B., Majkowska, A., Chiu, B., and Fast, M., "Dynamic response for motion capture animation," in *SIGGRAPH*, pp. 697–701, 2005.