ADAPTATION OF TASK-AWARE, COMMUNICATIVE VARIANCE FOR MOTION CONTROL IN SOCIAL HUMANOID ROBOTIC APPLICATIONS

A Thesis
Presented to
The Academic Faculty

by

Michael Joseph Gielniak

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Electrical and Computer Engineering

Georgia Institute of Technology
May 2012

ADAPTATION OF TASK-AWARE, COMMUNICATIVE VARIANCE FOR MOTION CONTROL IN SOCIAL HUMANOID ROBOTIC APPLICATIONS

Approved by:

Dr. Ayanna Howard, Advisor
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Andrea Thomaz, Co-Advisor
School of Interactive Computing
Georgia Institute of Technology

Dr. Patricio Vela
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. Lena Ting
School of Electrical and Computer Engineering
Georgia Institute of Technology

Dr. C. Karen Liu
School of Interactive Computing
Georgia Institute of Technology

Dr. Tom Habetler
School of Electrical and Computer Engineering
Georgia Institute of Technology

Date Approved: December 16, 2011

ACKNOWLEDGEMENTS

I would like to thank my advisor, Dr. Andrea Thomaz, for her constant support in my research. I am grateful for all the times when she gave me direction when I had none, for the countless occasions when she gave me feedback on my work, and for imparting upon me a research methodology that will stay with me for the rest of my life. I could not have asked for a better mentor, teacher, and guide.

I would also like to thank Dr. C. Karen Liu for her influence and guidance in my research. I am grateful for all the moments that we shared, discussing ideas, theories, and possible research directions. I will never forget that she took the time to care, even when she had nothing to gain.

I would like to thank the members of my dissertation committee: Drs. Ayanna Howard, Lena Tina, Tom Habetler, and Patricio Vela for their many helpful suggestions and support.

Finally, I would like to thank God for giving me the time and resources to follow my dreams. I am grateful for all the prayers that were answered along the way.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS OR ABBREVIATIONS

3-D	3-dimensional.
AE	All 13 motions exaggerated.
AIVV	Average Inter-Variant Variance.
ANOVA	Analysis-of-variance.
ASDOM	Average Square Deviation from the Original Motion.
AU	All 13 motions unexaggerated.
CL	Cartoon-like.
cm	centimeter.
CNR	Cumulative Energy Ratio.
DCGI	Dynamically Consistent Generalized Inverse.
DOF	degree–of–freedom.
DOFs	degrees–of–freedom.
DTW	Dynamic Time Warping.
EU	7 exaggerated motions then 6 unexaggerated.
EX	Exaggerated Motion Version.
F&F	Feedforward & Feedback Control.
HL	Human-like.
HNV	Hand Normal Vector.
HRI	Human-Robot Interaction.
IK	Inverse Kinematics.
KSE	Kolmogorov-Sinai Entropy.
LB	Liked Best.
LQ	Linear Quadratic.
LQR	Linear Quadratic Regulator.
MC	Mimicking Coordinated.
MH	Mimicking Human.
Mo-Cap	Motion Capture.

MR	Mimicking Retargeted.
NVBM	Number of Views Before Mimicking.
OC	Original Coordinated.
ODE	Open Dynamics Engine Simulation.
OH	Original Human.
OR	Original Retargeted.
OSC	Operational Space Control.
OV	Only the variance algorithm (unconstrained).
PCA	Principal Component Analysis.
PCL	Percentage Correctly Labeled.
PID	Proportional Integral Derivative.
post-opN	post-optimization (N = downsampling rate).
pre-opN	pre-optimization (N = downsampling rate).
PU	Participant Unconstrained.
RL	Robot-like.
RWTSN	Random White Torque-Space Noise.
SA#	Short Answer (# = question number).
SC	Spatial Correspondence.
SIK	Style-based Inverse Kinematics.
SIL	Simulation-in-the-Loop.
SIMM	Software for Interactive Musculoskeletal Modeling.
slerp	Spherical Linear Interpolation.
ST-Isomap	Spatiotemporal Isometric Mapping.
STC	Spatiotemporal Correspondence.
SVD	Singular Value Decomposition.
TAV	Task-Aware Variance.
TC	Temporal Correspondence.
TMB	Thought they Mimicked Best.
UE	7 unexaggerated motions then 6 exaggerated.

UN	Original Unexaggerated Motion Version.
VB	Variance algorithm with both internal & external constraints.
VI	Variance algorithm with internal constraints.
VP	Most Visually Pleasing.
WB	With the human-like optimization & constraints.
WC	With only constraints.
WH	With only the human-like optimization.
w/o	without.
w.r.t.	with respect to.

SUMMARY

Social robots are being developed for a variety of applications where they must interact with people, such as partners in factories or assistants for the elderly. For these applications, to avoid boredom and frustration for human partners, collaboration should proceed at comfortable speeds so that human partners are not waiting for robots to complete their actions. Robot and human turns in collaborative interaction should overlap when necessary to help facilitate smoother interaction. To reduce waiting time and transition time between human and robot turns, partners will need the ability to synchronize actions and respond quickly and fluidly to each other. This can be accomplished through improved communication between robots and humans. Robots need to clearly and transparently communicate intent and expectations to their partners so that collaboration with robots will be intuitive for human partners. To produce intuitive interaction, the field of natural human-robot interaction demands that when robots communicate with humans (i.e. human-robot interaction), they should communicate in the same manner that humans communicate with each other in human-human interaction. This applies to all possible communicative channels, including robot motion.

In social interactions, humans communicate with each other through their motion (e.g. gestures, changing eye gaze, reactions) [1, 2]. Social robots that interact with humans should also communicate with their partners via motion. Misclassification of motion information by human partners occurs when the communicative motion is too subtle, conflicting with other cues and channels, or displayed in an unexpected manner [3, 4]. Therefore, communication in robot motion must be added in a specific way to be interpreted correctly and benefit the interaction.

In this thesis, I explore the interaction benefits gained when robots communicate with their partners using a familiar mode: robot motion that is human-like. To achieve this, I have two concrete goals: (1) synthesize robot motion that is more human-like, and (2)

show that communicative robot motion has benefits for interaction with human partners. These two goals are motivated by the limitations with existing motion generation methods in robotics, such as heavy data dependence, and the problems caused by the use of these existing techniques for human-robot interaction, such as long wait times, ambiguous motions, and unsynchronized collaboration.

Unfortunately, the majority of existing motion generation techniques for social robots do not produce human-like motion. For example, retargeting human motion capture data to robots does not produce human-like motion for robots because the degrees-of-freedom differ in number or location on the kinematic structures of robots and humans. Also, in the rare instances when retargeting human motion to robots works well, it produces only one motion trajectory, rather than a variety of trajectories, which makes the robot move in a very repetitive way.

After presenting algorithms for the synthesis of three methods of communication in motion, I develop two algorithms that focus on explicitly making robot motion more human-like. When motion is more human-like, the motion is more familiar to human partners, and they can more easily identify the robot motion correctly. In order to measure success, I present and validate a metric that is used to measure human-likeness of motion, so that robot trajectories can be evaluated quantitatively, allowing for simple comparison of motion on the same robot or between multiple robots with different kinematics or dynamics.

When synthesizing communicative motion or modifying trajectories to appear more human-like, the tasks the robot must accomplish in the world must not be disrupted and real-world interaction should not be hindered (i.e. motion should be task-aware). In the literature, motion control is comprised of joint-coordination and planning [5]. Joint-coordination involves identifying and implementing programs to move, which is what my two prior goals (i.e. communication and human-like synthesis) accomplish. On the other hand, planning develops algorithms for combining motions for autonomous interaction with a dynamic environment, such as obstacle avoidance and manipulation . Thus, the last sections of my thesis are devoted to integration of the algorithms (i.e. composing motions)

and handling task-based constraints so the social robot can interact with the real-world.

My thesis discusses the results of 31 experiments, balanced between quantitative analysis and user studies, cumulatively involving 1,624 participants, and over 203,000 data points to support the interaction benefits of using my comprehensive algorithm to create task-aware, communicative, human-like motion for social robots. This large quantity of data is driven by the large number of experimental conditions per study and the expected variance of subjective opinions being large. Both these criteria lead to the requirement of many participants to achieve significant differences.

Discussion of all the results is distributed throughout the thesis, near the appropriate sections of the algorithm, but for example, my thesis concludes that the addition of three specific methods of communicating via motion (i.e. secondary motion, exaggeration, and anticipation) allow human partners to more quickly and more accurately tell what the robot is doing, feel more engaged in the interaction, and remember the interaction more accurately. Furthermore, communicative motion is shown to be more human-like and can be harnessed to direct the attention of the human partner.

CHAPTER I

INTRODUCTION

A social robot interacts with humans on a regular basis as a part of its functional goals, and often social robots are partners for humans in collaboration. Social robots are beneficial to humans for helping complete tasks, such as assembly in factories or assistants for the elderly, and therefore, these robots must leverage human social cues, such as eye gaze and gestures, to communicate with humans during cooperative tasks.

Specifically, my research focuses on a subset of communicative social cues for humanoid robots, i.e. those which can be communicated through hand and body motion, and the benefits they provide to human-robot interaction (HRI). My research establishes a systematic methodology for the creation and composition of communicative motion for humanoid robots that interact socially with humans. Novel algorithms for three specific types of communicative motion are set forth: secondary action, exaggeration, and anticipation. Facial gestures and expressions are not considered in my work, since they comprise a separate research problem.

However, synthesizing communicative motion does not completely solve many of the existing problems with robotic motion generation techniques. The problems will be discussed in detail in Section 1.1, but one main problem with existing techniques is that they do not account for motion recognition, which is essential for robots that interact with humans. For example, if the human partner cannot identify what action the robot is performing, they may pause to wait until this action become clear, or they may disrupt the robot in its task because they misinterpreted the robot action. These types of errors in human-robot interaction reduce the amount of time that interaction turns overlap. Thus, interaction efficiency and speed will decrease when social robot motion is unrecognizable to human partners. Roboticists may compensate for ambiguous robot motions in a controlled experimental environment by establishing conditions that help disambiguate, such

as spreading interaction objects far apart on a table to clarify pointing. However, these controlled conditions are unrealistic, and an effective robot motion synthesis technique must facilitate recognition of motion.

The easiest way to generate recognizable robot motion is to make robot motion familiar to humans by making it similar to human motion. My research identified a metric that can be used to synthesize and evaluate humanoid robot motion with respect to human-likeness. The metric is discussed in the framework of motion control, using quantitative and qualitative data from experiments to demonstrate the benefits of robot motion optimized according to this metric.

Furthermore, since humans exploit redundancy (i.e. having more than one way to accomplish something) to provide flexibility and overcome obstacles in the world, social robots can also benefit from possessing this capability. Thus, a novel algorithm is presented for the creation of an infinite number of motion variants from a single input exemplar. The amount of variance achieved in the resultant motions is quantified to establish the amount of difference between variants and the original motion.

The requirements for social robots include traditional interaction with the world, such as grasping objects. However, social robots have additional, non-traditional constraints, such as eye gaze and synchronization, that exist because they have social partners. In this thesis, I extend conventional projection algorithms to produce an even more powerful approach that respects physical and timing constraints, both static and dynamic.

Additionally, the benefits of each of these types of motion are established through experimental research with human participants using both qualitative and quantitative data. 31 experiments, balanced between quantitative analysis and user studies, cumulatively involving 1,624 participants and over 203,000 data points were performed to demonstrate the benefits to interaction between human and robots, when using my algorithms to generate robot motion. This large quantity of data is driven by the large number of experimental conditions per study and the expected variance of subjective opinions being large. Both these criteria lead to the requirement of many participants to achieve significant differences.

Finally, I demonstrate how to compose all the algorithms into a single algorithm, which is capable of providing all the functionality of the individual algorithms and transforming trajectories from any source (e.g. observed exemplars, animated trajectories, motion-captured projections, kinesthetically-taught motions). The following section will motivate why communicative, human-like motion generation for social robots is a challenging research problem.

1.1 Motivation

This research is motivated by the fact that problems exist with the techniques that are used to create motions for social robots. Those problems adversely affect human-robot interaction. Therefore, if I can overcome these problems, I will increase the robustness of interaction between a social robot and a human. Before I can describe the problems, I need to provide some background, so it will be easier to understand why certain motions are problematic.

1.1.1 Robust Interaction

A system is defined as anything that is connected or dependent, possibly working together, to form a complex whole. Systems are characterized by states, which identify their condition at any point in time. Each element within a system has one or more states because the parts of a system are often complex entities, such as a human or a robot. States of these dependent parts contribute to the overall system state. For example, a human interacting with a robot is a system. The human and the robot in this example can be defined by states that describe operating conditions, intentions, and many other variables. The objects involved in the interaction and the environment also have states such as mass or temperature. And the overall interaction has a state, which could be defined as progress toward the goal, for example. Since the two agents involved in the interaction are complex systems, complex dynamics describe how the interaction state evolves as a function of time for the cooperative task.

An interaction would be defined as more robust when it can handle more perturbations and more deviations and still yield success. The definition of a successful interaction is

not important for this discussion, but only that however success is measured, it does not deviate or vary significantly in a robust interaction between trials. In other words, the characteristic behavior of the interaction and the results are stable under wide variance of conditions.

Open-loop systems are highly subject to instability. However, traditional human-human interaction is not conducted in an open-loop manner (i.e. it is a robust system). Human communication is multimodal, involving multiple simultaneous feedback channels (e.g. speech, motion, attention), which results in increased transparency and observability of state information. Communication is the feedback mechanism by which interaction proceeds in a stable and useful way between partners. The use of multiple channels by humans is a form of redundancy to reinforce the message being communicated. Therefore, having a variety of different ways to communicate is a motivation for flexibility in robot motion (i.e. robots should be able to perform the same motion in more than one way).

Intuitively, when a human is interacting with a robot that does not display the expected multimodality, the interaction will not proceed as robustly since some of the communication is absent (as compared to human-human interaction). In fact, evidence and experience have shown that effectiveness of interaction increases when a machine is endowed with human-like traits, such as the ability to communicate with humans through speech or gesture [6, 7, 8]. Furthermore, anticipation of world states and earlier knowledge of collaboration partners' actions enable better teammate timing, which implies that robot actions must clearly and transparently communicate state information and expectations to partners [9, 10].

Stability and robustness of a system increase as more feedback loops are added, if all those feedback signals reinforce each other, communicating messages that produce a common response from the system. These increase the confidence that the system is measuring accurate data. Many social robots already employ speech and attention as feedback channels for communicating with human partners, and thus, they are not open loop systems. But there is a lot of room for improvement in human-robot interaction, especially in the domain of robot motion. I will discuss in greater detail in Section 1.1.3.6 that

motion is a largely under-utilized channel of communication in social robots. Therefore, I generate flexible, varied, communicative robot motion in order to demonstrate the benefits for interaction when humanoid robots move more similarly to their human partners.

1.1.2 Motion

A trajectory is defined as a variable or set of variables ordered in time. For example, position, velocity, acceleration, and torque are state variables that are often ordered as trajectories.

Humans and robots are systems that possess degrees-of-freedom (DOFs), such as joints, that allow their bodies to change position in a particular direction. State of a degree-of-freedom (DOF) can be represented by position, and any change in position state is motion. Thus, motion is often represented in the form of trajectories, which are a time-ordered sets of states. The term trajectory can be used to refer to a single DOF, a subset, or all DOFs that comprise a system that can undergo a change in position state (i.e. a function of time). The application of any generalized force to a body can result in motion. For example, actuators, such as motors or muscles, control the position of degrees-of-freedom between states by converting energy to motion.

1.1.3 Problems With Existing Motion Control in HRI

Motion is a simple concept, so why then is it so difficult to generate well for social robots? The problems that exist with current motion generating techniques for social robots motivate why I want robot motion to clearly communicate information such as intent or state variables.

1.1.3.1 Artifacts

Artifacts are undesirable bi-products of the motion control strategy. Some existing techniques that create robot motion induce artifacts in the output, such as self-collision or discontinuities in motion trajectories, that need to be resolved using pre- or post-processing steps to remove the artifacts [11, 12]. Artifacts are problematic because they stand out from the trajectory and tend to draw attention or the observer eye, such as unsmooth transitions

between states. A tic or a tremor in human motion would be considered artifacts. Robot actuators can have reduced life if they control to trajectories that contain artifacts for long periods of time; in other cases, such as self-collision, damage can occur by executing the trajectory just once.

Interpolation between two existing motions can create artifacts in the output, since the interpolated motion can cause the limbs of the robot to penetrate the body (i.e. self collide). Retargeting is another example of an algorithm that creates artifacts in the resultant trajectory, especially when the constraint data from motion capture (Mo-Cap) is noisy. Retargeting algorithms are usually accompanied by a smoothing process to remove the artifacts (see Section 2.4 for more information).

1.1.3.2 Strategies Don't Generalize

Another problem that many robot motion control strategies suffer from is the inability to easily generalize. This most commonly occurs when a model is created for one specific motion task, e.g. walking [13], and the model cannot apply easily to other tasks. In the case of the example walking robot, the complex dynamic equations are written only for one robot, so new equations must be written and solved to apply the model to other robots.

1.1.3.3 Data Dependence

A very common problem with robot motion creation, especially when attempting to synthesize human-like trajectories, is heavy data dependence. Databases are filled with robot trajectories, which are often created by retargeting human motion capture data to robot kinematic hierarchies optimally. The motions are annotated so that they can be recalled by a label (e.g. waving gesture) when they must be executed on the hardware. Variants are created from the database by blending the exemplars. However interpolation does not guarantee that human-likeness of the resultant motion will be preserved [14, 15].

When motion capture data is used, often dynamics are excluded (in favor of proportional integral derivative (PID) control) due to lack of accurate knowledge about robot parameters for dynamic equations (e.g. mass, inertia tensors). For example, most robot manipulation occurs at unrealistic velocities different than human manipulation since other

problems are being solved in manipulation [16]. However, the motion appearance has a huge impact on human interaction with robots in joint manipulation tasks such as hand-offs, which is why I am motivated to design human-like social robot motion.

1.1.3.4 Abstraction of State and Control

Many algorithms that create robot motion abstract the state variables away from the control variables. This means that the actuators require one variable (e.g. torque), but motion is generated from a model that operates on another variable, such as equations of motion. In other cases, data is stored abstractly, such as in a tree or graph structure [17]. Thus, abstraction requires transformation of data before motion can execute on the hardware. As long as this data transformation can be accomplished in real-time, abstraction is not an issue. However, for many algorithms, real-time operation is not feasible because the transformation from state to control variables takes too long [18, 19].

1.1.3.5 Manual Tuning

Some algorithms that generate robot motion also require manual user intervention to tune gains or adjust parameters. The ideal algorithm employs autonomous techniques to populate gains and parameters, so that behavior of the algorithm is consistent and the output does not vary based on the skill level of the person using the algorithm [20, 21].

1.1.3.6 Not Designed For Communication

The human body is a powerful representational tool, optimized for communicating spatial reference, demonstration, disambiguating speech, inquiring for feedback, influencing others' behavior, and directing attention [1, 2]. Zimmer hypothesizes that gesticulation was the original hominid language [22]. Even today, non-verbal communication constitutes approximately 60% of human interaction, and meaningful motion accompanies 75% of discourse [23]. When designing humanoid robots that interact with human partners, ideally the robots also capitalize on this advantage by possessing the ability to effectively use motion as a communication channel. However, the majority of existing methods for motion control of social robots ignore motion as a communication channel.

I believe that it is inappropriate to ignore motion as a communication channel. Humans naturally assign internal states, such as intent, to inanimate objects, such cartoon or video game characters, when those objects move by task-directed motion [24, 25, 26]. Thus, it is logical that robots will also be perceived to possess intent, when they are interacting with human partners toward a joint goal. Furthermore, one study on human motion found that nonverbal behavior cannot be regulated to convey no impression because observers ascribe intention to expression regardless of neutrality of the face and body [27]. These findings suggest that social robot motion must be carefully constructed to communicate the intended impression, otherwise the message that is inadvertently communicated by ignoring the motion channel can conflict with other social cues such as speech and confuse human partners. One study found that conflicting social cues in different channels of communication are reinterpreted and reconciled differently by the social partner [28]. By not designing the appropriate non-verbal behavior that complements and reinforces the desired message, adverse effects can result in the interaction, such as long wait times and misinterpretation of partner actions.

Numerous additional benefits exist by designing social robot motion to be communicative. Display of appropriate non-verbal communication behaviors results in increased perception of agent realism [29]. Multimodality increases attribution of social communicative mechanisms to the agent, such as social responses, communicative intent, and internal states [7, 30]. Therefore, to achieve more robust interaction, social robots must exploit every opportunity to increase the signal-to-noise ratio of communication signals expressed during collaborations with human partners. Social robots can do so by utilizing behaviors that are socially relevant to the people with whom they interact and increasing the transparency and observability of their own state information [31].

Robots should communicate with humans as humans communicate with each other. This is called natural human-robot interaction [32, 33]. Conveying information in a familiar and intuitive way in interaction may even be the key to widespread acceptance of robots beyond the domains of education and entertainment [34, 35, 36]. If robots become future collaborators, service workers, guides, instructors, and personal helpers then costs are

lower and less training is necessary when meaningful interaction can begin immediately because robot communication is embedded in user-friendly format [37, 38].

1.1.3.7 Lack of Realism

Retargeting human motion to social humanoid robots is a popular way to create trajectories for control of social robots. However, due to the DOF correspondence problem, for many social robots this does not create robot motion that appears human-like. The DOF correspondence problem is a mismatch between human and robot degrees-of-freedom in number or location on the kinematic hierarchy. Thus, when projecting human motion capture data onto the robot, information is lost, and that lost information causes the human motion to look bad when executed on the robot. The projection by retargeting cannot always compensate for missing degrees-of-freedom. For example, a human's ball-and-socket shoulder DOF translates. This 3-dimensional (3-D) translation cannot be captured by a robot that has no translatable DOFs. Thus, even the very best algorithms which begin with human motion as the input do not always guarantee realism.

Subjectivity is a significant challenge in the creation of human-like motion. Many researchers suggest that because motion is a communicative signal, it is highly subject to scrutiny. Even if observers cannot isolate the cause, the human eye is very sensitive when motion appears differently. For example, artificial limbs, limps, or kinematics that are not symmetric can cause walking to appear differently in humans [34]. On virtual characters, dynamic simulation with unrealistic parameters (e.g. gravitational acceleration constant) can cause motion to appear differently [39].

The most significant challenge to generating human-like motion is that consensus has not been achieved for definitions of subjective terms such as "human-like." However, since humans agree that motion can be used as a communication mechanism, my work demonstrates how communicative motion can positively impact human-robot interaction task performance, and success is measured by both qualitative and quantitative means to reinforce the conclusions made regarding the benefits of these algorithms.

1.1.3.8 Corrupting Interaction

The process of creating quality motions that satisfy the demands of natural human-robot interaction is difficult. The process is hard because the requirements are very stringent: (1) Interaction requires synchronization between channels of communication, and therefore, the motion control process must be able to handle constraints that synchronize timing. (2) The intent of the robot's actions must be very clear to collaborative partners, so they are not confused by gestures. (3) Ideally, interaction is a collaboration rather than a set of serial events (i.e. human and robot taking alternating, non-overlapping turns) because such discrete, independent turns between partners disrupt the natural flow of an interaction, causing it to take longer to accomplish the joint task. Motion control fails when the participant has to pause during an interaction to watch the robot motion to figure out what the robot is doing. (4) Gestures that disambiguate or reference spatially, such as gazing or pointing, need to be accurate, which suggests that precise control of the body is also a design requirement. When precise control does not exist, often compensation is required. For example, the workspace can be made much larger and objects are placed at larger distances apart to eliminate confusion and facilitate separability for the human partner when the robot points at or gazes at an object. (5) Furthermore, many collaborative tasks require rapid response and reaction timing between partners; therefore, long wait times for slow robot motions further corrupt interaction. In short, motion control for social robots is difficult because it has a very complex set of requirements dictated by natural human-robot interaction.

Existing techniques have advantages and disadvantages, but failure of an algorithm in any area that detracts from the fluidity of human-robot interaction is not acceptable. No motion control strategy that exists in the literature satisfies all the necessary functionality that produces a seamless interaction for social robots. And therefore, I was motivated to design my own motion control technique.

1.1.3.9 Repetitive Motion

Highly dynamic and unpredictable environments require flexibility in motion, which suggests that motion variance is a design requirement for robots. Existing techniques have many problems. For example, the majority of motion generation techniques create repetitive motions, which are not realistic; predefining trajectories for all potential motions and motion variants that social robots will require in the world is not feasible (e.g. with respect to memory storage); and conventional motion measurement devices (e.g. motion capture) create data that is subject to a retargeting problem (i.e. DOF correspondence) when applied to robots with different kinematic hierarchies [40].

Humans never move in exactly the same way twice. And the environment is not the only variable that induces variance in motion. Thus, repetitive motion is not human-like. Since natural HRI demands that human-robot interaction be the same as human-human interaction, then robot motion must also be non-repetitive. Even sophisticated techniques such as retargeting produce only a single trajectory, and therefore, do not meet the demands of natural human-robot interaction.

1.2 Approach Overview

Given all the above motivations, I present algorithms for adding communication to social robot motion that were inspired by the principles of animation [41, 42], which help cartoons communicate with their audience. These inspirations are discussed in detail in the accompanying sections of this thesis that outline my communicative motion algorithms. I generate human-like robot motion for HRI and measure the effects of these motion changes upon interaction success. These algorithms overcome limitations, such as heavy data dependence, and are capable of human-like motion synthesis regardless of kinematic differences between humans and humanoid robots.

Specifically, within my thesis, I provide evidence that communicative, human-like robot motion is easier for human partners to correctly tell what action the robot is performing. Since the motion is more human-like and communicative, they can perform this identification more quickly and more accurately identify motion, even in the absence of missing

context. The communicative motion output from my algorithm, which has an improved appearance, allows humans to more accurately predict robot state. Furthermore, coordination between human and social robots increases due to increased state transparency, which allows human partners increased time to react. Fluidity of interaction increases because there is greater redundancy in information transfer (i.e. motion exploited as a communication channel), and communicative signals exhibit less error (i.e. are incorrectly interpreted less often) between collaborative partners.

At a high level, I am motivated by the insufficiencies discussed in Section 1.1. However, further motivation accompanies each of the subsequent algorithms to give more in-depth insight into the reasons that each of the specific algorithms were included as part of the overall process to create communicative, task-aware, human-like motion for social robots.

1.3 Assumptions

The following assumptions were made to limit the scope of the research so significant contributions could be made to a tractable problem during the years when this research was being performed.

1.3.1 Environmental

At least three specific distinctions can be made in planning algorithms with respect to the environment in which the robot moves. The most trivial assumption is that free space is infinite, and no collisions exist in the world. A robot does not even need to worry about self-collisions, under this first assumption. The second domain assumption on free-space moves closer to the ideal situation, where a robot must interact with real-world, and therefore only a certain subset of objects in the world become salient to the robot. Employing this second assumption enables the robot to perform manipulation tasks. The third assumption is the most restrictive, in which the robot must account for highly-dynamic, time-varying environments where all obstacles must be avoided, including the robot's own body. Under the third assumption, the robot must account for any sensor inaccuracy; thus redundancy in sensing becomes important.

My work is capable of overcoming both the first and second assumptions on free-space,

and a significant percentage of situations that occur within third assumption are also taken into account by the algorithm set forth in this thesis. However, my work focuses upon improving local motion for HRI scenarios. Planning a path in a cluttered environment is navigation, which is a separate research problem that focuses upon global motion (i.e. over larger distances). Part of the reason that path planning wasn't addressed to a greater extent is due to the hardware platform available for the research, which leads to the next assumption.

1.3.2 Globally Static Hardware Platform

As opposed to global motion, local motion does not move the center of mass of the robot over large distances. All the work presented in this thesis is performed on a single hardware platform (see Section 2.5), which does not have a locomotive mechanism, such as legs or wheels, that can be used to translate its center of gravity large distances. Due to my upper-body hardware platform, the assumption of local motion was implicit in my research, which means that I cannot make any claims regarding performance of my algorithms on robots with locomotive means (e.g. wheels, legs). However, some of the algorithms presented herein were implemented in computer animation and simulation contexts, where they demonstrated very high levels of success on virtual characters that had legs and could globally translate. Since these results are beyond the scope of this thesis, no claims will be made about the applicability of my algorithms to robots' legs, wheeled bases, or other locative means.

1.3.3 Humanoid Robots

No claims can be made about the extension of the results of my research to non-humanoid robots, which includes the earlier discussion of wheeled robots. The assumption of a humanoid kinematic hierarchy was made to minimize the DOF correspondence problem. If success cannot be achieved where effects of this problem are smallest, benefits are more difficult to show on architectures that are non-humanoid. After benefits have been demonstrated for HRI and human-like motion is better understood in the humanoid domain, future work can study what human-likeness means (and how my algorithms generalize)

to other robots that have architectures that do not as closely fit the categorical descriptor of "humanoid."

1.3.4 HRI Focus

This thesis focused on benefits of human-like motion for HRI scenarios. Certainly, benefits exist beyond this domain, but they have not been explored because of the context that I selected for the scope of my thesis. An HRI focus as my research context is also part of the reason that the third domain of environmental constraints was not completely handled. Many human-robot interaction scenarios do not proceed at a very rapid pace (e.g. assisting the elderly, guiding a tour). Conversely, if the scope of my work had been robots that play contact sports with human athletes, such as football or soccer, interaction would have very different timing requirements with respect to communication and constraints.

1.3.5 Single Exemplar

Ideally, human-like motion could be created without any representative exemplars of any motions. However, throughout all my research I assumed that the robot had at least one exemplar of each motion type (e.g. wave, beckon). My research makes no assumptions about how the robot acquired this exemplar (e.g. the robot can observe this exemplar), but at a minimum one trajectory must exist. This is a reasonable assumption because humans also maintain a representation of each motion type. For example, humans can classify a wave when they observe one, or they can generate a representative wave upon request, provided that they have previously associated at least one motion to the label "wave." This is the same condition that I place upon use of my motion-generating algorithm. I.e. elsewhere in the robot control hierarchy there must be a process that receives the command for which motion is currently being requested, and delivers that trajectory to my algorithm. I make no assumptions about how this other process accomplishes its task (e.g. database, observation, learning by demonstration). My achievement of human-like motion production with a single exemplar alleviates the data dependence problem significantly over databases of motion that encompass a motion repertoire comprised of multiple exemplars of each motion type.

1.4 Structure & Format of This Thesis

The research question that is analyzed in this thesis is: how does communicative, human-like motion affect interaction in human-robot collaboration? Thus, given a robot motion, I want an algorithm to modify it to make it more communicative, to improve the interaction. To be useful for real robots, the algorithm must also be capable of creating continuous motion, given that new motions are provided in a serial manner. My focus is on social robots, and I must limit my conclusions only to humanoid robots, since these are the only robots on which I have only validated results. I consider facial expressions to be a different form of communication than hand and body gestures, and therefore, I limit my concentration only to the latter in this thesis.

Before I can analyze benefits of human-like, communicative motion for social humanoid robots, this motion must be produced. Figure 1 demonstrates how I accomplish this task. The diagram in Figure 1 will be refined in subsequent sections of this thesis to help identify how the individual algorithms contribute to the larger goal of my work.

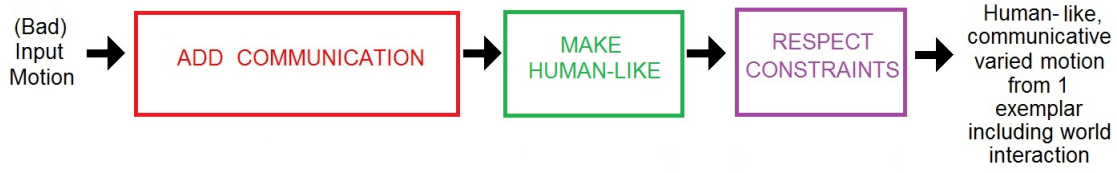

Figure 1: My vision for producing communicative, human-like motion.

First, in Chapter 2, theoretical background is provided on five subjects: operational space control (OSC), optimal control, Kolmogorov-Sinai Entropy (KSE), motion capture, and the hardware that was used to produce the results of this thesis. This background is provided as reference so that concepts will be clearer when they are discussed in detail in the sections that reference this background information. Operational space control will help clarify how constraints are satisfied; optimal control is useful to help understand how the human-like optimization and variance algorithms produce successful results; Kolmogorov-Sinai entropy background is provided so that it becomes easier to understand

why the human-like optimization works; motion capture and a discussion of the DOF correspondence problem are provided to aid in understanding retargeting and why its results are less than perfect; and finally, the hardware platform is discussed to provide background specifications of the robot used to produce results in this thesis and motivate some of the assumptions made in algorithm or experimental design.

Figure 2: My algorithm for generating communicative, human-like motion.

Chapters 3-5 are devoted to the three goals outlined in Figure 1: communication, human-likeness, and constraints, respectively. The subdivisions within each of these are shown in Figure 2, where one section or chapter is allotted one major algorithm.

In Chapter 3, three methods of generating communicative motion are presented: anticipation, exaggeration, and secondary motion. Each algorithm is accompanied by a set of user studies and quantitative analyses to demonstrate the benefits that each of these algorithms provide to human-robot interaction. Specifically, the anticipation algorithm benefits human-robot interaction by generating a variant of the original motion in which derivative information is preserved and the state that yields highest labeling accuracy occurs earlier, so that human partners are aware of robot intent sooner. The end result is increased human partner reaction time to anticipatory robot motion. The exaggeration algorithm benefits human-robot interaction by increasing the contrast between the task or intent of the motion (i.e. the primary motion) and the rest of the robot body. Exaggerated motion is shown to direct human partner gaze, increase the partner's perceived engagement in the interaction, and help partners to remember interaction details better. Secondary action provides the physics-based response to the task-based primary motion trajectory, so that the entire body moves in a manner consistent with Newton's third law of motion. Secondary motion benefits human-robot interaction by making robot trajectories

more human-like (both quantitatively and qualitatively). It also fills in missing context while communicating state parameters, so that robot and world state are more transparent to the human partner. As mentioned in the introduction and motivation (i.e. Chapter 1 and Section 1.1, respectively), these effects produce many collaborative benefits, such as more fluid human-robot interaction that proceeds at a pace closer to human-human interaction.

In Chapter 4, I present a metric for the synthesis and evaluation of human-like robot motion. This quantitative metric eliminates the subjectivity involved in evaluating trajectories to determine which appears more human-like. The metric reduces human-likeness to a single number so that comparisons can be made between robots, people, or any entity that can be represented as a hierarchical chain of degrees-of-freedom. My metric not only provides a framework to evaluate success of generating human-like motion without performing user-studies, but it also can generate a more human-like version of a motion, given a single input motion. Success of this metric as both a synthesis and an evaluation tool is quantified through subjective and objective data. Chapter 4 also discusses a technique for the generation of an infinite number of human-like motion variants from a single input motion, so that social robot motion is not repetitive. The variants are shown to be human-like, while maintaining the original motion intent throughout modulation. The algorithm is shown to generate similar and dissimilar variants frequently, so that a large portion of the task-space is represented (i.e. the algorithm produces motions that are always classified as the same motion type as the input, no matter how much variance is introduced).

In Chapter 5, operational space control is extended to handle timing constraints and synchronization constraints necessary for social robots. The original framework is validated to ensure that point, orientation, point-at, look-at, and all joint space variables (i.e. joint position, velocity, torque) can be maintained within acceptable bounds. Multiply satisfied constraints are applied to the robot to measure the introduced error as a function of constraint priority and proximity on the kinematic hierarchy. Multiple basic constraints can be composed simultaneously to handle larger constraints such as volume or proximity constraints. These constraints are composed with an input motion so the social robot can move in a human-like, communicative manner while not disrupting its task.

Chapter 6 discusses the rationale of composing the set of algorithm in a particular manner to synthesize overall motion that is communicative, human-like, and constrained. Since the algorithms are composed serially by making motion communicative and more human-like first, a select few experiments are performed to analyze how constraints affect the communication and human-likeness of the resultant output motion. The experiments show that variance and human-likeness are not statistically different when comparing motion with and without constraints.

Finally, in Chapter 7, I describe some extensions and changes that I have planned for the future of this algorithm. These are primarily based on the main constraints imposed upon work and the assumptions made in order to frame my work into a scope manageable in a five-year timeframe. This future work includes additional communicative algorithms, collision algorithms, handling rapidly dynamic constraints, and extending my work to other robots such as those that were designed with human-like dynamics and control or those non-anthropomorphic (i.e. non-humanoid robots). I conclude the thesis by restating some of my primary contributions.

1.5 Objective and Subjective Data

The experimental protocol used throughout this thesis is balanced between quantitative data. My algorithms are designed for human-robot interaction, and thus, often success in development of an algorithm or measurement of a particular variable that correlates with success can only be achieved through subjective data. Research methods such as questionnaires, Likert scales, and interviews are valid data that help discern success and future improvements. Accompanying other experimental data, I often report statistics, such as preference for one type of motion from a set of options. Since this is subjective data, not all humans agree, and under these circumstances, comparison to unweighted, statistical percentages (i.e. the inverse of number of options) is presented to indicate that the results of my work are preferred.

Statistical tests are the analysis method of choice with subjective data, and throughout this thesis I am demonstrating statistical significance, which yields a confidence in the

conclusions that I state herein. This research was covered by two institutional review board protocols written so that human subjects could be recruited on the campus of Georgia Institute of Technology for interacting with a social humanoid robot. These protocols covered all the experiments discussed in my thesis: in laboratory, web-based, and computer-based.

When recruiting participants, I sought a balance of genders and the full distribution of ages. However, since recruiting was performed on campus, the student population impacts the resultant distribution of participants. I.e. the majority of the Georgia Institute of Technology student population is male (approximately two-thirds) and aged between 18-23. When performing statistical testing, part of protocol is to test for bias in the data (e.g. order or gender effects). None of the work presented herein tested statistically significant for such bias, unless indicated specifically otherwise in the respective experiments.

CHAPTER II

THEORETICAL BACKGROUND

Chapter 2 discusses optimal control, operational space control, Kolmogorov-Sinai entropy, and motion capture, which provide the background required to understand the work presented in this thesis. None of these works are part of my research contributions, so they are devoted their own section and referenced when needed throughout the remainder of the thesis.

2.1 Operational Space Control

Operational space formulation of tasks provides a mechanism for composing motion primitives according to the dynamics of an articulated body. It is a unified framework for simultaneous motion and force control through the decomposition of joint torques into two dynamically decoupled control vectors: one that corresponds to forces acting on points on the robot body and the other is a set of joint torques from internal motions only.

Motion can be decomposed into a set of tasks (i.e. primitives) that must be simultaneously executed in a constrained actuation and control space. For example, these primitives might include constraints (e.g. contacts, joint-limits, and obstacles), operational tasks (e.g. manipulation), and trajectories (e.g. postures, gestures, residual motion) [43]. After prioritization of these tasks, task coordination then occurs from recursive projection of the robot dynamics onto the space associated with the task, in priority order, leaving a complement space for all other remaining tasks. This projection occurs via the dynamically consistent generalized inverse (DCGI), which is derived from relation of joint-space and Cartesian-space dynamics [44, 45, 46].

The process of application of the dynamically consistent generalized inverse will be demonstrated using two tasks, a primary task of highest priority and a secondary task of maintaining a posture, denoted in subsequent equations as task and posture, respectively.

The Jacobian in Equation 1 relates torques to forces, and can be derived for any task defined in terms of Cartesian forces,

$$\Gamma_{task} = J_{task}^T F_{task}, \quad \Gamma_{posture} = J_{posture}^T F_{posture} \tag{1}$$

where,

Γ = generalized force vector in joint-space for the primary task, posture, or any task

J = Jacobian matrix defined by the robot architecture

F = generalized force vector for the primary task, posture, or any task in Cartesian space

Beginning with the joint-space robot dynamics as in Equation 2, and following the derivation in [47], the configuration-dependent dynamically consistent generalized inverse of one task, without any other tasks can be written in Equation 3.

$$\Gamma = M(q) * \ddot{q} + V(q, \dot{q}) + G(q) \tag{2}$$

where,

$V(q, \dot{q})$ = vector of configuration- and velocity-dependent centrifugal and Coriolis forces

q = generalized state in joint-space

$\dot{q}$ = generalized velocity vector in joint-space

$\ddot{q}$ = generalized acceleration vector in joint-space

Γ = generalized joint force vector for any task

$M(q)$ = configuration-dependent mass matrix

$G(q)$ = vector of configuration-dependent generalized gravity forces

The quantity in brackets in Equation 4 is used to project the lower priority task into the null space of the higher priority task, which in this example defines the posture space that does not interfere with the task.

$$J_{dcgi,task} = M^{-1} J_{task}^T [J_{task} M^{-1} J_{task}^T]^{-1} \tag{3}$$

$$P_{task} = [I - J_{task}^T J_{dcgi,task}^T] \tag{4}$$

Thus, in the presence of a posture constraint and a single task, the actuation torques are determined by Equation 5.

$$\begin{aligned} \Gamma_{apply} &= \Gamma_{task} + \Gamma_{posture,non-disrupting} \\ &= J_{task}^T F_{task} + P_{task} \Gamma_{posture} \end{aligned} \tag{5}$$

However, social robots typically have more than just one task and a posture constraint. Under the presence of additional constraints, for each additional constraint, the higher priority tasks need to be combined into a composite dynamically consistent generalized inverse and composite projection, so that the torque trajectories of lower constraints can be projected into the null space of all higher priority constraints. This composition will be demonstrated by extending the Operational Space framework from the previous example to include one additional constraint (a subtask for the posture), since all additional lower priority constraints are composed in a similar way (i.e. the steps below are extensible to an infinite number of tasks).

The task and posture need to be composed to form the dynamically consistent generalized inverse and corresponding projection. The first step is to form the task-consistent posture Jacobian as in Equation 6, which defines a range space of posture motion that is consistent with the task.

$$J_{pt} = J_{posture}[I - J_{task}^T J_{dcgi,task}^T]^T \tag{6}$$

Hence, if the posture and primary task exhibit decoupled dynamics (i.e. do not conflict), J_{pt} has full rank and a direct solution exists for the dynamically consistent generalized inverse of both the primary and secondary tasks. The dynamically consistent generalized inverse in Equation 7 is a pseudo-inverse that defines the null space, since $J_{dcgi,pt}^T \Gamma_{task} = 0$.

$$J_{dcgi_{pt}} = M^{-1} J_{pt}^T [\Lambda]^{-1} \tag{7}$$

where, Λ is the inverse of the Cartesian space inertia matrix.

$$\Lambda_{decoupled} = J_{pt} M^{-1} J_{pt}^T \tag{8}$$

However, tasks rarely exhibit independent dynamics for an articulated body. Under the more common situation of coupled dynamics, Λ in Equation 7 is singular with rank k. Forces induced by the primary task in the posture space must be compensated. Eigendecomposition on Equation 8 provides the necessary variables to calculate the projection in Equation 9 that only induces forces along the eigenvectors of Λ and eliminates task space influence on the posture space.

$$\Lambda_{coupled} = U_r \Sigma^{-1} U_r^T \tag{9}$$

where,

U_r = l x k matrix of eigenvector columns corresponding to the non-zero eigenvalues of Λ

Σ = k x k matrix of non-zero eigenvalues of $\Lambda_{decoupled}$

l = number of DOF for posture task

Therefore, using either Equation 9 when task dynamics are coupled or Equation 8 when task dynamics are decoupled for Λ in Equation 7 generates the DCGI, which can multiply both sides of Equation 2 to determine the forces that act along only higher priority task-independent directions.

If more tasks exist in the prioritized hierarchy, a new projection matrix must be formed so that the steps outlined in Equations 6-7 can be repeated. Depending on whether the dynamics are coupled or independent, Equations 9 or 8 are used in Equation 7, to form a similar projection that prevents the posture subtask from disrupting either the task or the posture (i.e. higher priority constraints).

2.2 Optimal Control

The solution to an optimal control problem is the control policy that propagates the control variables in time, while minimizing a cost functional that is a function of state and control variables. Although optimal control problems exist in many variants, such as stochastic, nonlinear, discrete, and continuous forms, discussion will be limited to a special case of optimal control problem, called discrete linear quadratic (LQ) optimal control. The cost

functional for the LQ problems is shown in Equation 10, where penalty is incurred for more control or state energy and deviation from the final state x_N at the final time increment N.

$$J = \frac{1}{2} x_N^T S_N x_N + \frac{1}{2} \sum_{k=0}^{N-1} [x_k^T Q_k x_k + u_k^T R_k u_k] \tag{10}$$

where,

x_N = state value at final time increment

x_k = state value at discrete time increment k

u_k = control value at discrete time increment k

T_0 = initial time

N = final time increment

S_N = weight matrix for penalizing final state deviation

Q_k = weight matrix for penalizing state energy at discrete time increment k

R_k = weight matrix for penalizing control energy at discrete time increment k

An optimal control law satisfies Equation 11, which associates a control action to each state based on the state and control that yield minimum cost with respect to the functional and transition to the next state.

$$\pi(x) = argmin_{u \in U(x)} \{cost(x,u) + v(next(x,u))\} \tag{11}$$

where,

$v(next(x,u))$ = optimal value function for transition to the next state using control u

$\pi(x)$ = optimal control policy at state x

$U(x)$ = set of all possible control actions in state x

$cost(x,u)$ = cost functional for state x and control u

However, in the set of all possible control actions one control policy may not uniquely specify the minimum. In optimal control, the cost functional supplies the additional constraints that uniquely specify a trajectory from the set of all possible trajectories that propagate a state from current to final state. Therefore, the left-hand side of Equation 11 is replaced by $v(x)$, which is always unique, and *argmin* becomes *min*. Subject to the discrete time system

dynamics, $x_{k+1} = f(x_k, u_k) = A_k x_k + B_k u_k$, and using a quadratic optimal value function, $v(x,k) = \frac{1}{2} x_k^T S_k x_k$, Equation 11 becomes Equation 12.

$$\frac{1}{2} x_k^T S_k x_k = min_u \{ \frac{1}{2} [x_k^T Q_k x_k + u_k^T R_k u_k] + \frac{1}{2} (x_{k+1})^T S_{k+1} (x_{k+1}) \} \tag{12}$$

where $A_k = \frac{\partial f}{\partial x}$ and $B_k = \frac{\partial f}{\partial u}$ evaluated at the original state and control for time k.

The optimal control solution to Equation 12 is given in Equation 13. Using Equation 12 and an assumption for the final cost (e.g. $v(x_N) = 0$), Equations 13 and 14 can begin at the final state and calculate the control backwards for the entire trajectory [48].

$$u_k = -(R_k + B_k^T S_{k+1} B_k)^{-1} B_k^T S_{k+1} A_k x_k \tag{13}$$

$$S_k = Q_k + A_k^T S_{k+1} A_k - A_k^T S_{k+1} B_k (R_k + B_k^T S_{k+1} B_k)^{-1} B_k^T S_{k+1} A_k \tag{14}$$

2.3 Kolmogorov-Sinai Entropy

Spatiotemporal correspondence (STC) is known by many different names throughout the literature (e.g. DOF coupling, interdependence between degrees-of-freedom, motor coordination), but it refers to correlation between the trajectories of degrees-of-freedom that are located proximally on a kinematic hierarchy [49]. And similarity or correlation between two trajectories can be measured by information (e.g. entropy).

Kolmogorov-Sinai entropy, a task independent metric, measures the rate at which state information is lost over time (i.e. entropy rate). In the context of this thesis, the system is a set of motors interconnected through the rigid kinematic hierarchy of the robot hardware, each supplying torques to produce robot motion. Intuitively, if more spatiotemporal information is lost between past and future in the system, motors become less coordinated. Traditionally, KSE is used to evaluate an underlying signal in the presence of noise or measure the amount of determinism between different observations or measurements of the same signal. To construct the metric, a d-dimensional state space is divided into non-overlapping segments of size r^d, where r is the length element used to discretize state space. For each observation of the same signal, a joint probability that the trajectory is in

segment i_0 at time zero, segment i_1 at the first sample time, ...up to segment i_{d-1} at the final sample time can be written. KSE averages this probability over all samples as in Equation 15 for infinitesimal segment size and infinite number of dimensions.

$$KSE = -lim_{r\to 0}, lim_{d\to\infty} \frac{1}{d\Delta t} \sum_{i_0 \ldots i_{d-1}} P * lnP \tag{15}$$

where,

Δt = sample interval time

r = length of one side of the discrete phase space element (i.e. segment)

P = joint probability of the trajectory in an ordered time-sequence of segments

However, the formulation of KSE given in Equation 15 requires modification for use as a metric for human-like motion. Consider the same formulation now presented for finite spatial extent across all degrees-of-freedom of an articulated kinematic hierarchy (i.e. body of the agent), limited to the finite temporal extent of one motion, as in Equation 16, which is bounded from above by Equation 17.

$$KSE = -lim_{d_s\to\infty}, lim_{d_t\to\infty} \frac{1}{d_s d_t} \sum_{V(d_s,d_t)} P(V(d_s,d_t)) * lnP(V(d_s,d_t)) \tag{16}$$

$$K_2 = -lim_{d_s\to\infty}, lim_{d_t\to\infty} \frac{1}{d_s d_t} ln \sum_{V(d_s,d_t)} P^2(V(d_s,d_t)) \tag{17}$$

where,

$V(d_s,d_t)$ = a motion trajectory

d_s = spatial size of a motion (i.e. number of actuators used in the motion)

d_t = temporal extent of a motion (i.e. time length of a motion)

Equation 17 is an intuitive way to compare two motions since a lower value indicates more deterministic, more coordinated motion, and the values for d_t and d_s are easy to specify for any given motion. Using the correlation integral presented in Equation 18, which calculates the fraction of spatiotemporal delay vector pairs that are less than distance r apart, Equation 17 may be approximated by Equation 19 [50, 51, 52].

$$C_{(d_s,d_t)}(S,T,r) = \frac{1}{(T-1)T(S-1)S} \sum_{l=1}^{T} \sum_{j=1}^{T} \sum_{g=1}^{S} \sum_{h=1}^{S} \Theta(r - ||V_l^g - V_j^h||) \tag{18}$$

$$K_2(S,T,r) = ln(\frac{C_{d_s,d_t}(S,T,r)}{C_{d_s,d_t+1}(S,T,r)}) + ln(\frac{C_{d_s,d_t}(S,T,r)}{C_{d_s+1,d_t}(S,T,r)}) \tag{19}$$

where,

$\Theta(...)$ = Heaviside step function

V_i^k = $[w_i^k,...,w_i^{k+d_s-1}]$, spatiotemporal delay vectors

w_i^k = $[v_i^k,...,v_{i+d_t-1}^k]$, time delay vectors

v_i^k = element of time series trajectory for actuator k at time index i

S = number of actuators

T = number of motion time samples

2.3.1 My Adaptation

My adaptation for the KSE metric will be described in detail in Section 4.1, but here is an overview. As described above, traditionally, KSE measures the amount of information lost between two measurements of a chaotic signal. The metric above divides a d-dimensional state space into non-overlapping segments of size r^d. And thus, since a chaotic signal is not deterministic, it has a probability of being in each segment at each moment in time. And each time ordered path through the state space has a joint probability composed from the sequence of segments it traverses. KSE averages this joint probability as the segment size becomes infinitesimal and number of dimensions size approaches infinity (or total number of DOFs on the robot). Instead of using KSE traditionally to measure information lost between observations of the same signal, I use it to evaluate information difference between two different deterministic signals. Then, the KSE metric simply evaluates the fraction of spatiotemporal delay pairs that are less than a distance r apart. Thus, KSE (i.e. the K_2 metric) becomes a similarity metric for the difference between two trajectories in terms of timing and spatial magnitude (i.e. over all states).

In humans, muscles create coupling (i.e. STC). To synthesize human-like motion for robots, I emulate muscular coupling effects on a robot between DOFs that should be

connected by muscles (i.e. located kinematically proximal on the hierarchy). Then, KSE as estimated by the generalized entropy rate can be used as a metric for human-like motion, since it correlates to motor coordination and spatiotemporal correspondence of actuator states. For more information regarding how to use KSE as a metric or to synthesize human-like motion, see Chapter 4 of this thesis.

Spatiotemporal isomap (ST-Isomap) is an alternative algorithm which can also align a set of time series in terms of spatial and temporal similarity [53]. The advantages of KSE are that it is a task-independent metric, which allows for comparisons across different motion categories, and unlike ST-isomap, KSE lacks of ambiguity in the definition and specification because it requires no user input. It operates on periodic or chaotic signals, and the independent fractions in Equation 19 allow for divided, independent calculation of spatial and temporal metrics, which is an advantage that ST-isomap does not provide.

2.3.2 Implementation Note

To optimize a motion with respect to the KSE metric, there are two options: optimal control and dynamic time warping (DTW). The former algorithm includes undefined weights in the cost function. Also, to use optimal control, the metric is formulated in convex form to ensure convergence to a solution. Since $C_{d_s,d_t+1}(S,T,r) < C_{d_s,d_t}(S,T,r)$, $C_{d_s+1,d_t}(S,T,r) < C_{d_s,d_t}(S,T,r)$, and $0 < C_{d_s,d_t}(S,T,r) < 1$ for all $d_t > 1$ and for all $d_s > 1$, as will be the case for humanoid robots that are moving, Equation 19 is bounded from above by Equation 20, which is clearly convex [54].

$$K_2(S,T,r) < \left(\frac{C_{d_s,d_t}(S,T,r)}{C_{d_s,d_t+1}(S,T,r)}\right)^2 + \left(\frac{C_{d_s,d_t}(S,T,r)}{C_{d_s+1,d_t}(S,T,r)}\right)^2 \tag{20}$$

Motion is optimized with respect to KSE by replacing the cost function in the implementation by the respective spatiotemporal metric. Dynamic time warping optimizes through three basic operations: insertion, deletion, and hold. To improve the warp smoothness of the trajectory, after the warping algorithm completes, insertions are replaced by spherical linear interpolation (slerp) between endpoints of the insertion.

2.4 *Motion Capture*

Many different types of motion capture systems exist, such as inertial, mechanical, magnetic, and optical. Any trajectories generated in this thesis that are attributed to motion capture were generated using the 12-camera optical Vicon system in Dr. Karen Liu's laboratory at the Georgia Institute of Technology.

Motion capture works by overconstraining the human body using a number of reflective markers greater than the number of degrees-of-freedom on the motion model to which the data is being fit. A similar set of constraints or handles, which are positioned on the target kinematic hierarchy (i.e. the robot model), serve to sufficiently constrain the problem so the mapping between human motion capture markers and robot markers is one-to-one. The process of finding an optimal mapping for motion from one agent to another is called retargeting.

The agent whose motion is being captured (in this thesis, a human) wears a suit covered with markers that are calibrated to specific locations on the human body. After calibration of the model so that link lengths (i.e. bone lengths) are identical to the human and marker locations are known, infrared light impinges upon the markers in a dark room. The human moves in the suit, and cameras capture the reflected light at a sample rate of up to 120 Hertz. This provides a number of time varying point coordinates in 3-dimensional space that correspond to known locations on the kinematic hierarchy.

An optimization is solved, iterating over the human motion capture data as a function of time, aligning human and robot markers, to match a set of constraints, one for each marker, as in Equation 21, to solve for the joint angle trajectories for all angular degrees-of-freedom of the model, $\Omega(j)$, and positional degrees-of-freedom of the model, $P(j)$.

$$argmin_{P(j),\Omega(j)} \sum_{t=j*(s-o_l)}^{j*(s-o_l)+s-1} \sum_{k=1}^{C} dist(x_{data}(t,k), x_{model}(t,k)),$$
$$s.t.\ \ o_l < s, \quad \forall j = 0,1,...,floor(T/o_l) - 1 \tag{21}$$

where,

$dist(...)$ = distance function used in optimization

$x(t,k)$ = Cartesian marker location either recorded from equipment (i.e. actual data) or marker constraint location on the model for time sample t and constraint k

o_l = number of time samples in the data window that overlap

C = number of constraints

s = window size (for data smoothing)

$\Omega(j)$ = set of all angular degrees-of-freedom on the model (including center of mass rotational DOFs)

$P(j)$ = Cartesian position of center of mass

j = new time sample index for resampled, smoothed data

T = total number of samples in recorded data

The optimal mapping allows for scaling the overall size of the robot based on human participant's size, given that the proportions of the robot's parts remain constant with respect to each other. This ensures the maximum amount of information preservation over the retargeting process. Upon termination of the optimization, a set of time-varying point constraints exist on the robot body that align optimally with the human constraint markers from the motion-capture data. The time-varying point constraints on the robot create a motion trajectory, in joint angles, that can be executed on the robot.

To improve the optimization, filter windows that overlap in time are used so that smoother motion results. The optimization minimizes the distance between point coordinates of all of the marker locations from the sample data to the known locations on the kinematic model. To improve results, six degrees of freedom for the body centroid (3-D position, and 3-D rotation) are also included in the optimization.

2.4.1 DOF Correspondence Problem

When robot and human (or any two agents') DOFs differ in number, position, dimension, magnitude, and/or direction, this is called the DOF correspondence problem.

The DOF correspondence problem is most relevant when projecting human motion onto a robot kinematic hierarchy that is different with respect to degrees-of-freedom. This projection process is called retargeting, and it is accomplished typically through use of an

optimization like that described in Section 2.4. If I optimally retarget human motion onto a model which has the same kinematic structure and dynamic properties (e.g. torque and velocity limits), the motion looks excellent on the model (e.g. robot).

However, humans' and robots' joints typically do not correspond in terms of location and number of degrees-of-freedom. Since physical architectures differ, information is lost and motion appears different when attempting to directly use human motion on a robot. As two agents become more different in terms of kinematics or dynamics, the captured human motion looks much worse when projected and used on the target (e.g. robot). Logically, human motion retargeting works best on humanoid robots, but in my experience, even on Simon, which has no translatable DOFs, the retargeting process produces very poor motion.

The resultant projection between robot and human coordinate frames (i.e. architectures, kinematics) using retargeting is only scale invariant, and the scale factor must be equivalent in all Euclidean dimensions to preserve invariance. Since motion capture retargeting works by sufficiently constraining in Euclidean space through the collection of adequate number of markers on a human body, data is lost when the transformation between the target and human architectures is anything other than a scale factor. Retargeting algorithms do not solve the DOF correspondence problem. For example, projection of human translation at the shoulder will not survive a retargeting process, if the robot does not have a translatable shoulder DOF.

Typically, retargeting produces a input command trajectory for robots with motors that are controlled by PID control. Thus, it is a kinematic projection only. In other words, retargeting assumes that the robot dynamic DOF abilities exceed the human DOFs torque and velocity limits. This assumption, which is quite valid for virtual characters, is often untrue for robots. Therefore, motion capture trajectories for robots that were created by retargeting human motion often must execute much slower on the robot than they did on the human, so the robot can accurately track the human motion. If the trajectories are not executed slower, then the robot motion from a retargeted human trajectory appears even worse since there is more error in tracking the trajectories that did survive the retargeting

process.

2.5 *Research Platform*

The robot platform used in my research is an upper-torso humanoid robot called Simon (Figure 3). It has sixteen controllable DOFs on the body, four degrees-of-freedom per hand and per ear, three eye DOFs, and four neck degrees-of-freedom.

The robot operates on a dedicated ethercat network coupled with a real-time PC operating at a frequency of 1kHz. To maintain highly accurate joint angle positions, the hardware is controlled with PID gains of very high magnitude, providing rapid transient response.

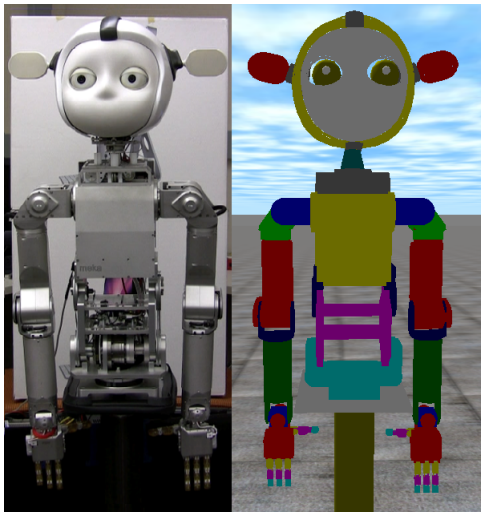

Figure 3: Hardware (left) and simulation (right) of SIMON.

A dynamic model of the robot hardware used for simulation is shown in Figure 3. The model was designed by importing 3-D meshes of the same Solidworks[1] files from which the robot parts were manufactured. Solidworks precalculates accurate centers of mass and the constant part of inertia tensors for all rigid bodies in the simulation. Motors are modeled in the simulation with identical gains for identical response between simulation and hardware. Separate gains are modeled for the separate motor control modes available in the hardware, such as position control and torque control.

[1]Solidworks is a registered trademark of Dassault Systmes SolidWorks Corp. All rights reserved.

The simulation environment that adds physics to the virtual world is implemented using Open Dynamics Engine (ODE), with a time-step corresponding to the update rate of the actual hardware. Bi-directional communication between hardware and software is possible in real-time due to efficient code and low network delay, which provides the ability to use either the robot or the simulation as either input or output at any given moment in time [55, 56].

Although the robot hardware is not equipped with force sensors beyond the torque sensors at DOFs, when external forces are applied to the hardware, I leverage simulation and hardware joint angle position, velocity, and acceleration differences to calculate the location, magnitude, and direction of the external force imposed in the real-world so it can also be applied to the simulation in real-time. The result is a full simulation loop and hardware read/write cycle in real-time, which is commonly called hardware-in-the-loop.

CHAPTER III

ADDING COMMUNICATION

For decades, traditional animation at Walt Disney Studios has been derived from twelve principles widely accepted to endow characters with natural behaviors, reactions, timing, and qualities [41]. In 1987, John Lasseter introduced the twelve principles of 2-D animation to 3-D animation, which have since been wholeheartedly accepted by the computer animation community to achieve better looking motion [42].

Since it is widely accepted in animation that these principles imbue a character with "life" or "naturalness," it is a reasonable assumption that these twelve principles can transcend the medium of application and bring the same qualities to robot motion. Effectiveness and engagement in an interaction increase when a machine exhibits human-like traits [57]. Work in conversational agents also supports the notion that natural motion of the agent improves the signal-to-noise ratio in communication with a human [30]. Inspired by initial findings by other researchers, I explore the direct benefits to adding communicative motion, inspired by these animation principles, to social robot motion beyond the benefit of improved motion appearance.

The goals then become (1) the adaptation of these principles to robot motion in a manner that overcomes the challenges that arise from real-world hardware implementations for social robots (2) establishing the benefits that each different principle has in HRI situations (3) showing that social robot motion appears more human-like with the inclusion of these principles and (4) measuring success of the principles as communication mechanisms.

The next three sections discuss three specific methods of communication, inspired by the principles of animation. Explicit hypotheses are set forth in each respective section, an algorithm is presented for the production of the communication mechanism, and experiments are conducted to test each of the hypotheses.

3.1 *Anticipation*

3.1.1 Introduction

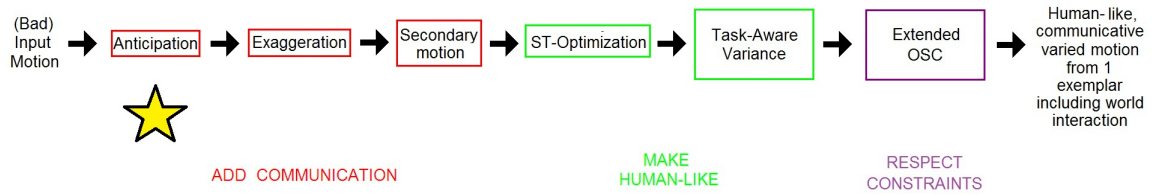

Figure 4: Section 3.1 discusses anticipation.

Anticipation is the first of the three communicative motion algorithms discussed in this thesis, as shown in Figure 4. It occurs first in the series of three in the high-level motion algorithm because it modifies the input motion temporally and spatially so that observers will be prepared for the task earlier. The order of these three communication algorithms in the high-level algorithm follow the progression of natural motion: preparation, task, and response [58].

Anticipation is first of three segments of a motion sequence, and the other two segments are the primary motion and follow-through. Anticipatory motion occurs before primary motion to prepare the viewer for a forthcoming action. Specifically, the act of an anticipatory stroke prior to motion communicates information to humans in the vicinity of the motion, such as the direction of a reaching motion or saliency of objects. A very common anticipatory motion accompanies the motion when humans change their eye gaze; humans direct their eyes first, closely followed by their head and body in that order because the majority of humans must localize targets prior to reaching. Other examples of anticipation include the wind-up action of a pitcher or a large inhale of air before a strong exhale. The concept of anticipation is familiar in the domain of computer animation, as it is one of principles of computer animation made famous by Thomas and Johnston [41]. However, even in the realm of computer animation, there are few autonomous algorithms to generate anticipatory motion.

Anticipatory motion should not be confused with the primary motion it accompanies.

For example, the goal of the *primary* motion is often task-based, such as locating a target or grabbing an object, in the cases of shifting eye gaze or reaching, respectively. However, the anticipatory motion that sets up these two tasks yields clues that a gaze shift or reach motion is about to occur and has very different goals than the primary motion that it precedes. I want to add this anticipatory motion to robot motion, so that human partners know what motion the robot will be performing earlier.

For humanoid robots, anticipatory motion is not created automatically by physics. I hypothesize that if anticipatory motion can be embedded into the motion of social robots in the appropriate way, humans can use these anticipatory motions to discern subsequent robot action (i.e. the primary motion it accompanies). If my hypothesis is correct, use of anticipatory motion can increase robot internal state and strategy transparency in human robot interaction.

Anticipation in human motion occurs for two primary reasons: to attract observer attention and to build up momentum. Thus, there are two types of anticipatory motion: communication-based and momentum-based. In general, the existing anticipatory motion algorithms focus upon the traditional type of anticipatory motion: motion preparation that sets up momentum for the following motion. Common examples of this type of anticipatory motion are in motions from sports, like the retraction of an instrument (e.g. bat, racket, or club) before swinging or the drawback of an arm before a throw.

Momentum-based anticipation has been created using an algorithm that retracts the center of mass and joint angles of the agent, modeled as a set of rigid bodies, in the direction opposite to the given subsequent motion to a retracted, extreme pose from the initial state using a weight-based strategy [59]. In this particular momentum-based anticipatory motion generation technique from the literature, an original, input (i.e. primary) motion is provided to the algorithm, and an optimization is solved to find a pose for the agent that retracts the center-of-mass of the agent in a direction opposite to the primary motion. If that solution is found, continuity of the trajectory and derivatives (e.g. velocity, acceleration) is maintained as the agent transitions from current pose, to retracted pose, and finally to initial pose of the given motion. This augments the given trajectory with

an anticipatory trajectory at the beginning. This particular algorithm does not work well without an initial guess to the pose before solving the optimization.

Since anticipation is most commonly associated with preparation of momentum and other researchers have already presented useful algorithms for momentum-based anticipatory motion, communicative anticipatory motion is the focus in my work. Social robot motion contains many gestures, which usually don't require large quantities of momentum and don't exhibit a large change in the robot's center-of-mass over the duration of the motion. Therefore, communicative anticipatory motion is more beneficial for robots that interact with people. Specifically, anticipatory motion in gestures can be used to communicate motion intent earlier than motion without anticipation. For example, robot handoffs will be more fluid if the human partner has more time to prepare, since they will be aware of the robot's intent sooner.

3.1.2 Insight

The key insight of my algorithm is that gestures used in social communication have a hand or body configuration that represents a *symbol*, which has a commonly accepted meaning, given the social context. If it is possible to extract that symbol and create a variant of the same motion which displays that symbol sooner, the motion becomes *anticipatory*, in that the human partner has advance knowledge of what motion the robot is performing. This will improve interactions (e.g., allowing the human partner to better coordinate with the robot in collaborative tasks [60]). For this work, facial gestures and motions for which anticipation is used for the sense of building momentum are excluded.

Thus, the symbol in my work is defined as state associated with the instant in time in the original motion that has the highest recognition accuracy. Anticipatory motion is any motion in which the symbol frame occurs earlier than it does in the original motion.

3.1.3 Goal

In Section 3.1.5, concrete hypotheses for the anticipatory motion experiments will be defined, but in general, robots that display anticipatory motion provide their human partners with greater time to respond in interactive tasks because human partners are aware of

robot intent earlier. Therefore, the goals are to:

- design an autonomous algorithm that creates an anticipatory variant of a given motion exemplar, from which human observers can discern motion intent sooner than motions without anticipation
- demonstrate the benefits of anticipatory robot motion; for example that humans can identify anticipatory motion intent with higher accuracy than non-anticipatory versions
- quantify the motion timing range when human partners can perceive anticipatory effects
- validate the algorithm using still image frames selected uniformly from motions to prove that this technique extracts the pose that is most useful in helping humans identify a motion

3.1.4 Algorithm

Inspired by a concept from computer animation called a motion-graph [61], which identifies points of transition between frames in large databases of motion to create new concatenated motion sequences, my algorithm creates anticipatory motion autonomously from a single motion exemplar by extracting hand and body symbols that communicate motion intent and moving them earlier in the motion.

The anticipatory motion algorithm begins with the assumption that a trajectory exists to which anticipation will be added. This original motion can be observed, come from a database, can be learned (through demonstration or otherwise), or can be provided by any standard means that trajectories are generated for robot actuators.

One frame is defined as $x(i) = \{x_1(i), x_2(i), \ldots, x_H(i)\}$, the set of all joint angles for a robot with H degrees-of-freedom at discrete time increment i. A trajectory, $x = \{x_1, x_2, \ldots, x_H\}, \forall i = 1, \ldots, T$, is defined as the set of all frames and all DOFs for all discrete time increments up to time T.

Creating a motion graph is similar to clustering. Frame $x(i)$ belongs to cluster C if and only if the distance between $x(i)$ and all other frames in C is less than some pre-determined distance threshold. Mathematically, $x(i) \in C$ if and only if $dist(x(i), x(g)) \leq dist_{threshold}, \forall x(g) \in C$. Then, one cluster or one node in the motion graph is defined as $C = \{\{x(i)\} : dist(x(i), x(g)) \leq dist_{threshold}, \forall x(g) \in C, \forall x(i) \in x\}$, i.e. the set of all frames with some distance less than or equal to some given threshold with respect to all other frames in cluster C. The distance measure need not be calculated in joint-space and will be discussed in detail in Section 3.1.4.3.

3.1.4.1 Determine Gesture Handedness: One or both hands?

Non-facial gestures for anthropomorphic robots are either one-handed or two-handed. Two-handed gestures represent a more constrained system, and they are commonly associated with a body posture that is part of the symbol. Waving and pointing are examples of one-handed gestures, whereas, shrugging ('I don't know') and bowing are two-handed.

Additionally, anthropomorphic robots usually have symmetric arms, with DOFs in the same locations relative to the end-effector. Logically, in one-handed gestures, the DOFs for one arm move much more than the DOFs of the other arm. Therefore, pairwise comparisons in variance (Equation 22) can be made for each DOF between both arm chains to determine if a particular DOF on one arm is moving significantly more than the corresponding DOF on the opposite arm.

$$v(x_m) = \sum_{i=0}^{N} \frac{(x_m(i) - \mu_{x_m})^2}{N-1} \tag{22}$$

where,

$v(x_m)$ = joint angle variance of arm DOF m

$x_m(i)$ = DOF m original joint angle value at time index i

μ_{x_m} = mean joint angle for trajectory of DOF m

N = number of samples in arm DOF m trajectory

Under the similar arms assumption, 'handedness' of the gesture reduces to a linear

regression of the variances. The least-squares minimization in Equation 23 is solved using pairwise left arm and right arm DOF data.

$$\hat{\beta} = \arg\min_{\beta_1,\beta_0} \sum_{m=0}^{M} (v_l(x_m) - (\beta_1 * v_r(x_m) + \beta_0))^2 \tag{23}$$

where,

$v_l(x_m)$ = joint angle variance of left arm DOF m

$v_r(x_m)$ = joint angle variance of right arm DOF m

β_0, β_1 = regression parameters

M = number of DOFs in one arm

If the correlation coefficient from the regression term in Equation 23 approaches 1.0, then the gesture is classified as two-handed. No two-handed motion in any of my anticipatory motion experiments had a correlation coefficient (R^2 value) below 0.998.

3.1.4.2 Find & Extract the Symbol

Since this work focuses on hand and body gestures, the *symbol* is a unique hand configuration that holds a social meaning. Thus, a representative hand configuration must be found from the input motion.

Gestures have a direction constraint, in that one or both hands during gestures are typically directed toward something; for example, consider stop gestures, waving, beckoning, or pointing, all of which make no sense if the hand changes orientation relative to some world constraint. Thus, the algorithm uses the hand normal vector (HNV, a vector directed outward from the plane that is parallel to the palm of the robot's hand) as the feature from which the motion graph is constructed. This unit vector (palm normal) is calculated for all discrete time increments in the trajectory, and it is represented in world coordinates, not a local vector relative to the hand orientation.

As shown in Figure 5, the hand normal vector is easily calculated from two world points: the hand centroid (x_{hc}, y_{hc}, z_{hc}) and a point one unit in the direction perpendicular outward from the palm $(x_{hnv}, y_{hnv}, z_{hnv})$ by subtraction. The original motion graph implementation, which clusters frames from a set of motions, was modified to use the hand

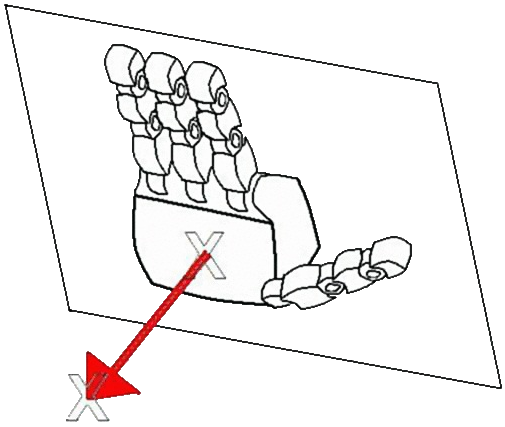

Figure 5: The hand normal vector extends outward from the palm perpendicular to the plane of the hand.

normal vector and a corpus of frames from a single motion rather than a set of motions [61]. Multiple features in addition to the HNV could be combined for posture extraction with my novel approach, but the given results demonstrate the power of this technique to extract the symbol frame using a single feature. Accuracy of symbol extraction is likely to increase as more features are added for posture extraction.

Given two hand normal vectors from frames f_1 and f_2 respectively, f_1 and f_2 belong to the same motion graph cluster if the criteria in Equation 25 is satisfied for appropriately sized increment thresholds for the two rotation angles, $\Delta\theta$ and $\Delta\phi$.

$$\begin{aligned} \theta &= \cos^{-1}(z_{hnv} - z_{hc}) \\ \phi &= \tan^{-1}\frac{(y_{hnv} - y_{hc})}{(x_{hnv} - x_{hc})} \end{aligned} \tag{24}$$

$$\Delta\theta > |\theta_{f1} - \theta_{f2}| \;\; and \;\; \Delta\phi > |\phi_{f1} - \phi_{f2}| \tag{25}$$

where,

(x,y,z) = coordinate of hnv or hc in Cartesian, world coordinates

hnv = denotes hand normal vector endpoint (i.e. vector head)

hc = denotes hand centroid point

ϕ_{f1} = ϕ formed from f_1 coordinates

θ_{f2} = θ formed from f_2 coordinates

Within each cluster, for each DOF, euler angles for all frames in the cluster are averaged together (DOF h shown in Equation 26). Then, a single representative frame for each cluster is composed from the set of averaged data for all DOFs, $p_{cluster} = \{p_{cluster_h}\}, h = 1, \ldots, H$, which creates frames in the anticipatory motion which do not occur in the original motion. If the HNV angular thresholds for clusters are set too high then the graph will have few clusters. If the threshold is too low, then the motion graph will simply devolve into the original motion.

$$p_{cluster_h} = \sum_{x_h(i) \in cluster} \frac{x_h(i)}{Y} \tag{26}$$

where,

$x_h(i)$ = joint position of DOF h at time index i

Y = number of frames in the cluster

i = time index of respective cluster frames from original motion

h = index for degrees-of-freedom

At least one node in resultant motion graph will contain a frame similar to the symbol in the original motion. To identify the symbol cluster (i.e. node) in the motion graph, the gesture (i.e. given input motion) is assumed to contain a large set of hand poses to ensure that the expressive message is received. Under this assumption, a large number of frames will contain the representative symbol hand configuration. After creating the motion graph, all of these frames should end up in the same node. Therefore, of all clusters in the motion graph, the symbol cluster is defined as the cluster in the motion graph with the largest quantity of frames.

3.1.4.3 Find Cluster-to-Cluster Transitions

The angular criteria in Equation 24 is extended to incorporate derivative information by using windows of frames to become the distance metric (Equation 28) for determining cluster-to-cluster transitions from the motion graph.

$$dist(x_{a1}, x_{bK}) = \sum_{k=1}^{K} |\theta_{x_{ak}} - \theta_{x_{bK+1-k}}| + |\phi_{x_{ai}} - \phi_{x_{bK+1-k}}| \quad (27)$$

where,

w, u = cluster indices being checked for transition ($w \neq u$)

x_{ai} = i^{th} frame in window beginning at original trajectory frame $x_{a1}; x_{a1} \in C_u$

x_{bj} = j^{th} frame in window ending at original trajectory frame $x_{bK}; x_{bK} \in C_w$

K = number of samples in transition window

Let x_{a1} be an arbitrary frame from the original trajectory that belongs to cluster C_u. Similarly, let x_{bK} be an arbitrary frame from the original trajectory that belongs to cluster C_w. Variables i and j represent index counters for two separate windows of frames of length K. One window begins at original trajectory frame x_{a1}, and the other window ends at original trajectory frame x_{bK}. Each frame in each cluster has an associated window of frames. After all distances for all possible pairs of windows between two clusters have been calculated according to Equation 27, a directed edge is added from cluster C_u to C_w if D (as calculated in Equation 28) is less than a predetermined distance threshold.

$$\begin{aligned} D &= dist(C_u, C_w) \\ &= \min_{x_{a1}, x_{bK}} dist(x_{a1}, x_{bK}), \forall x_{a1} \in C_u, \forall x_{bK} \in C_w \end{aligned} \quad (28)$$

where,

w, u = cluster indices being checked for transition ($w \neq u$)

x_{ai} = i^{th} frame in window beginning at frame $x_{a1}; x_{a1} \in C_u$

x_{bj} = j^{th} frame in window ending at frame $x_{bK}; x_{bK} \in C_w$

D = HNV angular distance metric

Intuitively, cluster transitions are calculated in this manner because all frames within a single cluster are similar according to the hand normal vector. The directed edge implies that motion can transition unidirectionally from one cluster to the other (i.e. from C_u to C_w

but not necessarily vice-versa) Thus, the distance metric is constructed to count forward from frame x_{a1} and backward from x_{bK}, the respective candidate cluster frames. As a result, pairs of frames after x_{a1} are compared with frames before frame x_{bK} to find out if velocity and acceleration in the original trajectory were similar. If the inter-cluster distance D is small between C_u and C_w, it means that directionally, the representative frames for these clusters, p_u and p_w (respectively) are good candidates for blending over a window of length K, since a similar transition existed in the original trajectory.

All possible pairs of representative frames in the motion graph are tested according to the distance metric in Equation 28 to find all clusters in the motion graph that should be connected by a directed edge. An edge between clusters represents two representative frames that are similar enough to be sequentially executed (or blended between) on the robot to create motion. Smaller values of D identify cluster pairs that are candidates for transition points. Low thresholds on D will create lower graph connectivity. Longer original trajectories have higher potential for creating motion graphs which have more node transitions. Higher graph connectivity allows the symbol to occur sooner in the anticipatory motion, as compared to a graph with lower connectivity.

The number of samples in the transition window, K, is set based upon the desired length of the transition between motions. One approach is to set K empirically [62], but a more dynamic approach would vary this transition time length based on a distance metric between frames being blended (see Section 3.1.4.4 for more details).

3.1.4.4 Compose the Anticipatory Motion

The anticipatory motion path through the motion graph is determined by beginning at the cluster that contains the initial frame from the original motion and following the path with fewest number of transitions to the symbol cluster (i.e. the node with the most frames in the graph), so the symbol occurs as soon as possible in the anticipatory motion. For motions with cyclic components, e.g. waving, the symbol cluster in the motion graph may be passed more than once. For cyclic motions, the resultant anticipatory motion is constrained to exhibit the same number of cycles as the original motion, which is easily

accomplished by observing the number of temporal discontinuities for frames in the symbol cluster. This is possible because the original motion is produced from a continuous trajectory for each DOF that has been discretely sampled, as is the case with all trajectories executed on computer-controlled hardware. After passing the symbol cluster the same number of times as in the original motion, the anticipatory motion can take any path to conclude at the cluster that contains the final frame from the original motion.

Since this particular algorithm creates motion from a graph structure, it allows for the creation of multiple anticipatory variants from one given motion. This flexibility is an advantage for my system, where creating human-like motion is a goal. For example, under certain circumstances one anticipatory variant might be better suited to meet the timing needs in particular context. In the experiments presented herein, the anticipatory variant with shortest time to the symbol was selected, to prove the benefits of anticipatory motion. But in general, the variant that best meets the needs of the system at that point in time should be selected. As long as the selected variant remains anticipatory (i.e. the symbol occurs sooner than in the original motion), the benefits discussed in my experiments with anticipatory motion will still apply.

$$x_{new_d}(t) = slerp(p_u(t), p_w(t), \alpha(t)) \tag{29}$$

where,

$\alpha(t)$ = weight function at index t, designed for continuity

d = d^{th} DOF in full body posture

t = frame index during transition, $-1 < t < K$

Anticipatory motion is synthesized in one of two ways. (1) When few or no candidate transitions exist that will allow an anticipatory variant of the motion to be extracted from the motion graph, splines are used between frames that represent the clusters' joint-space averages to generate the anticipatory variant to guarantee continuity of posture, velocity, acceleration, and other higher order state derivatives. (2) When motion needs to be generated using more of the frames from the original motion (e.g. when higher

frequency information would be lost in joint-space blending), the transition window from Section 3.1.4.3 can be utilized for spherical linear interpolation (Equation 29) with a properly designed blending weight function for continuity (see reference [61] for examples of weighting functions that offer C^1 continuity).

Regardless of the choice of (1) or (2), the anticipatory motion is reproduced using the joint angle data from all the DOFs. Since one frame of joint-angle data consists of all DOFs needed to generate motion, during motion synthesis redundancy is not an issue for this approach, as it might be if trajectories were represented for synthesis as sequences of HNVs, thereby creating a many-to-one mapping from Cartesian space to joint space.

3.1.5 Hypotheses

Other researchers have shown that event sequences from *human* movement generate prior expectancies regarding the occurrence of future events, and these expectancies play a critical role in conveying expressive qualities and communicative intent through the movement [63]. But, I have three hypotheses about the anticipatory motion for *robots* generated using my algorithm:

- H1: The symbol extracted using the procedure will yield the frame from the motion with the highest recognition accuracy of any frame in the trajectory.
- H2: In the anticipatory motions generated by the algorithm, human observers will label motion intent earlier than they can label the intent in the original motion.
- H3: Anticipatory motion is beneficial in helping observers predict motion intent only during a specific range in timing relative to the symbol. If an observer watches robot motion beyond the symbol frame, they will be able to predict motion intent with equal accuracy for anticipatory motion and the original counterpart.

Three experiments test these hypotheses, which were conducted on separate days using different sets of human participants. The intersection of these participant sets is a null set to eliminate bias in any of the three experiments.

3.1.6 Experiment 1: Symbol Validation

To test H1, I must determine if any other frame in the motion yields higher recognition accuracy (i.e. higher labeling accuracy) than the frame extracted by my algorithm, classified as the symbol. Thus, participants were shown still images taken from frames in motion trajectories and asked to label robot motions in the absence of motion.

3.1.6.1 Experimental Design

A set of thirteen original motions were obtained via animation with Maya[1] 3-D animation software or retargeting from human motion capture data. The gestures were: bow, beckon, shrug, point, stop, wave, fist bump, yes!, Mmm...tasty, cuckoo, knock, Shhh..., reach. These original motions were executed on the hardware and videotaped to access individual video frames. The algorithm was used to extract the symbol from each of these original gestures. The frame in the original motion that was nearest to the symbol was found, using a Euclidean distance metric in torque space. This became the representative symbol frame in the original motion. Some examples of symbols from Experiment 1 motions are shown in Figure 6.

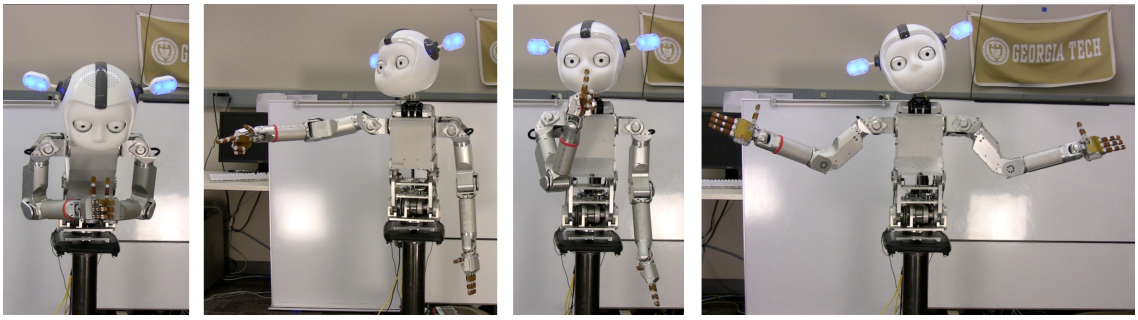

Figure 6: Example symbols on the Simon hardware extracted using the anticipation algorithm. Left to right: Bow, Point, Shhh..., I Don't Know

Since the goal of the experiment was to compare robot states (i.e. poses) based on visual differences, it was necessary to penalize the distance metric for pose variations that appear

[1]Copyright 2011 Autodesk, Inc. All rights reserved.

significantly different when viewed in Cartesian space. A joint-angle-space metric was insufficient because this treats all DOFs similarly when viewing poses in Cartesian space. For example, moving the wrist a small amount, does not make the robot configuration appear as different as moving the torso by the same amount. Furthermore, since gestures are predominantly free-space motions, payloads were irrelevant. Also weight-tuning was avoided for a weighted joint-space metric by using a torque-space metric. A torque-space metric more consistently gives a 'weighted' distance metric, which yields greater penalty for deviations in DOFs closer to the root (center-of-mass) in the chain, and produces better pose-dependent penalties in Cartesian space for gestures.

Once the symbol frame in the original motion was determined, the same distance metric was used to calculate the maximum composite torque-space distance from all other frames in the motion with respect to the symbol frame. This quantified the range of poses for a given original motion, which allowed them to be ordered according to the metric. The experiment cannot use all frames from all motions because some of original motions have over 300 frames. To do so would result in too many experimental conditions, necessitating too many human participants. Thus, this space was sampled uniformly to select six other uniformly-distanced frames. A true uniform sampling was not possible since the sample was selected from the finite number of frames that exist. Therefore, the selection of the frames was as close to uniform as possible. For all thirteen motions, selections included frames from before and after the symbol. All frames came from the original motion, since this experiment was testing whether the extracted frame is the frame of the original motion that produces the highest motion recognition accuracy from participants (i.e. the true symbol).

In this experiment, participants viewed one frame from each of the 13 motions in a random order, and were asked to label it with their best guess.

Participants were given the option to abstain from guessing, if they had no label for the motion. As a practice example, participants viewed one of seven possible frames from one of the 13 motions (randomly selected). After the practice example, the participants viewed only one of the seven possible frames from each of the remaining twelve other motions.

Motion order and frame were randomized. Participants were not allowed to go back to review a previous motion or change a label once they had guessed.

This experiment contained one independent variable, which is distance from symbol frame. Since seven still frames were used for each of the thirteen motions, 224 participants were recruited to participate, yielding a sample size of 32 per still image. All participants saw one of seven possible random frames from all thirteen motions. The high number of participants is necessary for increased data resolution, since results will be expressed in percentages of participants. 32 participants per condition results in a resolution of 3.125%.

In analysis of the data from Experiment 1, overall gesture recognition accuracy is irrelevant. The results should determine if the algorithm can pick out the "best" frame from the given set of all possible frames in a motion. Thus, the *relative* recognition accuracy between frames of the same gesture is the only important criterion. "Best" is the frame with highest recognition accuracy relative to all other frames.

3.1.6.2 Results

To demonstrate the relationship between frames in the motion relative to the symbol frame, the results depict the correlation between the distance metric (relative to the symbol) and percentage of participants who correctly labeled each still image. These results are shown for all thirteen motions from Experiment 1 in Table 1, where the numbers presented are the coefficients of determination from a monotonically decreasing power series fit of "percent correctly labeled" versus "distance from the symbol" ordered so that the symbol frame is far left and the frame furthest from the symbol is far right. A sample regressive fit is shown in Figure 7 for the bow motion, and the average of all motions is shown in Figure 8 with error bars. The fits for the other motions resemble these two plots. For the results presented in Table 1, any still images that had 0% correct recognition for any motion are excluded from the analysis.

A coefficient of determination of 1.0 means that the percent of participants who correctly labeled motion intent is perfectly correlated to distance from the symbol frame. Thus, the composite statistic (using the data from all thirteen motions) of 0.9944 indicates

Table 1: Coefficient of determination from monotonically decreasing power series fits of percentage correctly labeled versus distance from symbol frame. Maximum percentage correctly labeled (PCL) for any still image for each motion from anticipation Experiment 1.

Motion	Coeff. of Det.	Max. PCL
Bow	0.9025	93.8
Beckon	0.9804	59.4
I Don't Know	0.9336	62.5
Point	0.9269	87.5
Stop	0.8668	53.1
Wave	0.9934	81.3
Fist Bump	0.9836	56.3
Mmm...Tasty	0.8767	9.4
Knock	0.9966	40.7
Yes!	0.9634	40.7
Cuckoo	0.9734	15.6
Shhh...	0.8352	81.3
Reach	0.8825	25.0
Composite	0.9944	54.3

a strong correlation between torque-space distance from the symbol frame and ability of participants to accurately predict motion intent from still images. The high coefficient of determination and a fit that is monotonically decreasing with distance from the symbol is evidence that my algorithm extracts the true symbol frame. In short, based on the results from this experiment, when frames with a torque-space distance further from the symbol (which equates to poses less similar to the symbol) were shown to participants, they were less likely to predict the motion intent accurately.

The right-hand column in Table 1 shows the maximum percentage (for any frame) correctly labeled for each given motion. Even in motions where participants were unable to recognize a motion well on average, the high values of correlation still indicate that the algorithm was extracting frames that yield highest recognition. This reinforces the statement made previously that *relative* recognition accuracy between frames of the same gesture was more important than overall accuracy rate for motions in Experiment 1. Thus, use of less familiar or less common motions does not impact the experimental results.

Furthermore, in 12 of the 13 motions, the highest percent labeling accuracy occurred at

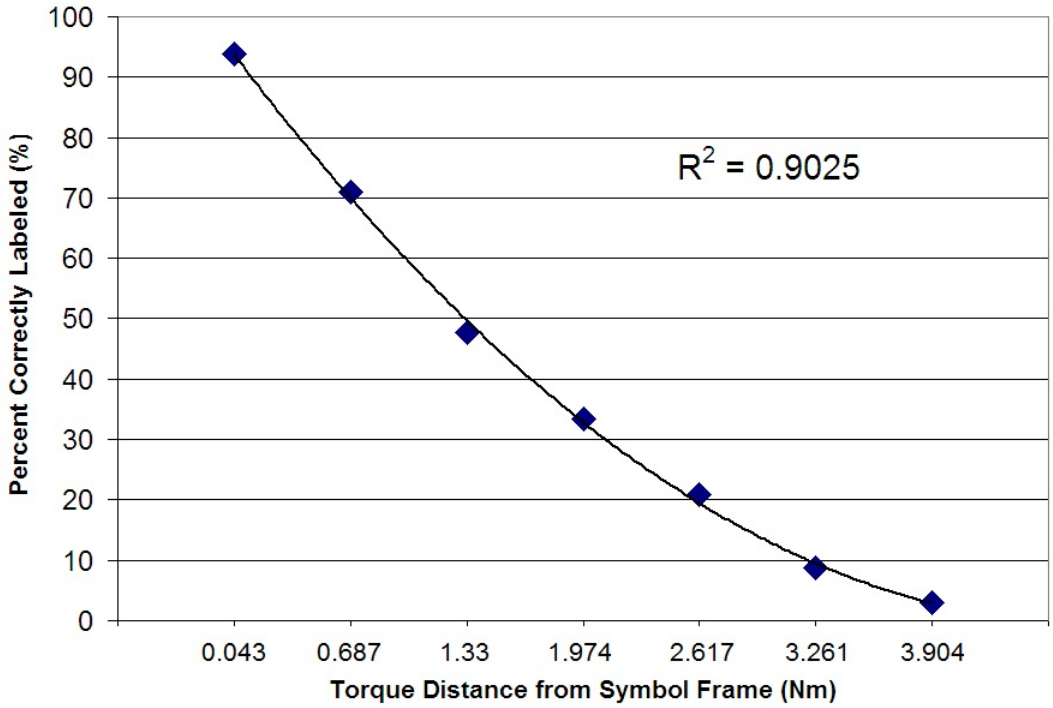

Figure 7: Sample regressive fit from anticipation Experiment 1. Percent of participants who correctly labeled each static frame of the bowing motion versus torque-space distance from the true symbol frame. High coefficient of determination and monotonically decreasing with distance from the symbol means that my algorithm extracts the true symbol frame.

the symbol frame. The exception was the 'reach' motion, where the labeling accuracy was 3% higher for the frame closest to the symbol. Reaching motion is a function of directionality. A reaching motion played forward looks like a 'placing' motion (without context), and a reaching motion executed backward is easily mistaken for a 'picking' or 'grabbing' motion. This directionality is absent in still frames, which suggests that prediction of intent for reaching depended more on context than the other motions in this study.

Given the high correlation between recognition accuracy and distance from symbol frame across motions and the fact that 12 of 13 motions had highest concentration of recognition accuracy near the symbol frame, I concluded that H1 holds true and my algorithm was extracting the true motion symbols.

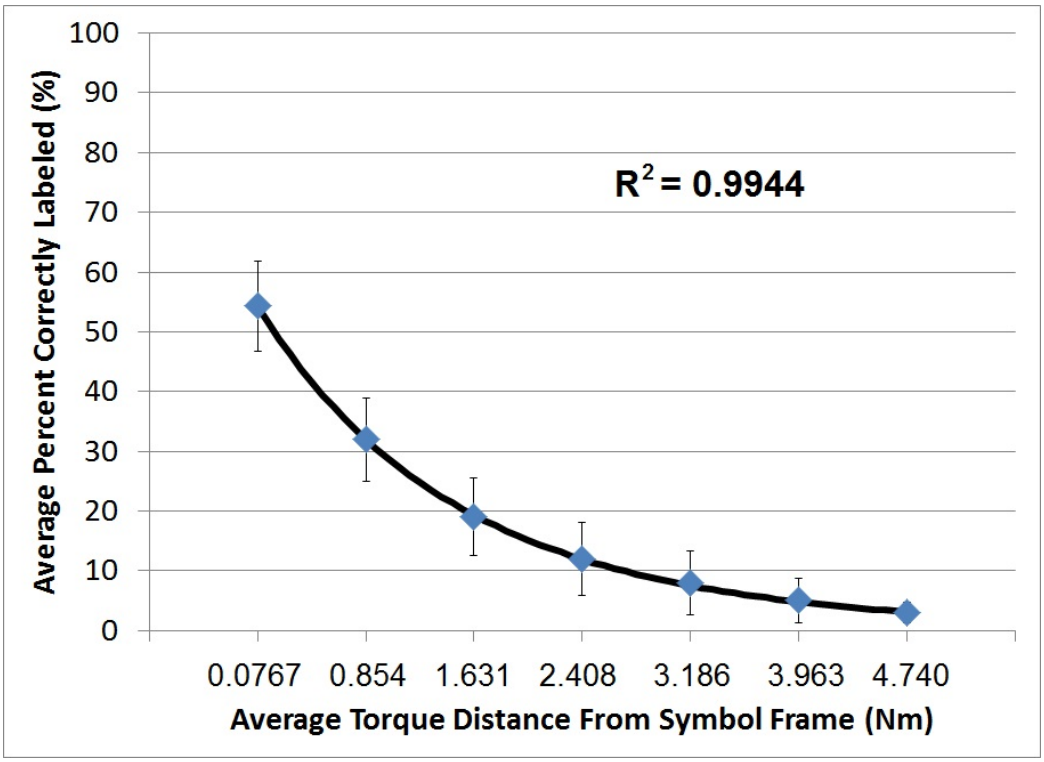

Figure 8: Average percent of participants who correctly labeled each static frame of all motions from anticipation Experiment 1 vs. average torque distance from the true symbol frame (Newton-meters). High coefficient of determination and monotonically decreasing with distance from the symbol means that my algorithm extracts the true symbol frame.

3.1.7 Experiment 2: Communication of Intent

Experiment 2 was designed to test whether humans can perceive motion intent sooner in anticipatory motion. This experiment also tested whether humans are confident enough to consistently guess a motion's intent prior to the symbol frame.

3.1.7.1 Experimental Design

Six motions from Experiment 1 were selected, and videos of these motions on the hardware were recorded. For the experiment, HTML and Javascript code was written that would progress through videos at random. To properly control the experimental conditions and measure accurate timing data, all participants accessed the code-based interface through the same computer, running the files locally on the computer, rather than over the internet. Six of the eight motions from Experiment 1 for which any frame was correctly labeled

by greater than 53.0% of participants, as shown in Table 1, were selected because for Experiment 2 overall gesture recognition accuracy mattered. Thus, common gestures and communicative motions that would be familiar to the largest number of participants were selected and less common motions such as 'Mmmm...tasty' which had a maximum of 9.4% correct recognition for any still image frame in Experiment 1 were eliminated. The six motions selected were: bow, beckon,'I don't know', point, stop, and wave.

Participants viewed each of the six motions only once, and for each motion, the motion version (anticipatory or original) was randomly selected. The participants were instructed to click the "stop video" button immediately, when they thought that they could label the motion. The stop video button was very large to minimize cursor localization time lag. After clicking, the screen changed to a blank screen with an empty prompt for typing a label. The code logged the time from start of video playing to click of the stop button. If the user didn't click the stop button and the end of the video was reached, the screen automatically transitioned to the page prompting for the motion label.

Videos of the robot hardware were used instead of the actual hardware for two reasons: safety and data integrity. It is safer to stop a video and have it disappear, than to have the real hardware freeze and hold a position upon press of a button. Second, the motion and all poses should disappear from view to ensure that participants were not relying more heavily upon the keyframe where motion ceased in decision making.

To encourage participants to watch as much motion as they needed, "no label" was not an option. If a participant left the response box blank or input characters that were not a label, this data was excluded from the experimental results. Since only the time when a human can first label a motion is important, if participants were unable to label the motion, then their answer provides no data for the experiment. Under ideal conditions, all participants would have labeled all motions correctly and waited only the minimum time necessary before pressing the stop button. Participants were not allowed to re-watch any videos.

To provide extra incentive, participants were told that only the two participants with the fastest times to correctly label all six motions would receive $10.00 each for their

participation in the experiment. The instructions were clear that only times when videos were actually playing counted toward their cumulative time total. They were allowed to spend as much time as they wanted typing in their labels for the motions.

3.1.7.2 Results

Eighty-two participants contributed to Experiment 2. This high quantity of participants is driven by two experimental conditions and the fact that only correct responses are valid data. I did not analyze data prior to experiment completion (to know how many responses were invalid data), and thus, to ensure adequate amounts of data were collected, many participants were required. Using only the correctly-labeled responses, Figure 9 clearly shows that average participant stop time is statistically significant for all six motions when comparing data from original and anticipatory motion versions. Since there are two versions of each motion and data is comprised of only the correct responses for each motion, the average stop times in Figure 9 are determined from 37-41 correct responses. The motions in Figure 9 are ordered from left to right in order of increasing symbol time difference between the original and anticipatory motions to demonstrate that predictability of motion intent from the symbol is not a strong function of how much the symbol moves relative to its timing in the original motion. Even with as little as 100 milliseconds of symbol timing difference in the beckon motion, intent was still more easily predicted with anticipatory motion. One possible reason is that motion at one time frame is dependent upon the previous frames (i.e. discrete states of motion trajectories are not independent). Thus, moving the symbol affects the previous frames to varying extents. By moving the symbol, human observers have more information about intent even before the symbol frame, and the symbol frame is not the sole means by which human perceive intent.

The majority of participants watched motions less than the symbol time before stopping to label the motion. Across all six motions, 74% of correct responses from anticipatory motions were labeled before the symbol, and 65.9% of original motions were labeled before the symbol. From this, I concluded that people are developing a mental model of motion intent while viewing motion. This prediction of intent via the motion communication

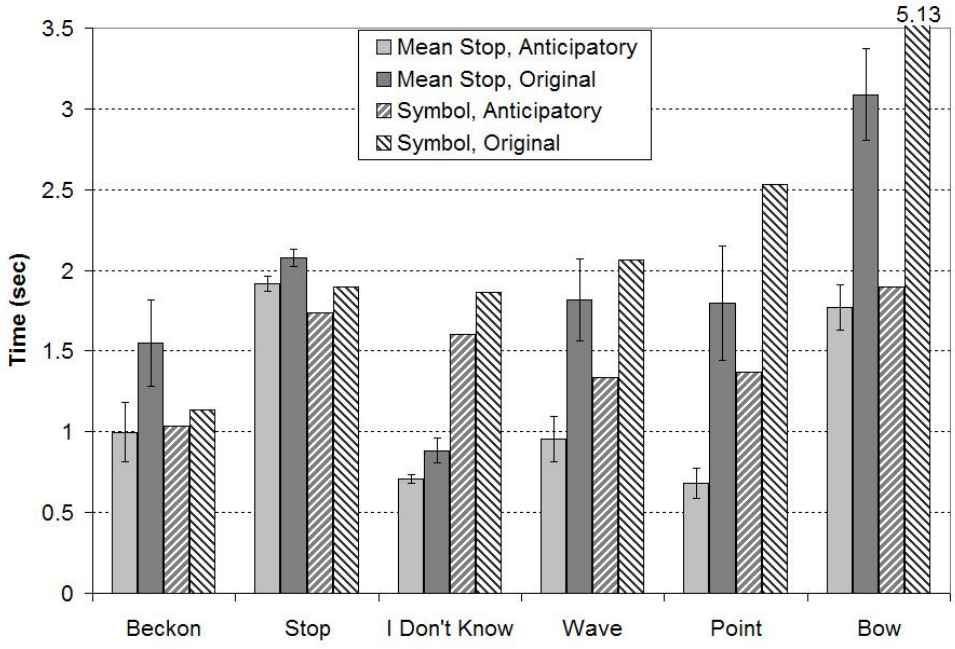

Figure 9: Average stop times and symbol times for all six motions, both anticipatory and original versions. From left to right, difference in original and anticipatory symbol times increase. Error bars show that average stop time between original and anticipatory motions is statistically significant for all motions.

channel could explain why turns can overlap in turn taking activities, or humans can react preemptively to partner motions in collaborative tasks.

On average, with the six anticipatory motions in Experiment 2, participants reacted 697 milliseconds sooner to anticipatory motion with correctly labeled responses for motion intent. This finding supports H2, and provides evidence that (1) when motions are familiar, humans can discern intent from motion, (2) the social cue for turns of dynamic collaboration is not restricted to action completion, (3) anticipatory motion leads to earlier correct labeling of motion intent than motions that are not anticipatory.

Comparing composite data for recognition accuracy of anticipatory motion against the original motion shows that by averaging over all six motions, anticipatory motion was labeled correctly 85% of the time versus 83% for the original motion. Treating all motions as six samples of a single distribution, the data for correct labeling fails statistical significance tests ($N=6$, $p>0.05$), and thus this is evidence that averaging motions that

terminate at arbitrarily different points in time is inappropriate. This is motivation and rationale as to why H3 is hypothesized with a timing relative to the symbol, instead of being a function of absolute time.

Finally, comparing the results from the first two experiments, overall response rates for correct labeling were higher for motion than for static images. This suggests that motion conveys more information about intent than a single frame. Even though both experiments were largely devoid of context, ambiguities in motion intent were better resolved by viewing more frames.

3.1.8 Experiment 3: Quantifying the Optimum Time Range

In the final anticipatory motion experiment, the time range over which anticipatory motion is beneficial for human-robot communication was quantified. In this experiment, generalizations across motions needed to made, and therefore a variable that is not specific to a particular motion was required. H3 is based upon the logic that if the symbol is the most important frame in the entire motion, then once an observer has seen it, they will gain very little (in terms of determining intent) from watching the rest of the motion. To generalize across motions, percent of symbol time was selected as the variable by which group categories were divided in this experiment. For example, 10% means that the [test video length] divided by the [time at which the symbol occurs] equals 0.1.

3.1.8.1 Experimental Design

There were two independent variables for this experiment: motion end time and motion type. Motion end time had one of seven values: videos that ended at 20% increments with respect to symbol time, up to 120%, and the entire motion. With each of these there were two possible motion versions that a participant could see: anticipatory or original motion.

Using web-based code running on a single computer (similar to the setup in Experiment 2), one of the two possible versions for each of six different motion videos, each of which ended at one of seven possible (randomly-selected) end times, were displayed in a random order. Participants watched one video to the predetermined concluding point, then the screen would blank and prompt them for a label for the particular motion they

had just watched. Participants were encouraged to label the motions, even if they were uncertain. If they had no guess, "no label" was an option. Only correctly labeled data was included in the analysis. The experiment concluded when a participant had watched all six gestures.

Since the symbol occurred at different points in time for different motions, using symbol time as the division variable means that based on robot velocity for a given motion, participants watched different amounts of motion for different lengths of time for the same 20% division across all six different motion videos.

3.1.8.2 Results

Two-hundred ten participants were recruited, which yields a sample size of fifteen per time division. For each of the seven stop time divisions (20%, 40%,...,120%, and entire motion) and two motion version groups (original and anticipatory), statistics were tallied across all motions for the average number of participants who correctly labeled the motion video. Each group in each division is an average of six motions, and all participants observed all six motions, where each motion was randomly selected as original or anticipatory.

The results in Figure 10 show that in the time range between 40% to 100% of the symbol, anticipatory motion is recognized more accurately ($p<0.05$). The data collected after the symbol time and too early in the motion is not statistically significant between anticipatory motion and the original motion. I speculate that for 20% of the symbol time, too little motion was seen to yield accurate guessing. As the motions progressed beyond the symbol time in each video, the anticipatory effects were not beneficial with respect to predicting intent because both the original and anticipatory motions had both shown enough representative motion. The data in Figure 10 supports H3. Moreover, it quantifies the time range relative to the symbol when benefits of including an early symbol in motion are gained: 40% to 100% of symbol time.

More participants correctly labeled motion when they were allowed to watch more motion. Consider Experiment 2 relative to Experiment 3. In Experiment 2, nearly all participants waited beyond 20% of symbol time to stop the video to label motion. In

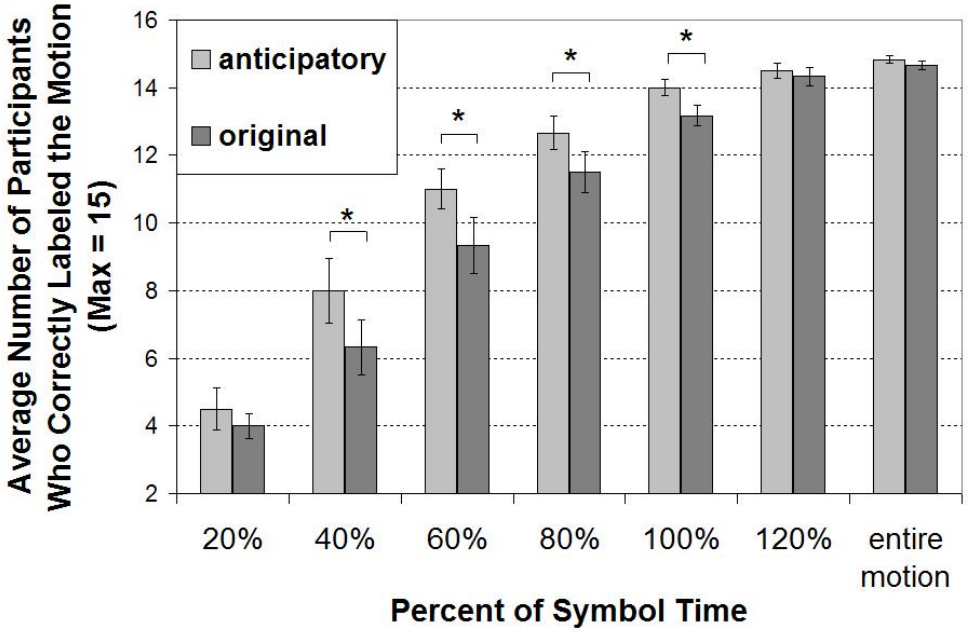

Figure 10: Average number of participants who correctly labeled the motions based on motion end time relative to symbol frame. Comparisons in each division for original and anticipatory motion. 6 motions total, 15 participants per motion. * = statistically significant ($p<0.05$).

Experiment 3, participants were forced out of their comfort zone, since they were not allowed to wait until they felt confident in their label for motion intent. However, results still demonstrate excellent performance when participants are asked to predict intent when viewing short motion segments. In contrast to Experiment 1, the 20% of symbol time correct response rates for viewing these short motion clips were often lower than that of certain static images. This can be partially attributed to viewing time. In Experiment 1, participants were allowed view a static image as long as they wanted before labeling without penalty. In Experiment 3, the short motion clips disappeared as soon as they were complete. Thus, to draw any conclusions about the precise reasons for these differences, viewing time would have had to have been a controlled variable. However, these conclusions are beyond the scope of my hypotheses.

Recognition accuracy was much higher for motions than for static images. For example, using the six motions from Experiment 3 compared to same six static symbol

frames from Experiment 1, the data shows that when participants watched all motion up to the symbol frame, recognition accuracy was 89.4% (average for all six motions). However, using only the static symbol frames, average recognition accuracy for all six motions dropped to 72.4%. Motion was easier to correctly label as more of it was seen.

3.1.9 Discussion

The value of communicative anticipatory motion is derived from the benefits of knowing motion intent sooner. In the experiments, the data showed 697 milliseconds earlier average reaction time with anticipatory motion. There are many examples of how earlier reaction time can be beneficial during interaction with humans. For example, in coupled interaction tasks, 697 extra milliseconds to respond can make the difference between not dropping the object that the human and robot are carrying together. Even in ordinary interactions, 697 extra milliseconds can possibly alter perceived responsiveness of the agent and ultimately be the difference between frustration and pleasure for the human partner in the interaction. The instrumental task utility of anticipatory motion is an important element of future work in this domain. For example, if humans interact longer because the interaction is more enjoyable with a more responsive agent, robots can learn more from humans and both partners benefit.

The main limitation of the anticipatory algorithm is that it depends on the motion having variance in the hand normal vector throughout the duration, as would be expected from human motion. For extremely simple motions, such as those where available DOFs on the robot rotate only about one axis, if the HNV is orthogonal to the sole axis of rotation, then all frames in the motion end up in the same cluster in the motion graph. In such cases, no anticipatory motion can be produced using the current formulation of the algorithm.

The other drawback for my anticipatory motion algorithm is that it relies upon presence of a hand and/or body symbol, which is not always present for all gestures. For example, yes, no, and head turning gestures will produce potentially nonsensical results from my anticipation algorithm because these gestures do not possess a single static time instant that captures the motion distinctly. Rather, in these gestures, information is spread

uniformly throughout the gesture, and the motion acquires its meaning primarily through movement.

3.1.10 Summary

This section presented an autonomous algorithm that creates communicative anticipatory motion variants from a single motion exemplar that has hand and body symbols as a part of its communicative intent. The algorithm was validated to ensure that it extracts the most salient frame (i.e. the true symbol) which is most informative about motion intent to human observers. Furthermore, the experiments showed that anticipatory variants created with the algorithm allow humans to discern motion intent sooner than motions without anticipation, and that humans are able to reliably predict motion intent prior to the symbol frame when motion is anticipatory. Finally, the time range for robot motion when the collaborative social benefits of anticipatory motion are greatest was quantified through the experimental data.

3.2 Exaggeration

3.2.1 Introduction

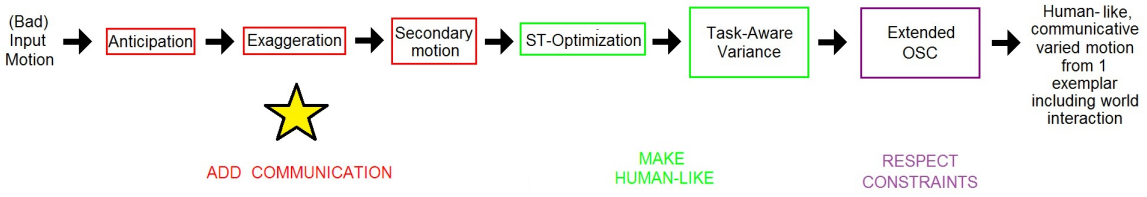

Figure 11: Section 3.2 discusses exaggeration.

Exaggeration is the second of the three communicative motion algorithms discussed in this thesis, as shown in Figure 11. It occurs second in the series of three in the high-level motion algorithm because it modifies the input motion spatially to emphasize the task to the appropriate extent. It should follow anticipation because anticipation creates a preparatory effect for the task, and secondary motion should follow exaggeration to respond appropriately to the emphasized task [64].

For a social robot to take advantage of its body like humans do, techniques are required to autonomously modify its instrumental task motion for communicative social task elements, such as directing attention. Currently, other than eye gaze and referential gestures (e.g. pointing), the relationship between robot motion and observer attention is not well understood. In addition to these explicit ways of directing attention, social robots should be able to communicate saliency and direct attention in all motions.

Exaggeration, a core principle of computer animation [41], is defined in abstract terms as developing the *essence of an idea*, where a moderate version of something is replaced by a more extreme version. In theory, if a social robot needs to attract attention to a certain body part or region, it can be exaggerated relative to the rest of its body. Exaggeration is especially important before rapid motion to ensure that observers are directed to receive information about to be communicated. Exaggeration can be accomplished by minimizing motion of less important body regions, amplifying motion of the salient regions, or both. Exaggeration can apply to size (i.e. spatial amplitude) or speed of a motion trajectory (i.e. temporal). For social robots, exaggeration is important in interactions with human partners who are unfamiliar with their channels of communication. Human interaction partners can overlook the methods that a robot uses to communicate with them, especially when communication modes are subtle. The added benefit of using motion as a communication channel is that it is a more natural method of communication than explicitly stating internal state information. The open research question is how a social robot can autonomously synthesize exaggerated motion, rather than relying upon *a priori* design.

When I discussed anticipatory motion in Section 3.1, I talked about anticipation, primary motion, and follow-through. The primary motion is the task or intent of the action that the robot is trying to accomplish, and some parts of the robot body predominantly contribute to primary motion, while others only contribute to secondary motion. I define exaggerated motion as any trajectory where there is greater contrast between body parts that contribute to primary and secondary motion. In doing so, the robot can communicate importance to human partners and observer eye will be drawn to the important action (i.e. the task the robot is performing).

3.2.2 Insight

The relationship between motion and subsequent observer attention has been studied in different contexts. For example, unexpected changes in motion direction will attract attention [65]. And when watching videos, movement features, such as magnitude and direction, impact observer eye movement [66]. Since I don't want to fundamentally change the motion type, I focus upon motion magnitude and its impact upon attention to harness this relationship for use in social robots. Thus, I developed a motion synthesis algorithm, the output of which produces reliable and consistent predictability in location of observer gaze.

3.2.3 Goal

In Section 3.2.5, concrete hypotheses for the exaggerated motion experiments will be defined, but in general, robots that display exaggerated motion direct the attention of their human partners to important body parts. Therefore, the goals are to:

- design an autonomous algorithm that creates an exaggerated variant of a given motion exemplar, that has predictable observer attention direction for longer periods of time than motions without exaggeration
- test whether human partners can perceive the difference in motion with exaggerated motion effects
- determine whether human express a preference for exaggerated or unexaggerated motion
- demonstrate the benefits of using exaggerated motion in human-robot interaction, such as directing the attention of the human partner to body parts with more energy, improving accuracy of contextual prediction, and increasing human partners' retention of interaction details
- understand whether exaggerated motion is more robot-like, human-like, or cartoon-like, than unexaggerated motion

3.2.4 Algorithm

Exaggerated motion techniques developed for cartoon or virtual characters cannot be directly applied to robots because fundamental differences exist in the real-world. For example, cartoon and virtual characters exist in worlds where physics often can be broken without consequences. In these environments, exaggerated motion is unlimited by real-world constraints such as torque or velocity limits of actual hardware. The extent of exaggeration that can be produced safely on a robot is less than on cartoon or virtual character due to hardware constraints. My algorithm was designed to transform motion through exaggeration, while respecting the boundaries of the real-world.

Exaggeration in joint-space can be produced by identifying the appropriate coordinates to modify for a given motion and then diminishing or amplifying these torque trajectories. Additional steps are necessary if the exaggerated motion must maintain certain features of the input motion (e.g. still be perceived as a representative exemplar of the input motion type). The nature and definition of exaggeration in other mediums such as cartoons and virtual animation provide insight into how to appropriately adjust a trajectory for exaggerated motion effect. For example, important body parts must be amplified and less important body parts should be diminished so that gaze is directed away from regions of the motion which are less important.

My algorithm leverages relative magnitudes and directions of torque in the given motion to exaggerate the trajectories without changing the representative exemplar type, since the relative actuation and timings of these degrees-of-freedom are known to produce one variant of the desired gesture. Logically then, relative DOF importance is determined by the amount of actuation in the degrees-of-freedom, which can be measured by torque.

Inspired by other research that examines the decomposition of motion by magnitude of actuation into coordinates in separate spaces [67, 68, 69], I exploit the variance in actuation among all degrees-of-freedom for a given motion to parameterize that motion according to a spectrum of actuation. In doing so, relative importance of all DOFs remains constant in my algorithm, which ensures that as long as physical limitations (e.g. joint limits, torque

limits) are not exceeded, the exaggerated version of the motion will also be of the same motion type as the input.

Let q_j be the torque trajectory for DOF j of the original input motion with T equidistant time samples. For a robot with M degrees-of-freedom, the torque trajectories are organized into a column-stacked matrix $\tau = [q_0, q_1, ..., q_M]$. To identify the actuation spectrum ordered by DOF for a given motion, a singular value decomposition (SVD) is performed on the covariance matrix in Equation 30 to obtain an M x M matrix of eigenvectors, denoted U, and an M x M matrix with eigenvalues along the diagonal, denoted Λ. The magnitude of the eigenvalues corresponds to a measure of the torque variance within the motion (i.e. larger eigenvalue indicates more torque variance for that specific DOF in the motion).

$$SVD((\tau - \mu)^T(\tau - \mu)) \tag{30}$$

where,

M = number of DOFs

μ_M = mean torque of DOF M for the entire trajectory

μ_1 = mean torque of DOF 1 for the entire trajectory

T = number of equidistant time samples in the original trajectory

μ = M x T stacked matrix, each column is $[\mu_1 ... \mu_M]^T$

τ = M x T column stacked matrix of command torques for the original trajectory

The largest gap in the distribution of the eigenvalues defines a threshold, λ_{th}, which separates the corresponding eigenvectors into mostly actuated and near-unactuated eigenvectors. For reference, typically less than ten eigenvectors exist in the mostly actuated set. The parameter α_{algo} quantifies the amount of exaggeration for all DOFs in torque space. Larger values of α_{algo} equate to more exaggeration. Since exaggeration has a "polarizing" effect upon motion, pushing motion toward extremes (i.e. highly actuated increase in actuation, near-unactuated become more diminished), all eigenvalues along the diagonal of Λ are modified according to the following four rules for DOF j, since $\alpha_{algo} \geq 1$:

1. If $\lambda_j < \lambda_{th}$ and $\lambda_j < 1$, then $\lambda_{j_{new}} = \lambda_j^{\alpha_{algo}}$.

2. If $\lambda_j < \lambda_{th}$ and $\lambda_j > 1$, then $\lambda_{j_{new}} = \lambda_j^{-\alpha_{algo}}$.

3. If $\lambda_j > \lambda_{th}$ and $\lambda_j < 1$, then $\lambda_{j_{new}} = \lambda_j^{-\alpha_{algo}}$.

4. If $\lambda_j > \lambda_{th}$ and $\lambda_j > 1$, then $\lambda_{j_{new}} = \lambda_j^{\alpha_{algo}}$.

These four rules are systematically designed to divide the two subspaces of a trajectory: primary and secondary motion. Rules one and two diminish the minimally existent actuation in near-unactuated torques in the original motion (i.e. diminish secondary motion); rules three and four exaggerate the torques in highly-actuated coordinates (i.e. amplifying primary motion). The composite effect of all four rules create the contrast necessary for exaggeration.

A new diagonal matrix, denoted Λ_{new}, is formed from the new eigenvalues by replacing the eigenvalues in Λ by their corresponding amplified or diminished versions. Along with Λ_{new}, the original eigenvectors are used to determine the new torque matrix of exaggerated and diminished torques as shown in Equation 31. τ_{new} is the torque trajectory that is commanded to robot actuators to produce exaggerated motion.

$$\tau_{new} = U^T \Lambda_{new} U + \mu \tag{31}$$

where,

τ_{new} = torque trajectory with exaggeration

Λ_{new} = M x M diagonal matrix of modified eigenvalues

U = M x M column stacked matrix of eigenvectors

The upper bound on α_{algo} is motion dependent and is a function of the maximum torque limits of the robot actuators. For safety, I set the upper bound on α_{algo} to be the minimum value that would cause any motor to exceed its torque limit.

3.2.5 Hypotheses

I have several hypotheses about the algorithm that I developed, the exaggerated motion it produces, and the benefits that it brings to social robot interaction with a human partner. These hypotheses are as follows:

- H1: Humans can perceive the difference between exaggerated and unexaggerated motion.
- H2: Exaggerated motion will appear to be more cartoon-like than unexaggerated motion.
- H3: Exaggerated motion increases the pleasure the human experiences from interaction with a social robot as compared to interaction with a robot that exhibits unexaggerated motion.
- H4: Exaggerated motion will enable human partners to remember more of the interaction details.
- H5: Exaggerated motion changes which body parts are salient for a given motion, and can be used to direct attention through exaggerating parts of the body that should be noticed and diminishing body parts that should be ignored.

3.2.6 Experiment 1: Testing Memory

To test the five hypotheses in Section 3.2.5, I conducted two experiments using a storytelling task. Storytelling was suitable for the experimental goals because it minimized the intellectual load of the human in the interaction, which allowed participants to devote more attention to the robot. It engaged the human enough to prevent overanalysis of motion. Furthermore, storytelling provided a context for the interaction, which facilitated testing the effects of exaggerated motion on memory, to show instrumental HRI benefits.

3.2.6.1 Experimental Design

For use in both experiments, one of Aesop's fables was adapted to include the robot as a main character. Thirteen motions were designed to match the computer-synthesized story spoken in the background, each motion was timed to a particular sentence, so that the robot acted out what the narrator said. Contrary to other work in the storytelling domain (e.g. [70]), I tested the effects of human gaze, when the robot was not being responsive to the human. The robot was like an actor in the story, rather than a storyteller or narrator. So

as not to elicit attention and potentially corrupt the human eye gaze data being recorded, the robot made no intentional eye contact with participants, nor did it speak any words. Furthermore, to test some of my hypotheses, context was completely omitted from the story. No objects were used, and no other characters in the story physically existed while the robot acted out the story.

The experiments included two versions of these 13 motions: the original unexaggerated motions (UN), and one exaggerated version per motion (EX), created using the algorithm described in Section 3.2.4.

The 13 motions were shown in the same order to all participants because the context of the experiment was a story. The 13 motions (in story order) were: walk, look up, reach, look down, beckon, scan, shucks, grrr..., phooey, leave, point, hands open, and wave.

The robot remained idle during the first two sentences of the story, to give participants a small period of time to get acclimated to the robot appearance and narrator voice. This period of time is excluded from my data and analysis, since the motions during this time were identical for all participants.

Four different experimental conditions were created for the story. They had identical story text, but differed in the motion types that accompanied the speech.

- AE: All motions in the story seen by participants were exaggerated.
- EU: The first seven motions were exaggerated, and the last six motions were unexaggerated.
- UE: The first seven motions were unexaggerated, and the last six motions were exaggerated.
- AU: All motions in the story seen by participants were unexaggerated.

Experiment 1 was designed to test whether exaggerated motion improves recollection of interaction with the robot. 54 participants (36 male, 18 female) between the ages of 19 and 30 were recruited, which yields 13-14 participants per experimental condition. 29.6%

of participants had previous experience with any robot, and 31.5% of participants had participated in a research study before.

Each participant sat in a chair approximately 193.5 centimeters (cm) away from the robot and watched one of the four different story conditions. A square table (86 cm per side) stood between the robot and human, so that 94 cm of free space existed for the robot to perform its motions. The table was not used in Experiment 1 but was necessary as a control, so that conditions would be as close as possible to Experiment 2. The participants were not aware of the story before the experiment (unless they knew the original Aesop version), and they were never shown text of what the narrator was saying until the experiment was over. Prior to beginning the story, the narrator volume was adjusted to each specific participant's preference level by sounding a click through the narrator's speakers.

After participants watched one of the four story conditions, they rated the story motions according to sixteen variables, each on a 7-level Likert scale. The sixteen variables were: subtle, entertaining, realistic, exaggerated, expressive, stiff, accentuated, cartoon-like, life-like, emphatic, emphasized, natural, noticeable, engaging, smooth, and human-like. The opposite end of each Likert scale was annotated with the corresponding variable preceded by the word 'not.' A Likert value of 1 corresponded to the 'not' condition for all Likert variables.

After completing the Likert questionnaire, participants answered six fill-in-the-blank questions and three short answer (SA#) questions. The six fill-in-the-blank questions were taken verbatim from the text that the narrator spoke, so that the objective was to fill in the exact word spoken from the story. Three questions were taken from each half of the story so that in conditions EU and UE, participants answered three fill-in-the-blank questions each from the UN and EX portions of the story. The answers were paired in both story halves so that these three answers were a location, an object, and an emotion. Unbeknownst to participants, the correct answer pairs were synonyms in the story context. I.e.,

- Location: High-above and vine.
- Object: Grapes and fruit.

- Emotion: Discouraged and unhappy.

These pairings tested the effect of exaggeration on memory substitution. In conditions where the same participant saw motions with and without exaggeration (i.e. EU and UE), I wanted to test whether people more frequently substitute one answer for the other when the accompanying motion is UN or EX. A full analysis of this effect requires a high failure rate on correct responses to the fill-in-the-blank questions or a large number of participants so that enough data is collected to use appropriate statistical techniques.

The three short answer questions were designed to test H4:

- SA1: What was the title of the story that the robot just told you?
- SA2: What was the moral of the story?
- SA3: What was the color of the object that the robot was trying to reach?

SA1 and SA3 were not explicitly mentioned during the course of the story, and they were designed as trick questions, also to test memory substitution. The moral of the story was explicitly mentioned during the 'hands open' gesture, which was the second to last motion in the story.

Following the short answer questions, participants were asked to tell their favorite part of the story and their favorite motion from the story, and they were prompted for reasons why they selected these as their favorites. Any blank answers on the short answer or fill-in-the-blank questions were followed-up during the post-story interview to clarify whether the participant intentionally left the answers blank. All participants were asked to estimate the percentage of speech they understood, and data from any participants who understood less than 85% of the speech (by their own estimate) was excluded.

At the end of Experiment 1, the following data existed per participant:

- Subjective ratings (1-7) for 16 Likert variables.
- Six fill-in-the-blank question answers.
- Three answers to short answer questions.

- Favorite part of the story and associated reason.
- Favorite motion from the story and associated reason.

3.2.6.2 Results

3.2.6.2.1 EX Improves Interaction Performance

In support of H3, human-robot interaction performance was measured by the success with which participants were able to remember the story over a short period of time. Thus, evaluation of performance was based upon the answers to the fill-in-the-blank and short answer questions.

Table 2: Correct fill-in-the-blank question answers and percent of participants in each of four conditions who correctly answered the question. Two columns furthest to the right (UN and EX) exhibit results for percent of correct fill-in-the-blank question responses grouped according to motion associations are shown cumulatively across all four conditions.

Answer	AE	EU	UE	AU	EX	UN
high above	71.4	84.6	30.8	42.9	77.8	37.0
grapes	92.9	76.9	38.5	64.3	85.2	51.9
discouraged	85.7	69.2	69.2	64.2	77.8	66.7
fruit	75.9	30.8	69.2	42.9	74.1	37.0
unhappy	85.7	61.5	84.6	71.4	85.2	66.7
vine	85.7	53.8	92.3	64.3	88.9	59.3

The 6 correct fill-in-the-blank question answers were verbatim from the narrator speech (Table 2). Any answer other than the exact text spoken by the narrator was counted as an incorrect answer. Each of the six fill-in-the-blank answers were associated to one sentence and one motion in the story. Each participant saw only either an exaggerated or unexaggerated motion for each specific sentence that accompanied each motion. As a result, question answers can be grouped based upon the type of motion seen during the story when the narrator spoke the associated sentence (i.e. UN or EX). The percent of correct fill-in-the-blank question responses grouped according to motion associations are shown cumulatively across all four conditions in the two columns furthest to the right in

Table 2 (i.e. the UN and EX columns).

Comparing the columns of conditions with some exaggerated motions (AE, EU, and UE) to the column of data from participants who saw only unexaggerated motions (AU), more participants remembered the story better when they watched some exaggerated motions. Observing the two columns on the right in Table 2 where the participant responses are grouped according to actual motion type (EX or UN) seen while the sentence with the correct answer in the story was spoken, the trend is stronger, which demonstrates the benefit that exaggerated motion helps people to remember the story better.

Table 3: Percentages of incorrect answers where fill-in-the-blank question answer pairs were substituted across categories of (1) location {high above, vine}, (2) object {grapes, fruit}, (3) emotion {discouraged, unhappy} in each of four story conditions. EX and UN columns are percent of incorrect fill-in-the-blank question responses substituted across contextual synonyms, grouped according to the motion type executing when the substituted word was heard. The left column still defines the question by the intended correct response.

Answer	AE	EU	UE	AU	EX	UN
high above	75.0	50.0	66.7	12.5	71.8	31.3
grapes	0.0	66.7	75.0	20.0	37.5	43.3
discouraged	100.0	50.0	75.0	20.0	87.5	35.0
fruit	66.7	77.8	75.0	25.0	72.2	37.5
unhappy	0.0	80.0	50.0	0.0	40.0	25.0
vine	100.0	66.7	100.0	20.0	83.3	60.0

In Table 3, EX and UN are accumulated over the motion type that was executing when the substituted word was heard; this is different than in Table 2 because for percent correct, the columns entitled UN and EX are tallied over the motion type associated with the text in the question.

In Table 3, when fill-in-the-blank answers were incorrect, the contextual synonym was more frequently substituted for the correct response, when the sentence of the synonym was associated with an exaggerated motion. These substitution results suggest that participants retained the words from a particular story sentence in memory better when an exaggerated motion was associated with that sentence. The higher percent incorrect and lower

substitution rates for associated unexaggerated motions imply that, in general, words and sentences are not retained as well when the associated motion is unexaggerated. Thus, exaggerated motion has benefits with respect to retaining interaction details and human memory in storytelling applications.

Similar trends exist for the data collected from the short answer questions. The moral of the story was explicitly mentioned during the 'hands open' gesture, which was the second to last motion in the story. The moral of the story was "Do not speak disparagingly of things that you cannot attain." Table 4 shows the percent of correct answers as a function of all four conditions and the two most common substitutions grouped according to the motion type seen when the participant-given answer was spoken by the narrator in the story.

Table 4: Participant answers for the short answer question regarding the moral of the story. EX and UN columns are question responses accumulated across all four conditions, grouped according to the motion type that was executing when the participant-provided answer was spoken by the narrator. Substitution 1 = word "attain" exchanged for the word "reach." Substitution 2 = words "speak disparagingly of" exchanged for "be discouraged by".

Answer	AE	EU	UE	AU	EX	UN
Verbatim Correct	35.7	7.7	30.8	0.0	33.2	3.8
Substitution 1	35.7	15.3	7.7	0.0	25.5	3.8
Substitution 2	7.1	30.7	23.1	7.1	19.0	15.1

Four instances of the word "reach" were heard in the story, and all occurred during the first half of the story. One instance of "discouraged" occurred in the story, and it occurred in the first half of the story. The two substitutions in Table 4 are "attain" replaced by "reach" and "speak disparagingly of" replaced by "be discouraged by." The short answer question about the story moral shows the same results as for the fill-in-the-blank questions. Exaggerated motions were associated to both (1) more correct responses and (2) a higher percentage of the answer substitutions for incorrect responses made with contextual-synonyms heard during exaggerated motions.

The two short answer questions that asked participants to tell the story title and color of

Table 5: Participant answers for the short answer question regarding the color of the object being reached for in the story. Not in Story = participant explicitly wrote that the narrator did not say.

Answer	AE	EU	UE	AU
Not in Story	21.4	0.0	7.7	0.0
Blank; Knew	64.3	15.3	7.7	0.0
Blank; Didn't know	0.0	7.7	0.0	42.9
Purple	14.3	61.5	38.9	7.1
Green	0.0	15.3	38.4	0.0
Other	0.0	0.0	0.0	42.9

the object that the robot was trying to reach were not explicitly given in the story. Across all four conditions, the participants who saw only EX had the highest correct response rates in both Tables 5 and 6. Furthermore, feasible answers for color of the grapes such as green or purple were concentrated in the conditions that had at least half of the motions exaggerated (i.e. AE, EU, and UE). Responses for the blank answers were sorted during the post-experiment interviews: *knew* (participant knew that the narrator did not say the answer) and *didn't know* (participant left the answer blank because they thought the narrator had mentioned the answer, and also thought that they couldn't remember). Participants who answered ambiguously or indicated uncertainty were placed into the "didn't know" category.

Table 6: Participant answers for the short answer question regarding the story title. Not in Story = participant explicitly wrote that the narrator did not say.

Answer	AE	EU	UE	AU
Not in Story	21.4	0.0	0.0	0.0
Blank; Knew	64.3	46.1	46.1	7.1
Blank; Didn't know	7.1	7.7	0.0	28.6
Adaptation of an Aesop Fable	7.1	46.1	46.1	14.3
The Robot & the [Sour] Grapes	0.0	0.0	7.7	42.8
Other Incorrect	0.0	0.0	0.0	7.1

The final short answer question that asked participants to tell the title of the story was not explicitly given in story. The correct responses and other responses are displayed in

Table 6. Across all four conditions, the participants who saw only exaggerated motions had the highest correct response rates. Responses for the blank answers were sorted during the post-experiment interviews. The qualifications on the blank answers (i.e. "knew" and " didn't know") clarify whether the participant actually knew that the narrator did not say the title. Participants who answered ambiguously with answers that indicated uncertainty were placed into the "didn't know" category.

Based on the results presented, interaction task performance was improved with exaggerated motion because exaggerated motion led to higher correct response rates for questions about the interactive experience with the robot. There is a distinct benefit of using exaggerated motion, when remembering details about the interaction is important.

3.2.7 Experiment 2: Testing Gaze Direction

3.2.7.1 Experimental Design

The same exact story, motions, experimental layout, and four experimental conditions were repeated in Experiment 2. However, the faceLAB[2] system was used to track participant eye gaze direction. Using a model of the real-world, calibrated through actual measurements, and using the exact robot model from which the hardware was manufactured, two time-varying trajectories were captured for the duration of the entire story, which allowed the determination of the exact location on the robot's body that each eye of the participant (left and right) was looking at during the story.

The two faceLAB cameras that track eye gaze direction were placed upon the table edge opposite from the human, 95 cm away from the robot center. The table height was set at the same height as the robot base. Participant height was adjusted before each trial to maximize the image of the participants' heads in the camera field-of-view. The cameras were adjusted angularly (not positionally) before each trial, and a calibration of both cameras was run for each participant prior to the story. Participants were asked to sit as still as possible during Experiment 2 to remain in the camera field-of-view. If they asked, participants were told that more could be explained about the cameras after the

[2]faceLAB is a trademark of Seeing Machines. All rights reserved.

experiment. The duration of the story was less than three minutes, so that it was easier for participants to minimize body motion for this duration.

While each participant was watching the story, the eye cameras captured the intersection of their eye gaze vectors with two virtual planes, (i.e. four data points in real-time for each time sample). These planes were static in the virtual environment, and upon completion of the camera calibration, the time varying human eye centroids were also known in the virtual world in real-time. The two planes were parallel to each other, both were perpendicular to a directly overhead view of the robot and participant, and the normal vector of each plane was aligned collinearly with two points: the robot base centroid and the centroid of the participant's chair. Thus, the planes were parallel with an eye vector of someone looking at the robot face-to-face from the front. The planes were located at horizontal distances of 99.5 cm (table edge) and 193.5 cm (robot centroid) from the participant. Prior to Experiment 2, a pilot study was conducted for equipment testing and data integrity. Data from fifteen participants, with and without glasses, was tested to ensure the accuracy of the models and eye gaze data.

Prior to beginning the story in Experiment 2, participants were given the story text to read so they would be familiar with the story. This was done to maximize attention on the motions and minimze off-robot glances while trying to listen intently to understand speech.

For Experiment 2, 68 participants (44 male, 24 female) between the ages of 19 and 30 were recruited, which yields 17 per experimental condition. 29.4% of participants had previous experience with any robot, and 30.9% of participants had participated in a research study before.

After the story finished, participants were asked to give their opinions and rate the robot according to each criteria using the same 16 variable Likert. Participants were then prompted for their favorite part of the story, their favorite motion from the story, and their reasons for both.

In place of the short answer and fill-in-the-blank questions from Experiment 1, participants were seated at a virtual model of the robot and given two controls: + (plus) and -

(minus). These buttons directly controlled the value of α_{exp}, which indirectly controlled the value of α_{algo} for the exaggeration using a mapping function. Participants were asked to use the buttons to select the value of α_{exp} that produced the motions that they considered to be:

- HL: Most Human-like
- CL: Most Cartoon-like
- RL: Most Robot-like
- VP: Most Visually Pleasing
- LB: They Liked Best

Order of all the five values was randomized for each participant, but each participant saw the same order for all motions. Motion order was randomized for each participant. In order to intentionally bias the experiment away from the expected results, the value of α_{exp} was always initially set at the value, which corresponded to unexaggerated motion. Each press of the plus or minus button was mapped to increment the value of α_{exp} by 0.01. The range on α_{exp} was limited between positive and negative one, which included a wide range of motions, such as motions where subsets of DOFs were static for all time (i.e. no motion) or motion at maximum exaggeration without exceeding the torque limits of any motor on the robot. This part of the experiment was not conducted on hardware for safety reasons.

At the conclusion of Experiment 2, the following data existed per participant:

- Subjective ratings (1-7) for 16 Likert variables.
- Four trajectories of time-varying left and right eye gaze data during the story.
- Five subjective α_{exp} settings per motion, which correspond to preferences for most human-like, most cartoon-like, most robot-like, most visually pleasing, and liked "best."

- Favorite part of the story and associated reason.
- Favorite motion from the story and associated reason.

3.2.7.2 Results

Data from both experiments is used to support the remaining four hypotheses presented in Section 3.2.5 because this data leads to conclusions not specific to either experiment. For example, discussion of the statistically significant variables from the Likert scales is distributed so that results are presented in support of the appropriate hypotheses. Combining data from both experiments also helps obtain the data quantities necessary to achieve statistical significance because variance of subjective data is often large. The mean values from the Likert variables are shown in Table 7.

Table 7: Mean subjective responses for the sixteen Likert scale variables in the two exaggerated motion studies. Likert scale ranges from 1 (not) to 7 (variable).

Variable	AE	EU	UE	AU
Subtle	2.63	3.48	3.40	5.35
Entertaining	5.20	4.02	3.92	3.67
Realistic	2.87	3.84	3.59	4.21
Exaggerated	4.86	3.82	3.86	3.13
Expressive	4.59	4.08	4.22	3.39
Stiff	3.18	3.26	3.31	3.49
Accentuated	4.62	2.88	3.55	2.90
Cartoon-like	3.34	2.40	2.72	2.50
Life-like	3.24	3.85	3.70	3.62
Emphatic	4.36	3.86	3.77	3.05
Emphasized	4.47	3.76	3.57	2.87
Natural	3.90	3.52	3.50	3.70
Noticeable	4.91	3.92	4.00	3.98
Engaging	5.24	3.84	3.67	3.53
Smooth	5.49	5.43	5.32	5.26
Human-like	4.12	3.56	3.72	3.67

3.2.7.2.1 EX and UN are Perceptibly Different

A good measure of success is to test the algorithm to ensure it accomplished the goal of creating motion that is exaggerated. Although many of the Likert scales achieved statistical

significance between participants in the AE and AU groups, three of the sixteen scales add evidence to support that my algorithm creates exaggerated motion: subtle, exaggerated, and accentuated.

First, ANOVAs were conducted on all the data from both experiments assuming that all four conditions (AE, EU, UE, AU) belong to the same distribution. F_{crit} for each variable in the Likert is 2.681. Respectively, the F-values achieved from these ANOVAs are 23.0 (subtle), 3.18 (exaggerated), and 4.85 (accentuated). Thus, for each measure at least one of the four experimental conditions was statistically different than the rest of the data.

Post-hoc pairwise t-tests were performed. For the subtle variable, three of the six possible pairings for all four test conditions exhibit statistically significant results ($p<0.01$): (AE, AU), (EU, AU), and (UE, AU), which means that participants who saw at least half of the motions modified by my algorithm indicated that the motions were less subtle than participants who saw only the original motions.

For both the accentuated and exaggerated Likert variables, three of the six pairings exhibit statistically significant results ($p<0.01$): (AE, EU), (AE, UE), and (AE, AU), which means that participants who saw only motions modified by my algorithm indicated that the motions were more accentuated and more exaggerated than participants who saw at least half of the original motions. Thus, I concluded that motions produced by my algorithm are less subtle, more accentuated, and more exaggerated than the motions input into the algorithm.

3.2.7.2.2 EX Appears More Cartoon-like Than UN

A good measure of success is whether the output of the algorithm maintains the same qualities and characteristics of its inspiration. H2 is a logical hypothesis, since the inspiration for exaggerated motion comes from animated and virtual characters. And by testing whether EX is more cartoon-like than UN, I also am evaluating whether my algorithm accomplished one of its most fundamental goals. The hypothesis H2, which claims that exaggerated motions produced by my algorithm are more cartoon-like, is supported by data from two of the Likert variables: realistic and cartoon-like.

ANOVAs conducted on all the data from both experiments assuming that all four conditions (AE, EU, UE, AU) belonged to the same distribution yielded F-values of 3.84 and 4.26 for realistic and cartoon-like respectively. Both of these values exceeded the F_{crit} of 2.681, which indicates that for both of these measures at least one of the four experimental conditions was statistically different than the rest of the data.

For the realistic Likert variable, three of the six possible pairings exhibited statistically significant results ($p<0.05$): (AE, EU), (AE, UE), and (AE, AU). These three results were the three pairings that compare participants who saw only motions produced by my algorithm with participants who saw at least half of the original motions. And for the cartoon-like Likert variable, one of the six pairings exhibited statistically significant results ($p<0.02$): (AE, AU). These results were the pair that compares participants who saw only motions produced by my algorithm with participants who only original motions. From these results, I concluded that motions exaggerated by my algorithm are less realistic but more cartoon-like than the motions input into my algorithm.

There is further quantitative evidence relevant to H2 from the data of Experiment 2 where participants selected α_{exp} values that pertain to their subjective ratings of most human-like, most cartoon-like, most robot-like, most visually pleasing, and liked best (HL, CL, RL, VP, and LB, respectively).

A singular ANOVA was performed using all the data in Table 8. The F-value of 59.71 from this analysis was greater than F_{crit} of 2.525, which means there was statistical difference between at least one of these measures from the rest of the data.

The analysis was then performed assuming that the qualities of HL, CL, RL, VP, and LB for α_{exp} are motion dependent. Thus, thirteen separate ANOVAs were performed, one for each motion from the story, assuming all five qualities belonged to the same distribution. 13 of 13 ANOVAs yielded F-values greater than F_{crit}, which means that for each motion, subjective settings for α_{exp} according to each of the five groups held at least one statistically independent group.

The results from the motion-by-motion analysis for α_{exp} led me to suspect that these subjective measures were motion independent. To show that these subjective measures for

Table 8: Average α_{exp} Participant Responses for Most Human-like (HL), Most Cartoon-like (CL), Most Robot-like (RL), Most Visually Pleasing (VP), and Liked Best (LB).

Motion	HL	CL	RL	VP	LB
Walk	0.131	0.910	0.056	0.877	0.844
Look Up	0.230	0.967	0.074	0.812	0.754
Reach	0.164	0.787	0.041	0.385	0.501
Look Down	0.607	0.959	0.434	0.836	0.869
Beckon	0.230	0.956	0.066	0.771	0.574
Scan	0.517	0.869	0.246	0.689	0.877
Shucks	0.680	0.851	0.098	0.762	0.844
Grrr...	0.443	0.967	0.197	0.911	0.899
Phooey	0.639	0.926	0.080	0.836	0.756
Leave	0.541	0.754	0.221	0.803	0.639
Point	0.508	0.853	0.148	0.836	0.910
Hands Open	0.623	0.951	0.180	0.885	0.541
Wave	0.320	0.861	0.107	0.615	0.525
Average	0.433	0.893	0.150	0.771	0.733

Table 9: P-values from post-hoc pairwise t-tests of the data in Table 8 for all five measures. Tests that fail statistical difference $p<0.05$ are shown in gray. x = captured elsewhere in the table.

	HL	RL	VP	LB
CL	0.008	0.004	0.024	0.016
RL	0.023	x	0.009	0.012
VP	0.021	x	x	0.072
LB	0.027	x	x	x

exaggerated motion were truly independent of motion when equalized on the scale of α_{exp}, 10 more pairwise grouping data averages across all motions were performed to compare all possible pairings of HL, CL, RL, VP, and LB. The p-values from these tests are shown in Table 9.

Table 9 shows that across all motions, the qualities of human-like, cartoon-like, and robot-like were statistically different from each other and all other variables in the study. Only statistical difference tests failed between visually pleasing and liked best (shaded gray in Table 9, which indicates that these two subjective measures for exaggerated motion may not come from independent distributions.

Previous statistical tests have already shown that human-like, cartoon-like, and robot-like were statistically independent of motion. For completeness, 10 post-hoc pairings were performed for each motion, using all possible pairs of the five subjective qualities, yielding a total of 130 post-hoc pairings. 122 of the 130 pairings were statistically significant ($p<0.05$). The eight pairings that failed statistical difference tests were all between visually pleasing and liked best for the motions of reach, beckon, scan, shucks, phooey, leave, hands open, and wave. From these results, I concluded that the values of α_{exp} (and α_{algo}) are not motion dependent. By modulating α_{exp} (and therefore, α_{algo} in the algorithm) I can consistently create motions which are more robot-like, more human-like, and more cartoon-like.

Average values across all motions for the α_{exp} setting from the experiment are shown in the bottom row in Table 8. For consistency, prior to the experiment, all unexaggerated, input motions that were to be used in the story were selected to have α_{exp} values between -0.2 and 0.0, and no α_{exp} for any exaggerated motion used in the story was less than 0.7. For reference, exaggerated motion values with an α_{exp} value of 1.0 correspond to exaggerated motion so that any more exaggeration would cause at least one motor to exceed its torque limits.

The average human subjective value for α_{exp} of 0.893 for cartoon-like is consistent with the Likert results, which also found that the exaggerated motions were cartoon-like; it indicates that exaggerating motions by using my algorithm adds a cartoon-like quality to the motions. Robot-like motion that was consistent with participants' expectations tended to lack the quality of exaggeration, which may help explain the results that are discussed in Section 3.2.7.2.3 regarding entertainment and engagement of exaggerated motion. Human-like motion tended to exhibit moderate levels of exaggeration, not near the torque limits for any motors, but also far from the values of robot-like exaggeration.

For the story, values of α_{algo} (and therefore, α_{exp}) were used to create each exaggerated motion that was shown to participants. Similarly, each unexaggerated motion in the story had a corresponding value of α_{exp}. I can use statistical tests based on the α_{exp} data collected in the last part of Experiment 2 to reinforce the conclusions drawn from the Likert scales

by testing whether values of α_{exp} used in the story coincide with participants' subjective settings. Therefore, I can determine whether my exaggerated or unexaggerated story motions are robot-like, human-like, cartoon-like, visually-pleasing, or liked best, without directly asking participants about the motions.

To do so, the final analysis performed in support of H2, was to evaluate 130 additional pairwise t-tests (10 per motion) for the α_{exp} values that correspond to the story motions and the distributions provided in Experiment 2 based upon HL, CL, RL, VP, and LB, to find out if the UN or the EX motions used in the story were statistically different from the human subjective measures' distributions. The ten pairings were all possible combinations of one member from each of the sets: {UN, EX} and {HL, CL, RL, VP, LB}. 65 of 65 pairwise t-tests for the UN motion pairs had $p<0.05$, which indicates that the choice of unexaggerated motion in the story did not coincide with participants' distributions of any of the five measures. The 26 motion pairs for EX with HL and EX with RL had $p<0.05$, which indicates that the exaggerated motions used in the story were not robot-like or human-like, using participants' subjective responses as the measures of these two variables. However, for 13 of 13 (EX, CL) pairs there was no statistical difference ($p<0.05$) between the exaggerated story motion and participants' expectations of cartoon-like motion.

3.2.7.2.3 Humans Prefer EX Over UN

Supporting H4, two Likert variables provided evidence that humans prefer exaggerated motion: entertaining and engaging. ANOVAs conducted on all the data from both experiments assuming that all four conditions (AE, EU, UE, AU) belonged to the same distribution yielded F-values of 3.56 and 3.17 for entertaining and engaging respectively. Both of these values exceeded the F_{crit} of 2.681, which indicates that for both of these measures at least one of the four conditions was statistically different than the rest of the data.

For the both the entertaining and engaging Likert variables, three of the six post-hoc t-test pairings exhibited statistically significant results ($p<0.05$): (AE, EU), (AE, UE), and (AE, AU). Participants who saw at least half of the story motions exaggerated through modification by my algorithm indicated that the motions were more entertaining and more

engaging than participants who saw only original motions. Thus, motions produced by my algorithm are more entertaining and more engaging than the motions input into the algorithm.

Returning to the subjective α_{exp} data discussed in Section 3.2.7.2.2, in which participants were asked to find values of α_{exp} that are most human-like, most cartoon-like, most robot-like, most visually pleasing, and liked best, only 3 of 13 (EX, VP) and 5 of 13 (EX, LB) pairs showed statistical significance $p<0.05$; thus, for the majority of motions exaggerated motion was not statistically different than the motions that participants found most visually pleasing and the motions they liked best.

Other data that supports H4 and a preference for exaggerated motion was derived from the interview questions regarding favorite part and favorite motion from the story. If all four conditions are included (i.e. AE, EU, UE, AU) from both experiments, then 62.2% of participants chose a favorite part of the story associated with an exaggerated motion, and 64.8% of participants selected an exaggerated motion as their favorite from all 13 motions in the story. However, in only two conditions did participants actually have a choice (i.e. for EU and UE). Excluding the other two conditions where participants had no choice, these percentages increase to 75% and 80% respectively. When participants selected an exaggerated motion, they described their favorite parts or motions as "animated," "expressive," or "emotional." Unexaggerated motions were typically selected as favorites because they seemed more "appropriate."

3.2.7.2.4 EX Can Be Used To Direct Attention

The final set of results was meant to address H5, and whether exaggerated motion can be used to direct attention to salient body parts. The gaze data collected was a pair of time-varying trajectories where the eyes intersect the robot body throughout the story. To appropriately analyze the data from the faceLAB system and draw meaningful conclusions from the collected gaze data, a measure of the amount of exaggeration that exists at locations on the robot body or for each robot body part was needed, since the hypothesis depends on determining whether people look at exaggerated body parts more frequently

or for greater lengths of time.

α_{algo} is not a sufficient measure for exaggeration because α_{algo} corresponds to exaggeration within joint-space. The exaggeration for a particular body part is a function of all of the parent joints on the hierarchy relative to the body part. Since exaggerating a DOF increases the energy of all the children body parts in the kinematic chain, the appropriate measure should account for the chain of exaggerated DOFs and their respective energy increases. Therefore, cumulative energy ratio (CNR, EX over UN) was used to represent exaggeration in Cartesian space (shown in Equation 32).

The robot hardware was discretized into 12 segments, each corresponding to non-overlapping regions. These twelve segments were named after the body parts they represent: left hand, right hand, left forearm, right forearm, left bicep, right bicep, torso top, torso mid, neck, head, left ear, and right ear. Using the data from Experiment 2, the intersection of each of the two eye gaze trajectories with the twelve body part segments was determined and accumulated over all participants. Using these trajectories, percent of total trajectory time that each participant spent watching each body part was determined for each motion in the story. Assuming the data for both groups of motions (UN and EX), all body parts, and all motions belong to the same distribution, the ANOVA yielded an F-value of 67.4, which was greater than $F_{crit} = 3.9$, and therefore post-hoc analysis was required.

Assuming each of the distributions are different according to body part and motion from the story, (UN, EX) data pairs exist. Each of these 156 pairs (12 body parts x 13 motions) represent distributions for the amount of time participants spent watching a particular body part for a particular motion. 151 of 156 pairwise t-tests were statistically significant ($p<0.05$), which indicates that participants watched the same body part in the same motion for different lengths of time when observing exaggerated motion instead of unexaggerated motion. The five tests that failed to yield statistical difference were (scan, left hand), (scan, torso top), (scan, torso mid), (scan, head), and (wave, left hand). Scan and wave were both performed with the left hand; during gestures, humans focus on the symbol formed by the salient hand. Additionally, the head is near the left hand for

a significant portion of the scan gesture, which could help explain why observation time was not statistically different for EX and UN during these two gestures.

Comparing groups UN and EX includes both between-participants and within-participants data. To make a stronger claim (i.e. one that shows temporal effect), two groups can be excluded. UE and EU groups are individuals who witnessed both types of motion. Comparing UE and EU uses only within-participants data. Thus, the data in the AE and AU conditions were excluded from the analysis to determine if the same participants will change their own behavior in a short period of time based on whether they are watching exaggerated or unexaggerated motion (i.e. to analyze temporal effect). The 156 pairwise t-tests were repeated using only data from EU and UE to create the distributions. This time, 153 of 156 pairwise t-tests were statistically significant ($p<0.05$). The three pairings that failed to be statistically different with respect to percent of gaze time for a specific motion and body part, when observing exaggerated motion instead of unexaggerated motion were (scan, left hand), (scan, head), and (wave, left hand). Thus, the *same* participant changed their own attention and behavior, when watching UN or EX.

The pairwise t-tests do not provide any information about the data trend with respect to exaggeration. To test whether attention was directed toward exaggerated body parts, I used my Cartesian measure for exaggeration and plotted the average percent time watching both exaggerated and unexaggerated motions against cumulative energy ratio for the point trajectory that represents the body part centroid as a function of time (Equation 32).

$$CNR = \frac{\sum_{t=0}^{t=T_f} (v_x(t)_e^2 + v_y(t)_e^2 + v_z(t)_e^2)}{\sum_{t=0}^{t=T_f} (v_x(t)_u^2 + v_y(t)_u^2 + v_z(t)_u^2)} \tag{32}$$

where,

CNR = cumulative energy ratio

T_f = final time sample for discrete motion trajectory at time t

$v_x(t)$ = velocity of body part centroid in x-direction at time t

$v_y(t)$ = velocity of body part centroid in y-direction at time t

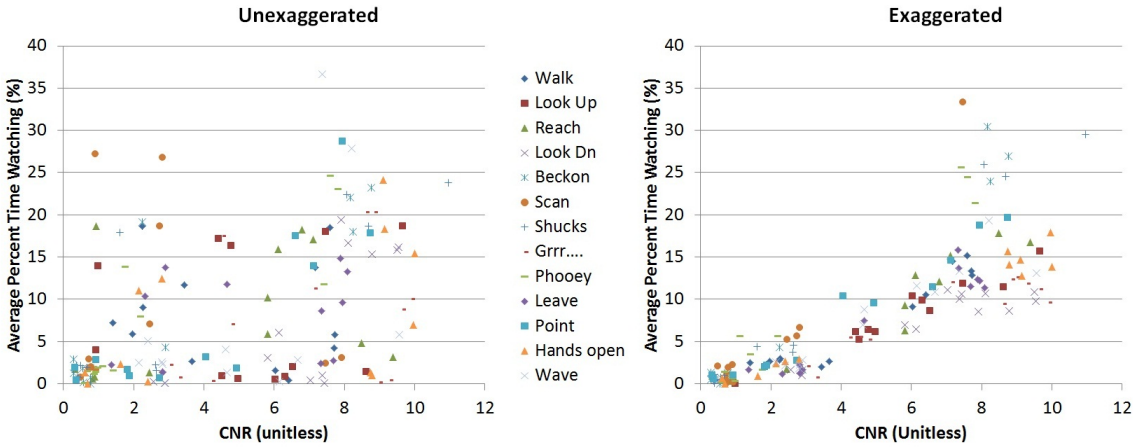

Figure 12: Average percent of time participants spent watching a specific body part in a trajectory (unexaggerated on left; exaggerated on right) vs. cumulative energy ratio for that body part centroid trajectory.

$v_z(t)$ = velocity of body part centroid in z-direction at time t

u = denotes unexaggerated motion trajectory

e = denotes exaggerated motion trajectory

The two plots of average percent time spent watching versus CNR for UN and EX are shown in Figure 12 (left and right, respectively). Since trajectories for body part motions with higher CNR are more exaggerated, the horizontal axis is a measure of increasing exaggeration. Each point on the plots represents the data for one body part from one motion. From Figure 12, on average the participants spent more time watching body parts with more exaggeration (right). The absence of a trend in Figure 12 (left) provides evidence that exaggeration produced this effect upon participant behavior.

Thus, the differences in trends exhibited in Figure 12 (left and right) indicate that exaggerated motion was used to direct attention to body parts that are exaggerated more, and attention was directed away from body parts with low exaggeration (i.e. motion that is diminished). I concluded that hypothesis H5 holds true.

3.2.8 Discussion

The algorithm presented in Section 3.2.4 provides only spatial exaggeration (i.e. exaggeration of the torques). However, I am developing temporal exaggeration, so that modifications in the timing of relative body parts can be performed autonomously in a way that also can be employed to increase engagement in the interaction and direction participants' attention to desired body parts.

Currently, only minor amounts of temporal exaggeration might be perceived when accelerations or velocities change between relative body parts. However, this is not true temporal exaggeration, which creates drastic timing changes between relative degrees-of-freedom. Temporal exaggeration would be able to emphasize salient velocity points during a trajectory, such as making zero velocity points during a gesture "pop" (i.e. stand out from the rest of the motion). Or, these points could be lengthened or shortened for dramatic pause effect, such as in a stop or shrug gesture. Furthermore, true temporal exaggeration would be able to change the trajectory time length, which currently doesn't occur with just spatial exaggeration.

Some might claim that the existence of a tunable parameter, α, is a disadvantage of my algorithm. Perhaps a systematic method for determining an "optimal" α for any motion is preferred. However, I believe that optimality is context-dependent (e.g. I provide optimal values that correspond to robot-like, human-like, and cartoon-like). Since I do not believe that one α value is optimal for all situations, α is tunable and can be exploited to control the amount of exaggeration at any given time based on whether a robot needs to attract attention or increase engagement of human partners.

3.2.9 Summary

My algorithm creates exaggerated variants of an input motion in real-time. Experimental data confirmed that (1) exaggerated motion is perceptibly different than the input motion, provided that sufficient exaggeration is added to the motion; (2) different levels of exaggeration in motion correlate to human expectations of robot-like, human-like, and cartoon-like motion; (3) use of exaggerated motion in experiments enhances the interaction through

the benefits of increased engagement and perceived entertainment. I provided statistical evidence that the added benefits of exaggerated motion also include increased retention of interaction details. Furthermore, since observers watched exaggerated parts for longer durations when motion is executing, I concluded that exaggerated motion changes the salient body parts and directs gaze away from body parts with less energy.

3.3 Secondary Motion

3.3.1 Introduction

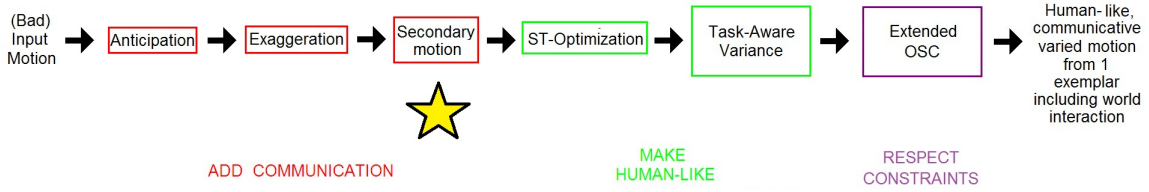

Figure 13: Section 3.3 discusses secondary motion.

Secondary motion is the last of the three communicative motion algorithms discussed in this thesis, as shown in Figure 13. It occurs third in the series of three in the high-level motion algorithm because it modifies the task spatially to add the appropriate response to the primary motion. It should follow exaggeration to respond appropriately to the emphasized task [68].

Secondary motion (a.k.a. secondary action), a principle of traditional animation, can be adapted for communication in motion. Despite usages in other literature and fields, in this thesis these two terms will be used interchangeably, and they are defined to be consistent with Lasseter's original intent. Secondary motion is the direct result of the primary, or task-oriented, action as defined by Newton's third law. From the standpoint of Newtonian physics, secondary motion results when forces on any two connected, articulated segments of a body are unequal in magnitude and/or direction. Then, motion of one body affects all other connected bodies to varying degrees and magnitudes. Secondary motion can be produced by both internal and external forces, and the effect is most noticeable when

the magnitude of the force difference between two connected bodies is large. In general, secondary motion can go unnoticed if it exists and is accurate, but it becomes noticeable when it is lacking [41, 42].

3.3.2 Insight

When robots are modeled as a hierarchy of rigid articulated bodies, secondary motion is the manifestation of Newton's third law on the articulated body, where forces from the primary motion produce coupled effects on other attached body parts. Secondary motion results from a force imbalance between parts of a body. It can also result from a difference in level of control between actuators that secure DOF positions. When the difference in level of control between body parts is greater, the secondary motion effect is more pronounced. Intuitively, DOFs that lack all control, i.e. passive DOFs, exhibit the most secondary motion.

Since secondary motion enforces a reaction in more passive DOFs from the dynamics of DOFs with greater actuation, it can be thought of as a mechanism to increase motor coordination or change the amount of influence DOFs have upon each other, just like coupling or synergies between joints on any hierarchy of articulated rigid bodies. In humans, this DOF coupling (i.e. secondary motion, coordination) occurs through muscle synergies, where sets of muscles between different joints act together to drive motion or hold the body against perturbations [71]. Coupling effects or secondary motion are also created simply due to the fact that DOFs are connected to each other. Other investigations have also concluded that an extremity body in a kinematic chain has a "natural" contribution to its motion from the parent joints in the chain [72]. Therefore, these observations support why adding secondary motion should make motion more human-like, and they yield insight as to how it might be algorithmically generated.

Secondary motion uses the natural physics of the entire kinematic chain to improve the realism by augmenting natural, passive motion to primary action. Secondary motion typically results from redundancy in a system where a subset of DOFs are unactuated or under-actuated. Consequently, a technique to produce secondary motion must exploit

the primary actuation of highly-actuated DOFs to change actuation either in magnitude or dimension in DOFs with less actuation. Logically, this reduces into a small set of subgoals that can be composed to create secondary motion: identify the passive or near-unactuated space, identify the primary or highly-actuated DOFs, and use the actuation from the highly-actuated space to move the near-unactuated space in a coordinated way with the rest of the motion (as much as possible, given system constraints).

3.3.3 Goal

Since secondary motion is a product of the laws of physics, in theory, robot movement should inherently exhibit perceptible secondary motion. If that were true, only the joints required for a particular primary motion would need to be actuated, and the others would passively exhibit secondary effects of the motion.

In reality, robots must overcome two main hardware and software constraints to produce noticeable secondary motion. First, actuator design and robot mass severely damp any secondary motion that can noticeably result from natural physics. Mechanical limitations, such as large mass, damp transients of motion [11]. And in reality, motor rotor inertia and frictional inefficiencies reduce secondary motion. Switching motors on and off during operation to attempt to emulate a more passive, unactuated response is often infeasible for safety considerations, especially in close proximity to humans. As a result, secondary motion must be induced while maintaining a finite, non-zero level of active control.

Second, the purpose of control systems is to modulate actuator response, thereby providing the internal forces necessary for motion. However, robots use control schemes that typically eliminate the nonlinear dynamics that humans advantageously exploit [73]. Furthermore, robot control systems, such as PID control, can induce artificial dynamics into trajectories (i.e. artifacts), which become perceptible in robot motion, dominating secondary motion effects, rather than producing the response consistent with Newton's third law. If hardware and control are fixed, then to overcome these challenges, the motion must be adapted to minimize exhibition of the artificial control system response (e.g. by using input command shaping) on the hardware or dynamically change control gains on

hardware (which is a safety issue).

Existing methods for generating natural motion that include secondary motion are expensive and time consuming. These methods tend to rely on using databases of natural motion trajectories with inherent secondary motion either from human motion capture data [61, 74] or created by a professional animator [75]. My goal, on the other hand, is to develop mechanisms to automatically generate natural motion for a robot. Then, any algorithm or animator can generate a primary or functional action for the robot to perform, and my techniques can automatically generate secondary motion for that action.

In Section 3.3.5, concrete hypotheses for the secondary motion experiments will be defined, but in general, my algorithm augments a given input motion with secondary motion. Therefore, the goals are to:

- create natural, compliant secondary motion from a single exemplar in real-time for humanoid robots that overcomes the hardware and software constraints that normally damp or mask secondary motion
- exploit simulation methods and passivity in actuation to create virtual secondary motion from both internal and external forces to be used as robot input command, while keeping all real-world hardware fully actuated for safety
- demonstrate that secondary motion can communicate both internal and external state parameters
- show that secondary motion can help fill-in missing context
- add evidence that motion is considered to be more natural when it includes secondary motion

3.3.4 Algorithm

I present three techniques that create natural, compliant, secondary motion for humanoid robots by overcoming the hardware and software constraints that normally damp or mask secondary motion. Advantages and disadvantages of each will be discussed in greater

detail in Section 3.3.12, but three techniques were developed because each algorithm of the three works better under different circumstances. In other cases, tools necessary for one technique might not be available, and therefore, one of the other algorithms must be used.

These approaches exploit dynamic simulation and passivity in actuation to create virtual secondary motion from both internal and external forces to be used as robot input command, while keeping all real-world hardware fully actuated. Additionally, by coupling a robot to an accurate, real-time simulation, the robot is used as both an input and output device in real-time; the direction, magnitude, point of application of an external force can be calculated; and the perceived hardware response to an external force can be altered by changing simulated characteristics such as mass, gravity, inertia tensors, or external input force magnitudes. Through cameras in the robot eyes, the robot perceives real objects in the world. Virtual objects are instantiated into the simulation based on what the cameras perceive. The simulation responds to virtual objects with characteristics different from the real world to produce different hardware responses. For example, a box with very small mass in the world can be simulated to appear heavy, and when the robot hardware lifts the box, the secondary motion will communicate the response consistent with the heavier box. Thus, secondary motion can be manipulated to communicate realistic or artificial information about the world, objects in the world, or robot capabilities.

My methods provide two key advantages. First, by simulating secondary motion virtually, I can manipulate the visual perception of physical properties of a robot, such as making a light-weight robot appear heavy or a highly actuated robot appear compliant to perturbations. Second, my methods produce secondary motion 'on-the-fly' based on the dynamic state of the robot in real-world, removing the need of authoring and storing a large database of pre-scripted motion clips.

3.3.4.1 Simulation-in-the-Loop (SIL)

As discussed in [68], motor rotor inertia and control system response effects impede the generation or physically-consistent secondary motion on robots. Thus, the communication

signal of secondary motion must be added to the input command that drives the actuators. Even if the input trajectory already includes secondary effects of the primary motion (e.g. because it was motion-captured), the secondary effects will need to be adjusted to communicate a particular internal state in the motion signal or more accurately reflect the robot's appropriate secondary motion response. A physics-based simulation coupled to the control system of the robot can accomplish this modification in real-time [68].

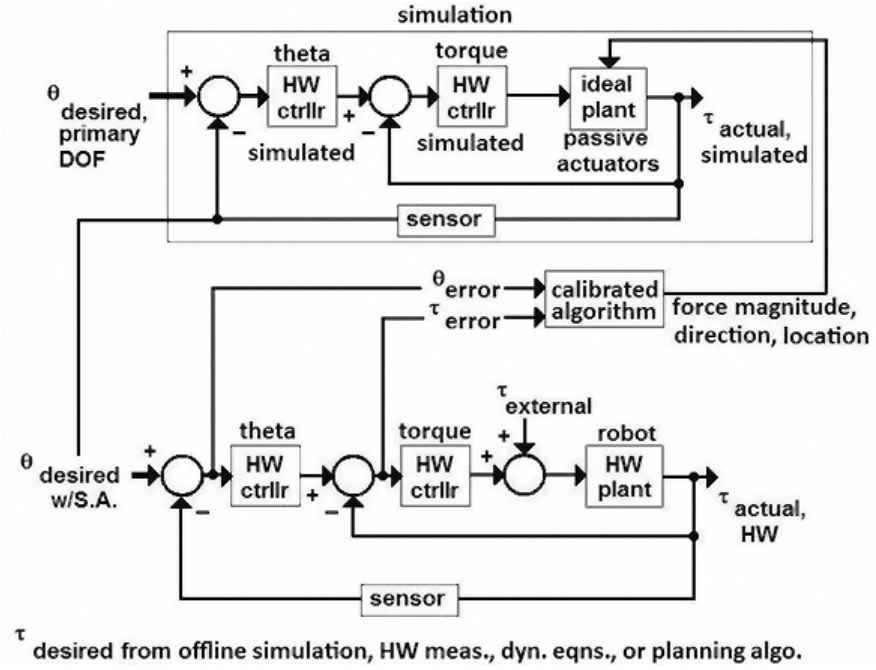

Figure 14: Block diagram of the simulation-in-the-loop technique for generating secondary motion.

The first technique is called simulation-in-the-loop because it determines the secondary motion that would exist in passive actuators by simulating frictionless motors unimpeded by rotor inertia that track primary trajectories and commanding the hardware to the simulated positions. A block diagram of this technique is shown in Figure 14. To enable compliant and fluid secondary motion in SIL, control of the virtual robot requires dynamic selection of passive actuators.

The need for a systematic way to select the underactuated motors and design of a

dynamic gain switching strategy are disadvantages of SIL. Ramping gains from low to high values can eliminate unsafe dynamic motor switching transients that might manifest in the output motion. But, the best results occur through selecting a subset of extremity motors and passively simulating them for the entire duration of a motion. Since passive extremity DOFs can be problematic in certain situations, this technique is not useful for creating secondary motion in all situations. Therefore, two additional techniques were developed to create secondary motion.

3.3.4.2 Feedforward & Feedback Control (F&F)

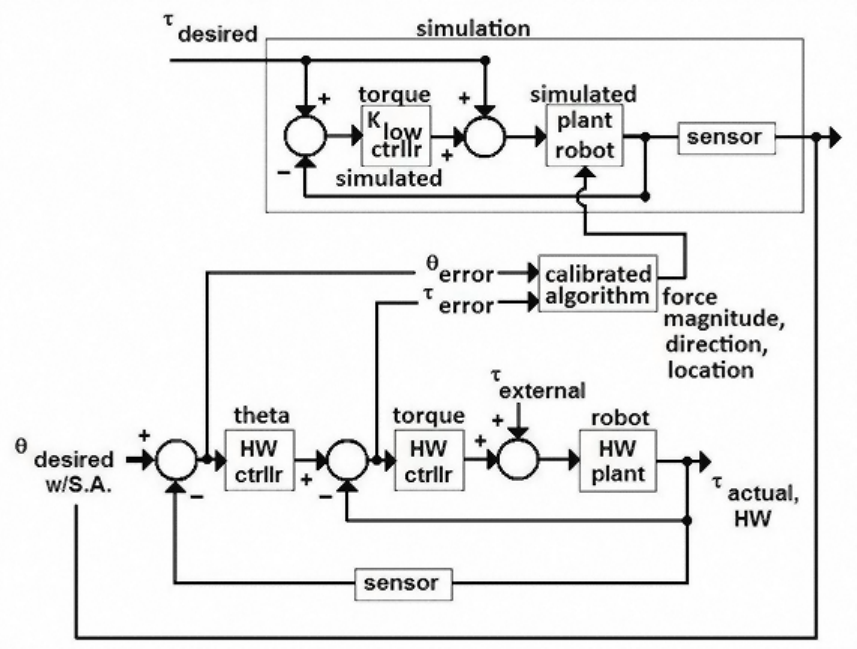

Figure 15: Block diagram of the feedforward and feedback control technique for generating secondary motion.

In feedforward control, the commands to achieve a trajectory are pre-computed or read from sensors and applied open-loop to produce system response. A well-known methodology of control is F&F control, where feedforward control is combined with very low gain state feedback to handle any drift or perturbations to the system that cause deviation from the desired state. This technique simulates secondary motion using F&F control and commands the robot to track the simulated trajectory. A block diagram of F&F

is shown in Figure 15. In my experience with PID control on the Simon robot hardware, simulated proportional motor feedback gains should be reduced less than 0.5% of actual hardware gains and noticeable secondary motion will result.

3.3.4.3 Eigenphysics

Based on Ye and Liu's method for animating responsive motion [67], I developed another new technique, denoted as "eigenphysics," to produce secondary motion on robots. Eigenphysics is motivated by the observation that less actuated degrees-of-freedom usually exhibit more pronounced secondary motion. Instead of determining these under-actuated components by heuristics and hand-tuning their physical parameters, eigenanalysis is applied on the primary motion to define a new set of coordinates, ranked by the level of joint actuation in the primary motion.

Eigenphysics provides a more principled way to identify underactuated coordinates (corresponding to eigenvalues close to zero) specific to each primary motion sequence. For each primary motion sequence, an offline algorithm is used to compute under-actuated coordinates.

Since torque is the mechanism for DOF actuation, secondary motion would logically occur in DOFs that have low variance in actuation (i.e. near-unactuated coordinates). Let q_j be the torque trajectory for DOF j of the original input motion with T equidistant time samples. For a robot with M degrees-of-freedom, the torque trajectories from the input motion are organized into an M x T column-stacked matrix $\tau = [q_0, q_1, ..., q_M]$. To identify the actuation spectrum ordered by DOF for a given motion, a singular value decomposition is performed on the covariance matrix in Equation 33 to obtain an M x M matrix of eigenvectors, denoted U, and an M x M matrix with eigenvalues along the diagonal. The magnitude of the eigenvalues corresponds to a measure of the torque variance in the motion.

$$SVD((\tau - \mu)^T(\tau - \mu)) \tag{33}$$

where,

μ_M = mean torque of DOF of index M for the entire trajectory

μ_1 = mean torque of DOF of index 1 for the entire trajectory

μ = M x T stacked matrix where each column is $[\mu_1 ... \mu_M]^T$

τ = M x T column stacked matrix of command torques for the original trajectory

M = number of DOF

T = number of equidistant time samples in the original trajectory

The largest gap in the distribution of the eigenvalues defines a threshold, which separates the corresponding eigenvectors into mostly actuated and near-unactuated eigenvectors. For all eigenvalues smaller than the threshold, the corresponding eigenvectors are considered part of the near-unactuated set. Remaining DOFs are in the mostly actuated set. For reference, typically less than ten eigenvectors were kept in the mostly actuated set (out of forty-one independent DOFs).

After projecting to the rotated space, back to the original space, and the adding the mean for back for each corresponding DOF as in Equation 34, the command torque trajectory that includes secondary motion is known. The result of these operations is a new torque trajectory that includes control torques in the subset of control space selected as near-unactuated (i.e. the secondary motion), computed from the torques of the mostly actuated DOFs set (i.e. the primary motion).

$$U^T U(\tau - \mu) + \mu \tag{34}$$

where,

U = P x M column stacked matrix of mostly actuated eigenvectors

P = number of eigenvectors kept in the mostly actuated set

Eigenvectors (i.e. principal components) corresponding to lower eigenvalues are less significant, less important dimensions of motion in terms of active body control [67]. Hence, eigenphysics can produce the secondary motion from internal forces and torques without significantly modifying the primary motion. Eigenphysics is coupled with the very low gain feedback to recover from drift and external forces. The resultant secondary motion

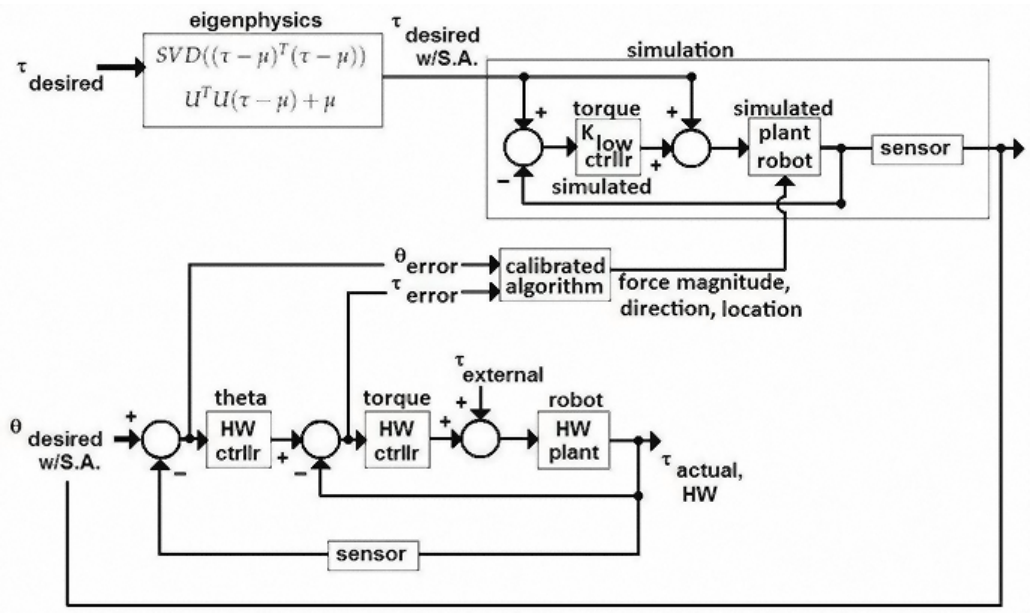

Figure 16: Block diagram of the eigenphysics technique for generating secondary motion.

impacts all DOF, even the support DOF, and does not require that any specific motors remain passive in simulation or in hardware. A block diagram of eigenphysics working in conjunction with a technique for secondary motion in response to external forces is shown in Figure 16.

3.3.5 Hypotheses

To measure success of my secondary motion algorithms, I established the following criteria in the form of five hypotheses.

- H1: Greater quantities of secondary motion are produced in near-unactuated DOFs than in highly actuated DOFs from my algorithm (i.e. the torque trajectory is significantly different for near-unactuated DOF)
- H2: Secondary motion can be used to communicate external and internal state parameters better than motion without a secondary effect
- H3: Secondary motion allows humans to better understand and predict missing context

- H4: Primary motion with secondary action is preferred over motion without secondary motion

- H5: Adding secondary motion helps coordinate the exemplar motion, by decreasing KSE (see Sections 2.3 and 4.1) and increasing human-likeness

H1 tests the success of my algorithm in accomplishing what it was designed to do. H2 and H3 are hypotheses about some types of information that secondary motion can communicate. Whereas, the experiments that evaluate H4 and H5 are designed to demonstrate that my algorithms produce human-like motion, which is inherently a statement about the quality of produced results. These hypotheses are given in general terms here, but they will become more concrete in subsequent sections, as corresponding experiments to test these hypotheses are outlined.

3.3.6 Experiment 1: Internal Forces

Experiment 1 was designed to exemplify secondary motion in response to the forces and torques created by the primary internal motion. Experiment 1 used a very specific exemplar motion so that the generated secondary motion was different and analysis was able to be performed easily on a subset of DOFs for all three secondary motion techniques.

3.3.6.1 Experimental Design

The motion used for Experiment 1 was a tennis swing wherein the right arm was highly actuated over large angle ranges and the left arm held a static equilibrium pose. Since secondary motion can be induced by internal forces transmitted between body segments during motion, a primary motion with large acceleration in the joint space results in more evident secondary motion. Therefore, a rapid swing motion with large disparity between the right and left arm motion was designed. The left arm keeping a static pose provided a convenient subspace of DOFs for a secondary motion analysis. Furthermore, the other scalar product in Newton's second law, i.e. mass or moment of inertia, was utilized to amplify the simulated secondary motion for hardware command.

Secondary motion was created for the tennis swing motion using each of the three techniques, and their differences were analyzed. Furthermore, in the convenient left arm subspace, statistical difference between the original and secondary motion trajectories was expected as discussed in hypothesis H1.

3.3.6.2 Results

For comparison of visual difference, keyframes are presented from the resulting trajectories. These keyframes are not captured at the same time instants in the different trajectories. Instead, the photos show expressive moments to characterize the full trajectory produced by each technique.

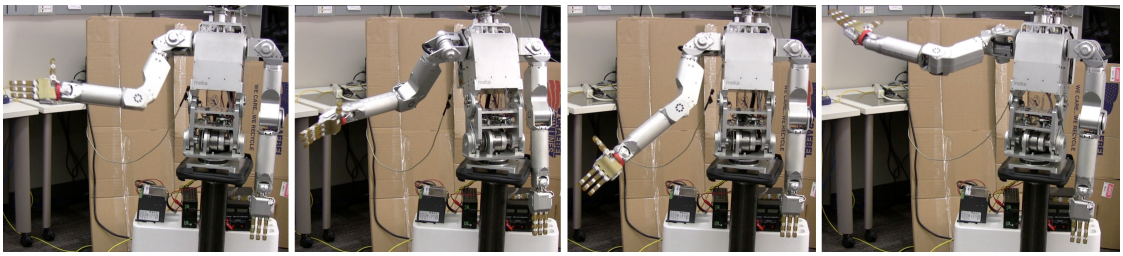

Figure 17: The tennis swing motion on the SIMON hardware actuates only the right arm to demonstrate the amount of internal secondary motion that results from left arm actuator passivity.

Figure 17 shows the primary action of a tennis swing motion designed with the right arm of the robot, leaving the left arm completely passive on the hardware. A large angle range was intentionally used on the right shoulder rotation to allow the hardware the range necessary for large accelerations and decelerations to produce larger amounts of internal secondary motion while remaining safe distances from joint angle limits.

My algorithm for secondary motion is motivated by the fact that secondary motion is not automatically created in passive hardware. To demonstrate this point, the left arm motors were turned off during the tennis swing, so that it could be used to demonstrate secondary motion due to internal forces when no algorithm is used to generate secondary motion. Figure 18 shows the insignificant amount of secondary motion that results when

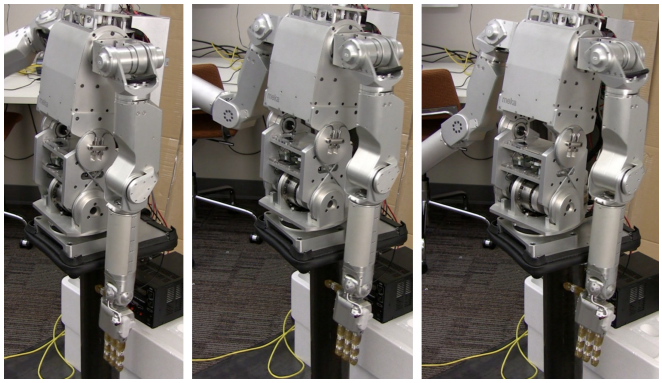

Figure 18: Close-up and side view of left arm during unmodified tennis swing motion to demonstrate hardware capability to create secondary motion without algorithmically modifying the trajectory.

the left arm chain hardware was turned off during the tennis swing.

In the closer view of the baseline trajectory (Figure 18), at the critical points in the motion, where the right shoulder DOFs changed direction of travel, the expected response was a deflection of the torso, which should subsequently cascade to the left arm. However, the left arm remained nearly vertical, only exhibiting small amounts of motion in the wrist that were not part of the primary action because actuator control and the large mass of the robot diminished this response.

In order to quantify the lack of internal secondary motion produced in the hardware without algorithmic modification, trajectories for each of the seven passive DOFs in the left arm were recorded for the time duration of one swing trajectory. No variance calculated from the measured data for any of the seven left arm DOFs was greater than 0.01 square degrees, which is empirical proof that passive hardware does not move and naturally generate secondary motion. Table 10 shows these variance values individually for each of the seven left arm DOFs. One interesting trend was that DOFs closer to the torso (nearer to the top of the table) exhibited less secondary motion when passive on the hardware. This trend was consistent with expectations because on the robot, larger motors (more massive) are required closer to the torso, which also have greater friction and rotational inertias to overcome. Thus, these DOFs should exhibit less secondary motion.

In simulation-in-the-loop, for the tennis swing motion, the left arm was passive in

Table 10: Variance (square degrees) for each of the seven passive left arm DOFs during the tennis swing trajectory. Higher variance indicates more secondary motion created due to internal forces for a particular technique. Orig = original trajectory; HW = passive hardware; Eig = eigenphysics. Shou = shoulder; Elbw = elbow; Wrst = wrist.

DOF	Orig	HW	SIL	F&F	Eig
Shou X	0	0.001	50.2	56.8	81.2
Shou Z	0	0.002	1.52	4.79	8.81
Shou Y	0	0.001	2.01	3.53	6.02
Elbw X	0	0.003	16.7	26.4	29.8
Wrst Y	0	0.002	0.78	0.66	1.49
Wrst X	0	0.007	24.6	32.8	35.7
Wrst Z	0	0.006	8.03	14.2	22.4

simulation. As shown in Figure 19, the left arm moved in response to simulated forces that were transmitted across the simulated torso. Compared to Figure 18, the shoulder, elbow, and wrist DOFs deflected and moved significantly. The left wrist exhibited movement away from the body as the torso rotated about the vertical axis, which was a result of centrifugal and Coriolis forces. SIL produced the smallest amount of noticeable secondary motion of the three techniques, in terms of joint angle ranges, because the torso remained under full virtual gains, which restricted force transmission across the body. Torso deflection backward caused the left arm, elbow, and wrist to also deflect in delayed response.

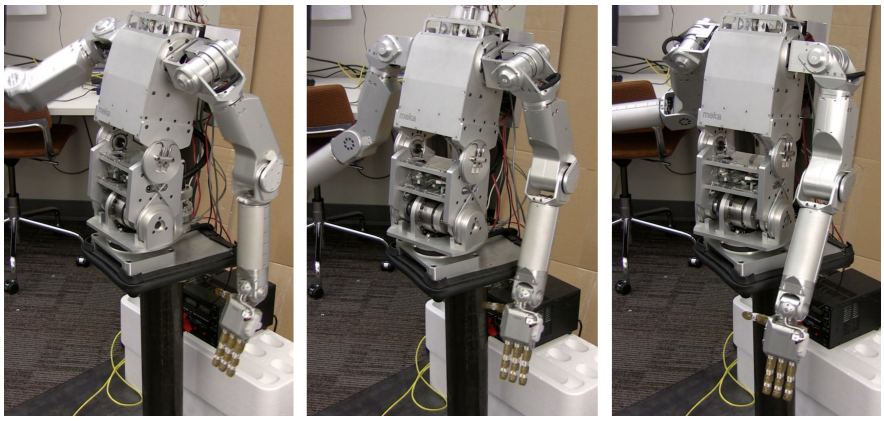

Figure 19: Close-up of the left arm during a tennis swing to show simulation-in-the-loop secondary motion.

In order to quantify the amount of internal secondary motion produced in the hardware with SIL algorithmic modification, trajectories for each of the seven passive DOFs in the left arm were recorded for the time duration of one swing trajectory. Variances for each of the seven DOFs are listed in Table 10, which shows that passive degrees-of-freedom gained secondary motion when simulating those DOFs as passive and commanding the simulated response to hardware.

Table 10 shows an interesting trend for SIL. The shoulder, elbow and wrist passive DOFs about x-axis demonstrated the largest quantity of secondary motion. This was expected since the dominant direction of primary motion should create torque about the x-axis.

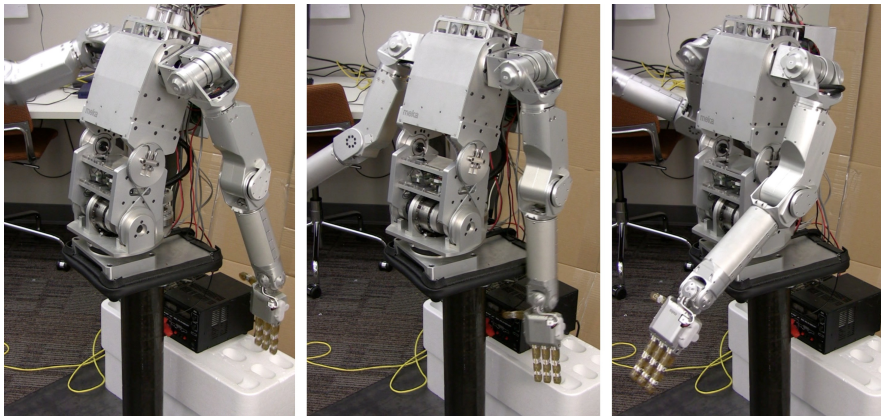

Figure 20: Close-up of the left arm during a tennis swing to show feedforward & feedback secondary motion.

Feedforward & feedback control in Figure 20 demonstrated that low virtual gains can produce secondary motion in body parts that support the robot such as the torso because the majority of actuation is open loop, feedforward command. Secondary motion in the simulation-in-the-loop technique cannot occur in the torso because it would require passive torso actuators. Passive torso actuators would not be able to supply sufficient torque to keep the robot upright (in simulation or hardware). In low gain feedback control, there is a slight temporal delay in the secondary motion due to control system response that does not exist when a DOF is completely passive.

To quantitatively compare the amounts of secondary motion created in non-passive DOFs for each of my three techniques, I defined a measure called average square deviation from the original motion (ASDOM), shown in Equation 35, which I used to calculate the deviation of the trajectory with secondary motion from the primary motion trajectory that was used to generate the secondary version, averaged over all time samples.

$$ASDOM_h = \frac{1}{N}(q_{pm} - q_i)^T(q_{pm} - q_i) \tag{35}$$

where,

q_{pm} = joint angle trajectory of the primary motion (i.e. original, given trajectory) of DOF h as a vector

q_i = joint angle trajectory of DOF h created by i = {SIL, F&F, or eigenphysics} as a vector

h = denotes DOF index

N = number of equidistant time samples in the original trajectory

$ASDOM$ is calculated just like variance, except that the mean in the variance calculation is replaced by the primary motion trajectory in the $ASDOM$ calculation. Thus, $ASDOM$ is a measure of how much a trajectory for a particular DOF on average deviates from another trajectory. This is a more appropriate measure to compare secondary motion in DOFs where motion existed in the primary motion, since secondary motion is relative to the original trajectory. If variance were used instead, incorrect conclusions could be drawn since secondary motion increases or decreases dynamics as necessary in already moving DOFs. Variance cannot capture this effect because the calculation is relative to the mean, instead of being relative to the values of another trajectory. The $ASDOM$ calculation was performed for all three techniques for the torso degrees-of-freedom, and the results are shown in Table 11.

Compared to SIL, the F&F technique used to generate secondary motion for the tennis swing trajectory in Figure 20 induced more secondary motion in the torso. For F&F, the large right arm swings pulled the torso back and forth under low simulated gains,

Table 11: Average square deviation from original trajectory (square degrees) for each torso DOF during the tennis swing trajectory. Higher average deviation indicates more secondary motion created due to internal forces for a particular technique. Eig = eigenphysics.

DOF	SIL	F&F	Eig
torso X	0.00	3.61	12.24
torso Y	0.00	1.42	4.47

transferring more forces across the body—the effect of Newton's third law. However, torso movement in F&F was damped by the low gains in the left arm. For SIL, the torso remained highly controlled and the left arm passive, whereas in F&F both the torso and left arm were at more median amounts of actuation. Also, using F&F, induced secondary motion in the torso DOF that rotates the vertical axis (i.e. torso Y DOF) is largest. Since the torso was supporting the right arm, it had to remain actuated in SIL, and therefore was not able to exhibit secondary motion.

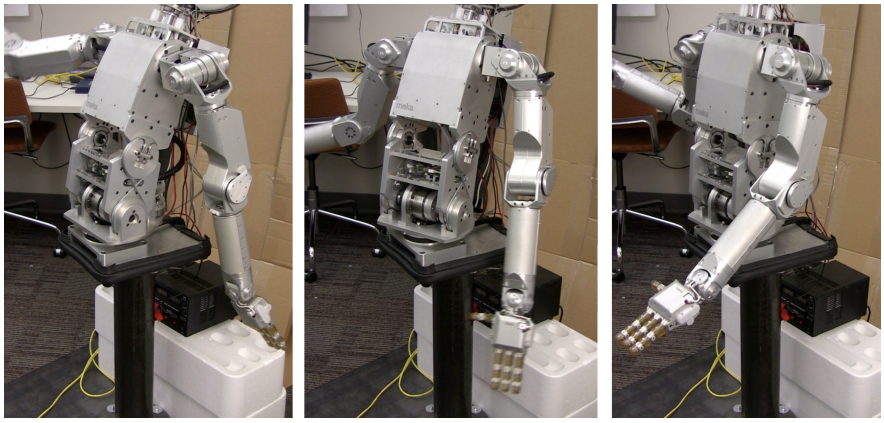

Figure 21: Close-up of the left arm during a tennis swing to show secondary motion from eigenphysics.

Eigenphysics in Figure 21 altered the trajectory of all DOF to create secondary motion everywhere, unlike SIL, which was constrained to create secondary motion in the DOFs simulated as passive. In eigenphysics, actuation of the left arm did not depend solely

upon the low gain feedback as in feedforward & feedback control. In eigenphysics, the actuation command for secondary motion was calculated directly and applied with the primary motion. The result was a left arm that was swinging in a coordinated, controlled, but passive-appearing way, synchronized to movement of the torso and the right arm. Additionally, the eigenphysics projection preserved most but not all of the primary motion, so even the primary trajectory was affected by this technique. In comparison to F&F, the actuation delay does not exist in either other technique.

For the tennis swing motion, secondary motion was dominantly produced in three DOFs: shoulder X, elbow X, and wrist X. Table 10, which shows variance induced by secondary motion in these three left arm DOF axes, illustrates the fundamental noticeable difference in the eigenphysics motion: the left arm DOFs spanned a larger range of joint angles with eigenphysics than with either SIL or F&F.

Furthermore, to add evidence that each of the three techniques produces meaningful secondary motion over the baseline of passive hardware, pairwise t-tests between each technique and the baseline were performed for velocity and acceleration trajectories for the seven left arm DOFs for the tennis swing motion. 14 of 14 of these pairwise t-tests exhibited statistical significance ($p<0.05$) for all three techniques (vs. baseline), which means that the dynamics of the trajectories in the left arm were statistically different than the dynamics without motion induced by the secondary motion algorithms.

To explore the differences between algorithms and the dynamics they induced in the space of near-unactuated DOFs, pairwise t-tests were performed between each of the three techniques for each DOF for both acceleration and velocity trajectories. Results from these pairwise t-tests are shown in Table 12, where the character 'b' indicates that both the velocity and acceleration trajectories were statistically significant ($p<0.05$), between the respective techniques. As expected, the dynamics of many left arm DOFs are different between techniques, and in other cases, there is no statistical difference between the dynamics (indicated by '0'). The former is partially attributable to the advantages of certain techniques over other techniques; and the latter is partially attributable to the fact that all

Table 12: Pairwise t-test results of left arm dynamics for each of three secondary motion techniques for the tennis swing trajectory. b = indicates that both velocity and acceleration trajectories exhibit statistical significance ($p<0.05$). 0 = neither velocity nor acceleration trajectory exhibits statistical significance ($p<0.05$) for the given pairing. Eig = eigenphysics. Shou = shoulder; Elbw = elbow; Wrst = wrist.

DOF	SIL v. F&F	F&F v. Eig	SIL v. Eig
Shou X	0	b	b
Shou Z	b	b	b
Shou Y	b	b	b
Elbw X	b	0	b
Wrst Y	0	b	b
Wrst X	b	0	b
Wrst Z	b	b	b

trajectories created by all three techniques derive from the same input motion, and each technique is meant to generate the response appropriate with physics (i.e. all techniques have the same target output trajectory).

Table 13: Average square deviation from the original trajectory (square degrees) for each arm DOF during the tennis swing trajectory. Higher average deviation indicates more secondary motion created due to internal forces for a particular technique. Results show that near-unactuated DOFs (left arm) have more secondary motion than highly-actuated DOFs (right arm). Eig = eigenphysics. Shou = shoulder; Elbw = elbow; Wrst = wrist.

	SIL		F&F		Eig	
DOF	Left	Right	Left	Right	Left	Right
Shou X	47.1	0.00	53.4	3.41	76.5	4.52
Shou Z	1.57	0.00	6.21	1.53	9.03	3.71
Shou Y	1.96	0.00	4.13	0.64	5.59	1.77
Elbw X	17.8	0.00	29.5	3.78	26.5	4.07
Wrst Y	0.63	0.00	0.35	0.19	1.12	0.89
Wrst X	22.2	0.00	33.2	3.32	34.2	4.66
Wrst Z	8.31	0.00	15.1	2.25	20.8	3.31

The final analysis in Experiment 1 tested H1 to find out if more secondary motion is created in near-unactuated DOFs. To test this hypothesis for secondary motion created from internal forces, $ASDOM$ was used to compare the right arm (highly-actuated) with

the left arm (near-unactuated) for the tennis swing. Table 13 shows that near-unactuated DOFs (left arm) had more secondary motion than highly-actuated DOFs (right arm), which is consistent with H1.

3.3.7 Experiment 2: External Forces

3.3.7.1 Experimental Design

To exemplify secondary motion produced from external forces for each of the three techniques, in Experiment 2, the robot received an external push from a human hand stimulus during a common pick-and-place task (i.e. lifting a object and moving it to another location). To emphasize the effect of external forces and allow external-force-induced secondary motion to be observed with minimal secondary motion produced via internal forces, the velocity and acceleration in this primary motion was kept low. The force magnitude of the push was estimated using the robot joint angle sensors and a calibrated algorithm that determines the exact location of the impact upon the robot exterior. For safety reasons, perturbations on the hardware were kept low, and when larger stimuli were required, the magnitude of the push was amplified in simulation. The external secondary response was then added to the commanded motion for the hardware in real-time. For Experiment 2, which is demonstrating that each of the three techniques have the ability to create and add *external* secondary motion, the resultant responses with secondary motion may appear disproportionate with respect to expectations for such a small push. However, this was intentional, and designed to demonstrate the techniques in an non-subtle manner without forcefully pushing the robot hardware.

3.3.7.2 Results

To demonstrate the real-time simulation and hardware loop, the robot was calibrated using differences between commanded and actual joint angles to detect velocity and position changes significant enough to be external hardware disturbances and perturbations. Robot position sensors were used to calculate velocity and force vectors (magnitude, direction, and location) of disturbances. Since the accompanying simulation was dynamic, the exact location of the disturbance was quickly identified on one specific region on one plane of

one rigid body in the simulation from the reaction of the hardware. The actual measured angles from the hardware were input into the simulation to solve for the magnitude of the application of the force because the response of the hardware nearly completely isolates a unique position on the robot body, provided that a sufficient response to the push manifests in the hardware.

In simulation, the secondary response to an external push was simulated using the three techniques outlined above, and the robot hardware exhibited secondary motion in response to external forces.

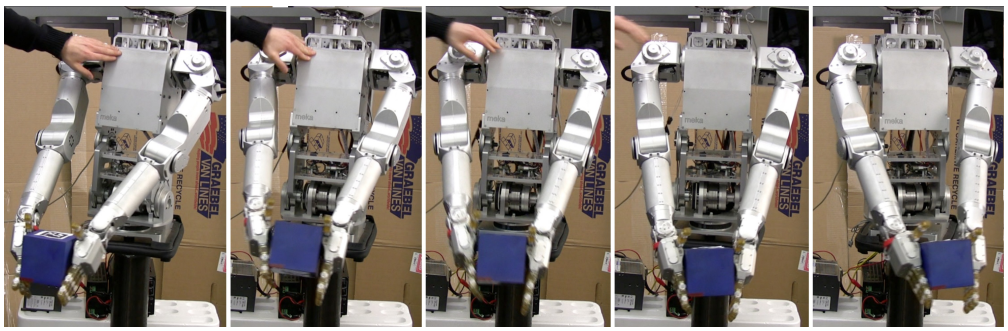

Figure 22: Pushing the robot during a box-moving trajectory to demonstrate the response without secondary augmentation. Without an algorithm to augment secondary motion, the torso deflects less than four degrees.

Figure 22 shows key frames of the box moving trajectory while the robot torso was pushed in the fore/aft direction in the initial frame. While under complete control, without any secondary motion technique, the torso deflected a maximum of four degrees when perturbed. The effect was very subtle and damped after one oscillation. No other DOFs were noticeably affected by the push without using one of the three secondary motion techniques.

To further quantify how little the push influenced the box-moving trajectory, the average square deviation from primary motion in Equation 35 was used to calculate $ASDOM$ for the torso trajectory with the push in reference to the same trajectory without the perturbation. For the torso DOF nearest to the push, $ASDOM$ was 1.586 square degrees, which was large enough to be noticeable, but much smaller in magnitude than was created by

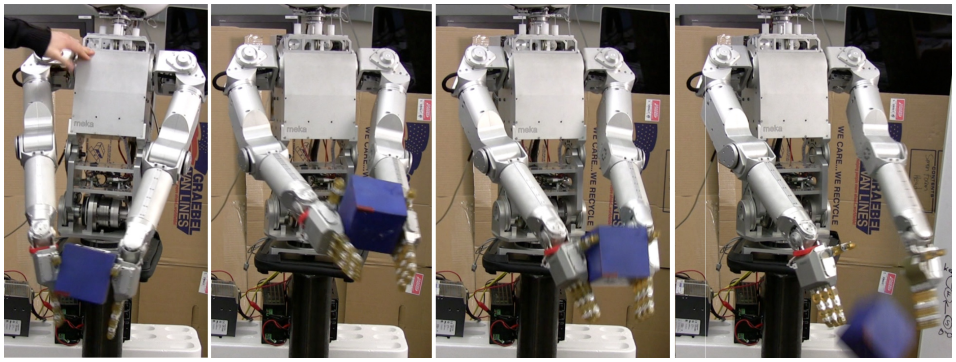

Figure 23: Pushing the robot during a box-moving trajectory to demonstrate the response using SIL. Simulation-in-the-loop technique exhibits secondary motion in the passive DOFs (i.e. wrists and elbows).

F&F or eigenphysics (as will be presented later).

For SIL, the elbows and two of the wrist DOFs were chosen to be unactuated in simulation. In Figure 23, the noticeable secondary effect of the push to the torso for SIL was that the elbows curl upward. Since the wrists remained unactuated in the motion and yet were holding a box, the wrists limply clutched the box, while the actuated arm DOFs supplied the force necessary to maintain grasp on the box. When the external push occurred on the robot, the wrists deflected upward from the joint angle limit of negative 40 degrees to close to zero, as can be seen in the second and third keyframes of Figure 23. The SIL response was more realistic than in Figure 22, where the hardware reacted to a similar perturbation through stiff arm response without algorithmic compensation.

Using the box-moving trajectory as a reference (i.e. original) motion, $ASDOM$ was calculated for seven DOFs in one arm and the two torso DOFs. This is sufficient to analyze where each technique creates the larger quantities of secondary motion because the box-moving trajectory has arm trajectories that are close to symmetric. Table 14 shows that for SIL, secondary motion was created from external forces only in DOFs that were passively simulated.

In the F&F technique for the box-moving trajectory (Figure 24), the push significantly deflected the torso which subsequently created a force imbalance that cascaded through both arms, as the low gains compensated for the external perturbation. The effect is most

Table 14: Average square deviation from primary motion (square degrees) for each of the seven passive left arm DOFs and the two torso DOFs during the perturbed box-moving trajectory. Higher deviation indicates more secondary motion created due to external forces for a particular technique. Eig = eigenphysics. Tors = torso; Shou = shoulder; Elbw = elbow; Wrst = wrist.

DOF	SIL	F&F	Eig
Tors X	0.000	16.9	18.2
Tors Y	0.000	0.37	0.76
Shou X	0.000	36.3	22.1
Shou Z	0.000	0.23	0.19
Shou Y	0.000	0.08	0.05
Elbw X	36.13	6.65	5.59
Wrst Y	0.000	0.04	0.04
Wrst X	0.066	0.07	0.07
Wrst Z	134.8	15.2	11.9

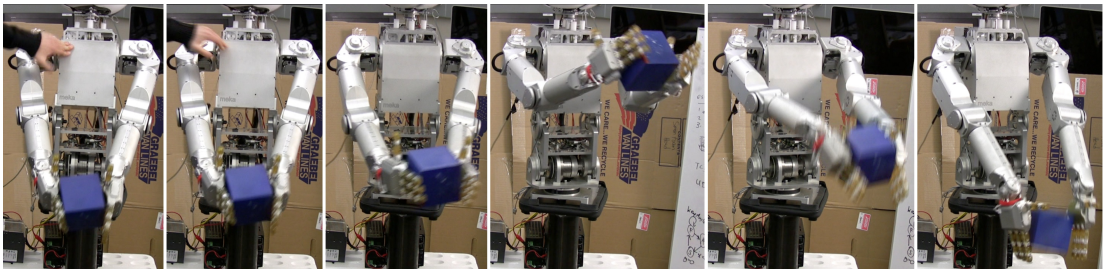

Figure 24: Pushing the robot during a box-moving trajectory to demonstrate secondary motion from external forces using feedforward & feedback control.

noticeable as the box was lifted much higher in the third frame of Figure 24 because unlike in SIL, low torso gains had to compensate for the external force. In general, as gains of a particular motor controller decrease, secondary motion will be more pronounced in that DOF, which is evident by the rapid damping of the push without algorithmic augmentation and that secondary motion was only created in the simulated passive DOFs in SIL.

Table 14 demonstrates why SIL is insufficient for generating secondary motion. Comparing the columns of SIL and F&F, the deviation in a wide range of DOFs was canceled in SIL because the DOFs where secondary motion can be generated needed to be pre-selected.

Thus, the secondary response to external forces was incomplete in the SIL technique, especially in DOFs where significant secondary motion was generated (e.g. torso X and shoulder X) as a result of the push.

Comparing row-wise elements in the SIL and F&F columns in Table 14 for the elbow X DOF and wrist Z DOF shows that the magnitudes of the average square deviation from the primary motion was less for both of these DOFs in F&F, which was the result of two causes: (1) low gains damped the secondary response in F&F as compared to the completely passive actuators in SIL; and (2) since only a few DOFs are passive in SIL, the secondary response to the external perturbation must be absorbed in these select few DOFs, which caused their secondary response to be larger. In F&F, the external push can be absorbed by the motion of more DOFs because all DOFs have low gain feedback.

Even though all three techniques dropped the box at some point (because it was part of the original trajectory to let go) the eigenphysics secondary motion in Figure 25 created a very fluid response after the box was released, evident by the roll over of the wrists at the end of the trajectory.

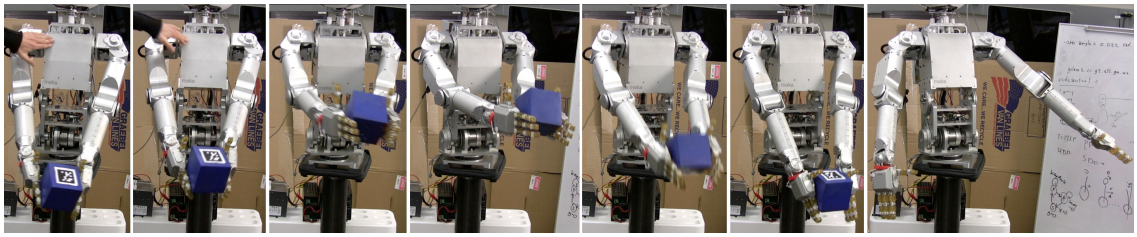

Figure 25: Pushing the robot during a box-moving trajectory to demonstrate secondary motion from external forces using eigenphysics.

In Figure 25, eigenphysics demonstrates the distinct advantage of creating secondary motion while all motors remained under normal gains. Making DOFs passive as with simulation-in-the-loop, or using low gains can cause the robot to fail the task (e.g. drop the box) in the presence of secondary motion because certain degrees-of-freedom can no longer supply the necessary forces to maintain the task (e.g. when external forces become too large). However, eigenphysics creates secondary motion in the null-space of the task.

Therefore, since the task was supplying the torques necessary to track the trajectory and move the box, eigenphysics caused minimal disruption of the task. With SIL, creating secondary motion the presence of a task is challenging because DOFs must be chosen that do not disrupt the task, which isn't always feasible.

A perfect comparison of the three techniques cannot be performed because the external push from the human does not occur at the exact same time instant or with the exact same magnitude. However, the timing and point of application of the external force is critical to determining how each technique will respond. The main difference between F&F and eigenphysics lies in how the feedforward control term is calculated. Because the actuation space of eigenphysics has lower dimension and is derived from the primary motion, it typically induces larger amounts of secondary motion in the "near-unactuated" coordinates and less in the "mostly-actuated" coordinates. For example, comparison of the torso in Figures 22-25 (and also in the torso X row in Table 14) shows that eigenphysics (Figure 25) caused the largest deviation from the box-moving trajectory for the torso fore/aft DOF (torso X), but less secondary motion in the arm DOFs than F&F. Eigenphysics typically induces more secondary motion in the "near-unactuated coordinates" than in the "mostly actuated coordinates." The force from the push combined with the actuation inherent from the primary motion or internal secondary motion to impact the same DOF in different ways for each of the different techniques.

The results of Experiment 2, especially in the calculation of $ASDOM$, do not describe the quality of the produced secondary motion, but rather these results examine only the quantities of motion induced in certain DOFs as a result of algorithmically exploiting passivity. Thus, further experiments were required to examine the qualitative features and benefits of the motion. The motion generated with my algorithms is secondary motion because it does not dominate the original, primary motion, and the generated actuation is concentrated in DOFs that were previously passive or near-unactuated.

Pairwise t-tests for statistical significance of the dynamics each algorithm induced in the space of near-unactuated DOFs were not performed for the external forces experiment because the perturbation on the hardware was not consistent and repeatable for all three

techniques. Instead, referring to Table 14, comparison of the respective rows in the same column for near-unactuated DOFs (e.g. wrist Z) and mostly actuated DOFs (e.g. wrist X and shoulder Z) shows that larger quantities of secondary motion were induced in the near-unactuated DOFs, and they deviated more from the primary trajectory. This was only true in possible DOFs, of course, since the SIL technique is restricted in its application of secondary motion to pre-selected DOFs. This finding is consistent with H2.

3.3.8 Experiment 3: Communicating State Parameters

Experiments 1 and 2 were targeted toward validating the creation of secondary motion from internal and external forces respectively, which was testing the algorithm to ensure that it worked as designed. However, Experiments 3-6 were performed to discern the benefits of adding secondary motion to robot motion.

Throughout the literature, a fundamental assumption often made about secondary motion is that it can communicate internal and external state parameters. This assumption is widely believed to be true since it is an involuntary cue that humans frequently use when sensing the world through vision. For example, the average person might claim that they can tell just by watching a man carry a box, whether the box is heavy or light and whether the man is strong or weak. The former (i.e. mass) is an example of an external state parameter since it measures a state external to the being performing the motion (e.g. some world state or state of an object in the world), and the latter (i.e. muscle strength) is an example of an internal state parameter since it measures a state internal to the being performing the motion. Specifically, Experiment 3 tested whether secondary motion can communicate internal and external state parameters.

It is important for a social robot to clearly communicate internal and external state parameters through its motion because this is increasing state transparency and increasing the redundancy of passing these messages to human partners by adding additional channel of communication. In the terms of control theory, if the human is a predictor and the robot is the system, whose internal state is being estimated, this increased observability leads to better measurements and more accurate predictability of the true system state

from those measurements by estimative processes. Redundancy increases the likelihood that the intended message will be received, since communication now occurs through multiple channels. Estimative processes (e.g. humans) that take advantage of multi-channel communication have higher confidence in their estimates of social robot partner state when messages or predictions from multiple channels reinforce each other.

3.3.8.1 Experimental Design

For Experiment 3, eight different tasks were created that lacked any secondary motion:

1. Turn: Robot reaches out, grabs a lever with the right hand, turns it 45 degrees, releases grasp, and retracts the arm.

2. Handoff: Robot reaches out with right arm, picks up a block off a table, lifts it, extends the arm to pass it to a partner, releases, and retracts the arm.

3. Wave: Robot performs a waving gesture with the left arm.

4. Beckon: Robot performs a come hither gesture with the right arm.

5. Push: Robot grabs a block with two hands that is idle at waist-height, pushes the block directly outward using the entire upper body, then the robot returns to an erect posture.

6. Slide: Robot grabs a block with the right hand, uses a combination of torso and arm movement to slide the block toward the robot's left side, then the robot retracts the arm.

7. Grab: Robot turns to its right side, reaches with the right arm to a shelf near head height, grabs the block with the right hand, lifts the block off the shelf, turns to place it on the table in front of itself, releases the block, and retracts the arm.

8. Lift: The robot grabs a block sitting idly in front of it with two hands, lifts the block upward using mostly torso and forearms, places the block back to its original position, and retracts its arms.

These tasks are categorized according to two different internal state parameters and two different external state parameters:

- Motor Temperature: Turn and handoff.
- Passivity (Motor Gains): Wave and beckon.
- Friction: Push and slide.
- Mass (Weight): Grab and lift.

Each of the state parameters were modulated to three different values, and then the secondary motion algorithm was used to generate secondary motion for the robot with these new values. Thus, for each of the eight motions, there are six variants in Experiment 3: three with and three without secondary motion, where the three correspond to low, medium, and high state values.

A dynamic simulation was used to create the six variants per task with and without secondary motion. Levers, blocks, tables, and surfaces that contact the robot during any of the eight tasks were added to the dynamic simulation as rigid bodies with appropriate parameters such mass, rotational friction, or surface friction. These task-based rigid body objects imparted force trajectories upon the modeled robot architecture in response to the motion. The original trajectory for each task was executed in the dynamic simulation after the corresponding internal or external state parameter was modified to one of the three values. Modifying this state parameter caused the original trajectory to appear different, especially for extreme values of the state parameter.

To create the three variants of each task that have secondary motion, my feedforward & feedback secondary motion algorithm was integrated into the dynamic simulation robot model so that the original trajectory now appeared different because (1) a state parameter had changed and (2) the robot motion responded to secondary motion forces and/or torques.

The feedforward and feedback method was selected because for these tasks where state parameters were changing, the entire input trajectory is not known in advance. Thus,

eigenphysics cannot be used. Simulation-in-the-loop was not used because secondary motion is desired in all DOFs, not a subset of preselected ones. The feedforward & feedback algorithm enables coupling between the changes in the internal or external state to interact simultaneously with the dynamics induced by the secondary motion produced at previous time steps. Use of F&F results in a secondary motion trajectory that is influenced by many different changes in dynamics that would otherwise remain independent: (1) internal state parameter changes (2) external state parameter changes (3) changes in task forces due to changes in secondary motion (4) changes in secondary motion due to changes in task forces (5) changes in changes in task forces due to state parameter changes and (6) changes in secondary motion due to secondary motion.

Since levers, heavy boxes, and surfaces of varying frictional coefficients are not easy to procure (and for safety reasons), trajectories were determined in simulation. Then, videos of the robot hardware performing each variant of each task were recorded from three different angles. The three different angles of each particular task variant were concatenated into a single video for each motion. Thus, forty-eight different videos were created for Experiment 3, modulated according to:

- Eight Tasks: Turn, handoff, wave, beckon, push, slide, grab, and lift.
- Three State Parameter Levels: Low, medium, and high.
- Two Secondary Motion Options: Present and absent.

All of the motions in the videos on the hardware lacked context. The missing context for each motion is summarized below.

1. Turn: Lever was absent in the videos.
2. Handoff: Block, table, and partner were absent in the videos.
3. Wave: Person the robot is waving to was absent in the videos.
4. Beckon: Person the robot is gesturing toward was absent in the videos.
5. Push: Block and surface were absent in the videos.

6. Slide: Block and surface were absent in the videos.

7. Grab: Block, shelf, and table were absent in the videos.

8. Lift: Block and table were absent in the videos.

For the motor temperature internal state parameter, using a dynamic simulation that modulates motor torque output based on motor temperature, the temperature of all the motors on the modeled robot plant were raised by a pre-determined number of degrees Fahrenheit. The task was then executed and the simulated robot turned a simulated lever or hands an simulated box. The output trajectory was recorded, run on the hardware, and videotaped from three different angles. This process was repeated to reflect internal state conditions of normal operating temperature and five degrees Fahrenheit below overheating.

Similarly, the other internal state parameter of passivity was modulated by changing gains of the simulated robot plant, which should not be confused with changing gains of the F&F control algorithm. Passivity was a special case, unlike the other state parameters, since the two tasks associated with passivity were free-space gestures (i.e. beckon and wave). They did not interact with the world, and therefore, there was no influence of *external* secondary motion forces upon the trajectories produced.

The external state parameters of friction and mass were modulated by changing the coefficient of friction or the mass of the simulated objects. These state parameter changes induced secondary motion changes through the influence of changing force trajectories imparted upon points of the simulated robot body in the plant.

The effects of the *state parameter level* changes on the original motion are best described as follows:

- Turn: The velocity of the robot's level turn decreases as motor temperature increases.

- Handoff: Robot maximum available torque is lower at higher temperatures, and therefore, since mass of the box being handed is constant, the torso is bent further over as the motor temperature increases. There is also an increased time delay before the robot lifts the object as motor temperature increases.

- Wave: There is an increased time delay between DOF direction changes in the waving arm (i.e. the DOFs are less synchronized) as gains are lower (i.e. more passivity).
- Beckon: When more motors are passive, the beckoning arm at each cycle is not stopped as quickly (due to lower available control energy), which results in more forces being transferred to the torso. Thus, the robot rocks back more during beckon cycles for the higher passivity.
- Push: The robot moves forward faster for lower surface frictional coefficients of the surface beneath the pushed object.
- Slide: The velocity of the slide increases for lower surface frictional coefficients beneath the slid object. Also, the robot's right wrist is pushed further back before the object starts to move when the surface frictional coefficient is higher.
- Grab: As mass increases, the robot lifts the box to a lower height off the shelf, and eventually, the robot just slides the box off the shelf at highest mass. Also, as mass increases, the robot's right hand with the object moves more quickly toward the floor after it is pulled off the shelf.
- Lift: The velocity of the lift is lower at higher masses. The joint position of the wrists bend more toward the floor as mass increases. The maximum height of the box lift is lower for higher masses.

Then, after *secondary motion* was added to the videos above, the noticeable effects can be described as follows:

- Turn: The left arm swings forward more as motor temperature increases to compensate for the forces the robot is applying to turn the lever with the right arm.
- Handoff: The torso dips further forward upon object release at higher temperatures. The left arm then follows by swinging forward more in response to the torso motion at higher temperatures. The torso also dips further down at higher temperatures when the robot is initially lifting the object in response to various DOFs reaching their maximum torque limits during the lift.

- Wave: The wave motion of the arm is not as tightly controlled at greater passivity, and therefore, the torso rocks in response to transmitted wave energy.
- Beckon: The neck, torso, and left arm exhibit greater amounts of secondary motion after each beckon cycle at higher passivity.
- Push: As surface frictional coefficient decreases, secondary motion causes more overshoot in the final location where the robot arms push the object before retracting back to the final location.
- Slide: The left arm, torso, and wrist exhibit more secondary motion at the point when the object starts moving when surface frictional coefficient is larger.
- Grab: When more massive boxes are dragged off the shelf, as gravity takes over, a jerk is applied to the robot arm, which responds with larger deflections for more massive boxes. There is also more secondary response in the form of follow-through, after the robot releases the more massive boxes on the floor.
- Lift: After the box is placed back to the floor, the secondary motion causes the hands and arms to follow through due to mass of the box. There are larger amounts of this secondary response evident by larger torso motion forward and larger arm downward displacement at higher masses.

Sixty-four participants were recruited for Experiment 3 to cover two experimental conditions and yield twenty-four data points per task. 50% of the participants had previous robot experience and considered themselves to be at or above a novice level with robot expertise. The other 50% of the participants had never previously used or interacted with a robot. Each participant watched six of the eight possible tasks, in one of two randomly selected secondary motion options. No participant saw the same secondary motion option for all six tasks.

One of the three videos for one of the two secondary motion options was randomly selected for each participant. After they watched this one video, participants were asked which motion they thought the robot was performing in the video. This data was collected

to supplement the data taken in Experiment 4. These results are included in the results analysis under Experiment 4.

After their task label answer was recorded, they were told the task that the robot was supposed to be performing, and all three videos for the same secondary motion option were presented to the participant. Each participant watched all three state parameter levels that correspond to the secondary motion option level randomly selected for that task. These three videos per task were shown in random order. Order of all tasks was randomized for each participant. After the participants watched all three state parameter levels for a given task and given secondary motion option, since context was omitted from the videos, they were posed two questions based upon the state parameter category of the task:

- Motor Temperature (Turn and handoff): In which of these three motions was the robot operating at the hottest temperature? In which of these three motions was the robot operating at the coolest temperature?
- Passivity (Wave and beckon): In which of these three motions was the robot most stiffly controlled? In which of these three motions was the robot least stiffly (i.e. most loosely) controlled?
- Friction (Push and slide): In which of these motions was the robot pushing/sliding the box against the most friction? In which of these motions was the robot pushing/sliding the box against the least amount of friction?
- Mass (Grab and lift): Which of these three motions was the robot grabbing/lifting the heaviest box? Which of these three motions was the robot grabbing/lifting the lightest box?

Each participant was allowed to watch the three motions as many times as necessary until they made their selections with respect to the motions that represented the extreme of a particular state parameter. To limit participant frustration and save time, order of the videos was only enforced for the first viewing of the set of three motions for each task.

3.3.8.2 *Results*

By selecting the two extreme exemplars of a particular state parameter level, the participants ordered the set of three videos with respect to that state parameter. Therefore, between-participants comparisons can be made for each task to see if accuracy of ordering with respect to a state variable improved when secondary motion was added to the task.

Sixty-four participants total provides twenty-four data points per task per secondary motion option, since each participant only saw six of the eight possible tasks, and these forty-eight total samples per task were divided equally between those with and without secondary motion (N=24).

To test hypothesis H2, the data was grouped into two sets of videos (with and without secondary motion), which can be ordered for each set according to the respective state parameter levels for each task. Rank was then applied to each of the videos using the distinct values selected by each participant when answering the two questions of most/least, hottest/coolest, heaviest/lightest, etc. for the extreme values of the state parameter level. Thus, for each participant the three videos were ranked so that lowest magnitude of rank is the highest state parameter level (i.e. 3=Low; 2=Medium, 1=High). A Friedman test was run on the rank data for all 24 participants' responses for each task and for each secondary motion option, to determine if any of videos within a given secondary motion option and for a given task were consistently ranked higher or lower than each other. If the Friedman test produced a p-value less than 0.05, this indicated that further analysis is required to identify which videos the participants could consistently ordinally define. The results for all sixteen Friedman tests are shown in Table 15.

The results in Table 15 show that for all eight tasks with secondary motion there was a statistically significant difference in perception of motion depending on state parameter level. $P<0.05$ indicates that participants were able to determine the correct order of at least two of the three videos with different state parameter levels. Participants were able to determine the correct order of at least two of the three videos in only six of the eight tasks without secondary motion. These two videos were both parameterized according to motor

Table 15: Friedman test results for all eight tasks and both secondary motion options. Data given in the form $\chi^2(2) = X; P = Y$, where the X, Y pairs are given in the table. P-values less than 0.05 indicate that participants could consistently rank at least two of the three videos with respect to the given state parameter level that was modulated for the respective task. (w/o = Friedman test run on three task videos ranked without secondary motion; with = with secondary motion).

Task	w/o	with
Turn	0.609, 0.738	21.333, 0.001
Handoff	2.044, 0.360	12.000, 0.002
Wave	11.389, 0.003	21.332, 0.001
Beckon	12.250, 0.002	25.074, 0.001
Push	17.583, 0.001	37.000, 0.001
Slide	12.000, 0.002	33.334, 0.001
Grab	19.011, 0.001	36.969, 0.001
Lift	17.958, 0.001	46.083, 0.001

temperature (i.e. turn and handoff tasks), which is preliminary evidence that secondary motion helps humans perceive internal state parameter levels from robot motion.

Post–hoc analysis with Wilcoxon Signed–Rank Tests was conducted with a Bonferroni correction applied, resulting in a significance level set at $p<0.017$. The p-values from each of the individual pairings are listed in Table 16.

Participants were able to discern the correct order for the pairing of low and high state parameters, but in general, intermediate state parameter values were not discernible as shown by the lack of statistical difference between LM and MH pairings. When secondary motion was added to the motion, the resolution of perceivable state parameter values increased, and all three levels were statistically different. The conclusion is that internal and external state parameters such as motor temperature, passivity, friction, and mass can be communicated through secondary motion. When humans are provided with discrete examples from a continuum of a particular state parameter, they are better able to rank order these motions according to the state parameter, when the motion contains secondary effects. In Experiment 4, stronger conclusions are sought by testing if humans can predict the actual magnitude of a state parameter when seeing only one motion, rather than ordering a set of videos.

Table 16: Post–hoc analysis with Wilcoxon Signed-Rank test results for all eight tasks and both secondary motion options. P-values given in the table less than 0.017 (gray) indicate that participants could consistently rank those two videos correctly with respect to the given state parameter level that was modulated for the respective task. (w/o = Wilcoxon test run on three task videos ranked without secondary motion; with = with secondary motion) and (LM = Low-Medium video pair; LH = Low-High video pair; MH = video pair of Medium and High state value parameter levels) x = Wilcoxon not run since Friedman did not achieve statistical significance.

	w/o			with		
Task	LM	LH	MH	LM	LH	MH
Turn	x	x	x	0.015	0.001	0.015
Handoff	x	x	x	0.014	0.014	0.014
Wave	0.260	0.001	0.037	0.010	0.001	0.001
Beckon	0.015	0.012	0.112	0.003	0.001	0.015
Push	0.033	0.001	0.030	0.001	0.001	0.001
Slide	0.034	0.001	0.034	0.001	0.001	0.001
Grab	0.031	0.001	0.026	0.001	0.001	0.001
Lift	0.062	0.001	0.021	0.001	0.001	0.001

To reinforce the conclusions derived from Table 16, the percent of participants who correctly ranked all three videos according to the respective state parameter level for all eight tasks and both secondary motion options is shown in Table 17. For all eight tasks, the addition of secondary motion caused more participants to correctly order the videos according to the state parameter, which reinforces that hypothesis H2 is correct.

One more analysis was performed on the data from Experiment 3 to support the conclusion that secondary motion better communicates internal and external state parameters than motion without a secondary effect. Each participant provided responses to two questions when selecting the extreme values of a particular state parameter for a given task. The responses to these two questions were coded as rank. No video was selected as the answer to both questions. Therefore, the maximum and minimum distances between any two ranks in Experiment 3 is 1, and no rank was repeated in Experiment 3 for a given task. By plotting mean participant rank vs. actual rank for each task, and linearly regressing the optimal line that fits this data, one additional measure of success is obtained. Ideally, all participants would rank the motion as actual rank, and the magnitude of the slope

Table 17: Percent of participants who correctly ranked all three videos according to the respective state parameter level for all eight tasks and both secondary motion options. (w/ o = videos ranked without secondary motion; with = with secondary motion).

Task	w /o	with
Turn	12.5	33.3
Handoff	29.2	75.0
Wave	41.2	75.0
Beckon	50.0	54.2
Push	37.5	75.0
Slide	4.2	66.7
Grab	45.8	75.0
Lift	41.7	95.8

Table 18: Slope magnitude of mean participant vs. actual rank of three videos according to the respective state parameter level for all eight tasks and both secondary motion options. Slope magnitude closer to one indicates that participants were better able to discern rank of the modulated state parameter level for the given task. (w/o = videos ranked without secondary motion; with = with secondary motion).

Task	w /o	with
Turn	0.083	0.667
Handoff	0.188	0.500
Wave	0.479	0.667
Beckon	0.500	0.708
Push	0.604	0.875
Slide	0.500	0.791
Grab	0.625	0.875
Lift	0.604	0.979

of the resultant regression would be 1. However, the distance between the actual slope magnitude of the regressed line and the optimal value of 1 is a measure of accuracy of participants in ranking motions with and without secondary effects. The results of these linear regressions are listed in Table 18.

In Table 18, all slope magnitudes from regressions of actual and average rank for motions with secondary effects yielded values closer to 1, which indicates that for these eight tasks, in support of H2, secondary motion increased human accuracy in predicting internal and external state parameters.

3.3.9 Experiment 4: Helping to Fill in Missing Context

Experiment 4 was designed to test hypothesis H3. When watching a mime perform, there is a complete lack of context because the goal of the mime is use only her body to draw the audience into the story being told. Size and weight of absent objects or forces applied by invisible objects only become evident when the mime uses the secondary motion response (i.e. Newton's Third Law of Motion) upon her own body to communicate how the objects in the world are affecting her and are affected by her. From this inspiration comes the hypothesis that secondary motion helps to fill in missing, lost, or unperceived context. With secondary motion, the trajectory is altered to reflect the natural world response, which could be considered a form of redundancy in communication because now the social robot motion and the objects being interacted with match more closely to each other.

Since the presence of internal and external secondary motion can be resolved to force vectors acting at contact points on the robot hierarchy, context (e.g. boxes, shelves, objects) can easily be omitted as long as the time-varying force vectors are known because they can be applied consistently in simulation. Experiment 4 shows trajectories of robot motion with and without secondary motion, in both cases without context, to support that in the former case, with secondary motion human observers are better able to predict the missing context.

3.3.9.1 Experimental Design

For Experiment 4, the same eight tasks and the same forty-eight videos without context were used. However, Experiment 4 was designed to be more difficult, to find out whether stronger conclusions could be made with respect to the relationship between secondary motion and context. Experiment 3 tested whether humans can better discern context from motions with secondary motion (i.e. a binary answer); whereas Experiment 4 was designed to quantify to what extent the inclusion of secondary motion improves context prediction.

300 participants were recruited for Experiment 4. 50% of the participants had previous

robot experience and considered themselves to be above a novice level with robot expertise. The other 50% of the participants had never previously used or interacted with a robot. Each participant watched all eight possible tasks in a randomly selected order. Each participant watched one of the six possible videos per task, randomly selected for each task (across state parameter level and secondary motion option). No participant saw the same secondary motion option or same state parameter level for all six tasks.

300 participants were necessary because there are six experimental conditions (3 state parameter levels and 2 secondary motion options), which yields a sample size of 50 participants per experimental condition. Furthermore, some results will be expressed in percentages of participants, and 50 participants provides a resolution of 2% for my data.

Web-based code was written and posted to a website so participants could perform Experiment 4 online. When participants were allowed to watch videos multiple times, replay buttons were integrated into the interface. The code restricted progression through the experiment unless answers were provided where required.

In Experiment 4, a series of qualitative and quantitative questions were posed to participants in the same serial order. Each question was designed based upon different resolution of the state parameter, and order of the questions was constant so as not to bias participants for subsequent questions. After each of the questions, participants were allowed to watch the video as many times as desired before submitting an answer to each question. Participants were required to watch a video a minimum of one time between questions.

At the beginning of Experiment 4, participants watched the randomly selected video for a given task only one time, and since context was omitted from the video, participants were asked which motion they thought the robot was performing in the video. After their task label answer was recorded, they were told the task that the robot was supposed to be performing and the same motion video for that task was again shown to the participant, and each participant was asked question one. For question one, participants were given three qualitative values that corresponded to different abstract ordinal categories for the state parameter level. Question one and the ordinal, categorical values given are shown

below, organized by state parameter categories.

- Motor Temperature (Turn and handoff): Based on the robot's motion during the task it just performed, do you think the robot was operating at normal temperature, hot, or near overheating?

- Passivity: (Wave and beckon): Based on the robot's motion during the gesture it just performed, do you think the robot was loosely controlled, normally controlled, or stiffly controlled?

- Friction: (Push and slide): Based on the robot's motion during the task it just performed, do you think the box it was moving against the floor was fighting very little friction, a medium amount of friction, or lots of friction?

- Weight: (Grab and lift): Based on the robot's motion during the task it just performed, do you think the weight of the box it moved was a small, medium, or large amount?

After recording the answer to question one, the participants were shown the same motion for the same task again (to refresh their memory) and then asked question two. For question two, participants were provided with a blank empty text box, the name of the state variable parameter associated with the task, and the units associated with the variable that they should enter. To increase intuition, mass was provided to participants as the variable "weight" in units of pounds-force through the interface, and after the experiment, it was converted to mass for all calculations and analyses.

- Motor Temperature (Turn and handoff): Robot Motors' Average Temperature, Degrees Fahrenheit. Low values correspond to a cold motor. High values correspond to an overheating motor.

- Passivity: (Wave and beckon): Median Proportional Gain for All Robot Motors, Unitless. Low values correspond to a loosely controlled robot. High values correspond to a stiffly controlled robot. This was not told to participants, but in order to represent the gains of N DOFs with one value, passivity was set to be a scalar value at the

median motor proportional gain, and all other motors were scaled proportionately relative to that value.

- Friction: (Push and slide): Coefficient of Friction between the block and the surface it's moving upon, Unitless. Low values correspond to low amounts of friction. High values correspond to large amounts of friction.

- Weight: (Grab and lift): Weight of the object lifted, pounds-force. Low values correspond to a light object. High values correspond to a heavy object.

After recording the answer to question two, the participants were shown the same motion for the same task again (to refresh their memory) and then asked question three. For question three, participants were given a slider with explicitly labeled endpoints. Participants were told that they should assume that the slider endpoints are the maximum and minimum values for the respective state parameter value. They were asked to place the value of the slider in the location that they thought corresponded to the state variable for the motion seen in the video.

After recording the answer to question three for a given task, the participants were shown the same motion for the same task again (to refresh their memory) and then asked question four. For question four, participants were provided with four options, each of which was labeled with a value from the state variable parameter associated with the task in the same units from question two. Participants were told to assume that one of the four given values was the correct value for the internal or external state parameter associated with the motion (or a missing object in the video) that they just saw. Participants were asked to select which of the four values they thought was the correct state parameter value associated with the video they just saw. The three other possible answers were selected so that they were distributed very tightly around the correct answer, so that if the participant had correctly labeled qualitatively the correct region of the state parameter spectrum in question one, this would force them to narrow their answer within that region.

This process then repeated beginning at the context question for the next randomly selected task until all eight tasks were completed.

Order of the questions was constant. This was important because I wanted to identify the strongest relationship possible from humans' ability to predict state parameters. Question one is ordinal and very broad, and it allowed participants to begin thinking about whether they think the state value should be large or small. However, question two asked participants to tell the exact correct answer without being given a spectrum. And since questions three and four could have potentially biased the answer for the most specific question, it was important to test for stronger conditions before they were biased. Thus, question two preceded questions three and four. Question three provided additional constraints to the participants, and was more helpful for participants who answered question two outside the valid spectrum range. In short, the questions were ordered by increasing answer specificity with varying levels of constraints.

- Q1: 3 broad and qualitative values. Low, Medium, High. Without given spectrum or limits.
- Q2: Text-box with units and variable name. Broad and quantitative. Open-ended text box. Without given spectrum or limits.
- Q3: Open-ended slider. Given spectrum and limits. Broad and quantitative.
- Q4: Fine resolution. One of four possible multiple choice answers. Specific and quantitative.

3.3.9.2 Results

3.3.9.2.1 Increases Context Recognition

From Experiment 3, each of the six possible videos per motion (3 state parameter levels x 2 secondary motion options) was labeled eight times for all tasks, and in Experiment 4 each of the six possible videos was labeled fifty times after the participants watched the motions only one time, giving a total of 58 labels per video. The percentages of correct recognition are shown in Table 19 organized according to low, medium, and high values of state parameter levels and both secondary motion options.

Table 19: Percentages of correct recognition organized according to state parameter levels and secondary motion options for secondary motion Experiment 4. Fifty-eight samples per video. Low = low amounts of state variable. Med = medium amounts of state variable. High = high value of state variable. with = with secondary motion. w/o = without secondary motion.

	w/o			with			w/o	with
Task	Low	Med	High	Low	Med	High	Avg	Avg
Turn	24.1	25.9	25.9	29.3	31.0	31.0	25.3	30.5
Handoff	8.6	8.6	8.6	12.1	17.2	15.5	8.6	14.9
Wave	75.9	74.1	70.7	82.8	75.9	77.6	73.6	78.7
Beckon	69.0	67.2	63.8	75.9	70.7	69.0	66.7	71.8
Push	37.9	34.5	31.0	44.8	43.1	31.0	34.5	39.7
Slide	62.1	58.6	55.2	63.8	65.5	60.3	58.6	63.2
Grab	48.3	48.3	51.7	48.3	55.2	56.9	49.4	53.4
Lift	81.0	86.2	89.7	86.2	91.4	96.6	85.6	91.4
Avg	50.9	50.4	49.6	55.4	56.3	54.7	50.3	55.5

The data in Table 19 shows that on average, adding secondary motion increased correct labeling and recognition for the same state parameter level by approximately 5%. Correct recognition rates are strong functions of the task (i.e. motion type), as can be seen by the large differences between recognition rates for different motions. In general, trends for state parameters vary based more on the state parameter being modulated, rather than varying as strongly with the motion type. For example, recognition rates tended to be higher at lower passivity, higher friction, and more massive objects regardless of secondary motion option; whereas motor temperature does not appear to exhibit a trend.

The fact that secondary motion aids in correct recognition supports H3 because the label for a given motion is part of the context that is not explicit. However, there is additional context to be derived from motion in the missing state parameters, which will be discussed in the subsequent sections.

3.3.9.2.2 Improves State Parameter Prediction

In Experiment 3, I found that people can correctly order videos of robot motion according to a spectrum of a state variable, when they see at least three points along that spectrum.

Table 20: Percentages of correct qualitative state parameter level association (i.e. question one) organized according to state parameter levels and secondary motion options for secondary motion Experiment 4. Fifty samples per video. Low = low amounts of state variable. Med = medium amounts of state variable. High = high value of state variable. with = with secondary motion. w/o = without secondary motion.

	w/o			with			w/o	with
Task	Low	Med	High	Low	Med	High	Avg	Avg
Turn	30.0	36.0	32.0	56.0	58.0	60.0	32.7	58.0
Handoff	38.0	34.0	30.0	40.0	46.0	58.0	34.0	48.0
Wave	42.0	38.0	40.0	68.0	82.0	88.0	40.0	79.3
Beckon	60.0	58.0	58.0	66.0	70.0	74.0	58.7	70.0
Push	40.0	36.0	28.0	84.0	86.0	92.0	34.7	87.3
Slide	36.0	38.0	34.0	78.0	70.0	76.0	36.0	74.7
Grab	60.0	60.0	64.0	84.0	90.0	94.0	61.3	89.3
Lift	56.0	68.0	70.0	88.0	94.0	96.0	64.7	92.7
Avg	45.3	46.0	44.5	70.5	74.5	79.8	45.3	74.9

In Experiment 4, I tested using a single point on the spectrum, to find out how accurate humans are at discerning state parameter magnitudes when context is missing for motions with and without secondary effects.

For question one, participants were classifying motion based on state parameter level into one of three possible qualitative categories: low, medium, or high. Based solely on chance, prediction should have been 33.3% if there is no trend according to the state variable or secondary motion option. However, the percent correctly classified data for question one listed in Table 20 shows that humans are better than chance classifiers for motion that varies based on state parameter level, regardless of whether the trajectory includes secondary motion (45.3% on average for motion without secondary effect, and 74.9% on average with secondary motion). These results mean that humans can discern missing context when the state parameter values manifest change in the motion due to the missing context alone. However, the addition of secondary motion clearly benefits the resultant prediction.

One interesting trend found in Table 20 was that without secondary motion, humans were slightly better at classifying median values of a state parameter (46% on average is

highest of all three levels without secondary motion). One possible explanation for this is that these median values tend to coincide with more typical state parameter values (e.g. normal operating temperatures, normal friction, boxes of average mass), which would result in motions more common or more familiar to humans (as compared to motion without secondary motion but exhibiting extreme values of state parameter levels). However, secondary motion tends to manifest more change in motion at extreme values of a particular state parameter level (e.g. when mass is heavy, when the robot is near overheating, when the actuators are very passive), and thus, classification of motion with secondary effects based on qualitative state parameter level was much better at extreme values of state parameters (79.8% on average at the high state parameter level).

The percent of correct responses improved by 29.7% on average when secondary motion was included. Furthermore, when the data in Table 20 was treated as independent distributions (all eight motions in the same distribution; three distributions per secondary motion option), the pairwise t-tests between the average correct response rates with and without secondary motion for each of the three state parameter levels showed that the distributions for all three state parameter levels were statistically significant ($p<0.05$). I concluded that the addition of secondary motion to trajectories with either the low, medium, or high state parameter level produces statistically significant increase in percentage of correct responses for participants attempting to discern state parameter magnitude just from watching the robot move. Therefore, the relative improvements of 25.3%, 28.5%, and 35.3% for the respective parameter levels of low, medium, and high demonstrate the benefit of secondary motion in that it improved human ability to predict contextual information and state parameters.

For question two, participants were asked the most difficult of all four questions because they were asked to apply a quantitative label for the state parameter without seeing any other motions, given only the state parameter name, one motion, and the units for the label. In many cases, it is feasible that participants might be unfamiliar with a particular unit of measurement or state parameter, and yet, they were still asked to guess a number.

The responses were accumulated into 48 distributions (8 motions x 2 secondary motion

Table 21: Percentages of error between average state parameter magnitude provided by participants (i.e. question two) and actual value used to synthesize motion. Data organized according to state parameter levels and secondary motion options for secondary motion Experiment 4. Fifty samples per video. Low = low amounts of state variable. Med = medium amounts of state variable. High = high value of state variable. with = with secondary motion. w/o = without secondary motion.

	w/o			with			w/o	with
Task	Low	Med	High	Low	Med	High	Avg	Avg
Turn	43.7	50.8	68.7	40.3	47.6	51.2	54.4	46.4
Handoff	44.3	56.4	65.0	39.7	51.3	43.4	55.2	44.8
Wave	606.2	249.9	1146.4	599.8	219.7	334.2	667.5	384.6
Beckon	429.8	300.7	1006.7	393.8	223.4	831.7	579.1	483.0
Push	319.0	246.5	156.9	290.6	195.6	131.1	240.8	205.8
Slide	977.0	221.3	161.3	882.1	166.2	138.7	453.2	395.7
Grab	334.1	67.9	51.6	309.9	63.3	41.0	151.2	138.1
Lift	402.2	41.0	44.2	376.4	37.4	29.7	162.5	147.8
Avg	394.5	154.3	337.6	366.6	125.6	200.1	295.5	230.8

options x 3 state parameter levels). Percent error between the mean of the participant-provided distribution of responses and the actual value is given in Table 21. The magnitudes of the percent error provided a measure of how familiar participants were with a given state parameter. High values of percent error indicate that participants were unfamiliar with the relationship between a state parameter and how it manifests in motion.

The data in Table 21 shows that humans were more accurate predictors of state parameters when secondary motion is included. However, based on the high values of percent error, I concluded that humans were not very adept at discerning exact magnitudes of state variables based on seeing motion alone. Experiment 4 was testing a combination of factors: (1) do humans understand the state parameter? and (2) do humans understand how changing the state parameter will affect the robot's motion? In certain cases it is difficult to tell based solely upon percent error because percent error is a strong function of the correct value, and very small correct values have more percent error bias when participants guess at an answer. However, from the data, it is clear that mass is an example of a state parameter that humans understand well and understand it's effect on motion

well. Friction also appears to be a state parameter that participants were familiar with, but the percent errors are biased high likely due to small actual values. Temperature seems to be an example opposite of friction. My participants understood temperature well, but not its effect on motion. And finally, the data suggests that passivity seemed to be a variable that participants did not understand, nor did they understand it's effect on motion well.

Each of the distributions for human-provided values of state parameters were compared against the actual value used in motion generation to determine statistical significance. 48 t-tests were performed to find out if the actual value for each motion fell within the human-provided distributions ($p<0.05$). 18 of 24 possible motions *without* secondary effects were statistically significant; whereas, 13 of 24 possible motions *with* secondary effects were statistically significant. This means that in 18 of the possible cases without secondary motion, the actual value for the state parameter was statistically different than the distribution of human-provided responses for that parameter, after watching the motion. However, for only 13 of the possible videos with secondary motion, the actual value for the state parameter was outside the distribution. This means that secondary motion improves humans' ability to predict state parameter values and fill-in missing context.

Question two was a very difficult task to expect humans to perform well, even when they understand the state variables and the effects on motion well, which is why the percent errors in Table 21 are so high. I expected that such a task would be difficult, which is why questions three and four existed for the study.

In question three, participants were given a slider, with bounded ends, and they were asked to place the slider at the location of the state variable they thought corresponded to the value used to create the motion in the video. This task revealed the quantitative spectrum for a given state parameter, which means that participants could have interpreted the endpoints as maximum and minimum acceptable values (even though they weren't explicitly told this). These bounds make the task of labeling the correct answer easier, especially in cases where the participants do not understand the state variable or have intuition about its units. For example, if they thought that the state variable is high, they could have placed the slider in the general area of the upper half of the spectrum and have

Table 22: Percentages of error between the average state parameter magnitude provided by participants (i.e. question three) and actual value used to synthesize motion. Data organized according to state parameter levels and secondary motion options for secondary motion Experiment 4. Fifty samples per video. Low = low amounts of state variable. Med = medium amounts of state variable. High = high value of state variable. with = with secondary motion. w/o = without secondary motion.

	w/o			with			w/o	with
Task	Low	Med	High	Low	Med	High	Avg	Avg
Turn	40.1	47.6	64.6	33.3	29.8	41.6	50.8	34.9
Handoff	48.4	43.6	59.1	29.5	30.3	39.2	50.4	33.0
Wave	69.7	68.7	52.1	18.9	33.7	31.2	63.5	27.9
Beckon	45.1	49.3	42.3	27.6	22.5	20.5	45.6	23.5
Push	50.5	55.9	54.4	19.4	29.4	28.6	53.6	25.8
Slide	62.6	65.0	68.1	25.9	35.4	45.2	65.2	35.5
Grab	32.9	33.8	40.9	35.8	26.4	26.1	35.9	29.4
Lift	34.9	30.1	31.6	31.3	29.3	27.8	32.2	29.5
Avg	48.0	49.3	51.6	27.7	29.6	32.5	49.6	29.9

a better guess than in question two, where state variable bounds are not provided.

The same analysis as in question two was repeated with the distributions provided by the sliders for question three. These results can be interpreted by the hypothetical statement of 'if all participants had answered question two knowing acceptable limits of all state variables, the results from question two would have been...'

The percent error values in Table 22 are drastically improved over the values in Table 21, which shows that by providing valid bounds on acceptable state parameter ranges, humans were better able to predict missing context. More importantly, humans had less error in their state variable value predictions when secondary motion was included.

The 48 t-tests were repeated to find out if the actual value for each motion fell within the human-provided distributions ($p<0.05$) from the sliders in question three. 13 of 24 possible motions *without* secondary effects were statistically significant; whereas, 1 of 24 possible motions *with* secondary effects was statistically significant. This means that in 13 of the possible cases without secondary motion, the actual value for the state parameter was statistically different than the distribution of human-provided responses for that

parameter, after watching the motion. However, for only 1 of the possible videos with secondary motion, the actual value for the state parameter was outside the distribution (23 of 24 distributions for state parameter estimates of motions with secondary motion were not statistically different than the actual value for that state parameter). This means that secondary motion improves humans' ability to predict state parameter values and fill-in missing context.

The percent error was improved by 19.7% on average when secondary motion was included. Furthermore, when the data in Table 22 was treated as independent distributions (all eight motions in the same distribution; three distributions per secondary motion option), the pairwise t-tests between the percent errors with and without secondary motion for each of the three state parameter levels, showed that the distributions for all three state parameter levels were statistically significant ($p<0.05$). I concluded that the the addition of secondary motion to trajectories with either the low, medium, or high state parameter level produced statistically significant improvement in percentage error for participants attempting to discern state parameter magnitude just from watching the robot move. Therefore, the relative improvements of 20.3%, 19.7%, and 19.1% for the respective parameter levels of low, medium, and high demonstrated the benefit of secondary motion in that it improves human ability to predict contextual information and state parameters.

In question four, the participants were presented with a four-option multiple choice question for each motion in the study. For each of the eight motions per participant, they were asked to identify the correct value for a given state parameter when given four different options. The values were tightly distributed around the correct value to increase difficulty and test humans' ability to discern state parameters at a very fine resolution. Since 25% is chance, any trend must be greater than this percentage. On average, participants' ability to discern the magnitude of a state parameter level was better than chance, regardless of whether the trajectory includes secondary motion, as shown in Table 23. This means that state parameters manifest perceptible change in the trajectory. However, it was necessary to test whether secondary motion improved the human ability to discern state parameter magnitude from motion.

Table 23: Percentages of correct quantitative state parameter level association (i.e. question four) organized according to state parameter levels and secondary motion options for secondary motion Experiment 4. Fifty samples per video. Low = low amounts of state variable. Med = medium amounts of state variable. High = high value of state variable. with = with secondary motion. w/o = without secondary motion.

	w/o			with			w/o	with
Task	Low	Med	High	Low	Med	High	Avg	Avg
Turn	25.8	19.2	27.4	19.3	16.1	17.4	24.1	17.6
Handoff	22.8	24.3	26.1	22.9	23.4	20.3	24.4	22.2
Wave	27.5	28.6	23.5	23.6	24.0	21.1	26.5	22.9
Beckon	19.9	22.8	25.6	23.2	24.7	22.9	22.8	23.6
Push	20.2	25.3	20.6	18.9	17.4	17.1	22.0	17.8
Slide	29.3	24.9	22.7	17.6	18.3	17.6	25.6	17.8
Grab	20.9	26.7	23.1	15.4	17.8	17.0	23.6	16.7
Lift	15.5	22.3	23.2	14.2	16.9	15.3	20.3	15.5
Avg	22.7	24.3	24.0	19.4	19.8	18.6	23.7	19.3

The percent of correct responses improved by 4.4% on average when secondary motion was included. Furthermore, when the data sets in Table 23 were treated as independent distributions, the pairwise t-tests between the average correct response rates with and without secondary motion for each of the three state parameter levels, showed that the distributions for all three state parameter levels were statistically significant ($p<0.05$). Just by watching the robot move, regardless of the state parameter level, participants were able to more accurately discern state parameter magnitude due to the presence of secondary motion when compared to motion that lacked secondary action. The relative improvements of 3.4%, 4.4%, and 5.4% for the respective parameter levels of low, medium, and high demonstrated that secondary motion is beneficial in improving human ability to predict contextual information and state parameters.

Overall, the data from all four questions supports hypotheses H2 and H3, that including secondary motion helps participants to predict state parameters and fill-in missing context. Regardless of how familiar the participants were with state parameters, their error rates and percentage of correctly predicted responses for state parameter values (qualitative and quantitative) improved in Tables 20 - 23. And in the three questions

where participants were given either options or the spectrum for a state parameter, the distribution of responses at all three motion levels were statistically different. Question two was challenging since it asks participants to predict the exact numerical value of a state parameter without being explicitly given boundaries of the distribution or any other constraints. Unlike the other three questions, where statistical significance could be found across motions and state parameters, for question two I can only draw conclusions for the same state parameter level and motion. In these pairwise t-tests, trajectories without secondary motion showed an increase in the number of occurrences of the actual value being statistically different from the human predicted distribution.

During the interviews following Experiment 4, one interesting point raised by participants was that certain assumptions have to be made for estimation of state parameters. For example, since mass (an external state variable) is not independent of the robot strength (an internal state variable), prediction requires that assumptions be made about dependent variables. Therefore, some participants said that they used physical size and appearance to fill in the missing (i.e. ungiven) context of robot strength. The robot is big, so it is likely strong. For clarity, it is helpful to make statements about dependent variables during the course of the study so that it is more controlled and participants do not need to make assumptions.

3.3.10 Experiment 5: Making Motion More Natural

In Experiment 5, participants were asked to state qualitative preferences while watching motion pairs with and without secondary motion. Secondary motion as created by my algorithm creates a response that is more consistent with physics (as discussed in Section 3.3.2). Since physics is the basis for natural motion of an articulated body, then in theory, humans should state a preference for motions with secondary motion over motion without the secondary motion dynamics. Also, humans are used to seeing motion consistent with physics that includes secondary motion dynamics. Motion without these secondary dynamics should, in theory, seem less natural.

3.3.10.1 *Experimental Design*

In Experiment 3, each participant only viewed six of the eight possible tasks. In order to maximize use of participants, during the final part of Experiment 3, each participant was asked to watch two of the possible six videos for each of the last two remaining tasks that the participant had not previously seen.

The two videos selected were one of three possible pairs, and each pairing contained one video with and one video without secondary motion. Since each of the trajectories without secondary motion had been used to create a version of that trajectory with secondary motion, pairings were naturally assigned based on origin.

Sixteen participants watched the 2 unseen tasks from Experiment 3, which yields an average of 5.33 viewings per video pair per task. Since larger data quantities were desired, an additional one hundred eighty-five participants were recruited to watch one video pair in a random order for all eight tasks. This provides an average of 67 viewings per video pair per task, which for results expressed as a percentage is 1.4925% data resolution.

In Experiment 5, participants watched the video pair in a random order as many times as they desired. Afterward, they were asked to tell which motion looked better and which motion looked more natural. This continued until all eight tasks were done in random order. When performed as a supplement to Experiment 3, only two video pairs were shown before the supplementary part of that experiment was complete.

3.3.10.2 *Results*

The data in Table 24 demonstrates that approximately seven out of ten people prefer trajectories with secondary motion, which supports H4. Furthermore, different trends exist for each of the different state variables. Low temperatures, high gains (low passivity), median amounts of friction, and heavy weights are preferred. Except for heavy weight, the other three state variables matched more closely with values typical of normal operating conditions.

Table 24: Percent of participants who chose trajectories with secondary motion as "looked better" and "looked more natural" as compared to trajectories without secondary motion. Low = low amounts of state variable. Med = medium amounts of state variable. High = high value of state variable.

	better				more natural			
Task	Low	Med	High	Avg	Low	Med	High	Avg
Turn	71.6	67.2	64.2	67.7	71.6	68.7	64.2	68.2
Handoff	74.6	74.6	62.7	70.6	73.1	65.7	65.7	68.2
Wave	53.7	64.2	71.6	63.2	56.7	76.1	77.6	70.1
Beckon	58.2	64.2	74.6	65.7	55.2	65.7	74.6	65.2
Push	67.2	58.2	76.1	67.2	77.6	68.7	74.6	73.6
Slide	89.6	65.7	85.1	80.1	71.6	70.1	80.6	74.1
Grab	62.7	71.6	74.6	69.7	64.2	67.2	77.6	69.7
Lift	68.7	76.1	79.1	74.6	74.6	76.1	77.6	76.1
Avg	68.3	67.7	73.5	69.8	68.1	69.8	74.1	70.6

3.3.11 Experiment 6: Improving Motor Coordination

In Section 4.1, a full analysis of a metric for human-like motion was performed, which was inspired by observations made during Experiment 6 with secondary motion.

Coordination of distributed, independent actuators results from a small number of controlled degrees-of-freedom, constituting a primary or goal-oriented motion, which exhibit small variance and common direction of motion [76, 49]. Since the definition of motor coordination sounds very similar to the low dimensional representation of actuated coordinates, I believe that a relationship between motor coordination and secondary motion exists. Thus, I exploited this controllable subspace to add evidence for my hypothesis that spatiotemporal correspondence (i.e. motor coordination) in task space is a metric for human-like communicative motion.

As discussed in Sections 2.3 and 4.1, Kolmogorov-Sinai entropy, which measures rate of system state information loss as a function of time, is a proven metric for spatiotemporal coordination of distributed actuators [51]. Dependent DOF variance in magnitude and direction is a characteristic of natural movements [77]. Also, computer animation has widely accepted for decades that primary motion coupled with secondary motion looks better

than the former alone [41, 42]. Thus, if secondary motion can be consistently correlated to more coordinated actuators, then my hypothesis that a relationship between Kolmogorov-Sinai entropy (i.e. motor coordination, correspondence, DOF inter-dependence) and secondary motion is correct. Also, as will be shown in Section 4.1, evidence suggests that Kolmogorov-Sinai entropy is correlated to human-like motion. Therefore, if a relationship between secondary motion and KSE can be found in Experiment 6, based on the data presented in Section 4.1, conclusions can be drawn about secondary motion and its relationship to both motor coordination and human-likeness of motion. Specifically, if secondary motion can be correlated to more coordinated actuators, by use of Kolmogorov-Sinai entropy, then I can add evidence that motion that includes a secondary response is more human-like than motion without it. The latter is a reasonable expectation because by virtue of Newton's third law, secondary motion is a natural element of motion.

The existence of a relationship between secondary motion and motor coordination is intuitive. Consider the extreme situation presented in Experiment 1, where a subset of DOFs were moving and a subset of DOFs were completely idle. In the tennis swing motion, these two distinct groups of motors were not coordinated because to create coordination requires that the actuated DOFs stop moving or the unactuated DOFs begin to move. The latter is the preferred method of creating coordination, since the objective is motion. In short, if the subset of idle DOFs exhibit motion, motor coordination should improve.

3.3.11.1 Experimental Design - Initial

As an initial test of hypothesis H5, I examined all the secondary motion algorithms for the tennis swing motion with respect to Kolmogorov-Sinai entropy presented in Experiment 1. Details of the KSE calcuation are outlined in Section 2.3 [51]. The robot hardware has finite spatial extent due to finite number of DOF with finite joint angle and torque limits, and finite temporal extent due to the motion trajectories being cyclic.

Kolmogorov-Sinai entropy was calculated using a measure called K_2, which has two distinct parts: spatial correspondence (SC) and temporal correspondence (TC). These measures assess coordination in magnitude and timing, respectively. One of the conclusions

of my thesis is that for K_2 to be used as a measure for human-likeness of motion quality, SC and TC are best kept as independent measures, since they provide more information in these separate numbers (see Section 4.1.6.2.3 for details). Thus, subsequent analysis will refer to SC and TC independently. For both of these measures, values closer to zero indicate more coordination.

3.3.11.2 Results - Preliminary

Perfect motor coordination exists when KSE is zero. By calculating baseline KSE without any secondary motion and comparing with KSE of each of the three techniques, any technique that improves the KSE over the baseline, also produces more coordinated motion. For the tennis swing motion, since the left arm is passive, these left arm DOF belong to the null space in task- and joint-space. The calculated baseline entropy rate of the original trajectory, using the correlation entropy estimate K_2 from [52] with fixed time delay corresponding to the duration of the motion and fixed spatial extent including all sixteen controllable hardware body DOF was 0.33 for SC and 0.27 for TC. Both simulation-in-the-loop and feedforward & feedback techniques improved K_2 to (0.24 for SC and 0.13 for TC) and (0.14 for SC and 0.18 for TC) respectively, indicating that simply by moving the left arm DOF the overall motion was more coordinated in actuation. Eigenphysics further improved the KS-entropy to 0.09 for SC and 0.10 for TC, suggesting that, in general, eigenphysics produces the most coordinated motion of all three secondary motion techniques.

These preliminary results support my hypothesis of task-space motor coordination, and the order of the KSE values for these results (i.e. SIL as least coordinated and eigenphysics as most coordinated) was expected because techniques that add secondary motion to more DOFs have greater impact on overall coordination through affecting more DOFs. SIL had the smallest number of DOFs affected since secondary motion was isolated to the left arm. After adding secondary motion, those DOFs moved in passive response to primary motion, which produced coordinated motion in spatial extent. F&F control allowed all DOF to deviate slightly from the feedforward command, restructuring the motion with more coordination, but still the effect had temporal delay since it was filtered

by the low gains. Whereas, eigenphysics affected all DOFs by producing new torques that are a linear combination of the old torques for all DOF, which allows for the possibility of more space and time coordination between actuators without the time-lag created by low gain control, which adversely affects temporal coordination of distributed actuators.

3.3.11.3 Experimental Design

To increase confidence that adding secondary motion improves human-likeness, KSE was calculated for 492 motions that were motion-captured from humans mimicking twenty unique motions (i.e. 24-25 examples per motion, the participant unconstrained mimicking examples from Section 4.1.6). Each of these motions represent a human attempting to perform their own variant of a specific type of motion (e.g. waving, shrugging). Thus, the calculated KSE from these examples provided a human-like distribution for KSE that captures a range of different ways that a particular motion type can be performed.

Since these 492 motions come from humans, secondary motion is likely to already be included in the trajectory. Therefore, one trajectory of each of the twenty unique motions was synthesized that did not include secondary motion. These variants were created through animation of a scale model of the robot in Maya 3-D animation software. Trajectories were recorded from the animations in Maya for execution on the hardware. Each of these twenty trajectories was augmented with secondary motion by the three different algorithms outlined in previous sections. Comparison of KSE for each of the four trajectories (i.e. without secondary motion, SIL, F&F, and eigenphysics) with the mean of the KSE for the human-like distributions can be used to provide insight as to whether KSE (i.e. motor coordination) is more human-like with or without secondary motion.

Furthermore, these same results can be used to find out if H5 holds true in general across a variety of motions. If KSE is consistently lower for trajectories with secondary motion, I will have higher confidence that secondary motion increases motor coordination.

The comparison criteria for motor coordination and human-likeness are different. Coordination is optimal at zero, which is unachievable for human motion because muscles supply finite torque and joints have limited angle ranges. Therefore, human-likeness is

optimal as close to zero KSE as possible given the constraints of the human body, which means that optimal KSE for human-likeness will always be some finite non-zero number.

As a follow-up calculation, the participant unconstrained data (i.e. set of 24-25 human motions per motion type) was projected onto the robot hardware to become the ground truth data set for human-like motion. Nearest-neighbor, with a Euclidean distance metric in joint angle space, was used to find the minimum distance between the each of the four categories of secondary motion (i.e. without, SIL, F&F, and eigenphysics) and a trajectory in the ground truth set. This analysis assumes that distance to a ground truth trajectory is a good metric for human-like quality of a given motion. Certain DOFs (eyes, eyelids, and ears) were excluded from the analysis because they are not measurable with the motion capture equipment.

3.3.11.4 Results

As a general rule, for one-handed gestures such as 1-handed bow, scan, wave, throw, and beckon, the DOFs in the entire arm not utilized in the gesture were selected as passive for SIL. Additionally, the wrist X and wrist Z DOFs on the arm involved in the primary motion were typically selected to be passive in simulation in SIL, unless they were part of a symbol for the gesture, such as support for the hand in waving. Thus, each of the twenty motions had 2-9 passive DOFs for SIL (of 16 total body DOFs), depending on what was deemed appropriate for the given gesture.

Table 25 shows the spatial correspondence for the twenty motion for human averages, as well as the animations without secondary motion and each of the three secondary motion algorithms. For all twenty motions, adding secondary motion (for all three techniques) reduced the value of SC, bringing it closer to the human mean value. By adding secondary motion and improving the SC value closer to zero, I concluded that the presence of secondary motion improves the motor coordination, which supports hypothesis H5.

Furthermore, 14 of 20 F&F values and 19 of 20 eigenphysics values of SC were within one standard deviation of the human mean SC. 13 of 20 of these SC values for F&F and 14 of 20 of the SC values for eigenphysics failed t-tests for statistical significance ($p<0.05$)

Table 25: Spatial correspondence (SC) for twenty different motion types. Human represents the *mean* SC from 24-25 human motion-captured examples. W/o = animated trajectory without secondary motion. SIL, F&F, and Eig represent KSE for the animated trajectory with secondary motion created by one of the three algorithms. (w/o = without secondary motion; Eig = eigenphysics)

Motion	human	w/o	SIL	F&F	Eig
Wave	0.108	0.424	0.239	0.181	0.117
Beckon	0.090	0.294	0.247	0.177	0.113
Shrug	0.109	0.308	0.290	0.139	0.087
Move Object	0.115	0.306	0.287	0.134	0.089
1-hand Bow	0.124	0.378	0.219	0.180	0.111
2-hand Bow	0.102	0.398	0.279	0.164	0.103
Scan	0.112	0.335	0.249	0.181	0.123
Look Around	0.088	0.297	0.261	0.167	0.104
Worship	0.113	0.328	0.269	0.167	0.102
Present	0.144	0.397	0.228	0.160	0.107
Air Guitar	0.157	0.422	0.279	0.168	0.112
Shucks	0.149	0.429	0.253	0.149	0.099
Bird	0.136	0.276	0.273	0.154	0.103
Stick 'em up	0.134	0.374	0.273	0.142	0.085
Cradle	0.129	0.438	0.276	0.149	0.099
Call/Yell	0.119	0.378	0.298	0.144	0.088
Sneeze	0.088	0.374	0.316	0.143	0.090
Cover	0.112	0.272	0.293	0.136	0.091
Throw	0.139	0.387	0.230	0.187	0.115
Clap	0.090	0.269	0.307	0.110	0.065

when compared with the human distribution for unconstrained versions of the gesture, which is statistical evidence that adding secondary motion makes a trajectory magnitude more human-like.

Table 26 shows the temporal correspondence for the twenty motion for human averages, as well as the animations without secondary motion and each of the three secondary motion algorithms. For all twenty motions, adding secondary motion (for all three techniques) reduced the value of TC, bringing it closer to the human mean value. By adding secondary motion and improving the TC value closer to zero, I concluded that the presence of secondary motion improves the motor coordination, which supports hypothesis H5.

Furthermore, 6 of 20 SIL values, 2 of 20 F&F values and 19 of 20 eigenphysics values of

Table 26: Temporal correspondence (TC) for twenty different motion types. Human represents the *mean* TC from 24-25 human motion-captured examples. W/o = animated trajectory without secondary motion. SIL, F&F, and Eig represent KSE for the animated trajectory with secondary motion created by one of the three algorithms. (w/o = without secondary motion; Eig = eigenphysics)

Motion	human	w/o	SIL	F&F	Eig
Wave	0.045	0.340	0.113	0.183	0.072
Beckon	0.021	0.217	0.129	0.266	0.048
Shrug	0.040	0.281	0.184	0.188	0.026
Move Object	0.049	0.344	0.180	0.257	0.024
1-hand Bow	0.067	0.333	0.106	0.298	0.037
2-hand Bow	0.042	0.234	0.179	0.252	0.045
Scan	0.068	0.328	0.136	0.166	0.059
Look Around	0.073	0.329	0.141	0.176	0.050
Worship	0.052	0.279	0.148	0.168	0.051
Present	0.060	0.202	0.109	0.184	0.025
Air Guitar	0.061	0.247	0.177	0.251	0.086
Shucks	0.107	0.335	0.141	0.225	0.030
Bird	0.052	0.262	0.168	0.186	0.024
Stick 'em up	0.052	0.231	0.171	0.189	0.031
Cradle	0.130	0.233	0.173	0.210	0.023
Call/Yell	0.067	0.253	0.194	0.221	0.062
Sneeze	0.140	0.248	0.196	0.228	0.036
Cover	0.083	0.321	0.185	0.241	0.062
Throw	0.062	0.253	0.118	0.296	0.050
Clap	0.039	0.326	0.198	0.201	0.031

TC were within one standard deviation of the human mean TC. 4 of 20 of these TC values for SIL and 15 of 20 of the TC values for eigenphysics failed t-tests for statistical significance ($p<0.05$) when compared with the human distribution for unconstrained versions of the gesture, which is statistical evidence that adding secondary motion makes a trajectory magnitude more human-like.

These results also illustrate the advantages and disadvantages of SIL. This technique improved the temporal motor coordination more significantly than the spatial coordination. The temporal advantage arises because SIL has no low gain feedback on the passive DOFs, and passive DOFs move without an artificially-induced temporal delay. Secondary motion was only allowed to influence at most nine degrees-of-freedom in the motions

for SIL in this analysis, which is just 56.25% of all possible controllable body DOFs (for any motion). Nearly all DOF trajectories require modification if a trajectory is to become human-like. However, this is dependent upon the initial trajectory to which secondary motion is added. If this primary trajectory is highly coordinated in the DOFs that SIL cannot affect, the results of SIL can be very good. However, in Experiment 6, these primary motions were animations, and to not bias the experiment, no attempt was made to ensure they were coordinated prior to use.

Tables 25 and 26 show that motor correspondence (i.e. coordination) is highly motion dependent. This result was expected because humans exploit the position, velocity, and acceleration of their degrees-of-freedom relative to each other to create different motions and gestures.

Table 27 shows the distance between the nearest neighbor in the human-like data set and original trajectory in each of the four possible forms of secondary motion (without, SIL, F&F, and eigenphysics). When comparing columns relative to each other, lower values in Table 27 indicate that for a particular motion, the technique with that type of secondary motion is more similar to a human example from one of the twenty human motions that were optimally projected onto the robot kinematic hierarchy.

For all twenty motions, all three secondary motion algorithms were closer in Euclidean distance to a human exemplar from the data set, than any motion without secondary motion. On average, the eigenphysics secondary motion was nearest to human-likeness, when using Euclidean distance to projected human examples in joint space as the measure of human-like motion.

3.3.12 Discussion

I presented three independent techniques for generating secondary motion. Each of the three techniques for producing secondary motion has advantages and disadvantages, and the following discussion supports identification of situations when each technique would be the appropriate implementation choice to create secondary motion.

Table 27: Distance of the motions from the three secondary-motion-generating techniques to its nearest neighbor in the human-like data set, for each of the twenty gestures. Values measure joint angle Euclidean distances in radians. (w/o = without secondary motion; Eig = eigenphysics)

	w/o	SIL	F&F	Eig
Wave	0.233	0.202	0.155	0.139
Beckon	0.274	0.193	0.184	0.155
Shrug	0.261	0.179	0.149	0.162
Move Object	0.212	0.167	0.161	0.141
1-hand Bow	0.239	0.189	0.172	0.167
2-hand Bow	0.312	0.199	0.182	0.172
Scan	0.272	0.204	0.178	0.181
Look Around	0.256	0.213	0.192	0.185
Worship	0.234	0.177	0.181	0.170
Present	0.256	0.164	0.156	0.149
Air Guitar	0.271	0.153	0.164	0.144
Shucks	0.239	0.188	0.183	0.136
Bird	0.241	0.196	0.171	0.172
Stick 'em up	0.244	0.172	0.172	0.145
Cradle	0.254	0.159	0.158	0.133
Call/Yell	0.271	0.182	0.147	0.179
Sneeze	0.261	0.191	0.177	0.161
Cover	0.281	0.194	0.206	0.182
Throw	0.290	0.186	0.172	0.149
Clap	0.214	0.166	0.159	0.121

SIL has the advantages of being simple and intuitive; it can be validated, easily amplified, and requires no *a priori* specification of the trajectory to create secondary motion from internal forces.

If an accurate dynamic model of the robot does not exist, then it is difficult to use SIL to generate secondary motion. Another consideration when using SIL is deciding which DOFs to select to simulate passively; the appropriate DOFs depend on the task. To appropriately select these DOFs for SIL, it is important to understand which ones play an essential role in the primary action. Generally, DOFs near the robot's extremities are best to select for secondary motion. However, if for example a wrist DOF is being used, then it is not ideal to have it in the secondary motion set. For example, the requirement of

simulated passive DOFs makes SIL an unfavorable choice for producing secondary motion during manipulation trajectories.

There are many advantages to F&F in creating secondary motion. The feedforward torques greatly reduce the control energy necessary to command a trajectory. Additionally, the very low control gains make the system less stiff. F&F requires a less strict model for the robot than that needed for SIL. Only accurate models of the feedforward terms are necessary to correctly control the robot and create secondary motion. These feedforward terms can also be read directly off the robot, which further reduces modeling requirements.

Additionally, F&F can be used even when a trajectory is unknown in advance, for example predicting the feedforward torques through online planning algorithms, which are pervasive in robotics. Even when there is no advance planning, the feedforward torques to hold against gravity for the current time step are a decent estimate for one time step in the future, thereby reducing the amount of actuation dependently supplied by low gain feedback.

One of the main drawbacks to the F&F technique is creating a new set of stable gains for the low gain feedback portion. Often gain tuning can be considered an art form, despite the many developed techniques to assist (e.g. bode, root locus, Ziegler-Nichols). However, this disadvantage needs to be overcome only one time, and thereafter, does not hinder usage of the technique. It is also a disadvantage that occurs in development and does not impose penalties at run-time or affect real-time performance. Thus, in terms of disadvantages, the need for an additional set of gains is relatively benign.

Eigenphysics is the most computationally costly of the three techniques, but I recommend this technique over the others when trajectories are known in advance. The presented results prove it can produce secondary motion similar to the other techniques without the complexity of determining unactuated DOFs as in SIL. And, the computation cost is moved offline when the trajectories are known in advance, instead of being calculated in real-time. Eigenphysics produces secondary motion in all DOF, which makes it an optimal choice for motions such as manipulation where the primary trajectory requires minimal interruption.

3.3.13 Summary

I presented three novel techniques to create natural, compliant, secondary motion for humanoid robots. The approaches exploit simulation methods and passivity in actuation to create virtual secondary motion from both internal and external forces that is used to command the robot, while keeping all real-world hardware fully actuated. The techniques overcome hardware and software limitations that reduce secondary motion created by natural means. By coupling a robot to an accurate, real-time dynamic simulation, perception of internal and external robot characteristics (or world object characterstics) can be altered.

I demonstrated resulting motions for internal and external forces, and illustrated the trade-offs between the three techniques. Furthermore, I have presented quantitative evidence to show that greater quantities of secondary motion are produced in near-unactuated DOFs than in highly actuated DOFs and that the torque trajectories are significantly different from the original motion for near-unactuated DOFs. Statistical evidence was presented to support the conclusions that secondary motion improves humans' ability to predict missing context, as well as internal and external state parameters. Subjective evaluations demonstrate a preference of robot trajectories with secondary motion, and trajectories with secondary motion were shown to decrease K_2, which is known to increase the human-likeness of robot motion.

Additional communication must be added to motion as secondary motion in the null space of the task by increasing task-space motor coordination to communicate as clearly as possible with the human partner during interaction scenarios. Based on the presented data, physical attributes of a robot or a projection of an internal state can be communicated in motion.

CHAPTER IV

HUMAN-LIKE MOTION

Human-like motion will be created through an optimization that adds coordination between robot motors, which emulates the effects of degrees-of-freedom coupled by muscles. The resultant motion appears more human-like, which is validated by a series of experiments. Furthermore, to avoid repetition, which is a quality absent in human motion trajectories, I designed a variant-generating algorithm that synthesizes an infinite number of exemplars from a single input motion.

4.1 Synthesis & Evaluation

4.1.1 Introduction

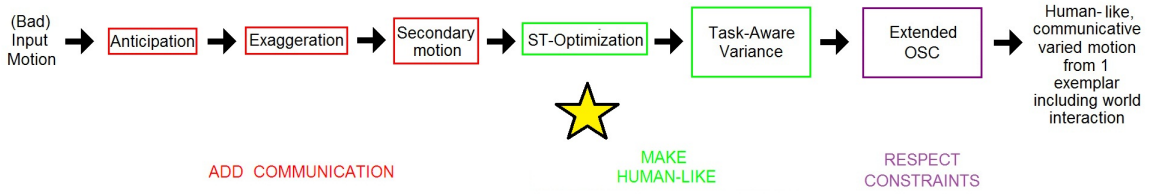

Figure 26: Section 4.1 discusses the human-like optimization.

The human-like optimization is the first of the two human-like motion algorithms discussed in this thesis, as shown in Figure 26. It follows the three communicative motion algorithms because the three communicative motion algorithms establish the appropriate task and bring the input motion closer to the desired state before optimizing. The spatiotemporal optimization precedes the variance algorithm because the variance algorithm outputs multiple motions; fewer computations need to be performed if the optimization is only performed once [78].

By using motion as a communication channel, redundancy and consistency in the signal being communicated increase. But multimodality and sharing a common "language"

are just a couple reasons why social humanoid robots should move like their collaborative counterparts. Communicating in a manner that is socially relevant and familiar to their human partners is called "natural human-robot interaction," and requires believable behavior to establish appropriate social expectations [33]. Thus, motion control for social robots must address this overall problem of how to generate human-like motion for an anthropomorphic robot.

Interaction and collaboration with human partners necessitates the goal of anthropomorphic social robots to communicate with humans as humans communicate with each other. Thus, robot motion must communicate desired messages to human partners without confusion or misinterpretation. For human partners, these goals are measured with recognition and accurate labeling of the motion performed or the information communicated. The challenge is to synthesize human-like motion, preferably with minimal input information, and then success of the synthesis procedure must be measured. A metric for human-like motion can accomplish both of these goals.

Human perception can be used as the metric for quality in robot motion, which means that motion with errors beyond the range of sensitivity of the human eye are effectively equivalent to the human eye [79, 80]. Studies that quantify this range are valuable as both synthesis and measurement tools. However, the primary problem with this type of metric is dependency on human input to judge acceptable ranges. Since these metrics depend upon quantifying the measurement device (i.e. human perception), the metric may not be extensible to all motions without new user studies that exhaustively test new motions.

Classifiers have been used to distinguish between natural and unnatural movement based on human-labeled data. If a Gaussian mixture model, hidden-Markov model, switching linear dynamical system, naive Bayesian, or other statistical model can represent a database of motions, then by training one such model based on good motion-capture data and another based on edited or noise-corrupted motion capture data, the better predictive model for testing would have a higher log likelihood of matching the test data under the model. This approach is inspired by the theory that humans are good classifiers of motion because they have witnessed a lot of motion. However, data dependence is the problem,

and retraining is necessary when significantly different exemplars are added [81].

To my knowledge, there is no widely accepted metric for human-like motion in the fields of robotics, computer animation, or biomechanics. By far, the most common validation efforts rely upon subjective observation and are not quantitative. For example, the ground truth estimates produced by computer vision algorithms are evaluated and validated widely based on qualitative assessment and visual inspection [82, 83, 84]. Other forms of validation include projection of motion onto a 2-D or 3-D virtual character to see if the movements *seem* human-like [83].

Just as with prior attempts at evaluation, problems exist with prior attempts at the synthesis of human-like motion for robots. Dependence on large quantities of data is often an empirical substitute for a more principled approach. Whether building a model of motion from a large number of exemplars [85, 86, 87], or relying upon a database to bias the solution toward realism, data and databases become the bottleneck for online planning, which can affect algorithm runtime and reduce practical usefulness [88].

Existing anthropomorphic robot motion often extrapolates characteristics derived from human motion to constrain robot actuators to create motion that appears more human-like [89, 90, 91]. This includes optimal criteria such as joint comfort, movement time, jerk [92, 93]; and human pose-to-target relationships [94, 95]. When motion capture data is used, often timing is neglected, causing robot motion to occur at unrealistic and non-human velocities [96, 90]. These models of natural systems are often only applicable for specific motions or body parts, and they fail to capture any interaction between degrees-of-freedom on an articulated body.

However, the large majority of robot motion algorithms just ignore the concept of naturalness in motion, since they are concerned with more environment-based objectives such as grasp orientations, manipulation without dropping, planning obstacle-free trajectories, etc. Thus, my metric-based approach can augment these base trajectories to increase their utility for HRI, so they accomplish environmental goals (Chapter 5) and appear human-like.

4.1.2 Insight

Human motion is characterized by *interdependent* degrees-of-freedom [49]. This implies that the underlying movement data has a level of coordination—the phenomenon that motions of connected bodies or joints are related in terms of space and timing. Humans have muscles, which inherently cause coupling, influence, or coordination between DOFs. This basic insight is the inspiration for my metric. Since robots have motors (naturally uncoupled), emulation of human DOF coupling in robot motion can produce human-like motion.

Thus, in theory, if you increase the amount of spatial and temporal coordination for a given robot motion, it should become more human-like. Intuitively, coordination could be thought of as correlation or similarity in space and time of two trajectories, but correlation and similarity already have specific mathematical definitions. Bernstein theorizes that the amount of coordination in motion may only be limited by constraints (e.g. kinematic & dynamic limits, environmental, task) [97], which is important because in Chapter 5 I demonstrate how my metric works in conjunction with constrained motion.

Since physical architectures differ, due to the DOF correspondence problem (Section 2.4.1), information is lost and motion appears different when attempting to directly use human motion on a robot. Since coordination depends on the interaction between DOFs, coordination is part of the motion information lost when trying to apply human motion to robots. For example, translation of the human shoulder will not appear on the robot, if the robot does not have a translatable shoulder DOF. Serial, one-dimensional motors on a humanoid robot cannot create the same motion as integrated, coupled DOFs, such as ball-and-socket joint motions, in real humans. So until technology invents a single motor that provides control of 3-DOFs simultaneously that also has specifications appropriate to fit on a humanoid robot, an alternative solution is required to achieve human-like motion.

In Sections 2.4 and 4.1.4.3, a process called retargeting is described, which allows human motion to be applied to a robot. However, even the best retargeting algorithms still lose data in the transformation process. If sufficient interdependence between DOFs

survives the retargeting process, my experiments show that optimizing the remaining trajectories with respect to my metric discussed in Section 4.1.4 will re-coordinate the motion given the constraints of the new, projected, robot kinematic chain (i.e. as much as possible given differences in joint angle limits and DOF locations).

4.1.3 Goal

In Section 4.1.5, concrete hypotheses for the human-like motion experiments will be defined, but in general, since I hypothesize that anthropomorphic robot motion will appear more human-like if coordinated spatially and temporally, my algorithm synthesizes a coordinated version from a given input motion. Therefore, the goals are to:

- add evidence to the hypothesis that spatiotemporal correspondence of distributed actuators (i.e. motor coordination) is a component of human-like motion
- demonstrate one method of generating human-like motion from a single exemplar
- present a metric useful for the synthesis and evaluation of human-like motion
- validate the results of the method using a user study based on mimicking
- show that humans can more accurately identify, recognize, and mimic STC-optimized motion than the retargeted version without modification
- present data that suggests that coordinating robot motion with respect to the given metric makes the motion more human-like
- discuss empirical data on potential reasons why coordinating motion increases recognition and human ability to mimic

4.1.4 Algorithm

The spatiotemporal correspondence problem has already been heavily studied and analyzed mathematically for a pair of trajectory sets, where there is a one-to-one correspondence between trajectories in each set (e.g. two human bodies, both of which have completely defined and completely identical kinematic hierarchies and dynamic properties)

[98, 50]. Given two trajectories $x(t)$ and $y(t)$, correspondence entails determining the combination of sets of spatial ($a(t)$) and temporal ($b(t)$) shifts that map two trajectories onto each other. In the absence of constraints, the temporal and spatial shifts satisfy the equations in 36, where reference trajectory $x(t)$ is being mapped onto $y(t)$.

$$\begin{aligned} y(t) &= x(t') + a(t) \\ t' &= t + b(t) \end{aligned} \tag{36}$$

where,

$x(t)$ = first reference trajectory

$y(t)$ = second reference or output trajectory

$a(t)$ = set of time-dependent spatial shifts

$b(t)$ = set of time-dependent temporal shifts

t = time

t' = temporally shifted time variable

The correspondence problem is ill-posed, meaning that the set of spatial and temporal shifts is not unique. Therefore, a metric or cost function is often used to define a unique set of shifts.

Spatial-only metrics, which constitute the majority of "distance" metrics, are insufficient when data includes spatial and temporal relationships. Spatiotemporal-isomap is a common algorithm that takes advantage of STC in data to reduce dimensionality. However, the geodesic distance-based algorithm at the core of ST-Isomap was not selected as the candidate metric in my work due to manual tuning of thresholds and operator input required to cleanly establish correspondence [53]. Another critical requirement for a human-like motion metric is nonlinearity, since human motion data is nonlinear.

Prokopenko et. al. used K_2 as a metric for STC for modular reconfigurable robots to identify optimal configurations that overcome environmental constraints on motion [51, 52]. I take advantage of their temporally extended version of the K_2 metric, which

is nonlinear and an upper bound of Kolmogorov-Sinai entropy (Section 2.3). In short, KSE describes rate of system state information loss as a function of time. Therefore, a lower value of K_2 is more optimal, since it indicates higher retention of system state information over time.

4.1.4.1 STC as a Synthesis Tool

In order to use K_2 to synthesize coordinated motion, two things are required: an optimization algorithm that can optimize with respect to a cost (or objective) function and one input motion to use as the basis (or reference) trajectory for the optimization. The K_2 metric presented in Equation 38 becomes the cost function used in the optimization, with a K_2 value of zero being the target for the optimal solution.

Optimal control and dynamic time warping are 2 examples of well-known approaches that can yield a solution for the optimal set of spatial and temporal shifts that solve the correspondence problem with respect to a cost or objective function, given the reference trajectory [99]. For example, if these were selected to generate optimized (i.e. coordinated) motion, Equation 38 would be used as the cost function for either optimal control or DTW. Dynamic time warping is solved via dynamic programming [99].

During development, motion was optimized using both these algorithms. Although nonlinear optimal control formulations can be difficult to solve, the optimal control solution is known if the cost function is quadratic [49]. Given the constraints of robot actuators (e.g. finite spatial extent) and the constraints of the input motion (e.g. finite temporal extent), the required convexity for an optimal control solution is handled by squaring the individual spatial and temporal terms on the right-hand side of my metric presented in Equation 38. The new squared version still satisfies K_2 as an upper bound of KSE. However, the squared version is a less strict bound, which means that the minimum value of Equation 38 is higher for optimal control due to the constraints of optimal control motion synthesis. In other words, an optimal control problem formulated by squaring individual K_2 terms has less potential to optimize coordination.

After implementation of both algorithms, DTW demonstrated clear advantages over

optimal control. Due to the formulation of the optimal control problem, DTW produced motion that was more coordinated than the motion produced with optimal control. As a result, DTW was selected over optimal control for synthesis of a trajectory with respect to the human-like metric (subject to constraints).

The K_2 metric presented in Equation 38 constrains the amount of warping in its three parameters: r, S, and T. The value r can be thought of as a resolution or similarity threshold. Every spatial or temporal pair below this threshold would be considered equivalent and everything above it, non-equivalent and subject to warping. I empirically determined a 0.1 N.m. threshold for r on the Simon robot hardware.

Since my study, which is described in Section 4.1.6, used predefined motions, temporal extent, T, varied based on the sequence length for a given motion.

$$C_{(d_s,d_t)}(S,T,r) = \frac{\sum_{l=1}^{T}\sum_{j=1}^{T}\sum_{g=1}^{S}\sum_{h=1}^{S}\Theta(r-||V_l^g - V_j^h||)}{(T-1)T(S-1)S} \tag{37}$$

$$K_2(S,T,r) = ln(\frac{C_{d_s,d_t}(S,T,r)}{C_{d_s,d_t+1}(S,T,r)}) + ln(\frac{C_{d_s,d_t}(S,T,r)}{C_{d_s+1,d_t}(S,T,r)}) \tag{38}$$

where,

$\Theta(...)$ = Heaviside step function

V_i^k = $[w_i^k,...,w_i^{k+d_s-1}]$, spatiotemporal delay vectors

w_i^k = $[v_i^k,...,v_{i+d_t-1}^k]$, time delay vectors

v_i^k = element of time series trajectory for actuator k at time index i

d_s = spatial embedding dimension

d_t = temporal embedding dimension

S = number of actuators

T = number of motion time samples

r = correspondence threshold

Originally, the algorithm warped with a spatial extent, S, equal to all actuators on the robot, which means that the robot motion would be coordinated to all other DOF

trajectories in the original motion. However, preliminary results showed that this created motion which appeared too fluid, such as hair or a flag flapping in the wind. The algorithm was modified because only DOFs coupled by muscles should influence each other. To emulate the local coupling exhibited in human DOFs (e.g. ball-and-socket joints) on an anthropomorphic robot, which typically has serial DOFs, the spatial parameter, S, was set at a value that optimizes only based upon parent and children degrees-of-freedom, in the hierarchical anthropomorphic chain.

The optimization begins at the "root" DOF (typically a rotationally motionless DOF, like the pelvis), and it extends outward toward the fingertips. For Simon, this "root" DOF represents the rigid mount to the base. In other robots, the DOF nearest to the center-of-gravity is a logical selection for beginning of the optimization, which can extend outward along each separate DOF chain.

In this context, the term "spatial" warping is synonymous with torque magnitude, since the given reference trajectories are torques for each DOF as a function of time.

4.1.4.2 STC as an Evaluation Metric

In order to use STC as a mechanism to evaluate motion quality with respect to human-likeness, the spatial and temporal correspondence numbers on the right-hand side of Equation 38 are evaluated for a trajectory. Then, the procedure outlined under Section 4.1.4.1 is used to optimize that trajectory with respect to spatial and temporal correspondence. Since coordinated motion is more human-like, the difference between the optimal and original spatial and temporal numbers from Equation 38 indicate "human-likeness" of the original trajectory. If the difference is small, the original trajectory is closer to being human-like.

The advantage of this technique is that the values for spatial and temporal correspondence are evaluated in separate domains (i.e. robot and human DOFs may be different), and yet the metric provides results that are comparable.

4.1.4.3 STC Application

Any motion can be spatially and temporally coordinated using the algorithm described above, but in my evaluation I use trajectories collected from human motion-capture equipment that are subsequently retargeted to the Simon robot. This retargeted motion is optimized with respect to STC as described in Section 4.1.4.1. My hypothesis, which will be tested in Section 4.1.6, is that this makes the motion more human-like on the robot hardware.

The motion-capture trajectories include position data from the 28 upper-body markers on the suit. During the process of retargeting, a similar set of constraints or handles, which are positioned on the target kinematic hierarchy (i.e. the robot model), serve to sufficiently constrain the problem so the mapping between human motion capture markers and robot markers is one-to-one. An optimization problem is solved, iterating over the human motion capture data as a function of time, aligning human and robot markers. The optimal mapping allows for scaling the overall size of Simon based on human participant's size, given that the proportions of Simon's parts remain constant with respect to each other. This ensures maximum amount of information preservation over the retargeting process. Upon termination of the optimization, a set of 28 time-varying point constraints exist on the robot body that align optimally with the human constraint markers from the motion-capture data. The time-varying point constraints on the robot create a motion trajectory, in Simon joint angles, that can be executed on the robot [100].

As mentioned previously, the resultant projection between robot and human coordinate frames (i.e. architectures, kinematics) using retargeting is only scale invariant, and the scale factor must be equivalent in all Euclidean dimensions to preserve invariance. Since motion capture retargeting works by sufficiently constraining in Euclidean space through the collection of adequate number of markers on a human body, data is lost when the transformation between the target and human architectures is anything other than a scale factor. Retargeting algorithms do not solve the DOF correspondence problem. For example, projection of human translation at the shoulder will not survive a retargeting

process, if the robot does not have a translatable shoulder DOF.

4.1.5 Hypotheses

In Experiment 1, more hypotheses were tested than the fundamental hypothesis that I wanted to investigate with Experiment 1 (i.e. hypothesis H3). This was done for completeness and to rule out competing hypotheses. These hypotheses are explained in greater detail later and will become clearer in Section 4.1.6.2.2, after variables and data are introduced. But for now, they are explained in general terms.

- H1: Human motion is not independent of constraint. All human motion capture data comes from the same distribution.
- H2: Mimicked motion is not independent of constraint. All mimicked (i.e. constrained) data comes from the same distribution.
- H3: Coordinated motion is indistinguishable from human motion in terms of spatial and temporal coordination.
- H4: Optimizing human-like motion with respect to spatial and temporal coordination does not significantly affect the trajectories when the target model of the motion capture projection is nearly identical to the human from which the motion data was collected.
- H5: Optimizing downsampled human-like motion with respect to spatial and temporal coordination will recover information lost in the downsampling process when the target model of the motion capture projection is nearly identical to the human from which the motion data was collected. As the sampling rate increases (i.e. more information is lost), less information will be recovered by the SC and TC metrics.

Hypotheses H1-H3 are analyzed in Experiment 1. H4 and H5 are investigated in Experiments 2 and 3 respectively.

4.1.6 Experiment 1: Mimicking

The purpose of Experiment 1 was to quantitatively support that spatiotemporal correspondence of distributed actuators is a good metric for human-like motion. Since human motion exhibits spatial and temporal correspondence, robot motion that is more coordinated with respect to space and timing should be more human-like. Thus, I hypothesize that STC is a metric for human-like motion.

In order to test this hypothesis, a quantitative way to measure human-likeness is required. Distance measures between human and robot motion variables (e.g. torques, joint angles, joint velocities) in joint space cannot be used without retargeting (i.e. a domain change) due to the DOF correspondence problem. Thus, I designed an experiment based on mimicking. In short, people mimicked robot motions created by different motion synthesis techniques and the type of motion that humans were able to mimic the "best" (to be defined later) was the technique that generates the most human-like motion. Thus, this experiment relied on the idea that a human-like motion should be easier for people to mimic accurately. And, since there is a documented effect of practice on coordination (i.e. practice makes perfect) [101], only people's first mimicking attempt was valid data. I theorized that awkward, less natural motions would be harder to mimic, and participants' motions would be less coordinated with the motions from the technique that produced more awkward results.

4.1.6.1 Experimental Design

Differences in mimicking performances across the three experimental groups of stimulus motion were examined. The following are the groups of motion that participants in the experiment watched and attempted to mimic:

- OH: Twenty motions were captured by a single male. This data is denoted as "original human".
- OR: The human motions (i.e. OH set) were retargeted to the Simon hardware using the position constraint process described in Section 4.1.4.3. This is the "original

retargeted" data set.

- OC: The retargeted motions (i.e. OR set) were then coordinated using the algorithm and metric described in Section 4.1.4. This set of motions is called "original coordinated."

Before the experiment it was necessary to create the above groups of motions. Twenty motions were used in the experiment including common social robot gestures with and without constraints such as waving and object-moving, but also nonsense motions like "air-guitar." The full set of the motions used is: shrug, one-hand bow, two-hand bow, scan the distance, worship, presentation, air-guitar, shucks, bird, stick 'em up, cradle, take cover, throw, clap, look around, wave, beckon, move object, sneeze, and call/yell. The latter four were constrained with objects for gesture directionality or manipulation, such as a box placed in a certain location to move or wave toward. The air-guitar motion was unconstrained because when humans perform an air-guitar motion, they do not have a guitar in their hands.

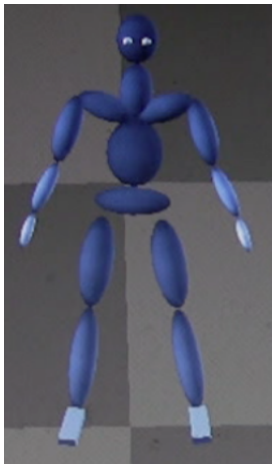

Figure 27: Virtual human model used in human-like Experiment 1.

In the experiment, participants were shown a video of a motion and then asked to mimic it. The OR and OC motion trajectories were videotaped on the Simon hardware from multiple angles for the study. Similarly, the original human motion was visualized on

a simplified virtual human character (Figure 27) and also recorded from multiple angles. Each video of the recorded motion contained all recorded angles shown serially. There were 60 input (i.e. stimulus) videos total (20 motions of 3 groups).

Forty-one participants (17 women and 24 men), ranging in ages from 20-26 were recruited for the study. Each participant saw a set of twelve motions from the possible set of twenty that were randomly selected for each participant in such a way that each participant received four OH, four OR, and four OC motions each. This provided a set of 492 mimicked motions total (i.e. 164 motions from each of three groups, with 8-9 mimicked examples for each of 20 motions).

4.1.6.1.1 Part One - Motion Capture Data Collection

Each participant was equipped with a motion capture suit and told to observe videos projected onto the wall in the motion capture lab. They were instructed to observe each motion as long as necessary (without moving) until they thought they could mimic it exactly. The videos looped on the screen showing each motion from different view angles so participants could view each DOF with clarity. Unbeknownst to them, the number of views before mimicking (NVBM) was recorded as a measure for the study. When the participant indicated they could mimic the motion exactly, the video was turned off and the motion capture equipment was turned on, and they mimicked one motion. Since there is a documented effect of practice on coordination [101], they were not allowed to move while watching and only their initial performance was captured. This process was repeated for twelve motions. Prior to the twelve motions, each participant was allowed an initial motion for practice and to get familiar with the experimental procedure. Only when a participant was grossly off with respect to timing or some other anomaly occurred, were suggestions made about their performance before continuing. This happened in only two cases.

Constraints accompanied some motions, such as objects or eye gaze direction. These constraints were given to the participants when necessary to facilitate ability to mimic accurately. For example, a cardboard box and two stools were given for the "move object"

motion, so the participant could move the box from one stool to the other. For all participants, the constraint locations and the standing position of the participant were identical. When constraints were given, they were given in all cases (i.e. OH, OR, and OC).

After mimicking each motion, the participant was asked if they recognized the motion, and if so, what name they would give it (e.g. wave, beckon). Participants did *not* select motion names from a list. After mimicking all twelve motions, the participant was told the original intent (i.e. name) for all 12 motions in their set. They were then asked to perform each motion unconstrained, as they would normally perform it. This data was recorded with the motion capture equipment, and in my analysis it is labeled the "participant unconstrained" (PU) set.

While the participants removed the motion capture suit, they were asked which motions were easiest and hardest to mimic; which motions were easiest and hardest to recognize; and which motion they thought that they had mimicked best (TMB). They were asked to give their reasoning behind all of these choices.

Thus, at the conclusion of part one of the experiment, the following data had been collected for each participant:

- Motion capture data from 12 mimicked motions:
  - 4 "mimicking human" (MH) motions.
  - 4 "mimicking retargeted" (MR) motions.
  - 4 "mimicking coordinated" (MC) motions.
- Number of views before mimicking for each of the 12 motions above.
- Recognition (yes/no) for each of the 12 motions.
- For all recognizable motions, a name for that motion.
- Motion capture data from 12 "participant unconstrained" (PU) performances of the 12 motions above.
- Participant's selection of:
  - Easiest motion to mimic, and why.

  - Hardest motion to mimic, and why.
  - Easiest motion to recognize, and why.
  - Hardest motion to recognize, and why.
  - Which motion they thought that they mimicked the best, and why.

4.1.6.1.2 Part Two - Video Comparison

After finishing part one, participants watched pairs of videos for all twelve motions that they had just mimicked. Each participant watched the OR and OC versions of the robot motion serially, but projected in different spatial locations on the screen to facilitate mental distinction. The order of the two versions was randomized. The videos were shown once each and the participants were asked if they perceived a difference. Single viewing was chosen because it leads to a stronger claim if difference can be noted after only one comparison viewing. Then, the videos were allowed to loop serially and the participants were asked to watch the two videos and tell which they thought looked "better" and which they thought looked more natural. The participants were also asked to give reasons for their choices. Unbeknownst to them, the number of views of each version before deciding "better" and more natural was also collected. Video order for all motions and motion pairs was randomized.

Thus, at the conclusion of part two of the experiment, the following data had been collected for each participant:

- Recognized a difference between OR and OC motion after one viewing (yes/no); for each of 12 motions mimicked in part one (Section 4.1.6.1.1)
- For motions where a difference was acknowledged,
  - Selection of whether OR or OC is "better"
  - Selection of whether OR or OC is more natural
- Rationale for "better" and more natural selections
- Number of views before better and more natural decisions

4.1.6.2 Results

4.1.6.2.1 Increases Recognition

The results from the study allowed me to conclude that STC-optimized motion makes robot motion easier to recognize. The data in Table 28 represents the percentage of participants who named a motion correctly and incorrectly, as well as those who opted not to try to identify the motion (i.e. not recognized). This data is accumulated over all 20 motions and sorted according to the three categories of stimulus video: OH, OR, and OC. Coordinated robot motion was correctly recognized 87.2% of the time, and was mistakenly named only 9.1% of the time. These are better results than either human or retargeted motion. Additionally, coordinating motion lead human observers to try to identify motions more frequently than human or retargeted motion (i.e. not recognized = 3.7% for OC). This suggests that coordinating motion makes the motion more familiar or common.

Table 28: Percent of motion recognized correctly, incorrectly, and not recognized by participants.

	Human (OH)	Retarg (OR)	Coord (OC)
% correct	72.1	46.6	87.2
% incorrect	19.4	42.3	9.1
% not recog.	8.5	11.0	3.7

In Table 28, on a motion-by-motion basis, percent correct was highest for 16 of 20 coordinated motions and lowest for 17 of 20 retargeted motions. In 17 of 20 motions percent incorrect was lowest for coordinated motions, and in a different set of 17 of 20 possible motions, percent incorrect was highest for retargeted motion. These numbers support the aggregate data presented in Table 28 suggesting that naming accuracy, in general, is higher for coordinated motion, and lower for retargeted motion. Comparing only coordinated and retargeted motion, percent correct was highest for 19 of 20 possible motions, and in a different set of 19 of 20, percent incorrect was highest for retargeted motion. This data implies that relationships for recognition comparing retargeted and

coordinated robot motion are maintained, in general, regardless of the particular motion performed. For reference, overall recognition of a particular motion (aggregate percentage) is a function of the motion performed. For example, waving was correctly recognized 91.7% of the all occurrences (OH, OR, and OC), whereas imitating a bird was correctly recognized overall only 40.2% of the time.

The subjective data also supports the conclusion that coordinated motion is easier to recognize. Participants were asked which of the 12 motions that they mimicked was the easiest and hardest to recognize. Table 29 shows the percentage of participants that chose an OH, OR, or OC motion, indicating that 75.3% of participants chose a coordinated motion as the easiest motion to recognize, while only 10.2% chose a coordinated motion as the hardest motion to recognize. A significant majority of participants (78.3%) selected a retargeted motion as the hardest motion to recognize.

Table 29: Percent of responses selecting types of motions as easiest and hardest motion to recognize.

	Human (OH)	Retarg (OR)	Coord (OC)
Easiest	14.8	9.9	75.3
Hardest	11.5	78.3	10.2

When asked, participants claimed that coordinated motion was easiest to recognize because it looked better, more natural, and was a more complete and detailed motion. On the other hand, retargeted motion was hardest because it looked "artificial" or "strange" to participants.

The majority of participants agreed that coordinated motion is "better" and more natural. In 98.98% of the trials, participants recognized a difference between retargeted and coordinated motion after only one viewing. When difference was noted, 56.1% claimed that coordinated motion looked more natural (27.1% chose retargeted), and 57.9% said that coordinated motion looked "better" (compared with 25.3% for retargeted). In the remaining 16.8%, participants (unsolicited) said that "better" or more natural depends on

context, and therefore they abstained from making a selection. Participants who selected coordinated motion indicated they did so because it was a "more detailed" or "more complete" motion, closer to their "expectation" of human motion.

Statistical significance tests for the results in Tables 28 and 29 were not performed due to the nature of the data. Each number is an accumulation expressed as a percentage. The data is not forced choice; all participants were trying to correctly recognize the motion; some attempted and failed, and some did not attempt because they could not recognize the motion.

4.1.6.2.2 Makes Motion Human-like

Four sets of motion-capture data exist from Experiment 1 part one (Section 4.1.6.1.1): mimicking human (MH), mimicking retargeted (MR), mimicking coordinated (MC), and participant unconstrained (PU) motion. Analysis must occur on a motion-by-motion basis. Thus, for each of the 20 motions, there is a distribution of data that captures how well participants mimicked each motion. For each participant, I calculated the spatial and temporal correspondence according to Equation 38, which resolved each motion into two numbers, one for each term on the right-hand side of the equation. For each motion, 8-9 participants mimicked OH, OR, and OC. Three times more data exists for the unconstrained version because regardless which constrained version a participant mimicked, they were still asked to perform the motion unconstrained. Thus for the analysis, I resolved MH, MR, MC, and PU into distributions for SC and TC across all participants. There are separate distributions for each of the 20 motions, yielding 4 x 2 x 20 unique distributions. The goal was to analyze each of the SC and TC results independently on a motion-by-motion basis, in order to draw conclusions about MH, MR, MC, and PU. I used ANOVAs to test the following hypotheses:

- **H1:** Human motion is not independent of constraint. In other words, all the human motion capture data sets, (MH, MR, MC, and PU) do not have significantly different distributions. The F values, for all twenty motions, ranged from 7.2-10.8 (spatial) and

6.9-7.6 (temporal) which is greater than F_{crit} = 2.8. Therefore, I concluded at least one of these distributions is different from the others with respect to SC and TC.

- **H2:** Mimicked motion is not independent of constraint. In other words, all mimicked (i.e. constrained) data, (MH, MR, and MC) do not have significantly different distributions. In these ANOVA tests, values for all 20 motions ranged between 6.1-8.6 (spatial) and 5.3-6.6 (temporal), which are greater than F_{crit} = 3.4-3.5. Therefore, I concluded that at least one of these distributions for mimicked motion is statistically different.

- **H3:** Coordinated motion is indistinguishable from human motion in terms of spatial and temporal coordination. MH, MC, and PU sets do not have significantly different distributions. $F_{observed}$ of 0.6-1.1 (spatial) and 0.9-1.9 (temporal), which are less than F_{crit} of 3.2-3.3, meaning that with this data there was insufficient evidence to reject this hypothesis for all twenty motions.

Since the above analysis isolated that retargeted motion is different from the other spatial and temporal correspondence distributions in mimicked motion, at this point, pairwise t-tests were performed to determine the difference between data sets on a motion-by-motion basis. Table 30 shows the number of motions for which there is a statically significant difference in spatial correspondence (the table for temporal correspondence is identical but not shown). For example, when participants mimicked retargeted motion, twenty motions were statistically different than the original retargeted performance. However, for the data when participants mimicked human or coordinated motion, the distributions failed to be different from their original performance for both spatial and temporal coordination (H3). From this, I concluded that humans are not able to mimic retargeted motion as well as the coordinated or human motion.

Since the above statistical tests do not allow me to conclude that the distributions were identical (H3), I performed a regression analysis of the data across all twenty motions to determine how correlated any two variables are in the study. For the purpose of this regression analysis the variables are either the mean or the standard deviation of SC, TC,

Table 30: Number of motions with $p<0.05$ for pairwise spatial correspondence comparison t-tests for the indicated study variables. Note: Table is identical for temporal correspondence.

	OH	OR	OC	MH	MR	MC	PU
OH	X	20	0	0	20	0	0
OR	X	X	20	20	20	20	20
OC	X	X	X	0	20	0	0
MH	X	X	X	X	20	0	0
MR	X	X	X	X	X	20	20
MC	X	X	X	X	X	X	0
PU	X	X	X	X	X	X	X

or STC, for each of the distributions (OH, OR, OC, MH, MR, MC, PU). However, OH, OR, and OC are only one number (not a distribution) so they were not included in the standard deviation analysis. The intuition for this analysis is that if two variables are highly correlated with respect to both mean and variance, then it is further evidence that their distributions are similar. Specifically, results showing high correlation between the human and coordinated motions were expected.

The R^2 values from the linear data fits, are shown in Tables 31 and 32. This data shows that participants mimicking coordinated and human motion were highly correlated (line 14 in Table 31 and line 2 in Table 32, lightly shaded), whereas the data from when participants mimicked retargeted motion was less correlated to all other data including the original human performance (lines 2, 6, 10, 13, and 16 in Table 31). When two variables have high correlation in a linear data fit, it means that either variable would be a excellent linear predictor of the other variable in the pair. These higher correlations between human and coordinated motion are further evidence that coordinated motion is more human-like than retargeted motion.

Furthermore, the standard deviation correlation on line 3 in Table 32 is low for the spatial and temporal components when regressing mimicked human and participant unconstrained data, which shows that mimicking does in fact constrain people's motion. Variance increases for the PU distribution because motion is unconstrained and humans

Table 31: R^2 value from linear regression analysis on spatial (SC), temporal (TC) and composite mean correspondence (STC) for pairs of variables. R^2 = 1 (perfectly correlated); 0 = (uncorrelated). Note the high correlation between mimicked human and coordinated motions seen in row 14.

	Variables	SC	TC	STC
1	OH v. MH	0.9783	0.9756	0.9914
2	OH v. MR	0.6339	0.0427	0.5483
3	OH v. MC	0.9792	0.965	0.9933
4	OH v. PU	0.9859	0.9378	0.9843
5	OR v. MH	0.0103	0.0009	0.0022
6	OR v. MR	0.0915	0.008	0.0526
7	OR v. MC	0.0001	0.0002	0.0004
8	OR v. PU	0.0011	0.0003	0.0001
9	OC v. MH	0.9494	0.9626	0.9819
10	OC v. MR	0.6084	0.0491	0.5176
11	OC v. MC	0.9834	0.962	0.9918
12	OC v. PU	0.9836	0.9414	0.9795
13	MH v. MR	0.6412	0.0421	0.5612
14	MH v. MC	0.9531	0.9749	0.9809
15	MH v. PU	0.969	0.9271	0.9756
16	MR v. MC	0.6728	0.0516	0.5365
17	MR v. PU	0.6414	0.017	0.5076
18	MC v. PU	0.9881	0.9144	0.9822

Table 32: R^2 value from linear regression analysis on standard deviation of spatial, temporal and composite correspondence for pairs of study variables. R^2 = 1 (perfectly correlated); 0 = (uncorrelated). Variables not shown have a standard deviation of 0. Note the high correlation between mimicked human and coordinated motions seen in row 2.

	Variables	SC	TC	STC
1	MH v. MR	0.1005	0.1231	0.3507
2	MH v. MC	0.8847	0.7435	0.9842
3	MH v. PU	0.0674	0.0906	0.8348
4	MR v. MC	0.0746	0.1749	0.346
5	MR v. PU	0.5002	0.0002	0.2239
6	MC v. PU	0.0986	0.096	0.8537

Table 33: Average number of times participants watched each motion before deciding that they were prepared to mimic the motion.

Motion	Human (OH)	Retarg (OR)	Coord (OC)
Wave	2.8 (0.33)	3.9 (1.04)	3.5 (0.75)
Beckon	2.9 (0.24)	4.9 (1.32)	3.3 (0.95)
Shrug	1.2 (0.11)	2.2 (0.75)	2.0 (0.47)
Move Object	3.4 (0.94)	5.4 (1.54)	3.7 (0.83)
1-Hand Bow	2.1 (0.53)	3.9 (0.88)	2.2 (0.58)
2-Hand Bow	1.2 (0.22)	1.8 (0.70)	1.6 (0.39)
Scan	2.7 (0.61)	4.7 (1.11)	2.9 (0.85)
Look Around	3.1 (0.83)	3.6 (0.89)	3.2 (0.90)
Worship	4.4 (0.91)	6.0 (2.03)	5.1 (1.07)
Presentation	1.5 (0.33)	3.1 (0.89)	1.4 (0.65)
Air Guitar	1.7 (0.77)	2.5 (0.86)	1.5 (0.63)
Shucks	1.1 (0.48)	1.8 (0.78)	1.2 (0.44)
Bird	4.2 (0.98)	5.5 (1.57)	4.4 (0.94)
Stick 'em Up	2.9 (0.23)	4.0 (1.01)	3.3 (0.90)
Cradle	2.7 (0.54)	4.8 (1.42)	3.0 (1.10)
Call/Yell	2.4 (0.48)	3.1 (0.91)	2.7 (0.64)
Sneeze	1.7 (0.47)	2.9 (1.21)	1.7 (0.72)
Cover	1.5 (0.33)	2.3 (0.95)	1.7 (0.78)
Throw	3.8 (0.89)	5.3 (1.22)	4.0 (0.66)
Clap	1.4 (0.25)	2.5 (1.01)	1.8 (0.67)
Average	2.43 (1.02)	3.71 (1.32)	2.71 (1.10)

are free to perform the motion as they please. This validates my premise in this study that mimicking performance is a method by which to compare motion.

The data shown in Table 33, which was collected for number of views before mimicking, also supports the claim that coordinated motion is more human-like. On average, humans viewed a retargeted motion more times before they are able to mimic (3.7 times) as compared to coordinated motion (2.7 times) or human motion (2.4 times). Pairwise t-tests between these distributions, on a motion-by-motion basis for NVBM, showed that 19 of 20 retargeted motions exhibited statistical significance ($p<0.05$) when compared with human NVBM whereas only 3 of 20 coordinated motions NVBM were statically different ($p<0.05$) from human NVBM. This suggests coordinated motion is more similar to human motion in terms of preparation for mimicking.

Table 34: Percent of responses selecting types of motions as easiest and hardest motion to mimic.

	Human (OH)	Retarg (OR)	Coord (OC)
Easiest	14.6	9.8	75.6
Hardest	31.7	56.1	12.2

Of the 12 mimicked motions, each participant was asked which motion was easiest and hardest to mimic. Of all participant responses, 75.6% of motions chosen as easiest were coordinated motions, and only 12.2% of participant responses chose a coordinated motion as hardest to mimic (Table 34). In my assertion stated earlier, I claimed that a human would be able to more easily mimic something common and familiar to them. These results suggest that coordination adds this quality to robot motion, which improves not only ability to mimic, as presented earlier, but also *perception* of difficulty in mimicking (Table 34).

During questioning of participants in post-experiment interviews, I gained insight into people's choices of easier and harder to mimic. Participants felt that human and coordinated motion were "more natural" or "more comfortable." Participants also indicated that human and coordinated motion were easier to mimic because the motion was "more familiar," "more common," and "more distinctive." In comparison, some people selected retargeted motion as being easier to mimic because fewer parts are moving in the motion. Others said retargeted motion is hardest to mimic because the motion felt "artificial" and "more unnatural."

4.1.6.2.3 SC and TC are better than STC

In Equation 38, the individual terms (spatial and temporal) on the right-hand side can be evaluated separately, rather than summing to form a composite STC. In my analysis, when the components were evaluated individually on a motion-by-motion basis, 20 of 20 retargeted motions exhibited statistical difference ($p<0.05$) from the human mimicked data and 0 of 20 coordinated motions exhibited correspondence that is not statistically

different ($p>0.05$) than human data distribution (Table 30). However, with the composite STC used as the metric, only 16 of 20 retargeted motions were statistically different than the original human performance ($p<0.05$). Since the results were slightly less strong when combining the terms and using composite STC as the metric rather than analyzing SC and TC individually, I recommend that the SC and TC individual components be used independently as a metric for human-likeness.

4.1.7 Experiment 2: Human Motion

The purpose of Experiment 2 was to further quantitatively support that spatiotemporal correspondence of distributed actuators is a good metric for human-like motion. Since human motion exhibits spatial and temporal correspondence, if SC and TC are good metrics for human motion, then optimizing human motion with respect to these metrics should not significantly change the spatial and temporal correspondence of human motion.

4.1.7.1 Experimental Design

Experiment 2 was designed to test hypothesis H4. In Experiment 1, I tested retargeting human motion onto two different models: a human model and a robot model (i.e. Simon). In Experiment 2, I focused more on the human model to add evidence that SC and TC are good metrics for human motion. I did not explain the details in Experiment 1, but the human model that was used for the OH, MH, and PU data sets was actually 42 different human models (i.e. one for each participant and one for the original human). The optimization for SC and TC in Equation 38 uses torque trajectories, and each human participant in my study was physically different (e.g. mass, height, strength). In order to get accurate dynamic models for my human data, when Experiment 1 was performed, I collected basic anthropometric data from my participants: height, weight, gender, and age. Using the model scaling functionality of the Software for Interactive Musculoskeletal Modeling (SIMM) Tool[1] and the dynamic parameters I collected, I was able to produce accurate dynamic models of all humans from which I collected motion capture data.

[1]Trademark of MusculoGraphics Inc. All rights reserved.

In Experiment 1, human subjects were visualizing the OH data on the simplified human model shown in Figure 27. Although the SIMM tool provides trajectories for 86 DOFs, in the motion capture data, markers were not placed to sufficiently capture all 86 DOFs because I focus on body motion in my research. Thus, the number of DOFs in the human model was also simplified. Degrees-of-freedom in the eyes, thumbs, toes, ears, and face of the human model were removed since motion capture markers were not placed to capture sufficient motion in these areas when I collected data. Additional DOFs were removed in the human model in locations such as the legs, since the majority of these degrees-of-freedom were not significantly animated in the 20 motions from Experiment 1. The final human model was comprised of 34 joints, concentrated in the neck, abdomen, and arms. When the motion capture data was retargeted to the human model, 45 markers were used as constraints to animate the 34 DOFs.

The human motion capture data was retargeted to this 34-DOF simplified human model optimally, after each model was scaled for dynamic parameters unique to each participant (thereby creating 42 human models). After this optimal retargeting, I had the OH and MH data sets. Since each participant in Experiment 1 mimicked 4 OH motions, I had distributions of MH data formed from 8-9 human performances per motion. For Experiment 2, I combined these OH and MH data sets to create a data set that I call pre-optimization (pre-op), which consists of 9-10 similar human examples for each of 20 different motions.

To test H4, I optimized each of these 184 trajectories (4 MH motions per participant x 41 participants + 20 OH motions) according to the procedure in Section 4.1.4. This provided a comparison dataset of 9-10 similar human examples for each of 20 different motions optimized according to the STC metric. This set is called post-optimization (post-op).

The SC and TC were evaluated for each of the trajectories in the pre-op and post-op data sets to create paired distributions of SC and TC for each motion. According to hypothesis H4, if SC and TC are good metrics for human-like motion, the optimization should not significantly affect them when the models used for retargeting and optimization are identical.

4.1.7.2 *Results*

For the initial analysis, on a motion-by-motion basis, twenty pre-op and twenty post-op distributions were combined to create twenty distributions (one for each motion). H4 states that SC and TC will not be affected by my human-like optimization. In other words, the SC and TC of the pre-op and post-op data come from the same distribution.

Twenty F-tests were performed, i.e. one for each combined motion distribution. The F values, for all twenty motions, ranged from 0.7-1.3 (spatial) and 0.8-1.4 (temporal) which were less than F_{crit} = 4.4-4.5. Therefore, the data for all 20 motions was not statistically different before and after the optimization with respect to SC and TC. This data does not allow me to prove H4 definitively, but it is a first step toward understanding the performance of my metric. Since I believe that my metric is valid, I can increase confidence that the pre-op and post-op data belong to the same distribution with a regression analysis.

Assuming a normal distribution, high correlation between the pre-op and post-op distributions' mean values and standard deviations would increase confidence in H4. Using the twenty independent data points (one for each motion), I formed a linear regression from the pre-op and post-op distributions' mean values and standard deviations. After performing these four linear regressions, the correlation coefficient of the means was 0.9874 (SC) and 0.9657 (TC), and the standard deviations resulted in a correlation coefficient of 0.9791 (SC) and 0.9580 (TC). Ideally, the correlation coefficient of both of these regressions would be 1.0, which would indicate that they are identical distributions. The corresponding slopes of these two lines were 0.9954 (mean, SC), 0.9879 (mean, TC), 0.9481 (standard deviation, SC), and 0.9823 (standard deviation, TC). The ideal slope is 1.0, and these results show that the optimization did not significantly affect SC or TC of human-like motion when the models used for retargeting and optimization were identical.

4.1.8 Experiment 3: Downsampling Human Motion

Experiment 3, which was designed to test H5, is an extension of Experiment 2. In Experiment 2, I provided evidence that the optimization does not significantly affect SC and TC for human-like motion when the model is identical for retargeting and optimization.

With Experiment 3, I wanted to test the strength of the optimization in "retrieving" lost information and data. In other words, after Experiment 2 I knew that the optimization would function as expected under ideal conditions (i.e. ideal model and ideal data), but in Experiment 3 I wanted to test the metric to see whether it would function as expected when used in non-ideal conditions (i.e. in the manner in which it was originally created).

Recall that KSE is a metric from chaos theory that estimates the information lost as a function of time for a stochastic signal. I used KSE to estimate information difference between two deterministic signals. Thus, in Experiment 3, I intentionally eliminated some information in the motion signal for an optimal model by downsampling the motion signal, and I tested how much of that information the optimization process was able to return for human motion.

4.1.8.1 Experimental Design

For Experiment 3, the same human model and same two data sets from Experiment 2 (pre-op and post-op) for all 20 motions were used. I created eight new datasets, each of which represents the pre-op data set uniformly decimated to remove information from the 184 trajectories. There were eight new data sets because the subsampling occurs at eight distinct rates: downsampling by rates in half integer intervals from 1.5 to 5.0 (i.e. 1.5, 2.0,...,5.0). These data sets are denoted by the label "pre-op" followed by the sampling rate (pre-opN, e.g. pre-op1.5 refers to the pre-op data set of trajectories with each trajectory decimated by a rate of 1.5). For reference, the original data set, pre-op, from Experiment 2 represents pre-op1.0.

Then, I optimized each of the 1,472 trajectories (184 x 8) according to the procedure in Section 4.1.4 to create an additional eight new datasets for each of 20 motions. This provided paired comparison datasets of 9-10 similar human examples of each of 20 different motions for each of 8 unique sampling rates optimized according to the STC metric. These eight data sets are denoted by the label "post-op" followed by the sampling rate, i.e. generically as post-opN, where N refers to the sample rate (e.g. post-op1.5 refers to the pre-op1.5 dataset after optimization).

For each trajectory in these 16 new datasets (2,944 trajectories), SC and TC were evaluated. According to hypothesis H5, if SC and TC are good metrics for human-like motion, the optimization should compensate for SC and TC lost in the downsampling process when the models used for retargeting and optimization are identical. Also, H5 states that SC and TC for post-opN trajectories with subsampling rates closer to 5.0 will be less similar to the pre-op (a.k.a. pre-op1.0) and post-op (a.k.a. post-op1.0) datasets for each motion.

4.1.8.2 Results

Since there are a large number of variables in the statistical significance tests (which result in many combinations of statistical tests), I will omit the details of the intermediate series of numerous tests that begin from the most broad hypothesis (i.e. that sampling rate has no effect on the pre-op1.0 trajectories; this means that the optimized, downsampled, and original data (pre-op1.0, post-op1.0, pre-op1.5, post-op1.5,..., pre-op5.0, post-op5.0) come from the same distribution). The intermediate set of tests led me to conclude the following:

1. Downsampled trajectories prior to optimization (pre-opN) were statistically significant ($p<0.05$) from pre-op1.0 and post-op1.0 for all sample rates greater than 1.0 (i.e. $N>1.0$) on a motion by motion basis for all 20 motions (with respect to both SC and TC). I concluded that decimation causes a motion trajectory to lose spatial and temporal information.

2. Downsampled trajectories prior to optimization (pre-opN) were statistically significant ($p<0.05$) with respect to downsampled trajectories after optimization (post-opN) for all evaluated sample rates greater than 1.0 (i.e. $N>1.0$) on a motion by motion basis for all 20 motions (with respect to both SC and TC). I concluded that my human-like optimization significantly changed the spatial and temporal information of downsampled trajectories when the downsample rate was greater than or equal to 1.5.

3. Downsampled trajectories prior to optimization (pre-opX) were statistically different ($p>0.05$) from each other (pre-opY) on a motion by motion basis for all 20 motions

(with respect to both SC and TC) for all sample rates where X≠Y. I concluded that downsampling at higher sample rates caused more spatial and temporal motion information to be lost.

The remainder of the tests are captured in Table 35. In Table 35 the data from human-like Experiment 2 (i.e. pre-op1.0 and post-op1.0) is combined into a single distribution and compared against the SC and TC data from downsampled datasets after optimization (i.e. post-opN, where N>1.0).

Table 35: Number of motions with $p<0.05$ for pairwise spatial and temporal correspondence comparison t-tests for the composite data set of pre-op & post-op (from human-like Experiment 2) vs. post-opN, where N = downsample rate.

N	Spatial	Temporal
1.5	0	0
2.0	0	0
2.5	0	0
3.0	7	5
3.5	13	11
4.0	17	18
4.5	19	19
5.0	19	20

Numbers closer to 20 in Table 35 indicate that more motions were unable to be recovered and restored with respect to SC or TC, as compared to the respective distributions without downsampling. Too much information was lost at higher sample rates, and the optimization was not able to compensate for the lost data, which provides evidence for the second half of H5. For lower values of downsampling rates, the optimization was able to perform well and recover lost information, but as downsampling rate increased less information was recoverable.

To increase confidence in distribution similarity for the data in Table 35, linear regressions on the distribution means and standard deviations (pre-op1.0 & post-op1.0 vs. post-opN) were performed. The slopes and correlation coefficients, as shown in Table 36, indicate that for motion distributions that were not statistically different, the pre-op1.0,

Table 36: Slopes and correlation coefficients (R^2) of mean (μ) and standard deviation (σ) for linear regressions of SC and TC of pre-op & post-op (from human-like Experiment 2) vs. post-opN, where N = downsample rate. Only data from non-statistically significant motions is included in regressions. Sample rates not shown did not include enough data to regress a non-trivial line.

	Spatial, SC				Temporal, TC			
N	μ, Slope	μ, R^2	σ, Slope	σ, R^2	μ, Slope	μ, R^2	σ, Slope	σ, R^2
1.5	0.983	0.995	0.942	0.954	0.969	0.975	0.980	0.971
2.0	0.976	0.957	0.941	0.958	0.951	0.942	0.947	0.943
2.5	0.945	0.965	0.972	0.970	0.995	0.947	0.994	0.951
3.0	0.967	0.955	0.953	0.975	0.982	0.967	0.969	0.973
3.5	0.950	0.961	0.996	0.957	0.963	0.972	0.987	0.972

post-op1.0, and post-opN distributions are similar (slopes close to 1.0 with high correlation coefficient). This means that for downsampled trajectories where sufficient information content survives the downsampling process, after optimization, the distributions of SC and TC of these trajectories (on a motion by motion basis) appeared similar to the distributions where there is no information loss (pre-op and post-op from Experiment 2, without downsampling data).

4.1.9 Discussion

During part one of Experiment 1, all participants were asked which motion they thought they mimicked the best and why. On average, 63.4% of responses selected a coordinated motion as the motion that they thought they mimicked the best (as compared with 26.8% that chose human and 9.8% that chose retargeted motion). Comparing the results of the participants' TMB motion selections to the actual data, I discovered that humans are not reliable predictors of their own ability to mimic. Using SC and TC as the metrics to gauge ability to mimic, I found that only 12 of 41 participants (29.3%) successfully labeled as TMB the motion (of the 12 motions given) with SC most similar between their own and the original performance; only 9 of 41 participants (22.0%) successfully labeled as TMB the motion (of the 12 motions shown) with TC most similar between their own and the original performance.

The rationale behind why humans chose a particular motion as the one they thought that they mimicked the best is very similar to the rational for easiest to mimic. This rationale favors comfort, naturalness, and familiarity for human and coordinated motion, but simplicity for retargeted motion.

The statistics for number of views before deciding "better" and "more natural" were not discussed above because in 9 of 20 motions the data was statistically significant ($p<0.05$) when comparing the distributions for coordinated and retargeted motion. Additionally, there was no consistent trend for one particular type of motion over the other (e.g. coordinated motion was always watched fewer times before people decide they like it "better"). Thus, this data does not add any conclusive value to my experiment. My hypothesis is that number of views before deciding "better" or "more natural" is motion dependent, rather than being a general characteristic of coordinated or retargeted motion.

Although for these experiments, the input motions came from motion-capture data, they do not need to come from motion capture or retargeting; it can come from any source (e.g. animation of any quality, dynamic equations, online planning algorithm). If retargeting is used as the mechanism to generate the trajectories used for optimization, more interaction between DOFs will survive the retargeting process if the robot is as similar as possible to the living entity whose motion is captured (e.g. human motion capture retargeting works better on humanoid robots, when the human is more physically similar to the robot).

4.1.10 Summary

If the goal of anthropomorphic robots is to communicate with humans as humans communicate with each other, then robot motion can be improved with my STC metric to create human-like motion. In addition to presenting spatial and temporal correspondence (i.e motor coordination) as an quantitative metric for human-like motion, I presented objective and subjective data to support my claims that motion optimized with respect to SC and TC is more human-like and easier to recognize, and therefore has benefits for human-robot interaction. This metric is useful as both a tool for human-like motion synthesis for

anthropomorphic robots and as a measurement instrument for the evaluation of human-like motion.

4.2 *Adding Variance*

4.2.1 Introduction

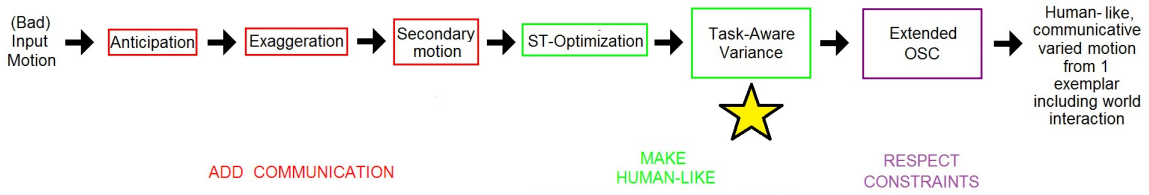

Figure 28: Section 4.2 discusses the variance algorithm.

The variance algorithm is the second of the two human-like motion algorithms discussed in this thesis, as shown in Figure 28. It follows the spatiotemporal optimization because this sequence requires fewer computations. Also the variance algorithm can easily be combined with the subsequent external constraint-preserving algorithm, which allows for greater algorithmic efficiency [102].

Human motion at any instance in time is the result of a huge number of variables (personality, age, sex, culture, environmental conditions, experiences, intents, beliefs, etc.) [103, 2, 31, 104]. This makes repetitive motion in humans highly unlikely. Several works find that human-like qualities in robots will facilitate a more effective interaction [6, 7, 29]. Thus, for a social robot, having the ability to easily produce variants of its exemplar motions or gestures, is important for generating human-like behavior.

Chapter 3 describes my contributions in adding communication to motion. However, communication tasks require meaningful motion variance so that the motions are not repetitive. Social robots, which interact with humans as a part of their functional goal, can benefit from motion variance because non-repetitive gestures and motions will be more natural and intuitive for the human partner. Sometimes, natural variance arises from interaction with the environment and motion preparatory action [105]. But, social

robots require a systematic method of variance generation that does not solely depend upon environmental and state conditions external to the robot.

Motion variance can be produced by dependence upon a database [106, 88]. Under these circumstances, one from a set of pre-annotated gestures is selected randomly when the system commands a specific motion. In most online applications, there is a trade-off between the richness (i.e. quality) of variations and the size of dataset.

Even when databases are large, variance is only possible when more than one motion exemplar for a specific motion type (e.g. wave, beckon) exists in the database. Ideally, an infinite number of variants of any motion type could be generated from one exemplar. The motions in the database come from either professional animators or motion capture data (e.g., [107]), both of which have significant lead-times to generate new exemplars. In general, unless environmental constraints, such as obstacles, induce variability, most robotic systems exhibit repetitive motion. For example, on one walking robot, variance is limited by joint angle constraints based on anthropometric data. This specific restriction is imposed for stability reasons [13].

The field of computer animation has explored creating variations in animation, for example, by building models of style and variance from human motion capture data for subsequent interpolation to generate new motions [108]. Or, given an input motion sequence, a simple way to generate variations is to introduce random noise in the joint trajectories [109, 110]. These methods can be computed efficiently in real-time applications, but the results highly depend on careful parameter tuning for the stochastic models. Other techniques that rely upon direct manipulation of trajectories are not applicable because they require manual intervention, which affects online application [111, 112, 113].

4.2.2 Insight

The simplest solution to generate variance is to inject random white noise into the joint angle trajectories for each degree-of-freedom. However, applying this technique to robots creates problems. First, the robot will appear to vibrate and shake, and these unsmooth trajectories may damage actuators. But I overcome this problem by applying noise in the

torque domain. The second problem is that purposeful motion is disrupted by random noise. This issue requires algorithms that induce variations in motion, while respecting the "task" the robot is performing.

Since all existing methods of variance generation exhibit problems undesirable for social humanoid robotic applications, I introduce a new approach for synthesizing variance in both the presence and absence of constraints using a stochastic process. This algorithm was originally published under the name task-aware variance (TAV), which combined the algorithms in Section 4.2 and Chapter 5, thereby showing how to generate variance and respect constraints simultaneously [102]. However, the part of the original TAV algorithm that I designed to respect constraints is useful in a more general sense for robot motion control. Thus, constrained motion synthesis is allotted the entirety of Chapter 5 so it can be discussed in a larger, more general context without redundancy. Furthermore, situations exist when variance is desired without constraints (or vice-versa), and so independent presentation is more useful.

My variance algorithm is based on insight from optimal control. The linearized version of the optimal control problem presented in Section 2.2 provides an optimal value function with time-varying Hessians as the solutions. The Hessian represents the curvature of the state space. The shape of the value function yields insight into the control policy's tolerance to perturbations. High curvature is large cost of deviation from tracking the trajectory optimally, and the eigenvector of the Hessian corresponding to the largest eigenvalue is a direction in state space that would cost the most in terms of tracking the reference trajectory. Since the eigenvalue-eigenvector pairs provide local information regarding cost and corresponding directions in the task space, I exploit this information to add noise in low-cost directions, in which injecting noise will minimally disrupt the task.

For the purposes of discussion in Section 4.2, for an algorithmic or mathematical standpoint, a task is defined as a cost function that measures motion execution (e.g. desirability of joint position, joint velocity and torque usage). In Chapter 5, this definition for "task" will be extended to include constraints. My algorithm adds biased torque to the optimal control torques according to a defined cost function. The resultant torque preserves the

characteristics of the input motion encoded in the cost function, but stochastically produces motion that is visually different from the input.

I also define a "task" from a functional perspective for the purposes of discussion in the same section (i.e. Section 4.2), as the intent or action of the original input motion that is fed into my algorithm. Thus, all trajectory modulation (i.e. variance) must be "task-aware" and maintain the original intent or motion type. Any variants produced by my algorithm must still be perceived as and classified as the same gesture of the input motion.

4.2.3 Goal

In Section 4.2.5, concrete hypotheses for the variance experiments will be defined, but in general, my algorithm synthesizes an infinite number of variants from a given input motion. Therefore, the goals are to:

- generate an infinite number of motion variants from one input motion in real-time that resemble the kinematic and dynamic characteristics of the input motion sequence
- synthesize variance in the presence of full posture constraints (e.g. free gestures)
- introduce a stochastic method to generate smooth but nondeterministic transitions between arbitrary motion variants
- evaluate the human-likeness quality of task-aware variants relative to other algorithms that can induce variance into motion

4.2.4 Algorithm

Given a sequence of poses (i.e. time sequence of joint angles), q_t, $0 \leq t \leq N$ that constitute a motion, a reference state trajectory $\bar{x}$, and its corresponding reference control trajectory $\bar{u}$ are constructed. The state trajectory consists of both q_t and $\dot{q}_t$, while the control trajectory $\bar{u}$ consists of joint torques computed from an inverse dynamics process. The core of the algorithm computes a time-varying multivariate Gaussian $\mathcal{N}(0, \Sigma_t)$ and a feedback control policy, $\Delta u_t = -K_t \Delta x_t$. The time-varying covariance of the Gaussian, Σ_t, is shaped using

a special variable (derived below), which results in the creation of task-aware variance. The characteristics of the input motion are represented by the Gaussian and the feedback policy (described in Section 4.2.4).

Variation to a given input motion is generated online by applying the following operations at each time step.

1. Draw a random Gaussian sample from a shaped distribution: $\Delta x_t \sim \mathcal{N}(0, \Sigma_t)$; $\Sigma_t = S_t^{-1}$
2. Compute the corresponding control force via the feedback control policy: $\Delta u_t = -K_t \Delta x_t$
3. Apply $\bar{u}_t + \Delta u_t$ as the current control force

Given an input reference state and control trajectory, $\bar{x}_t$ and $\bar{u}_t$, respectively, $t = 1, ..., N$, an optimization is formulated to track the reference trajectory. $\bar{x}_t \in \Re^{2n}$ contains the joint angles and velocity at frame t while $\bar{u}_t \in \Re^m$ contains the joint torques. The task can be viewed as minimizing the state and control deviation from the reference trajectory, subject to discrete-time dynamic equations, as shown in Formulation 39.

$$\min_{x,u} \quad \frac{1}{2}\|x_N - \bar{x}_N\|^2_{S_N} + \sum_{t=0}^{N-1} \frac{1}{2}(\|x_t - \bar{x}_t\|^2_{Q_t} + \|u_t - \bar{u}_t\|^2_{R_t}) \tag{39}$$

$$\text{subject to} \quad x_{t+1} = f(x_t, u_t)$$

where,

S_N = final state, final time Hessian matrix (i.e. state weight matrix at final frame)

Q_t = time-varying state weight matrix for $0 < t < N - 1$

R_t = time-varying control weight matrix for $0 < t < N - 1$

N = discrete final time and number of samples in given reference trajectory

S_N, Q_t, and R_t are positive semidefinite matrices that indicate the weight between different objective terms. In Formulation 39, the shorthand $\|x\|^2_Y = x^T Y x$ is used. Similar shorthand is used in subsequent equations. The method by which these matrices are

determined will be discussed shortly. To solve an optimal control problem, it is convenient to define an optimal value function, $v(x_t)$ that measures the minimal total cost of the trajectory starting from state x_t. The optimal value function can be written recursively as shown in Equation 40.

$$v(x_t) = \min_u \frac{1}{2}(\|x_t - \bar{x}_t\|^2_{Q_t} + \|u_t - \bar{u}_t\|^2_{R_t}) + v(x_{t+1}) \tag{40}$$

The optimal value function is the key to Bellman's optimality principle. If the optimal value function can be evaluated, then an optimal control policy can be defined that maps a state to an optimal action: $\Pi : X \rightarrow U(X)$.

I use the optimal value function to derive probabilistic models for generating motion variance. The key insight is that the shape of this value function reveals important information about the tolerance of the control policy to perturbations. With this information, a perturbation can be selected that causes minimal disruption to the task execution while inducing visible variation to the reference motion.

However, optimizations can be very difficult to solve when the problem is nonlinear. One notable exception is the linear-quadratic-regulator (LQR) which has a linear dynamic equation and a quadratic cost function. A common practice in approximating a nonlinear dynamic tracking problem is to linearize the dynamic equation around the reference trajectory and substitute the variables with the deviation from the reference, Δx and Δu. The reformulation of the original optimization is shown in Equation 41.

$$\begin{aligned} \min_{\Delta x, \Delta u} \quad & \frac{1}{2}\|\Delta x_N\|^2_{S_N} + \sum_{t=0}^{N-1} \frac{1}{2}(\|\Delta x_t\|^2_{Q_t} + \|\Delta u_t\|^2_{R_t}) \\ & \text{subject to} \quad \Delta x_{t+1} = A_t \Delta x_t + B_t \Delta u_t \end{aligned} \tag{41}$$

$$A_t = \frac{\partial f}{\partial x}|_{\bar{x}_t, \bar{u}_t}$$
$$B_t = \frac{\partial f}{\partial u}|_{\bar{x}_t, \bar{u}_t}$$

The primary reason to approximate the problem with a time-varying LQR formulation is that the optimal value function can be represented in quadratic form with time-varying

Hessians, as shown in Equation 42.

$$v(\Delta x_t) = \frac{1}{2}\|\Delta x_t\|^2_{S_t} \tag{42}$$

The Hessian matrix S_t in Equation 42 is a symmetric matrix. At time step t, the optimal value function is a quadratic function centered at the minimal point $\bar{x}_t$. Therefore, the gradient of the optimal value function at $\bar{x}_t$ vanishes, while the Hessian is symmetric, positive semidefinite, and measures the curvatures along each direction in the state domain. A deviation from $\bar{x}_t$ along a direction with high curvature causes large penalty in the objective function and is considered inconsistent with the task. For example, the perturbation in the direction of the first eigenvector (corresponding to the largest eigenvalue) of the Hessian induces the largest total cost of tracking the reference trajectory. My algorithm uses this bias to induce more noise in the dimensions consistent with the tracking task. Using the Cholesky decomposition and a vector of standard normal samples, random samples are drawn from a zero-mean Gaussian with a time-varying covariance matrix defined as the inverse of the Hessian: $x_t = sample(\mathcal{N}(0, S_t^{-1}))$. The matrices S_t can be efficiently computed by the Riccati equation, shown in Equation 43, which exploits backward recursive relations starting from the weight matrix at the last frame S_N. The subscript t is omitted on A, B, Q, and R for clarity (detailed derivation in [114]).

$$S_t = Q + A^T S_{t+1} A - A^T S_{t+1} B(R + B^T S_{t+1} B)^{-1} B^T S_{t+1} A \tag{43}$$

The random sample, Δx_t, drawn from the Gaussian $\mathcal{N}(0, S_t^{-1})$ indicates deviation from the reference state trajectory $\bar{x}_t$. Directly applying this "task-aware" deviation to joint trajectories will cause vibration. Instead, my algorithm induces noise in control space via the feedback control policy derived from LQR: $\Delta u_t = -K_t \Delta x_t$. In the discrete-time, finite-horizon formulation, the feedback gain matrix K_t is a $m \times 2n$ time-varying matrix computed in closed-form from Equation 44.

$$K_t = (R + B^T S_{t+1} B)^{-1} B^T S_{t+1} A \tag{44}$$

4.2.4.1 Singular Hessians

Occasionally, the Hessians of the optimal value function become singular. In this case, singular value decomposition is applied to the Hessian to obtain a set of orthogonal eigenvectors E and eigenvalues $\sigma_1 \cdots \sigma_n$ (because S_t is always symmetric). For each eigenvector e_i, a one-dimensional Gaussian with zero mean and a variance inversely proportional to the corresponding eigenvalue is defined: $N_i(0, \frac{1}{\sigma_i})$. For those eigenvectors with zero eigenvalue, I set the variance to a chosen maximal value (e.g., the largest eigenvalue of the covariance matrix in the entire sequence). The final sample Δx_t is a weighted sum of eigenvectors: $\Delta x_t = \sum_i w_i e_i$, where w_i is a random number drawn from N_i.

4.2.4.2 Determine weight matrices

The cost weight matrices, S_N, Q_t R_t, can be selected manually based on prior knowledge of the input motion and control. Intuitively, when a joint or actuator is unimportant, a small value should be assigned to the corresponding diagonal entry in these matrices. Likewise, when two joints are moving in synchrony, it is intuitive to give them similar weights. Q and R in theory can vary over time, but most practical controllers hold Q and R fixed to simplify the design process. Still, when the robot has a large number of DOFs, tuning these weight matrices manually can be difficult.

I propose a method to automatically determine the cost weights based on the coordinations observed in the reference trajectory. Principal Component Analysis (PCA) is applied to the reference motion $\bar{x}$ and on the reference control $\bar{u}$ to obtain respective sets of eigenvectors E and eigenvalues Σ (in diagonal matrix form). The weight matrix for motion can be computed by $Q = E\Sigma E^T$. By multiplying Δx on both sides of Q, Δx is transformed into the eigenspace, scaled by the eigenvalues. As a result, Q preserves the coordination of joints in the reference motion, scaled by their importance. R can be computed in the same way. In my implementation, I set S_N equal to Q.

My variance algorithm can also integrate with existing techniques that organize multiple motions into graph structures or state-machines. As a practical implementation detail, a method to generate nondeterministic transitions between arbitrary motion sequences is

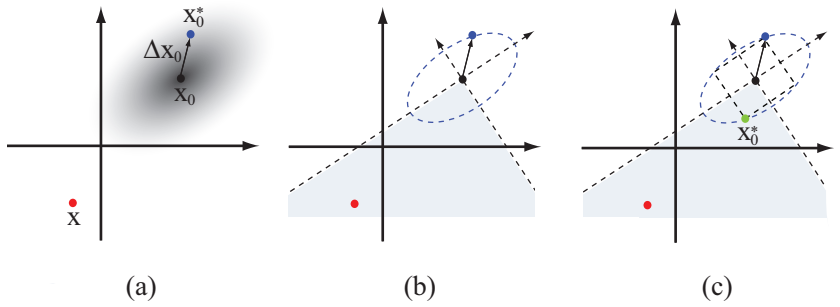

Figure 29: (a): If the transition-to pose is directly set to $x_0^* = \bar{x}_0 + \Delta x_0$, it is possible to generate an awkward transition when x_0^* is further away from the current pose x than from $\bar{x}_0$. (b): All poses with the same likelihood as $\bar{x}_0 + \Delta x_0$ form a hyper-ellipsoid. (c): Δx_0 defines a hypercube aligned with the eigenvectors of the covariance matrix. Select x_0^* as the corner that lies in the same quadrant as x to find motion transitions that span shorter distances and appear more realistic in Cartesian space.

described in Section 4.2.4.3.

4.2.4.3 Nondeterministic transitions

In this section, an algorithm is described that enables my technique to transition between multiple motion sequences. It is possible to transition to the next desired motion from a wide range of states. The state from which the next motion begins is termed the "transition-to pose." Furthermore, to make the transition reflect natural variance as well, the transition-to pose for the next motion is stochastically selected online. I call this technique *nondeterministic transition* because the transition motion between the same two given motion sequences is different each time.

Once the next motion is selected, the algorithm selects a transition-to pose x_0^* via a stochastic process, so that the robot does not always predictably transition to the first frame of the next motion. This can be viewed as a task-aware deviation from the first frame of the next motion $\bar{x}_0$. The Gaussian $\mathcal{N}(0, S_0^{-1})$ computed for the next motion is reused to get a random sample Δx_0. When the transition-to pose is directly set to $x_0^* = \bar{x}_0 + \Delta x_0$, it can generate an awkward transition if x_0^* is further away from the current pose x than from $\bar{x}_0$ (Figure 29 (a)).

To overcome this issue, the current state x is taken into account when selecting the

transition-to pose x_0^*. Since Δx_0 is drawn from a symmetric distribution, poses with the same likelihood form a hyper-ellipsoid in multi-dimensional space. To bias x_0^* toward the x, a state from this hyper-ellipsoid should be selected that lies in the same quadrant in the coordinates defined by the eigenvectors of the covariant matrix S_0^{-1} (Figure 29 (b)). To speed up computation, I use Δx_0 to define a hypercube aligned with the eigenvectors and select x_0^* to be the corner within the same quadrant as x (Figure 29 (c)). This particular corner of the hypercube produces a more natural, more realistic transition because it is closer to the end-point of the motion from which the robot is transitioning (i.e. the current pose at the start of the transition).

After determining the transition-to pose, spline interpolation and PID-tracking move the robot to this pose from the current pose. This interpolation works well because the transition-to pose is both consistent with the next task and biased toward the current pose.

4.2.5 Hypotheses

To measure success of my variant-generating algorithm for free-space gestures, I established the following criteria in the form of three hypotheses:

- H1: Variants produced by my algorithm significantly differ from the original, given, input motion in DOF important for expressing the gestures
- H2: Variants produced by my algorithm significantly differ from each other
- H3: Variants produced by my algorithm rival or surpass the quality of variants generated with other existing techniques

H1 and H2 are more quantitative because they make claims about whether the algorithm actually modifies motion, without regard to quality of the produced results. Whereas, the experiment that evaluates H3 was designed to demonstrate that my algorithm produces human-like variants, which is inherently a statement about the quality of produced results. These hypotheses are given in general terms here, but they will become more concrete in subsequent sections, as corresponding experiments to test these hypotheses are outlined.

4.2.6 Experiment 1: Variance in Unconstrained Gestures

Although the best way to demonstrate the amount of variance that can be created with my algorithm is to view the algorithm running in real-time on the robot, Experiment 1 was designed to quantify the amount of variance that can be generated. However, to fully quantify the algorithm, there were two variances that had to be measured: variance from the original motion and variance from other variants.

At the most fundamental level, Experiment 1 cannot evaluate quality of the variants. Instead, it only quantified the range-space of motion produceable from a single exemplar. This latter criterion is important to support arguments that the amount of variance produced by my algorithm is not merely *minimal* variance and is significant enough to produce noticeable variants. "Noticeable" will be further quantified with subsequent experiments.

4.2.6.1 Experimental Design

In the unconstrained case, the capability of my algorithm was demonstrated through two common social robot gestures: waving and an "I don't know" gesture. To characterize algorithm-generated variation, average variant square deviation from the original motion using 12 generated variants was calculated for each DOF individually using Equation 45. Equation 45 is Equation 35 averaged over K motions (i.e. variants).

$$\sigma_h{}^2 = \frac{1}{K}\frac{1}{N}\sum_{i=1}^{K}(\bar{q} - q_i)^T(\bar{q} - q_i) \tag{45}$$

where,

$\sigma_h{}^2$ = average square deviation of DOF h with respect to original, input motion

q_i = the joint angle trajectory of DOF h from variant i as a vector

$\bar{q}$ = the joint angle trajectory of DOF h from the original, input motion as a vector

K = number of variants used in the analysis

N = number of time samples in given reference trajectory

Equation 45 has the potential to show that my algorithm produces motions different from the original, but it does not show that variants are different from one another. To

verify that all variants are sufficiently different, the original motion in Equation 45 is replaced with variants from the set, which gives Equation 46. Similarly, average variance between variants (i.e. average inter-variant variance, AIVV) using 12 generated sequences was calculated for each DOF individually using Equation 46.

$$AIVV_h = \frac{1}{(K-1)^2}\frac{1}{N}\sum_{i=1}^{K}\sum_{j=1}^{K}(q_j - q_i)^T(q_j - q_i),\ \ i \neq j \tag{46}$$

4.2.6.2 Results

For the purposes of this analysis, only the left arm DOFs were evaluated. This is valid for a couple of reasons. The waving gesture used in this analysis was performed with the left hand. Furthermore, the cost function used in the optimal control formulation of the algorithm ensures that more variance is created in DOFs where there is more torque (i.e. it is "task-aware"). This is consistent with the expectation for a variance-generating algorithm, since the goal was to create more variance in the gesture (e.g. the waving hand) as opposed to the other hand which would only exhibit secondary motion. Also, the original "I don't know" gesture was symmetric, and analysis of the right arm would have provided negligible additional information. Thus, the left arm was sufficient to characterize the output of a variance-generating algorithm with these input gestures.

For the waving motion, shoulder Y, wrist Y, and wrist X were the DOFs with most actuation. For the shrug ("I don't know") motion, all the arm DOFs were moving with near-equal actuation, except for the elbow X and wrist Y DOFs, which were moving significantly more than the other DOFs. Table 37 shows that large variance was created in the DOFs with most actuation for different motions. But at this point, the numbers in Table 37 allow me only to conclude that my algorithm creates variants that are quantifiably, measurably different from the input motion. If these numbers were close to zero, it would mean that my algorithm for these particular gestures was unable to create significant variance in the important DOFs for the given gestures (e.g. waving motions should be different (have more variance) in the arm that is waving). This analysis was important to perform because it may not be immediately intuitive that the small amounts of torque noise injected by my

Table 37: Average variance and standard deviation from the original motion, of joint angles in the left arm. Averages taken over first 12 exemplars generated by the algorithm. Variance in square degrees. Larger numbers indicate more average deviation from the original motion.

Left Arm DOF	"Waving" var. (st. dev.)	"I don't know" var. (st. dev.)
shoulder X	364 (309)	974 (900)
shoulder Z	170 (137)	751 (229)
shoulder Y	1198 (488)	1238 (684)
elbow X	185 (176)	5993 (2412)
wrist Y	649 (57)	5354 (347)
wrist X	664 (384)	1742 (290)
wrist Z	177 (111)	834 (358)

algorithm could give rise to anything beyond minimal output variance in DOFs important for specific gestures. However, I have only examined the lower bound of variance with Experiment 1. Subsequent experiments will evaluate my algorithm to ensure that the output variance is "noticeable."

A variance-synthesizing algorithm is not very useful if the produced variants always resemble each other. The results in Table 38 show that on average the variants generated by my algorithm are different from each other, especially in the "I don't know" gesture.

Furthermore, as a general rule, the variances computed in Tables 37 and 38 should not only have moderate variance in important DOFs (as has already been discussed), but standard deviation should also be moderately-high. "Moderate" depends on the units used for the analysis. A standard deviation near zero in Table 37 would mean that variance from the original motion is fairly uniform regardless of the variant (i.e. all variants differ to a similar extent from the original motion). This is a measure similar to the information in Table 38, and in general, a good variance-producing algorithm should produce motions both similar to and different from the original input motion (which is what a moderately-high standard deviation indicates). Similarly, a standard deviation near zero in Table 38 would indicate that variance between variants was uniform, which can lead to repetitive produced motion (i.e. variants differ from each other to the same extent). The variance

Table 38: Average variance between variants (i.e. inter-variant variance) and standard deviation for joint angles of the left arm (in sq. degrees). Averages taken using the first 12 motions generated by my algorithm.

Left Arm DOF	"Waving" var. (st. dev.)	"I don't know" var. (st. dev.)
shoulder X	294 (143)	3740 (1847)
shoulder Z	90 (81)	1078 (867)
shoulder Y	2143 (1637)	2902 (2348)
elbow X	1627 (712)	2846 (2288)
wrist Y	704 (383)	643 (520)
wrist X	1470 (756)	235 (189)
wrist Z	510 (437)	202 (173)

algorithm was created to avoid repetitive motion. Neither standard deviation in Table 37 or Table 38 should be too high because this can indicate "task" corruption. It is important that the variants do not corrupt the original gesture intent, but I chose not to quantify the exact upper bounds for standard deviation or variance in Experiment 1; my subsequent experiments will demonstrate that my algorithm does not corrupt the "task."

As mentioned previously, my algorithm is capable of generating infinite variants of a motion in the absence of constraints. But, the best way to see this difference is visually. Some of the variants produced for the "I don't know" gesture are shown in Figure 30.

The disadvantage of the analysis of Experiment 1, is that deviation in that experiment is being determined in joint space, which does not linearly correlate to the amount of

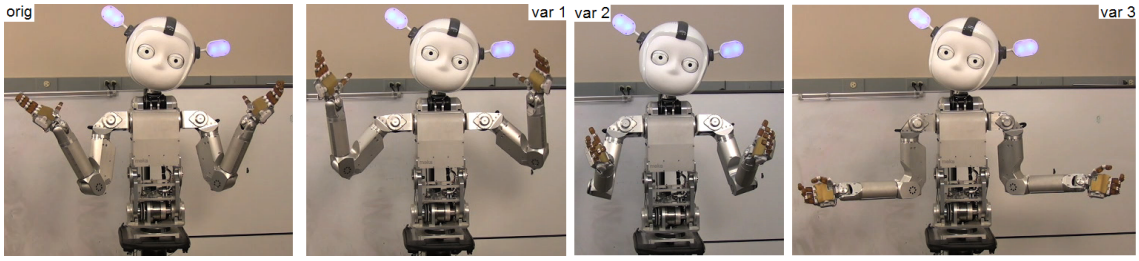

Figure 30: "I don't know" motion. Far left is the original motion. Other three are the inflection points in three variants generated by my algorithm.

variance in Cartesian space. Amount of perceived variance in Cartesian space is important because this is the space in which motions are observed. Since joint space is nonlinear, redundancy in degrees-of-freedom can cause large deviation in joint space but small deviation in Cartesian space, which could result in misleading conclusions from the analysis performed in Experiment 1. Thus, Experiments 2 and 3 will focus on variance in Cartesian space to show that variance is noticeable and significant but without corrupting the task.

4.2.7 Experiment 2: Variance is Task-Aware

Recall that "task" is used to refer to the original motion that the robot performed, and therefore, "task-aware" means that all variants should also be classified as the same motion type, gesture, or action as the original motion input to the algorithm, from which the variants were created. In other words, when the robot moves using a variant, observers should perceive or label the robot intent to be the same as the intent they would perceive or label if the robot were moving using the input motion.

Experiment 2 was motivated to demonstrate that the amount of variance produced by my algorithm is not so significant that it distorts the "task" to the extent that variants would be labeled as a gesture differently than the original, input motion. Since quantifying an upper bound on the variance calculations in Equations 45 and 46 is motion-specific, determining these upper-bounds is not the most intelligent means to confirm that my algorithm does not produce excessive variance. Thus, to show that excessive variance is not produced and that variants remain "task-aware," I designed Experiment 2.

4.2.7.1 Experimental Design

In Experiment 2, 153 participants were asked label one version of five separate motions. The five motions were: shrug (i.e. "I don't know"), beckon, wave, bow, and point. Participants watched videos of motions executed on the hardware so that the experiment could proceed more quickly, be more repeatable, and be well controlled. Participants watched the videos through web-based code, sequentially, and the experiment was conducted over the internet. After a video stopped playing, the screen blanked and prompted the user for a label. The next motion video appeared after the label was entered.

153 participants provided only 10 samples per variant, since the original motion was allotted higher samples rates for data resolution. In the results, variants will be compared against the original motion with respect to percentages of participants, and the data resolutions of each are 10% for individual variants, 0.833% for cumulative variants, and 3.03% for the original motion.

The order of the five motions was randomized, and each participant saw either the original motion version or one of 12 variant versions created using my algorithm. For Experiment 2, the 12 variant versions were approximately uniform selections based on inter-variant variance for each motion, so that the range of variants produced by my algorithm could be evaluated for "task" corruption. The next experiment, Experiment 3, will eliminate this bias of selecting particular variants for evaluation, and select variants randomly generated in the natural order produced by my algorithm.

Thirty three participants saw the original motion version for a particular motion, leaving ten people who saw each variant version of that same motion. All orders and motions were randomized. No participant saw only original motion versions for all five motions.

4.2.7.2 Results

Recognition was the metric used for Experiment 2, to provide evidence that my variance algorithm does not distort the "task." Earlier I defined the task as a cost function that measures motion execution (e.g. desirability of joint position, joint velocity and torque usage). In other words, the task is to produce the free-form gesture that was given as the input motion. Thus, recognition (i.e. labeling of the variants with the same label as the original motion), was an appropriate metric to measure the success of not disrupting the "task."

By comparing the bottom two rows in Table 39, in four of five motions, percent correct recognition is slightly higher for the variants than for the original motion. Additionally, percent correctly labeled is never more than 0.8% worse for the variants generated with my algorithm. Therefore, I conclude that my variance algorithm does not distort the original intent or meaning of the gesture (i.e. the original "task").

Table 39: Percent correctly labeled (%) for 12 variants and the original motion of five different gestures. (10 samples per variant, 33 samples for each original motion). Average for all motion variants as compared to the original motion, show that recognition does not decrease for generated variants (i.e. the algorithm does not corrupt the task of producing variants that are consistently labeled as the same motion type as the original motion).

Motion	Beckon	Bow	Point	Shrug	Wave
Variant 1	100	100	100	90	100
Variant 2	100	100	100	90	100
Variant 3	100	100	100	100	90
Variant 4	100	100	100	100	100
Variant 5	90	90	100	100	100
Variant 6	100	90	100	100	100
Variant 7	100	100	100	100	100
Variant 8	100	100	100	100	100
Variant 9	100	100	100	100	100
Variant 10	90	100	100	100	100
Variant 11	100	90	100	100	100
Variant 12	100	100	100	100	100
Var. Avg.	98.3	97.5	100	98.3	99.2
Original	93.9	93.9	100	87.9	100

4.2.8 Experiment 3: Frequency of Noticeably-different Variants

A variance-producing algorithm is less useful if noticeably-different variants are produced rarely. Therefore, Experiment 3 is designed to show two things: (1) my algorithm does not produce a series of similar variants in a row, and (2) noticeably-different variants are produced frequently.

4.2.8.1 Experimental Design

In Experiment 3, one hundred sixty five participants were asked to watch video pairs of five separate motions. The five motions were: shrug (i.e. "I don't know", beckon, wave, bow, and point. The first twelve variants of each motion produced by the algorithm were executed on the hardware and recorded. These variants were labeled variant 1, variant 2, etc. The videos were then paired for a particular motion based on order. I was trying to determine the average minimum number of variants created from my algorithm that a participant must watch before they see a noticeable difference between generated variants.

Thus, videos were paired according to their generated sequence.

Let the original motion be equal to variant i=0. Then, the following three sets of motion pairs were used in the experiment, from the first twelve variants produced using my algorithm (represented by i=1 to i=12).

- Set 1: Variants produced sequentially. $(i, i+1)$, 12 pairs
- Set 2: Variants generated one apart. $(i, i+2)$, 11 pairs
- Set 3: Variants synthesized two apart. $(i, i+3)$, 10 pairs

There were twelve possible pairings of videos in Set 1: (0,1), (1,2), ...,(11,12). There were eleven possible pairings of videos in Set 2: (0,2), (1,3), ...,(10,12). There were ten possible pairings of videos in Set 3: (0,3), (1,4), ...,(9,12). One hundred sixty five participants yields five viewings of each of the 33 possible pairs of videos in Experiment 3 (for a given motion type). For Set 2 and Set 3, the participants were still only watching two motions; videos of the intermediate variants are not shown to participants.

By examining the responses, the percentage of participants who saw noticeably-different motion in each variant, in every other variant, and in every third variant was calculated. Additionally, I calculated the average number of variants produced with my algorithm that a participant had to watch before seeing noticeably-different motion.

Participants watched videos of motions executed on the hardware so that the experiment could proceed more quickly, be well controlled, and be more repeatable. Participants watched the videos through web-based code, sequentially, and the experiment was conducted over the internet since speed was not an issue. Participants were told that both motions that they would see in a video pair were the same length in time, and therefore, differences in speed or timing should be neglected in their responses. After a video pair stopped playing, the screen blanked and the user was prompted with a single question. "Did you notice a difference between the motions in pair of videos you just watched?" Only two possible responses were given: "yes" and "no." A blank text box was provided in cases where users wanted to explain the difference or their answer. However, this text

box was optional. The next motion video appeared only after the forced selection was completed. The experiment was complete after a participant watched five pairs of videos (one pair for each motion).

For Experiment 3, order of the pair of videos for a particular motion mattered. Therefore, only the order of the five motions and the selection of the particular video pair (for a given motion) were randomized.

4.2.8.2 Results

Table 40 shows the data for all twelve pairs of motion videos of variants that were generated sequentially. On average, the data indicates that noticeably-different motion variants tend to occur one-after-another. My motion algorithm produces them frequently and does not produce many similar variants in a row. It varies slightly by motion, but at least 85% of participants recognized motion difference in sequential motion variants.

Table 40: Percent (%) of participants who noticed a difference in back-to-back motion variants for five different gestures. (5 samples per video pair). Average for all variant pairs (V1-V12) shows that the majority of participants see noticeable motion difference in subsequent motion variants. V0 = original motion.

Video Pair	Beckon	Bow	Point	Shrug	Wave
V0 - V1	100	100	100	100	100
V1 - V2	60	80	100	100	100
V2 - V3	60	100	80	100	100
V3 - V4	60	100	100	100	100
V4 - V5	100	80	80	100	100
V5 - V6	80	100	100	100	80
V6 - V7	100	100	80	100	100
V7 - V8	100	100	100	80	100
V8 - V9	80	80	100	100	100
V9 - V10	100	100	60	100	100
V10 - V11	100	60	100	100	100
V11 - V12	80	100	80	100	100
Avg. V1-V12	83.6	90.9	89.1	98.2	98.2

Since the data in Table 40 supported the argument that each motion variant produced using my algorithm is noticeably different, there is no reason to expect different results

Table 41: Percent (%) of participants who noticed a difference in alternating motion variants for five different gestures. (5 samples per video pair). Average for all variant pairs (V1-V12) shows that the majority of participants see noticeable motion difference in alternating motion variants. V0 = original motion.

Video Pair	Beckon	Bow	Point	Shrug	Wave
V0 - V2	80	100	80	100	100
V1 - V3	100	80	100	100	100
V2 - V4	100	80	100	80	100
V3 - V5	80	100	100	100	80
V4 - V6	100	100	80	100	100
V5 - V7	100	80	100	100	100
V6 - V8	100	100	80	100	100
V7 - V9	100	100	100	100	100
V8 - V10	80	80	80	100	80
V9 - V11	100	100	100	80	100
V10 - V12	100	80	100	100	100
Avg. V1-V12	96.0	90.0	94.0	96.0	96.0

from alternating motion variants or every third variant produced using my algorithm. If different results were produced, it would indicate a bias in my algorithm. Based on the data in Tables 41 and 42, a trend similar to that shown in Table 40 is apparent. The majority of participants (greater than 90%) noticed that the generated variants are different.

From the data in Tables 40, 41, and 42, I concluded that there is a recognizable difference in each motion variant. Generalizing across motions, 92.0% of people noticed that variants differed after seeing the next motion; 94.4% of people saw noticeable difference between variants in alternating motions; and 93.8% of participants noticed variance from my algorithm in every-third variant. If efficiency of this algorithm can be measured by how frequently it produces noticeably-different variants, then in general, my algorithm will produce noticeable variance 93.4% of the time, which is the average of the bottom row in Table 43. H2 was supported by this data, and variants are different from each other as measured by human perception.

Table 42: Percent (%) of participants who noticed a difference in every third motion variant (from the sequence of the first twelve motion variants) for five different gestures. (5 samples per video pair). Average for all variant pairs (V1-V12) shows that the majority of participants see noticeable motion difference in every third motion variant. V0 = original.

Video Pair	Beckon	Bow	Point	Shrug	Wave
V0 - V3	100	60	100	100	100
V1 - V4	100	100	100	100	100
V2 - V5	100	100	80	100	100
V3 - V6	80	80	100	100	100
V4 - V7	100	100	100	100	100
V5 - V8	100	100	100	80	100
V6 - V9	80	80	80	100	100
V7 - V10	80	100	100	100	80
V8 - V11	80	100	60	100	100
V9 - V12	100	80	100	100	80
Avg. V1-V12	91.1	93.3	91.1	97.8	95.6

Table 43: Average percent (%) of participants who noticed a difference in motion variants (from the sequence of the first twelve motion variants) for five different gestures organized by the three sets.

Set	Beckon	Bow	Point	Shrug	Wave
Set 1	83.6	90.9	89.1	98.2	98.2
Set 2	96.0	90.0	94.0	96.0	96.0
Set 3	91.1	93.3	91.1	97.8	95.6
Average	90.0	91.3	91.3	97.3	96.7

4.2.9 Experiment 4: Creating Human-like Variants

H1 and H2 ignore the quality of the produced motion, and therefore, it was necessary to demonstrate that my variance algorithm produces motions that are as human-like as the input motion. This is further evidence that it does not corrupt human-like characteristics of the motion (i.e. it is task-aware).

4.2.9.1 Experimental Design

In order to prove that my variance algorithm produces human-like variants, an objective, comparative evaluation against human motion was required. A data set of "waving" and

"I don't know" gestures were collected from 24 different humans with motion capture equipment. Each of these 48 motions were projected onto the Simon robot architecture using an optimization that calculates Simon's joint angle values as a function of time from the 28 upper-body position constraints collected from human motion capture markers and a similar set of constraints positioned on the robot body. The optimal mapping allows for proportional scaling the overall size of Simon based on a human participant's size. This procedure creates a motion trajectory that the robot can execute [100].

This data set of 48 human motions projected onto the robot hardware was the ground truth for human-like motion. Motions generated from my algorithm were compared to three alternative variance-inducing algorithms, yielding the following four test cases:

- My variance-generating algorithm. Hereafter, for simplicity of reference it will be referred to as task-aware variance (TAV).

- Random White Torque Space Noise (RWTSN). This is a naïve variance approach, that generates smooth motion based on the idea of allowing unbiased torque noise to alter trajectories without intelligent shaping. To produce unbiased torque noise, step two in Section 4.2.4 was replaced by $\Delta u_t \sim \mathcal{N}(0, I)$. Subsequent results show that purposeful motion is distorted and disrupted by random noise.

- Operational space control + RWTSN. Random torque noise was projected onto the robot body after applying two time-varying Cartesian end-effector point constraints (one per hand) calculated from the average of all the human motion-capture data. For point constraints and partial posture constraints, white noise torques projected into the null space of the robot architecture creates one of two outcomes: task disruption or lack of naturalness.

- Style-based inverse kinematics (SIK). I implemented a basic form of style-based inverse kinematics, in which human motion capture data is represented in a low dimensional space. By learning probability density functions of the data and interpolating in the low-dimensional model space, motion variants were generated [115]. The disadvantages of SIK are that variance is limited by the database and motion

variants generated may appear unnatural when interpolating between drastically different exemplar models. The implementation details for style-based IK can be found in [115].

For each of the four test cases, 20 variants of both "waving" and "I don't know" gestures (40 total) were generated. Nearest-neighbor, with a Euclidean distance metric in joint angle space, was used to find the minimum distance between the technique-generated variant and a trajectory in the ground truth set. This assumes that distance to a ground truth trajectory is a good metric for human-like quality of a given motion. For this analysis, all techniques were constrained to robot joint angle limits, i.e., torque noise was not applied to any DOF which would force it to exceed a joint angle limit. Additionally, certain DOFs (e.g. eyes, eyelids, and ears) were excluded from the analysis because they were not measurable with the motion capture equipment.

4.2.9.2 Results

The results are presented in Table 44 for "waving" and "I don't know" gestures. The means presented are averages for all times during the trajectory, for all DOFs, and for all 20 motion variants. For the style-based IK technique, cross validation was performed ten times using random sets of 12 exemplars to test and 12 exemplars to model the probability density functions. In Table 44, the averages for the cross validation are presented.

Table 44 shows that for two common social robot gestures, OSC and random torque

Table 44: Average distance of resultant motions from the four variance generating techniques to its nearest neighbor in the human-like data set, for two gestures. Results are an average of 20 variants, and values measure joint angle Euclidean distances in radians. SD = standard deviation.

	RWTSN	OSC	SIK	TAV
Waving Mean	0.723	0.502	0.198	0.155
Waving (SD)	(0.246)	(0.176)	(0.178)	(0.074)
I don't know Mean	0.628	0.424	0.217	0.151
I don't know (SD)	(0.254)	(0.146)	(0.142)	(0.144)

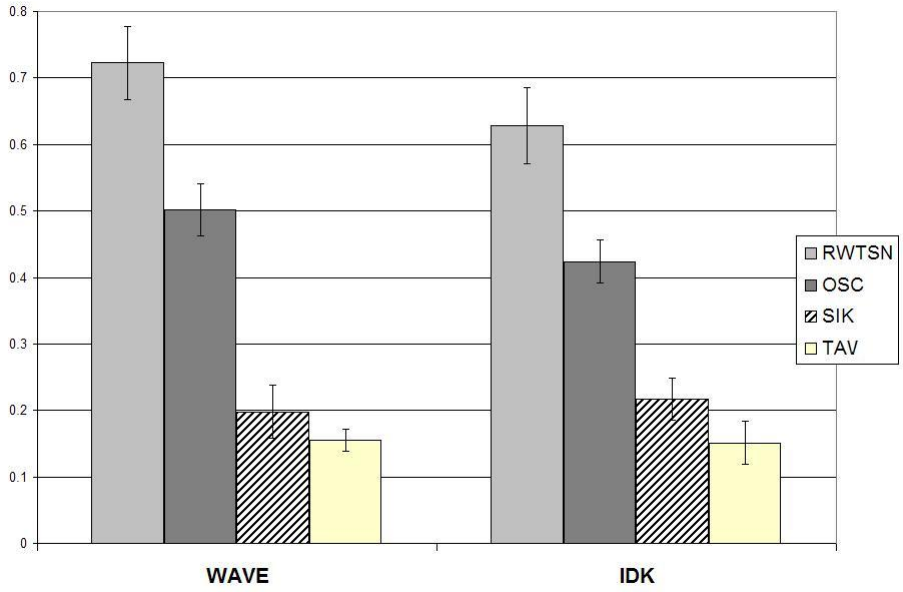

Figure 31: Average distance to nearest human neighbor in human-like set. Techniques are random white torque space noise (RWTSN), operational space control (OSC), task-aware variance (TAV), and style-based IK (SIK). T-tests results with error bars for "waving" and "I don't know" gestures. TAV is statistically different from RWTSN and OSC in both gesture cases. TAV and SIK are not significantly different with respect to the utilized human-likeness measure.

noise are less human-like than task-aware variance and style-based IK. As shown in Figure 31, paired t-tests showed statistical significance between all techniques ($p<0.01$) for both gestures, except task-aware variance and style-based IK. This suggests that TAV provides all the variance benefits of a high-quality technique like style-based IK, without segmenting, time-warping, annotating data, and other preprocessing steps necessary for model synthesis. Unlike SIK, which requires dozens of input exemplars, TAV produces motion without training a model and uses only one exemplar.

4.2.10 Discussion

My algorithm is capable of producing an infinite number of motion variants from a single exemplar. Based on human sensitivity to difference (i.e. what is "noticeable" to the human eye), a subset of these variants will be effectively equivalent to the original input motion.

Similarly, a different subset of generated variants will be effectively equivalent to each produced variant. Human sensitivity to difference between variants causes subtle variance to appear effectively equivalent. This human sensitivity manifests in the data in Experiment 3. However, based on the design of my experiments, it was not necessary to quantify this human sensitivity.

Earlier I stated that a good variance-generating algorithm produces both motions similar to and different from the original. In Experiment 3 the numbers were high for noticeably different exemplars (more than 93% of variants appear different); these high percentages can mean either that (1) the variants were significantly different or (2) the variants were subtlely different, but people are acute at noticing difference in generated motion. Regardless, the important result is that humans perceive this difference because that is the true measure of success of any variant-generating algorithm.

4.2.11 Summary

In an effort to create better human-robot interaction, I add variance to create human-like motion in social robots. I address the problem of creating variability in a set of exemplar gestures for a robot by presenting task-aware variance, an autonomous algorithm capable of generating an infinite number of human-like motion variants in free-form gestures. Furthermore, I use the solution to the optimal control problem to create variance in the initial condition of the trajectory by directing the selected sample to be closest to the current point in the state space; the latter is useful when transitioning between motions to avoid initial condition redundancy.

I quantified the range of variance capable from my algorithm for a subset of common robot gestures and proved that the variants maintained the intent of the original gesture (i.e. the algorithm is task-aware). Finally, I showed that my algorithm creates motion variants more human-like than two other variance-generating algorithms, in real-time using only one exemplar.

CHAPTER V

CONSTRAINED MOTION

5.1 Extended Operational Space Control

5.1.1 Introduction

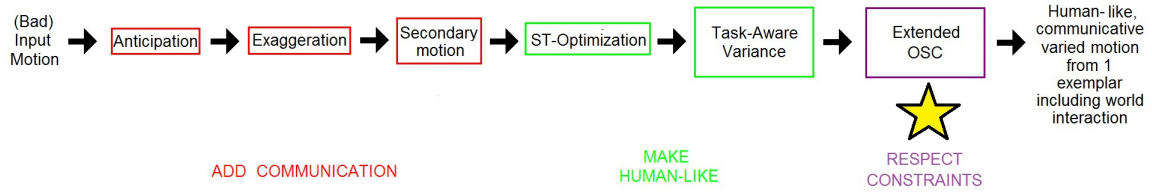

Figure 32: Chapter 5 discusses constraints.

The constraint-preserving algorithm is the final algorithm discussed in this thesis, as shown in Figure 32. It follows all other algorithms because external constraints must be satisfied for the robot to function in the world. Changes induced by the communication or human-like algorithms should not disrupt the robot's ability to interact with the world; and thus, since satisfying constraints is the most important task, motion is adapted for constraints last in the algorithmic sequence.

One of the most essential aspects of social robot motion control involves interaction with the environment. My communication, human-likeness, and variance algorithms were designed assuming free-form gestures as trajectories. However, social robots interact with humans and their environment constantly. Therefore, the challenge is to design a method that allows these algorithms to work in conjunction with the other "tasks" that a robot must perform, under the entire set of "constraints" imposed upon robots functioning in the real-world without interrupting the hierarchy of other actions being simultaneously performed.

In this section, the words "task" and "constraint" will be used interchangeably, and

a precise definition is needed. In Section 4.2.2, the word "task" was defined by a cost function that measured how well a given motion could be tracked. But Section 4.2.2 discussed adding variance to motion, and variance can be thought of as an "internal" constraint imposed upon motion (i.e. trajectories should not be perfectly, repetitively tracked). Based on the real-world needs of social robots, the definition of the word "task" needs to be extended to include anything internal and/or *external* to the robot that affects its motion. This new definition implies that a task is measured both by (1) the cost function for motion tracking and (2) how closely a set of physical criteria (i.e. constraints) on motion can be maintained. And when multiple "tasks" are competing for control of robot motion, they must be composed on a single architecture. Thus, robot motion is essentially a composition of "tasks," resulting from simultaneous competing goals, algorithms, and demands on the system. The process of composing these factors (e.g. communication, variance, constraints) will be described in detail in Chapter 6.

The following basic constraints will be addressed in detail: trajectory (i.e. posture) constraints, kinematic point constraints, orientation constraints, point-at constraints, look-at constraints, velocity constraints, synchronization constraints, timing constraints, and dynamic constraints. Also, I will discuss how more sophisticated constraints (e.g. volume constraints, proximity constraints) can be handled by the simultaneous application of the more basic constraints.

5.1.2 Insight

Constraints can be specified as trajectories of forces, velocities, torques, positions, angles, or accelerations depending on the implementation of a particular robotic system. Hence, to be universally applicable and composable on a singular architecture, they need to be represented in a singular domain.

Techniques such as inverse kinematics (IK) that transform between domains are the insight for my technique to satisfy constraints and handle the diverse set of "tasks" that social robots require. Trajectories, forces, velocities, etc. specified in Cartesian space can be transformed to joint space by inverse kinematics using the Jacobian for redundant systems.

Through the use of techniques such as IK, external kinematic constraints are transformed into internal constraints, which then allows all constraints (both internal and external) to be handled in a single, consistent manner.

Redundancy exists on social robot architectures when the number of dimensions in joint space is greater than the number of degrees-of-freedom in Cartesian space. This can be described by a nonlinear, many-to-one mapping between spaces. Since there is a discrepancy between the number of degrees-of-freedom in these spaces, trajectories specified in the lower-dimensional space (i.e. real-world coordinates, three-dimensional Cartesian space) map to many different possible trajectory solutions in the higher-dimensional space (i.e. robot joint actuators, joint space). This underspecification does not fully constrain the higher-dimensional space and the range of possible mappings creates a null space in joint space. For example, point-to-point end-effector trajectories allow the arm to trace any path in Cartesian space to the target location, with any timing, using multiple configurations for any given hand position, due to a redundant set of DOFs.

The minimum intervention principle is a motor control strategy wherein feedback is used only to correct trajectory deviations that interfere with task goals, thereby constraining motor variability to the uncontrolled manifold [116]. Additionally, through coupling control variables (i.e. exploiting redundancy and synergies), tasks can constrain sums of state variables to increase variability in redundant dimensions [49]. Analysis of people hopping in place at constant frequency demonstrates that joint torque variance (i.e. redundancy) can be exploited for functional, task-specific control. Fore/aft corrections to vertical hopping are used to minimize cycle-to-cycle ground reaction force variance. This human data has shown that inducing meaningful end-effector variance in one direction can be used to maintain a task constraint (minimum variance) in another degree of freedom [117].

Inspired by the hopping experiments, I can maintain constraints on a redundant robotic architecture via a projection of forces to cancel influence in dimensions that would interrupt the task. For example, consider a robot performing a shrug motion while trying to hold its left hand at a specific point in Cartesian space. Based on the shrug torque trajectory,

a trajectory of forces can be calculated that will keep the hand at the point. These three-dimensional forces are transformed into a set of torques for the robot actuators using the Jacobian. The shrug torques are projected to cancel actuation in dimensions that would disrupt the point constraint.

The composition of constraints in my formulation arises through successive projections into the null space on the robot architecture. Each added constraint reduces the space of redundant coordinates, successively leading to a smaller null space. In general, as more constraints are imposed upon the robot architecture, less motion occurs. Thus, the key to my constrained motion lies in the exploitation of the null space and redundancy to satisfy a set of prioritized constraints in an optimal (i.e. least-squares) sense, given the order of applied constraints.

5.1.3 Goal

In general, my algorithm demonstrates how a constraint or task, represented as a torque trajectory can be applied to a robot without violating the state space dimensions required to maintain the constraint. Therefore, the goals are to:

- demonstrate how to transform constraints and tasks into the torque domain
- present a mathematical formulation that exploits redundancy and the task null space
- show how to apply constraints to posture tasks (i.e. motion)
- formulate the composition of multiple constraints
- describe Operational Space Control to maintain kinematic constraints
- modify my task-aware algorithm (from Section 4.2) to satisfy dynamic constraints (e.g. synchronization, timing, velocity)

5.1.4 Algorithm

To handle the diverse set of "tasks" that social robots require, it is beneficial to divide these "constraints" into two categories:

- External: any task or constraint that can be resolved into Cartesian variables (e.g. position, velocity, acceleration, force trajectory)

- Internal: any task or constraint that is represented in joint-space variables (e.g. position, velocity, acceleration, torque trajectory)

The method to maintain and compose each of the two distinct types will be discussed separately, since they are handled differently in my formulation. Before discussing the representative algorithms, I will discuss how to formulate each of the distinct constraints in both categories into torque trajectories, since both types need to be represented in this consistent domain.

5.1.4.1 Internal Constraints

Internal constraints can be formulated to encompass a wide breadth of common scenarios that social humanoid robots encounter in the real-world. Internal constraints could be considered "soft" constraints because unlike external constraints there is no guarantee that they will always be *perfectly* satisfied. However, the algorithm that will be presented to maintain internal constraints is designed with the flexibility (i.e. weighting) so that tasks can be strictly or loosely constrained.

Internal constraints are always provided to the framework in the form of an input or given motion. Internal constraints are timing or dynamic constraints where a certain region (i.e. time period) of the trajectory is constrained to be synchronized to the input motion. Internal constraints typically come from joint position or its derivatives:

- Posture: The most basic form of internal constraint is a posture constraint where the joint angle position of a set or subset of DOFs are synchronized to the given original, input trajectory. This trajectory is a given motion that the robot should perform in addition to all other tasks.

- Velocity: Similarly, velocity of the robot joints can be constrained over a certain time region to match or synchronize with the velocity trajectory of the given motion.

- Acceleration (or Torque): And finally, the joint acceleration trajectory (i.e. torque trajectory) of the given motion can act as a constraint for the robot. Thus, the task would be trying to maintain this acceleration trajectory for a set of DOFs for a specified time range, in the presence of all other robot tasks.

For all these different internal constraints, the time range can be as short as one time sample or as long as the time range for the specified input motion.

5.1.4.2 Algorithm - Internal

The algorithm for maintaining internal constraints is derived by solving a discrete optimal control problem. Given a sequence of poses q_t, $0 \leq t \leq N$ that constitute a motion, a reference state trajectory $\bar{x}$, and its corresponding reference control trajectory $\bar{u}$ are constructed. The state trajectory consists of both q_t and $\dot{q}_t$, while the control trajectory $\bar{u}$ consists of joint torques computed from an inverse dynamics process.

In optimal control, an optimization is formulated to track the reference trajectory. $\bar{x}_t \in \Re^{2n}$ contains the joint angles and velocity at frame t while $\bar{u}_t \in \Re^m$ contains the joint torques. The task can be viewed as minimizing the state and control deviation from the reference trajectory, subject to discrete-time dynamic equations, as shown in the finite time LQR Formulation 47.

$$\begin{aligned} \min_{\Delta x, \Delta u} \quad & \frac{1}{2}\|\Delta x_N\|^2_{S_N} + \sum_{t=0}^{N-1} \frac{1}{2}(\|\Delta x_t\|^2_{Q_t} + \|\Delta u_t\|^2_{R_t}) \\ \text{subject to} \quad & \Delta x_{t+1} = A_t \Delta x_t + B_t \Delta u_t \end{aligned} \tag{47}$$

$$A_t = \frac{\partial f}{\partial x}|_{\bar{x}_t, \bar{u}_t}$$
$$B_t = \frac{\partial f}{\partial u}|_{\bar{x}_t, \bar{u}_t}$$

In Formulation 47, the shorthand $\|x\|^2_Y = x^T Y x$ is used. Similar shorthand is used in subsequent equations. In the discrete-time, finite-horizon formulation, the feedback gain matrix K_t is a $m \times 2n$ time-varying matrix computed in closed-form from Equation 48.

$$K_t = (R + B^T S_{t+1} B)^{-1} B^T S_{t+1} A \tag{48}$$

Solving backward from an appropriate initial choice of S_N, I exploit the formulation of optimal control, which tracks a trajectory according to a cost function, in order to satisfy soft constraints. The state in the optimal control formulation contains both joint position and velocity. Thus, all the synchronization constraints reduce to increasing the penalty against violating all important features increases closer to the point in time when those features must be maintained. For example, by introducing very high weights for all constrained DOFs at the constrained velocity trajectory points, the output torques from the optimization $u_t = -K_t x_t$ will be very close to the velocities at the same time points in the given input trajectory. The reason that this algorithm can preserve velocity and time-dependent task features is because the optimal control policy accounts for the cost of the entire trajectory; the cost of a future state will affect the action of the current state.

Multiple internal constraints are easily composed since the weight matrices are distinct for each constraint in the optimal control formulation. The weight matrices are independently set higher for constrained position, velocity, or torque times, and the optimization is solved only once. Thus, the composition of all internal constraints occurs simultaneously when the LQR problem is solved. A "pseudo-priority" can be imposed upon internal constraints based on relative constraint importance with respect to each other through the actual values imposed on joint position, velocity, and torque at different points in time. Higher weights will act as higher-priority constraints, since they will be more strictly satisfied. Solution of the LQR formulation ensures that the resultant constrained motion remains smooth.

As a starting point for the definition of weight matrices, a baseline of values is computed in Section 4.2.4.2. Position, velocity, or acceleration weights should be set lower than this baseline for timing ranges when they are unconstrained and when synchronization is unimportant. Constrained times may need higher weight values than this baseline, but it depends on the relative timing of other constraint weights'. For example, if the velocity is constrained at sample indices 1 and 5, then the weight for constrained velocity at sample index 3 does not need to be as high as it would have to be if indices 1 and 5 were unconstrained.

5.1.4.3 External Constraints

Four primary external constraints will be discussed and defined in the subsequent sections that can be formulated to encompass a wide breadth of common scenarios that social humanoid robots encounter in the real-world.

- Point Constraints
- Orientation Constraints
- Point-At Constraints
- Look-At Constraints

Sections 5.1.4.3.1 to 5.1.4.3.4 will describe how each of these four constraints can be represented as forces upon the robot kinematic hierarchy, so that they can be subsequently transformed into torques for use in my algorithm.

5.1.4.3.1 Point Constraints

Point constraints describe any situation where a 3-D point location in Cartesian space on the robot's body (hereafter denoted a robot point) must be held to another 3-D point location in world coordinates (hereafter denoted as world point). Both the robot and world points can be varying in time and space relative to the world origin, and they are specified in world coordinates. The world point can be a trajectory in Cartesian space. The most common application of point constraints is for specifying motion in terms of end-effector trajectories; for example to make the humanoid robot pick up an object or touch a certain location in the world.

Formulations to calculate forces for constraints exist that use a single point instead of two distinct points. Under ideal conditions (i.e. when the constraint is perfectly satisfied) both the world and robot point are equivalent, and under these conditions the two approaches converge. However, during interim periods when constraints are applied or released on real hardware, transient states occur while the hardware transitions rapidly

to maintaining the constraint. This is due to the limitations of real hardware implementations (e.g. motor torque limits). Thus, my approach that considers two points is more general and readily handles transient states, whereas when forces are calculated on a single point for the constraint, the robot is assumed to move infinitely fast with infinite available torque. With my two point formulation, three distinct constraint force contributions exist: (1) forces due to changing the commanded constraint (i.e. world forces), (2) forces due to changing the robot commanded position, velocity, acceleration (i.e. robot forces), and (3) forces to overcome positional difference between the two points (i.e. transient forces).

Let $x_{world}(t)$ be a given desired trajectory that pre-specified point on the robot's body, denoted $x_{robot,local}(t)$ in local coordinates, must follow for all time, t. $x_{robot,local}(t)$ is transformed from coordinates in kinematic hierarchy (i.e. local reference) into world coordinates, $x_{robot}(t)$, using forward kinematics at all time t. In forward kinematics, a series of rotation and translation matrices are used with the current (i.e. measured versions of commanded angles) joint angle positions of the robot to change domains. Since forward kinematics is a well-known algorithm, the details of this transformation can be found here [17].

Both the robot and world point trajectories are given here as functions of time. The robot point is a trajectory once it is specified in world coordinates because the robot may be performing a motion in addition to maintaining this constraint, and the world point need not be specified statically.

Through successive derivatives of the world point trajectory, $x_{world}(t)$, velocity and acceleration trajectories are formed. The second derivative represents an acceleration trajectory and the first derivative represents a velocity trajectory for the point constraint. From Equation 49, the force contributions from the given point constraint trajectory, $F_{world}(t)$ can be computed. Effective masses and inertias are used according to the operational space formulation [118].

$$F(t) = \Lambda(x(t)) * \ddot{x}(t) + \mu(x(t), \dot{x}(t)) + g(x(t)) \tag{49}$$

where,

$\mu(x(t), \dot{x}(t))$ = Centrifugal & Coriolis force vector

$g(x(t))$ = gravity force vector

$\Lambda(x(t))$ = kinetic energy matrix

$F(t)$ = 3-dimensional force vector in Cartesian space

$\ddot{x}(t)$ = time-varying acceleration trajectory in Cartesian space

$\dot{x}(t)$ = time-varying velocity trajectory in Cartesian space

$x(t)$ = time-varying position trajectory in Cartesian space

For the derivation and full specification of the matrices and vectors in Equation 49, see reference [46].

Similarly, velocity and acceleration trajectories are calculated through successive derivatives of the robot point trajectory, $x_{robot}(t)$; this can be accomplished by representing or modeling the robot hierarchy as a set of moving rigid bodies. The second derivative represents an acceleration trajectory and the first derivative represents a velocity trajectory for the robot body point. From Equation 49 and using the appropriate kinetic energy matrix, Centrifugal & Coriolis force vector, and gravity force vector, the force contributions from the given robot body point trajectory, $F_{robot}(t)$ can be computed.

The torque trajectory for one point constraint is then calculated from forces acting upon the robot body at the location of the robot body point, as in Equation 50.

$$\Gamma_{point}(t) = -J_{point}^{T}(F_{robot}(t) + F_{world}(t)) \tag{50}$$

where,

$\Gamma_{point}(t)$ = joint-space torques for the point constraint task as a vector

$F_{robot}(t)$ = time-varying force vector from robot internal motion in Cartesian space

$F_{world}(t)$ = time-varying force vector from specified point constraint trajectory in Cartesian space

J_{point} = Jacobian matrix defined by the robot architecture (including the constrained robot body location)

5.1.4.3.2 *Orientation Constraints*

Body part orientation constraints (as opposed to joint orientation constraints) are used in any situation where a part of the robot's body must be held at a specified angle with respect to a plane in the world. For example, orientation constraints are commonly used in for holding the palms of the robot's hands parallel to a table plane for grasping objects on the table.

Unlike point constraints, orientation constraints are constraints that represent the alignment of two planes: one on the robot body and one relative to the world. However, orientation constraints are very similar to point constraints because they can be satisfied by the coincidence of points in world coordinates (i.e. two pairs of points, with a distance of zero between the points in each pair).

The key to handling body part orientation constraints as kinematic constraints is to represent the two planes that need to be aligned by their respective unit normal vectors. Any vector (including normal vectors) in 3-D Cartesian space can be defined by one pair of points (i.e. vector head and tail). The base point of the world vector (i.e. the world vector tail) can translate to any point in space without loss of meaning, and therefore, to simplify calculations, at every time step, the tails of both vectors should coincide at the world location of the robot body vector tail. Since both vectors are unit normal vectors (i.e. same length), dimensionality of the problem of satisfying orientation constraints is halved by selecting the arbitrary base point of the world vector to always align with the robot vector tail (i.e. instead of aligning two pairs of points in Cartesian space, only vector head points need now be aligned).

Furthermore, the head of robot body vector is easily implemented as an imaginary extension to the robot kinematic hierarchy (i.e. one unit in the direction perpendicular to the robot plane). The location of the head of robot body vector can then be determined using the translation and rotation matrices in forward kinematics, which facilitates constraint calculations.

Neither the robot body vector nor the world vector need to be defined statically. Since

the tails of these vectors are placed at the same Cartesian location, by defining $x_{world}(t)$ to be the point corresponding to the head of the world vector and $x_{robot}(t)$ to be the point corresponding to the head of the robot body vector (both in world coordinates), the calculations found under Section 5.1.4.3.1 can be reused to find the torque trajectory for orientation constraints. However, the Jacobian needs to be extended to include the imaginary unit robot body vector on the kinematic hierarchy since the head of this vector is the point of application of the forces.

For example, consider placing an orientation constraint on the palm so that it always points directly down in the world (i.e. $x_{world}(t)$ would always be one unit beneath the hand centroid, regardless of what configuration or orientation the hand was in, as the robot moves). The $x_{robot}(t)$ point would be rotating about the hand centroid as the arm, wrist, elbow, and body DOFs change commanded position. The Jacobian in Equation 50 is extended to include the imaginary $x_{robot}(t)$ as part of the kinematic hierarchy. By aligning the tails of both the plane normal vectors, the generalized forces calculated in Equation 50 would be computing the forces required to apply at the point $x_{robot}(t)$ onto the robot body to align the $x_{robot}(t)$ and $x_{world}(t)$ points, which also aligns the two requisite planes in the orientation constraint.

5.1.4.3.3 Point-at Constraints

Point-at constraints are used in any situation where the social robot wants to point with its finger at a 3-D location in the world. They are a collinearity constraint for three points (two on the robot finger and one in the world). The two points on the robot index finger are located at the fingertip and at the proximal knuckle joint, so the index finger can be extended in a straight line.

The key to handling point-at constraints as kinematic constraints is to constrain the knuckle motors of the fingers (especially the index finger) so that the hand remains always in a pointing pose. In this configuration, two of the three necessary points always remain collinear.

For point-at constraints, the kinematic hierarchy is augmented with an imaginary segment extending from the tip of the finger. This imaginary segment always points in the direction collinear with the line created by the two index finger locations (fingertip and proximal knuckle). The length of this imaginary kinematic extension (calculated by Equation 51) is time-varying since the robot fingertip and/or point-at location in the world can also be time-varying.

$$Dist(t) = \sqrt{(x_{pointAt}(t) - x_{fingertip}(t))^T (x_{pointAt}(t) - x_{fingertip}(t))} \tag{51}$$

where,

$x_{fingertip}(t)$ = Cartesian coordinates of the robot's fingertip in vector form

$x_{pointAt}(t)$ = Cartesian coordinates of the world point to point-at in vector form

Since both the robot body and the point-at location can be changing position with time, by defining $x_{world}(t)$ to be $x_{pointAt}(t)$and $x_{robot}(t)$ to be $x_{fingertip}(t)$, the calculations found under Section 5.1.4.3.1 can be reused to find the torque trajectory for point-at constraints. However, the Jacobian needs to be extended to include the imaginary fingertip vector on the kinematic hierarchy as if it were a real and rigid segment, since the forces are applied to this imaginary fingertip extension at a point of distance $Dist(t)$ (calculated by Equation 51 with the variable substitutions mentioned above).

5.1.4.3.4 Look-at Constraints

Look-at constraints are used when the social robot with a humanoid head and eyes (or cameras) needs to look at a 3-D location in the world. They are a collinearity constraint for three points per eye (two on each robot eye and one in the world). The two points on each robot eye are located at the centroid of the eye and at the center of the pupil. For the following discussion, constraints for only one eye will be discussed. However, the technique applies to both left and right eyes.

Two of the three necessary points always remain collinear as long as the robot eyes are

able to rotate about the centroid of the eyes. For look-at constraints, the kinematic hierarchy is augmented with an imaginary segment extending from the center of the pupil. This imaginary segment always points in the direction collinear with the line created by the two eye locations (centroid and pupil center). The length of this imaginary kinematic extension is calculated by Equation 51, where the variable $x_{fingertip}(t)$ is replaced by $x_{pupil}(t)$, and the variable $x_{pointAt}(t)$ is replaced by $x_{lookAt}(t)$. The length, $Dist(t)$ is time-varying since the robot pupil can be moving and/or look-at location in the world can be time-varying.

Since both the robot body and the look-at location can be changing position with time, by defining $x_{world}(t)$ to be $x_{lookAt}(t)$ and $x_{robot}(t)$ to be $x_{pupil}(t)$, the calculations found under Section 5.1.4.3.1 can be reused to find the torque trajectory for look-at constraints. However, the Jacobian needs to be extended to include the imaginary eye extension vectors on the kinematic hierarchy as if they were real and rigid segments, since the forces are applied to these imaginary eye extensions at a point of distance $Dist(t)$ (calculated by Equation 51 with the variable substitutions mentioned above).

5.1.4.3.5 Algorithm - External

The initial point for the algorithm that handles external constraints is the trajectory of time-varying forces represented as the force summation on the right-hand side of Equation 49, $F_{robot}(t) + F_{world}(t)$. The appropriate variable substitutions were discussed and derived for point, orientation, point-at, and look-at constraints under their respective sections previously so that the developed framework would be generic and applicable to a variety of kinematic constraints useful for social robots.

First, one external constraint will be considered in the presence of a lower priority torque trajectory (e.g. a time-varying posture constraint). For the following discussion, assume that this trajectory been given in joint space, represented by the trajectory of poses, $q(t)$. In Section 5.1.4.4, the framework will be extended to demonstrate how to compose multiple constraints, so that a robot can perform many simultaneous tasks while concurrently respecting many constraints.

Since every external constraint that was discussed previously involved the calculation

of forces on two points, for simplicity the quantity $F_{task}(t)$ will be defined in Equation 52 as this summation of forces. The same appropriate variable substitutions as defined above allow it to be applicable to any of the four external constraints.

$$F_{task}(t) = F_{robot}(t) + F_{world}(t) \tag{52}$$

where,

$F_{world}(t)$ = Cartesian space time-varying forces on one point for a external constraint

$F_{task}(t)$ = time-varying constraint forces in Cartesian space

$F_{robot}(t)$ = time-varying forces on the other of the two points for a external constraint in Cartesian space

Applying the Jacobian for a given task using the generic $F_{task}(t)$ on the right-hand side in Equation 50 , $\Gamma_{task}(t)$ will be used to represent a generic constraint's time-varying torque trajectory. The Jacobian of the constraint is defined by Equation 53.

$$J(t) = \frac{\partial x_{robot}(t)}{\partial q(t)} \tag{53}$$

where,

$x_{robot}(t)$ = time-varying Cartesian space vector of the constraint point associated with the robot kinematic hierarchy

$q(t)$ = time-varying vector of degrees-of-freedom representing a lower priority joint-space motion

Based upon a formulation called operational space control for controlling robotic agents to achieve simultaneous tasks [119] (Section 2.1) at each iteration, I define a projection matrix, $P(t)$, that maps the torque for the lower priority motion, $\Gamma_{lower}(t)$, to the appropriate control torques that do not interfere with the given external constraint.

$$P(t) = I - J(t)^T J_{dcgi}(t)^T \tag{54}$$

where,

$J_{dcgi}(t)$ = dynamically consistent generalized pseudo-inverse of the Jacobian

$J(t)$ = the Jacobian as defined by Equation 53

I = appropriately sized identity matrix

In Equation 54, $J_{dcgi}(t)$ is one of the many pseudo-inverse matrices of $J(t)$. In the operational space control formulation, $J_{dcgi}(t)$ is selected to be the "dynamically consistent generalized inverse," as in Equation 55, according to [119].

$$J_{dcgi}(t)^T = \Lambda_t J(t) M_t^{-1} \tag{55}$$

where,

Λ_t = Cartesian space inertia matrix

M_t = joint space inertia matrix

By applying the projection matrix $P(t)$ to a torque vector, it removes the components in the space spanned by the columns of $J_{dcgi}(t)$, where the torque will directly affect the constraint. Thus, to maintain *one* constraint in the presence of lower priority motion, the final applied torque will be $\Gamma_{apply}(t) = \Gamma_{task}(t) + P(t)\Gamma_{lower}(t)$.

5.1.4.4 Composing Multiple Constraints

Multiple constraints may be composed on the robot hardware by establishing a prioritized order for all the tasks the robot must perform simultaneously. Each lower priority task or constraint is projected into the null space of the higher priority tasks, using the projection defined explicitly for the specific higher priority tasks and constraints. Each projection assumes the form in Equation 54. However, the Jacobian is formed using the uniquely defined $x_{robot}(t)$ for each constraint and task. The unique Jacobian will tailor $J_{dcgi}(t)$ so that dynamics of the task are respected. Thus for each external constraint defined by a unique $x_{robot}(t)$, there is an associated unique $P(t)$.

The following example is the easiest way to illustrate the approach. In the following example, the numerical subscripts denote the priority of the task (1 is the highest priority, assigned to the task that is most important to maintain). For example, J_1 is the Jacobian for task with priority 1, as calculated by Equation 53; $F_2(t)$ would denote the force trajectory

for task with priority 2, as calculated by Equation 52; $P_3(t)$ would designate the projection matrix for task with priority 3, calculated by Equation 54, using the corresponding $J_3(t)$.

1. For the highest priority constraint or task (i.e. number 1), the null space is equivalent to the entire torque space of the robot hierarchy. Thus, task number one does not need to respect any other forces on the robot body or any other torque trajectories (i.e. no projection exists for the highest priority task). The torque to apply to robot actuators to maintain only task one is $\Gamma_{apply}(t) = \Gamma_1(t) = {J_1}^T(t)F_1(t)$.

2. The torque trajectory for the second highest priority constraint or task (i.e. $\Gamma_2(t) = {J_2}^T(t)F_2(t)$), cannot disrupt any of the torques from task 1. Therefore, components along these coordinates are removed from the task before its application to actuators through the use of the projection matrix from the higher priority task (i.e. $\Gamma_2(t) = P_1(t){J_2}^T(t)F_2(t)$). Then the torque to apply is the sum of both torque trajectories (i.e. $\Gamma_{apply}(t) = \Gamma_1(t) + \Gamma_2(t)$).

3. Considering just one more task so the trend becomes apparent, the third-highest priority task needs to respect both task 1 and task 2. Components of this trajectory along all of these coordinates need to be removed before application to ensure that the higher priority tasks are respected (i.e. $\Gamma_3(t) = P_2(t)P_1(t){J_3}^T(t)F_3(t)$). Then the torque to apply is the sum of both torque trajectories (i.e. $\Gamma_{apply}(t) = \Gamma_1(t) + \Gamma_2(t) + \Gamma_3(t)$).

4. Generically, this pattern is defined for a task of n^{th} priority as $\Gamma_n(t) = (\prod_{i=0}^{i=n-1} P_i(t))$ ${J_n}^T(t)F_n(t)$. Under this formulation, $P_0(t) = I$, an appropriately sized identity matrix.

5. After the appropriate number of applied tasks, the lowest two tasks are always motion composed from internal constraints and variance (if they exist). Assume that $\Gamma_{internal}(t)$ represents this former trajectory. If $q(t)$, a joint position trajectory, is

given, the corresponding torque trajectory, $\Gamma_{internal}(t) = -K_t x_t$, by following the procedure described in Section 5.1.4.2. From Section 4.2, the torque from the variance-generating algorithm is defined as $u_t = -K_t \Delta x_t$. Thus, these two trajectories are composed in exactly the same way all other tasks are composed. However, no Jacobian exists for either of these two tasks. Let the variable ϵ represent the index for these two lowest priority constraints, which can be composed simultaneously, since they are not external constraints. Therefore, $\Gamma_\epsilon(t) = (P_{\epsilon-1}(t)P_{\epsilon-2}(t)\cdots P_1(t))(-K_t\Delta x_t + \Gamma_{internal}(t))$.

5.1.4.5 Constraint Transients

The existing formulation of operational space control in the literature [118] does not handle transient performance when external constraints are applied and released. The algorithm is torque-based and designed to move as quickly as robot actuators will allow on real hardware to reach constraints when applied or return to a posture when constraints are released. Due to high torques, it is unrealistic and dangerous to satisfy constraints as fast as the hardware is capable. A more principled, fluid, and elegant approach is required.

To avoid burnout and unnecessary replacement of actuators, filtering of torque application when constraints are applied and released is one useful option. If filtering is used, feedback should be implemented between the hardware and the higher level control system to communicate when constraints are reached in real-world applications. Such feedback might be necessary for subsequent action. For example, the robot needs to close its hand (subsequent action) after the hand reaches (constraint transient) a target (point constraint) to grab an object.

However, to eliminate the need for communication and guarantee safety, I implemented a technique to handle constraint transients. Flash & Hogan are famous for their use of bell-shaped velocity profiles based on human reaching experiments [93]. These profiles are biologically inspired and ensure smooth transition to an external constraint target.

At the time instant when an external constraint is applied, the maximum velocity (or amount of transition time to the constraint), the current $x_{Robot}(t)$, desired $x_{World}(t)$ (desired

end constraint point from Section 5.1.4.3.1), and the symmetric bell-shaped speed profile are used to define a force profile to reach constraint. Knowing the two points in Cartesian space that correspond to the endpoints of the constraint transients, the profile uniquely defines the force trajectory when either the maximum velocity or amount of transition time is given [93]. The constraint will be satisfied at the end of the transition period.

Until release, the standard formulation of OSC with an augmented Jacobian and hierarchy (i.e. augment so it can handle all external constraints as point constraints) are used to keep look-at, point-at, or orientation constraint points satisfied. Upon release, another symmetric bell-shaped velocity profile and maximum velocity (or time-to-target) are used to command the robot the desired number of discrete time increments to reach a posture without the released constraint.

5.1.5 Hypotheses

In general, my hypotheses are constructed to test that the two algorithms for internal and external constraints preserve the constraints specified. Subsequent experiments will test these hypotheses individually, and then a few combinations of constraints are tested to demonstrate composite constraint performance.

- H1: My external constraint algorithm maintains point, orientation, look-at, and point-at constraints to a level of less than 1% error in deviation from the constraint when executed as the highest priority constraint.

- H2: My internal constraint algorithm maintains position, velocity, and torque constraints to a level better than the smallest angular measurement tolerance on the robot hardware.

- H3: External constraints closer in proximity on the robot architecture will exhibit more error in the lower priority constraint than for external constraints further apart on the robot architecture. Distance between constraints measured through connecting robot link lengths only (i.e. in robot-relative hierarchy space, not in Cartesian world space).

- H4: As constraint priority decreases, error in the maintenance of the constraint increases. This effect is due to a smaller null space, and is more pronounced as the number of simultaneous constraints increases.

5.1.6 Experiment 1: Point Constraints

The following quantitative experiment tested the operational space framework for satisfying point constraints in the presence of lower priority motion. Experiment 1 quantified the maximum performance of point constraints using the framework, since point constraints were implemented as the highest priority task, with a single subtask (i.e. a posture constraint) for the experiment.

5.1.6.1 Experimental Design

To test H1, 400 random point constraints were selected in the reachable space for the robot that did not collide with the robot body. Half the points were tested with each hand centroid (200 with the left hand; 200 with the right hand). All 400 point constraints were tested in both simulation and on the robot hardware. The 400 point constraints occurred at different times throughout the lower priority posture constraint to add variability into the experiment and verify that constraint satisfaction is not specific to one posture. Five different lower priority motions (i.e. five different posture constraints) were used. They were gestures of shrugging (symmetric), right-arm pointing, left-arm waving, right-arm stop, and two-handed bowing (symmetric). Since the experiment was performed using three different methods, a specific point constraint location was paired to the same lower priority posture constraint for all 400 constraints to ensure data consistency.

In addition to the results presented from simulation of the algorithm, the measurements of the point constraints' actual locations in the real-world were performed in two ways: (1) capturing the measured joint angles from robot sensors and calculating the point location with forward kinematics and (2) physical measurement (all measurements by one person). To increase confidence, each measurement of the actual position of the hand centroid (i.e. joint angles or distance between hand centroid and constraint location) was taken three times, and all 1,200 measurements were taken in randomized order.

In simulation, the 3-D world locations of the hand centroid and the position constraint are known exactly. Therefore, the distance between them was precisely and easily measured with appropriate timing. Since the constraint technique requires application of torques to the robot actuators, the simulation used was a dynamic simulation developed using Open Dynamics Engine (see Section 2.5).

5.1.6.2 Results

Table 45 shows that simulation was the most accurate measurement method since everything is calculated under ideal conditions. Furthermore, Table 45 shows that symmetric motions tend to have similar error for both arms (i.e. right arm vs. left arm). In the posture constraints that were asymmetric, the arm with less dynamics in lower priority motion had slightly less error. This means that the calculation that projects and eliminates disruption in the null space is not perfect; This is most likely due to inertia matrix errors (actual hardware vs. that used in calculations). This asymmetric error was more pronounced when there was more velocity or acceleration in joint space because the inertia matrix has a more significant impact when motion dynamics are larger.

Table 45: Average distance measured from random point constraints in reachable space (i.e. point constraint error) for five different posture constraints (i.e. lower priority motions). Averages taken over 400 measurements, measured three times each, in three different ways (Sim = simulation; Joint = hand centroid calculated from measured actual robot joint angles; Human = measured by human). Distance in centimeters. Standard deviations given in parentheses.

	Left Hand Centroid			Right Hand Centroid		
Motion	Sim	Joint	Human	Sim	Joint	Human
Bow	0.039 (.009)	0.104 (.034)	3.36 (1.68)	0.023 (.005)	0.093 (.031)	2.54 (1.27)
Point	0.174 (.043)	0.202 (.067)	1.65 (0.83)	0.213 (.053)	0.289 (.095)	2.77 (1.38)
Shrug	0.044 (.011)	0.089 (.029)	4.06 (2.01)	0.027 (.007)	0.056 (.016)	3.01 (1.49)
Stop	0.224 (.056)	0.354 (.115)	2.17 (1.05)	0.252 (.063)	0.376 (.124)	2.55 (1.24)
Wave	0.314 (.078)	0.478 (.150)	2.98 (1.42)	0.263 (.063)	0.413 (.133)	2.81 (1.39)
Avg.	0.159 (.121)	0.245 (.170)	2.84 (1.02)	0.156 (.126)	0.245 (.172)	2.74 (0.51)

Table 45 allows me to conclude that point constraints using the operational space framework as developed in Section 5.1.4.3.1 are maintained at less than 3 centimeters (for all different measurement techniques) from the desired constraint position on average in the presence of a variety of common social robot tasks. Although no explicit percent error can be calculated for the distances measured, 3 centimeters provides evidence in support of H1, since it quantifies maximum deviation for point constraints given the different measurement techniques. Very low error was expected for point constraints because the operational space framework has been in existence for over 30 years to satisfy task constraints with lower priority posture constraints.

5.1.7 Experiment 2: Orientation Constraints

The following quantitative experiment tested the operational space framework for satisfying orientation constraints in the presence of lower priority motion. Experiment 2 quantified the maximum performance of orientation constraints using the framework, since orientation constraints were implemented as the highest priority task, with a single subtask (i.e. a posture constraint) in the experiment.

5.1.7.1 Experimental Design

To test H1, 1,400 total orientation constraints were generated. 100 orientation constraints per body part were randomly selected from all possible vector directions in Cartesian space for 14 different right or left arm body parts and applied to the centroid of the corresponding body part. All 1,400 orientation constraints were tested on both simulation and on the robot hardware, and measurements were taken to determine the error between the commanded and actual constraint. Each measurement of the actual position of the DOF angle was taken three times, and all 4,200 measurements were taken in randomized order. Only one orientation constraint was imposed upon the robot architecture at any given instant in time. The 1,400 orientation constraints occurred at different times throughout the lower priority posture constraint to add variability into the experiment and verify that constraint satisfaction is not specific to one posture. Five different lower priority motions (i.e. five different posture constraints) were used. They were gestures

of shrugging (symmetric), right-arm pointing, left-arm waving, right-arm stop, and two-handed bowing (symmetric). A specific orientation constraint was paired to the same lower priority posture constraint on both the real hardware and the simulation for all constraints to ensure data consistency.

In simulation, the joint angles that satisfy the constraint are known exactly. The same dynamic model used for Experiment 1 was used also for Experiment 2 to ensure model kinematic and dynamic accuracy. In the dynamic simulation, two invisible rigid bodies that represented the position of head of the robot body vector and the head of the world vector were added. Therefore, the error between commanded constraint and simulated value was precisely and easily calculated via distance between these two bodies with appropriate timing.

The measurements of the orientation constraints' actual locations on the hardware were captured by measuring the joint angles from robot sensors. Since there was no way to easily instrument the robot measure angles in Cartesian space quickly, the actual joint angles on the hardware were measured instead. These joint angles were tracked in a simulation that was augmented with the two points that represented head of the robot body vector and the head of the world vector. Then error was calculated as the difference between these two points using the measured data instead of simulated data for joint angles that satisfy the constraint.

Prior to running Experiment 2, all orientation constraints were simulated to ensure that each constraint could be met without robot self-collision. Any constraints where the robot collided with itself in order to satisfy the constraint were excluded, and a new constraint was selected in place of the one that caused self-collision.

5.1.7.2 Results

In order to test orientation constraints applied to all parts of the robot (and not just end-effector orientation constraints), the robot body *in simulation* was augmented with robot body vectors at the centroid of every rigid body on the left and right arms.

Recall that orientation constraints are satisfied by matching the head of the robot body

Table 46: Average *simulated* error between commanded orientation constraints and actual values for five different posture constraints (i.e. lower priority motions). Averages taken over 100 measurements per DOF, measured three times each. Error in degrees multiplied by 1000 (x1000). Standard deviations given in parentheses multiplied by 1000 (x1000). (l = left arm; r = right arm; s = shoulder DOF; e = elbow DOF; w = wrist DOF)

DOF	Bow	Point	Shrug	Stop	Wave	Average
leX	15 (5)	3 (3)	2 (5)	7 (5)	13 (6)	8 (8)
lsY	31 (9)	20 (5)	13 (4)	18 (5)	24 (6)	21 (10)
lsZ	47 (14)	32 (9)	21 (6)	29 (8)	35 (9)	33 (16)
leX	65 (19)	37 (11)	27 (8)	32 (9)	39 (12)	40 (21)
lwY	72 (20)	53 (15)	62 (18)	55 (16)	51 (17)	59 (22)
lwX	79 (23)	71 (23)	84 (12)	62 (19)	77 (17)	75 (21)
lwZ	86 (23)	81 (24)	95 (22)	81 (20)	84 (21)	85 (25)
rsX	28 (8)	15 (4)	9 (2)	15 (4)	17 (5)	17 (9)
rsY	35 (11)	23 (6)	19 (6)	23 (6)	28 (8)	25 (13)
rsZ	55 (16)	34 (10)	22 (6)	30 (8)	36 (10)	36 (17)
reX	68 (12)	47 (14)	46 (13)	32 (10)	40 (12)	47 (15)
rwY	72 (20)	66 (19)	68 (20)	58 (17)	54 (16)	64 (21)
rwX	86 (22)	83 (24)	71 (21)	77 (23)	65 (22)	76 (24)
rwZ	85 (25)	84 (25)	72 (21)	76 (22)	70 (19)	77 (27)

vector (i.e. the unit normal of a plane attached to the robot kinematic hierarchy) with the head of the unit normal vector for the constraint. For Experiment 2, the tails of both of these vectors were placed at the centroid of a rigid body on either the left or right arm of the robot in the simulation. In the subsequent tables of results, data is listed by joint angle. This joint angle uniquely identifies the rigid body centroid from which the two vector tails extended. Since one and only one rigid body exists between each DOF on the robot arms, the rigid body for a given DOF corresponds to the rigid body nearest to the DOF, but further away from the kinematic root (i.e. closer to the end-effector). For example, for DOF shoulder Z, the hierarchy was augmented at the centroid of the bicep rigid body. And for all constraints listed under DOF shoulder Z in the tables, the robot body vector at the bicep was constrained to the world vector. Similarly, for DOF wrist Z, the hierarchy was augmented at the centroid of the hand rigid body. Most of these orientation constraints have little practical value and would rarely be used in practice. However for the sake of thorough testing, they are included in the subsequent tables.

Table 47: Average *measured* error between commanded orientation constraints and actual values for five different posture constraints (i.e. lower priority motions). Averages taken over 100 measurements per DOF, measured three times each. Error in degrees multiplied by 1000 (x1000). Standard deviations given in parentheses multiplied by 1000 (x1000). (l = left arm; r = right arm; s = shoulder DOF; e = elbow DOF; w = wrist DOF)

DOF	Bow	Point	Shrug	Stop	Wave	Average
leX	44 (15)	41 (13)	47 (15)	37 (17)	40 (16)	42 (22)
lsY	48 (18)	44 (14)	53 (14)	48 (15)	44 (11)	47 (21)
lsZ	47 (19)	45 (18)	51 (16)	49 (17)	51 (12)	49 (23)
leX	51 (19)	51 (18)	57 (18)	52 (17)	49 (16)	52 (25)
lwY	52 (20)	57 (19)	62 (18)	55 (16)	52 (14)	56 (23)
lwX	56 (19)	60 (17)	60 (17)	66 (15)	67 (15)	62 (21)
lwZ	58 (22)	65 (24)	61 (21)	70 (17)	63 (17)	64 (20)
rsX	41 (18)	45 (24)	49 (12)	50 (19)	47 (15)	37 (22)
rsY	39 (16)	43 (16)	41 (17)	48 (18)	48 (19)	45 (19)
rsZ	44 (12)	54 (19)	39 (18)	43 (14)	46 (21)	46 (26)
reX	45 (11)	44 (18)	41 (19)	48 (12)	50 (18)	41 (22)
rwY	49 (24)	46 (19)	58 (20)	50 (13)	60 (17)	54 (22)
rwX	50 (21)	52 (21)	51 (21)	47 (13)	65 (21)	56 (29)
rwZ	55 (20)	51 (23)	50 (26)	54 (22)	69 (20)	60 (30)

Table 46 compared to Table 47 shows that on average orientation constraint error for the simulation is smaller than measured, as was expected due to the ideal conditions of the simulation. In Table 46 orientation constraint error slightly increases in the DOFs toward the end effector. However, this same effect was not evident for the constraint angles measured on the hardware. The reason this effect occurred in simulation is that error in the values for the mass and inertia matrices can stack up as calculation includes more of the kinematic chain. However, when using measured joint angle values from the hardware to calculate the locations of the head of the robot body vector, the simulation is used for tracking only, and this small trend disappears.

Using Equation 56 to calculate the percent error for the orientation constraints averaged over all degrees-of-freedom and all gestures, the average simulated percent error for orientation constraints was 0.42% and average percent error measured from the hardware was 0.53%, both of which are consistent with H1. For reasons mentioned previously (i.e. inertia tensor accuracy), simulated percent error was lower closer to the root DOF and

measured joint angles lead to lower percent error in orientation constraints applied closer to the end-effectors. This result is advantageous because typically orientation constraints will be used at the end-effectors on hardware, which is where they are most accurate.

$$Error = \frac{|commanded - measured|}{commanded} \times 100 \tag{56}$$

5.1.8 Experiment 3: Point-At Constraints

The following quantitative experiment tested the operational space framework for satisfying point-at constraints in the presence of lower priority motion. Experiment 3 quantified the maximum performance of point-at constraints using the framework, since point-at constraints were implemented as the highest priority task in Experiment 3, with a single subtask (i.e. a posture constraint).

5.1.8.1 Experimental Design

To test H1 for point-at constraints, 400 point-at constraints were randomly selected from all 3-D locations in Cartesian space not occupied by the robot body volume. Each point-at constraint was satisfied with both arms (but not simultaneously) to force the robot to turn when necessary to satisfy the constraint. The only exceptions for when a constraint was applied to a single arm were either when applying that constraint to a particular arm caused self-collision or body penetration.

Prior to running Experiment 3, all point-at constraints were simulated to ensure that the constraint could be met without robot self-collision. Any constraints where the robot collided with itself in order to satisfy the constraint using a particular were excluded for that particular arm, and a new constraint was selected in place of the one that caused self-collision. A new constraint was also selected for all point-at locations that penetrated the robot body volume during motion (i.e. a constraint was specified that was inside the volume occupied by the robot body at the time of constraint application).

At any given instant in time, only the right arm or left arm was constrained because the criteria of the experiment dictated that single constraints be tested in the presence of

only a posture constraint. Applying two simultaneous point-at constraints would violate this criteria. All 400 point-at constraints were tested on both simulation and on the robot hardware, and measurements were taken to determine the error between the commanded and actual constraint. Each measurement of the actual constraint position was taken three times, and all 1,200 measurements per arm were taken in randomized order. The 400 point-at constraints occurred at different times throughout the lower priority posture constraint to add variability into the experiment and verify that constraint satisfaction is not specific to one posture. Five different lower priority motions (i.e. five different posture constraints) were used. They were gestures of shrugging (symmetric), right-arm pointing, left-arm waving, right-arm stop, and two-handed bowing (symmetric). A specific point-at constraint was paired to the same lower priority posture constraint on both the real hardware and the simulation for all constraints to ensure data consistency.

The same dynamic model used for Experiment 1 was used again for Experiment 3 to ensure model kinematic and dynamic accuracy. In the dynamic simulation, two invisible rigid bodies that represented the position the robot is pointing at and the constraint position were added. Therefore, the error between commanded constraint and simulated value were precisely and easily calculated via distance between these two bodies with appropriate timing.

To simplify and increase the speed of measurement for 3-D pointing location in the real-world, small targets were created with concentric rings of radial distance pre-written on them. Since the 3-D locations in the world were known in advance, the targets were placed at the locations in advance. A small laser pointer was fixed to the index finger on each of the robot's right and left hands.

5.1.8.2 Results

Table 48 shows that on average point-at constraint error for the simulation was smaller than measured on the hardware by humans, as was expected due to the ideal conditions of the simulation. The measurement scheme is not ideal for human measurement of point-at constraints on the hardware, but the results are accurately satisfied on the hardware.

Table 48: Average distance measured from random point-at constraints (i.e. point-at constraint error) for five different posture constraints (i.e. lower priority motions). Averages taken over 400 measurements per hand, measured three times each, in two different ways (Sim = simulation; Human = human measured. Distance in centimeters. Standard deviations given in parentheses.

Motion	Left Sim	Left Human	Right Sim	Right Human
Bow	0.136 (0.064)	1.69 (0.87)	0.149 (0.051)	2.14 (1.04)
Point	0.197 (0.081)	2.09 (1.12)	0.177 (0.033)	1.71 (1.30)
Shrug	0.223 (0.060)	1.99 (1.01)	0.184 (0.090)	1.66 (1.49)
Stop	0.157 (0.049)	1.71 (1.54)	0.232 (0.047)	1.59 (1.71)
Wave	0.179 (0.052)	1.42 (1.33)	0.192 (0.056)	1.92 (0.88)
Average	0.178 (0.085)	1.78 (1.63)	0.187 (0.094)	1.80 (1.24)

The results demonstrate that the point-at constraints on the hardware were accurate to within less than 2.15 centimeters for the robot pointing.

Although no explicit percent error can be calculated for the point-at constraints measured (i.e. error would be a function of the reference point), 2.15 centimeters provides evidence in support of H1, since it quantifies maximum deviation for point constraints given the different measurement techniques. Considering the ultimate goal of a point-at constraint, which is to add the functionality for the social robot to indicate location of something in the world, accuracy within 2.15 centimeters is sufficient for human partners to discern location or which object the robot is pointing toward.

5.1.9 Experiment 4: Look-At Constraints

The following quantitative experiment tested the operational space control framework for satisfying look-at constraints in the presence of lower priority motion. Experiment 4 quantified the maximum performance of look-at constraints using the framework, since look-at constraints were implemented as the highest priority task in Experiment 4, with a single subtask (i.e. a posture constraint).

5.1.9.1 *Experimental Design*

To test H1 for look-at constraints, 500 look-at constraints were randomly selected from all 3-D locations in Cartesian space not occupied by the robot body volume. Each look-at constraint was satisfied with both eyes, since single eye constraint satisfaction is not common in HRI.

Prior to running Experiment 4, all look-at constraints were simulated to ensure that the constraint could be met without robot self-collision. Any constraints where the robot collided with itself in order to satisfy the constraint using a particular arm were excluded for that particular arm, and a new constraint was selected in place of the one that caused self-collision. A new constraint was also selected for all look-at locations that penetrated the robot body volume during motion (i.e. a constraint was specified that was inside the volume occupied by the robot body at the time of constraint application). Furthermore, any look-at constraints outside the visible range were also discarded (e.g. behind the robot's head).

All 500 look-at constraints were tested on both simulation and on the robot hardware, and measurements were taken to determine the error between the commanded and actual constraint. Each measurement of the actual position of the look-at constraint was taken three times, and all 1,500 measurements were taken in randomized order. The 500 look-at constraints occurred at different times throughout the lower priority posture constraints to add variability into the experiment and verify that constraint satisfaction is not specific to one posture. Five different lower priority motions (i.e. five different posture constraints) were used. They were gestures of shrugging (symmetric), right-arm pointing, left-arm waving, right-arm stop, and two-handed bowing (symmetric). A specific look-at constraint was paired to the same lower priority posture constraint on both the real hardware and the simulation for all constraints to ensure data consistency.

The same dynamic model used for Experiment 1 was used again for Experiment 4 to ensure model kinematic and dynamic accuracy. In the dynamic simulation, two invisible rigid bodies that represented the position the robot is looking at and the constraint position

Table 49: Average distance measured from the specified look-at constraints (i.e. look-at constraint error) for five different posture constraints (i.e. lower priority motions). Averages taken over 500 measurements, measured three times each, in two different ways (Sim = Simulation; HW = Hardware. Simulation distance in centimeters. Hardware distance in pixels. Standard deviations given in parentheses.

Motion	Left Sim	Left HW	Right Sim	Right HW
Bow	0.147 (0.189)	51.9 (23.7)	0.176 (0.161)	46.5 (27.1)
Point	0.188 (0.154)	47.3 (28.2)	0.164 (0.135)	44.3 (23.2)
Shrug	0.202 (0.127)	27.5 (15.7)	0.221 (0.190)	35.4 (19.0)
Stop	0.195 (0.132)	38.6 (25.4)	0.264 (0.203)	40.1 (22.9)
Wave	0.276 (0.198)	41.3 (23.3)	0.199 (0.187)	42.2 (28.6)
Average	0.202 (0.235)	41.3 (33.6)	0.205 (0.241)	41.7 (31.2)

are added. Therefore, the error between commanded constraint and simulated value was precisely and easily calculated via distance between these two bodies with appropriate timing.

To simplify and increase the speed of measurement for look-at location in the real-world, the robot eye cameras were used. Additionally, this helps test whether the robot was actually looking at the desired constraint location. Since the 3-D locations in the world were known in advance, steel bearings were placed at the constrained world locations in advance. The camera image for eyes at the moment corresponding to the constraint timing was saved for each constraint during the experiment execution. Then, distances for the constraints were measured on the hardware in units of pixels as calculated by the difference between the center of the image and the 2-D center of the ball bearing in the captured image. Each eye camera is 640 x 480 pixels.

5.1.9.2 Results

Table 49 does not allow comparison of average look-at constraint error for the simulation and the hardware, since the units are different. However, on average the location the robot was commanded to look-at on the hardware was approximately 6.5% of the camera image distance away from the center of the image for both eyes. This supports H1 because the robot will always be looking at the commanded constrained look-at location in 3-D space.

From a functional standpoint, as long as the full object remains in the camera image, the look-at constraint is successfully maintained. The alternative measure of success, wherein the *centroid* of the object must remain in the camera image, was used instead in Experiment 4. The functional goal of measuring if the entire *object* being looked-at was within camera image was a goal that was not tested in the Experiment 4, since small steel bearings were used in place of real objects that social robots frequently interact with. Steel bearings were used in place of common social robot objects because their small size better approximate points in Cartesian space. Using objects that better approximated points in Cartesian space facilitated the measurement process for look-at constraint error. The data in Table 49 shows that the centroid of the object being looked-at was, on average, approximately 200 millimeters from the direct line of sight of the constrained eyes or 41 pixels from the centroid of the camera.

5.1.10 Experiment 5: Two Commonly Combined Constraints

The following quantitative experiment tested the operational space framework for satisfying orientation and position constraints in the presence of lower priority motion (i.e. a posture constraint) simultaneously. Experiment 5 quantified the maximum performance of these simultaneous constraints using the framework. These two specific constraints were selected because grasping, a very common social robot task, was accomplished through simultaneously constraining the robot hand with one point constraint and at least one orientation constraint in the experiment.

5.1.10.1 Experimental Design

To test H1, 400 random point and orientation constraint pairs were selected in the reachable space for the robot in locations that did not collide with the robot body. Half the constraints were tested with each hand centroid (200 with the left hand; 200 with the right hand). All 400 constraint pairs were measured in the simulation and on the robot hardware.

The measurements of the point constraints actual locations in the real-world were performed in two ways: (1) capturing the measured joint angles from robot sensors and calculating with forward kinematics and (2) physical measurement by one person. In the

latter case, since the random point constraint positions were known before the experiment began, steel bearings were placed at the pre-measured world constraint locations to facilitate human measurement.

For Experiment 5, the constraint pairs were applied for the entire duration of a lower priority motion. Since the framework takes a finite time to reach a constraint, to measure only steady state performance of satisfying constraints, each lower priority motion was executed serially three times with smooth transition between runs. During the first execution, the constraint was applied, and the remainder of motion timing was allotted for the hand to reach both of its constraints. During the second serial execution of the motion, all the actual hardware joint angles and a human recorded maximum distance between the hand centroid and the steel bearing during the entire trajectory (instead of instantaneous distance at a predefined point in time as in Experiment 1) were measured for the entire duration of the second execution. For Experiment 5, the same person performed all measurements. During the third serial motion execution, the constraints were released to allow sufficient time to set up for the next constraint pair.

Five different lower priority motions (i.e. five different posture constraints) were used. They were gestures of shrugging (symmetric), right-arm pointing, left-arm waving, right-arm stop, and two-handed bowing (symmetric). A specific point constraint location was paired to the same lower priority posture constraint in both the real hardware and the simulation for all constraints to ensure data consistency. Each of the five different posture constraints were different lengths of time.

All measurements for the hand constraints (i.e. joint angles or distance between hand centroid and constraint location) were taken three times, and all 1,200 sets of measurements were taken in randomized order.

When taking measurements in simulation, the 3-D world locations of the hand centroid and the position constraint are known exactly. The robot model was also augmented with the appropriate "invisible" rigid bodies to directly measure orientation constraints in simulation (see Section 5.1.7). Therefore, all constraints were precisely and easily measured with appropriate timing in simulation. Since the constraint technique requires application

of torques to the robot actuators, the simulation used was the same dynamic simulation from Experiment 1.

The measurements of the orientation constraints' actual locations on the hardware were captured by measuring the joint angles from robot sensors. Since there was no way to easily instrument the robot measure angles in Cartesian space quickly for humans, the actual joint angles on the hardware were measured instead. These joint angles were tracked in a simulation that was augmented with the two points that represent head of the robot body vector and the head of the world vector. Then maximum orientation constraint error during the execution of the lower priority trajectory was calculated from the difference between these two points using the measured data instead of simulated data for joint angles that satisfy the constraint.

Prior to running Experiment 5, all constraint pairs were simulated to ensure that the constraint could be met without robot self-collision. Any constraints where the robot collided with itself in order to satisfy the constraint were excluded, and a new constraint was selected in place of the one that caused self-collision.

For Experiment 5, priority was randomized so that for 50% of each the cases for per hand point constraints are implemented as the highest priority task; and in the remaining 50% of trials in the experiment the orientation constraint for the palm has the highest priority; the single posture constraint subtask always was assigned the lowest priority since this is true whenever my algorithm is used.

5.1.10.2 Results

Comparing Tables 50 and 51, the maximum error in the point constraints during any of the five lower priority posture constraints was slightly higher when the point constraint was the lower priority. This finding is consistent with H4.

Furthermore, the Tables 52 and 53 exhibit a similar trend for maximum orientation constraint error during all posture constraints, in that the error was slightly larger for all measurement techniques when the orientation constraint was lower priority.

The trends evident from Experiment 1 are also evident in all four tables in the results

Table 50: Average of maximum distance between the hand centroid and actual specified point constraints (i.e. maximum point constraint error) over the duration of entire lower priority motion for five different posture constraints (i.e. lower priority motions). Averages taken over 200 measurements per hand, measured three times each, in three different ways (Sim = simulation; Joint = constraint calculated from measured actual robot joint angles; Human = human measured). Distance in centimeters. Standard deviations given in parentheses. Data taken with an *orientation constraint as second-highest priority*.

	Left Hand			Right Hand		
Motion	Sim	Joint	Human	Sim	Joint	Human
Bow	0.149 (.089)	0.371 (.124)	4.21 (2.06)	0.133 (.101)	0.402 (.110)	3.51 (1.27)
Point	0.162 (.093)	0.332 (.181)	3.78 (1.91)	0.148 (.190)	0.367 (.144)	2.96 (1.58)
Shrug	0.178 (.123)	0.318 (.211)	3.19 (1.82)	0.169 (.177)	0.354 (.131)	3.25 (1.78)
Stop	0.166 (.141)	0.370 (.201)	3.62 (1.74)	0.154 (.164)	0.382 (.156)	3.76 (2.04)
Wave	0.148 (.097)	0.400 (.191)	3.42 (1.87)	0.172 (.081)	0.369 (.133)	3.36 (2.02)
Avg.	0.161 (.209)	0.358 (.220)	3.64 (2.42)	0.155 (.217)	0.375 (.254)	3.37 (2.31)

Table 51: Average of maximum distance between the hand centroid and actual specified point constraints (i.e. maximum point constraint error) over the duration of entire lower priority motion for five different posture constraints (i.e. lower priority motions). Averages taken over 200 measurements per hand, measured three times each, in three different ways (Sim = simulation; Joint = constraint calculated from measured actual robot joint angles; Human = human measured). Distance in centimeters. Standard deviations given in parentheses. Data taken with an *orientation constraint as highest priority*, and the point constraint as second highest priority.

	Left Hand			Right Hand		
Motion	Sim	Joint	Human	Sim	Joint	Human
Bow	0.201 (.160)	0.452 (.215)	4.18 (2.16)	0.220 (.154)	0.471 (.209)	4.15 (2.51)
Point	0.234 (.139)	0.422 (.220)	4.30 (2.61)	0.214 (.204)	0.430 (.212)	3.90 (2.31)
Shrug	0.251 (.188)	0.441 (.233)	4.21 (2.09)	0.264 (.165)	0.432 (.226)	4.10 (2.05)
Stop	0.228 (.133)	0.451 (.218)	3.99 (2.18)	0.230 (.151)	0.439 (.251)	3.88 (2.35)
Wave	0.271 (.180)	0.418 (.217)	4.01 (2.22)	0.209 (.167)	0.414 (.229)	4.11 (2.34)
Avg.	0.237 (.219)	0.437 (.276)	4.14 (2.93)	0.227 (.245)	0.437 (.239)	4.03 (2.81)

Table 52: Average of maximum error between the commanded and actual specified orientation constraints over the duration of entire lower priority motion for five different posture constraints (i.e. lower priority motions). Averages taken over 200 measurements per hand, measured three times each, in two different ways (Sim = simulation; Joint = constraint calculated from measured actual robot joint angles). Error given in degrees. Standard deviations given in parentheses. Data taken with a *point constraint as second-highest priority*.

	Left Hand		Right Hand	
Motion	Sim	Joint	Sim	Joint
Bow	0.071 (0.103)	0.105 (0.124)	0.084 (0.107)	0.117 (0.122)
Point	0.073 (0.095)	0.119 (0.108)	0.069 (0.076)	0.132 (0.090)
Shrug	0.071 (0.054)	0.127 (0.127)	0.091 (0.087)	0.115 (0.129)
Stop	0.076 (0.081)	0.117 (0.126)	0.093 (0.079)	0.127 (0.119)
Wave	0.088 (0.090)	0.131 (0.130)	0.066 (0.083)	0.132 (0.089)
Average	0.076 (0.156)	0.120 (0.167)	0.081 (0.149)	0.125 (0.171)

Table 53: Average of maximum error between the commanded and actual specified orientation constraints over the duration of entire lower priority motion for five different posture constraints (i.e. lower priority motions). Averages taken over 200 measurements per hand, measured three times each, in two different ways (Sim = simulation; Joint = constraint calculated from measured actual robot joint angles). Error given in degrees. Standard deviations given in parentheses. Data taken with a *point constraint as highest priority*.

	Left Hand		Right Hand	
Motion	Sim	Joint	Sim	Joint
Bow	0.132 (0.121)	0.171 (0.142)	0.121 (0.145)	0.169 (0.191)
Point	0.119 (0.177)	0.182 (0.172)	0.113 (0.149)	0.192 (0.146)
Shrug	0.111 (0.134)	0.188 (0.204)	0.127 (0.197)	0.178 (0.199)
Stop	0.106 (0.142)	0.181 (0.183)	0.123 (0.156)	0.166 (0.166)
Wave	0.188 (0.161)	0.148 (0.142)	0.126 (0.139)	0.156 (0.159)
Average	0.131 (0.227)	0.174 (0.243)	0.122 (0.295)	0.172 (0.225)

from Experiment 5. In all of these tables the simulated error is lowest and human measured error is highest. This was caused by the ideal conditions of the simulation.

The error values in all four tables are very small, regardless of which technique was used to measure the error in position or orientation constraint. With less than 4 centimeters of point constraint error and less than 0.2 degrees of orientation constraint error, I concluded that these two constraints can be combined successfully.

To prove that these two constraints work well together under their most common application (i.e. grasping), 72 point constraints were commanded with both arms to reach onto a table to grasp rectangular blocks from either side or the top (three total different orientation constraints). Using successful grasps as the measure of success of the constraint-based framework, each of the 72 x 3 combinations was executed on the robot hardware five times (i.e. 1,080 total attempted grasps). Of the 1,080 total attempted grasps, 997 were successful on the robot hardware (89.7% successful from the right side; 98.3% from the top; 88.9% from the left side). 92.3% success for this task-based success measure gives an indication of how well these two constraints work well together.

5.1.11 Experiment 6: Joint Position Constraints

The following quantitative experiment tested my optimal control-based algorithm for satisfying joint position constraints. Experiment 6 quantified the average performance of these constraints for a variety of arm and torso DOFs.

5.1.11.1 Experimental Design

Five different motions (i.e. five different posture constraints) were used as the basis from which joint position constraints were defined. They were gestures of shrugging (symmetric), right-arm pointing, left-arm waving, right-arm stop, and two-handed bowing (symmetric). The data for Experiment 6 was collected from both simulations and on the real hardware by recording actual joint angles, and thus, the same constraints were run on both to ensure data consistency.

Three joint position constraints per DOF per motion trial were randomly selected. Each motion was executed 150 times with the three joint position constraints at different timings.

Table 54: Average *simulated* error between commanded joint position constraints and simulated values in five different motions. Averages taken over 450 measurements per DOF, per motion, measured once each. Error shown in degrees multiplied by 1000. Standard deviations (x1000) given in parentheses. (l = left arm; r = right arm; t = torso; s = shoulder DOF; e = elbow DOF; w = wrist DOF)

DOF	Bow	Point	Shrug	Stop	Wave	Average
tX	63 (41)	13 (10)	91 (11)	57 (49)	52 (52)	55 (67)
tY	22 (42)	44 (24)	72 (47)	14 (21)	39 (26)	38 (48)
lsX	88 (68)	91 (31)	91 (59)	60 (57)	25 (32)	71 (26)
lsY	93 (28)	25 (15)	56 (18)	99 (46)	95 (24)	74 (35)
lsZ	64 (63)	40 (36)	23 (31)	66 (21)	94 (53)	57 (33)
leX	22 (47)	26 (56)	58 (25)	96 (39)	77 (28)	56 (21)
lwY	29 (42)	68 (35)	57 (45)	70 (53)	48 (28)	54 (29)
lwX	62 (41)	25 (22)	12 (16)	53 (45)	76 (66)	46 (29)
lwZ	66 (30)	13 (17)	16 (8)	20 (15)	42 (60)	31 (18)
rsX	69 (31)	51 (48)	92 (24)	36 (42)	10 (15)	52 (68)
rsY	20 (47)	98 (50)	83 (66)	74 (21)	60 (14)	67 (57)
rsZ	56 (35)	56 (17)	65 (67)	57 (19)	28 (32)	52 (64)
reX	62 (45)	46 (66)	69 (37)	62 (34)	97 (65)	67 (36)
rwY	67 (25)	31 (39)	71 (31)	72 (38)	44 (27)	57 (19)
rwX	39 (17)	41 (10)	10 (22)	67 (29)	91 (25)	50 (43)
rwZ	24 (36)	95 (19)	15 (14)	12 (69)	82 (30)	46 (50)
Avg	53 (94)	48 (114)	55 (86)	57 (82)	60 (78)	55 (71)

With 16 total DOF (2 torso, and 7 per arm), this yielded 36,000 total position constraints for the entire experiment.

5.1.11.2 Results

Based on the data in Table 54, joint position constraints were met to within 0.1 degrees of the commanded values on average for five common social robot gestures in simulation. On the hardware, joint position constraints exhibited less than 0.5 degrees of error on average, as shown in Table 55. If the gains were increased for the optimal control formulation, these values could be improved. However, the hardware error is approaching the value of the resolution of the joint angle sensors, which supports H2. Therefore, there is little need to change the gains and improve these results.

The trend seen in previous experiments where simulated values performed better than

Table 55: Average *measured* error between commanded joint position constraints and values measured off the hardware in five different motions. Averages taken over 450 measurements per DOF, per motion, measured once each. Error in degrees multiplied by 1000. Standard deviations (x1000) given in parentheses. (l = left arm; r = right arm; t = torso; s = shoulder DOF; e = elbow DOF; w = wrist DOF)

DOF	Bow	Point	Shrug	Stop	Wave	Average
tX	153 (67)	158 (76)	133 (86)	194 (85)	161 (75)	160 (20)
tY	17 (22)	115 (68)	186 (35)	33 (57)	34 (37)	77 (88)
lsX	130 (23)	124 (76)	53 (56)	18 (48)	168 (36)	99 (31)
lsY	51 (36)	143 (56)	147 (52)	171 (15)	16 (12)	106 (58)
lsZ	174 (84)	138 (48)	120 (78)	53 (71)	170 (72)	131 (89)
leX	88 (37)	108 (36)	114 (99)	81 (57)	196 (72)	117 (43)
lwY	109 (50)	189 (34)	68 (59)	138 (48)	20 (61)	105 (85)
lwX	38 (63)	101 (34)	26 (77)	48 (62)	116 (95)	66 (12)
lwZ	117 (46)	151 (49)	41 (41)	29 (51)	92 (71)	86 (65)
rsX	193 (16)	73 (23)	164 (49)	150 (35)	166 (51)	149 (45)
rsY	106 (75)	119 (19)	60 (33)	69 (85)	35 (63)	78 (94)
rsZ	46 (41)	144 (72)	140 (60)	108 (48)	82 (46)	104 (32)
reX	72 (42)	144 (90)	55 (23)	95 (77)	60 (68)	85 (42)
rwY	155 (46)	196 (23)	189 (26)	59 (98)	172 (29)	154 (86)
rwX	78 (36)	171 (14)	32 (56)	181 (50)	130 (58)	118 (41)
rwZ	139 (49)	109 (91)	165 (95)	184 (83)	157 (48)	151 (58)
Avg	104 (114)	136 (137)	106 (195)	101 (180)	111 (96)	112 (148)

measured values on the hardware is also evident in Tables 54 and 55. As mentioned previously, this is most likely due to the inaccuracies of the inertia matrix.

5.1.12 Experiment 7: Joint Velocity Constraints

The following quantitative experiment tested my optimal control-based algorithm for satisfying joint velocity constraints. Experiment 7 quantified the average performance of these constraints for a variety of arm and torso DOFs.

5.1.12.1 Experimental Design

Five different motions (i.e. five different posture constraints) were used as the basis from which joint velocity constraints were defined. They were gestures of shrugging (symmetric), right-arm pointing, left-arm waving, right-arm stop, and two-handed bowing

Table 56: Average *simulated* error between commanded velocity constraints and simulated values in five different motions. Averages taken over 450 measurements per DOF, per motion, measured once each. Error in degrees per second multiplied by 1000. Standard deviations (x1000) given in parentheses. (l = left arm; r = right arm; t = torso; s = shoulder DOF; e = elbow DOF; w = wrist DOF)

DOF	Bow	Point	Shrug	Stop	Wave	Average
tX	64 (57)	120 (99)	219 (68)	120 (21)	48 (106)	114 (83)
tY	116 (107)	144 (48)	135 (32)	169 (99)	144 (71)	142 (40)
lsX	230 (14)	205 (45)	51 (29)	211 (35)	63 (69)	152 (109)
lsY	225 (120)	193 (67)	48 (26)	116 (66)	129 (26)	142 (108)
lsZ	157 (100)	233 (20)	155 (11)	132 (81)	235 (34)	182 (37)
leX	105 (46)	106 (49)	234 (114)	216 (24)	223 (73)	177 (67)
lwY	179 (83)	181 (53)	54 (24)	210 (94)	137 (20)	152 (32)
lwX	150 (34)	142 (34)	97 (120)	112 (102)	126 (47)	125 (98)
lwZ	149 (83)	157 (66)	210 (74)	126 (81)	98 (111)	148 (100)
rsX	115 (30)	168 (31)	184 (69)	160 (30)	135 (13)	152 (99)
rsY	154 (85)	103 (50)	168 (25)	91 (107)	152 (120)	134 (108)
rsZ	250 (28)	165 (97)	131 (79)	145 (74)	159 (97)	170 (107)
reX	161 (18)	79 (79)	128 (117)	56 (50)	181 (39)	121 (90)
rwY	165 (81)	139 (61)	130 (43)	120 (87)	141 (28)	139 (34)
rwX	118 (57)	122 (56)	127 (116)	90 (105)	131 (74)	118 (120)
rwZ	108 (115)	215 (33)	133 (79)	87 (25)	149 (76)	139 (54)
Avg	153 (180)	154 (112)	138 (175)	135 (134)	141 (149)	144 (153)

(symmetric). The data for Experiment 7 was collected from both simulations and on the real hardware by recording actual joint angles, and thus, the same constraints were run on both to ensure data consistency.

Three joint velocity constraints per DOF per motion trial were randomly selected. Each motion was executed 150 times with the three joint velocity constraints at different timings. With 16 total DOF (2 torso, and 7 per arm), this yielded 36,000 total velocity constraints for the entire experiment.

5.1.12.2 Results

Based on the data in Table 56, joint velocity constraints were met to within 0.14 degrees per second error of the commanded values on average for five common social robot gestures in simulation (1.76% of maximum velocity for any of the five motions). On the hardware,

Table 57: Average *measured* error between commanded joint velocity constraints and values measured off the hardware in five different motions. Averages taken over 450 measurements per DOF, per motion, measured once each. Error in degrees per second multiplied by 1000. Standard deviations (x1000) given in parentheses. (l = left arm; r = right arm; t = torso; s = shoulder DOF; e = elbow DOF; w = wrist DOF)

DOF	Bow	Point	Shrug	Stop	Wave	Average
tX	396 (207)	453 (95)	398 (200)	420 (222)	430 (237)	419 (202)
tY	469 (236)	375 (194)	228 (125)	494 (59)	311 (91)	375 (142)
lsX	456 (98)	183 (249)	461 (153)	143 (59)	193 (250)	287 (232)
lsY	189 (98)	193 (204)	165 (167)	196 (178)	146 (163)	178 (90)
lsZ	146 (99)	177 (104)	463 (68)	351 (93)	167 (183)	261 (177)
leX	448 (126)	407 (85)	243 (180)	235 (160)	285 (86)	324 (155)
lwY	121 (133)	313 (186)	170 (135)	205 (133)	188 (68)	199 (79)
lwX	451 (150)	161 (186)	210 (201)	498 (136)	331 (195)	330 (230)
lwZ	347 (233)	188 (220)	395 (205)	119 (126)	419 (105)	294 (124)
rsX	111 (204)	105 (176)	364 (229)	272 (170)	494 (31)	269 (244)
rsY	336 (112)	370 (48)	170 (164)	210 (164)	315 (150)	280 (82)
rsZ	238 (118)	247 (115)	314 (92)	438 (158)	232 (232)	294 (106)
reX	214 (185)	218 (216)	277 (132)	248 (166)	130 (153)	217 (93)
rwY	265 (209)	207 (111)	469 (111)	244 (184)	332 (202)	303 (139)
rwX	225 (241)	183 (96)	354 (107)	292 (242)	354 (87)	282 (227)
rwZ	286 (165)	410 (145)	386 (172)	285 (159)	384 (152)	350 (233)
Avg	294 (269)	262 (331)	317 (314)	291 (303)	294 (256)	291 (328)

joint velocity constraints exhibited less than 0.30 degrees per second of error on average (3.77% of maximum velocity for any of the five motions), as shown in Table 57. If the gains were increased for the optimal control formulation, these values could be improved. However, the hardware error is approaching the limit imposed by the resolution of the joint angle sensors, which supports H2. Therefore, there is little need to improve these results.

The trend seen in previous experiments where simulated values performed better than measured values on the hardware is also evident in Tables 56 and 57. As mentioned previously, this is most likely due to the inaccuracies of the inertia matrix.

5.1.13 Experiment 8: Multiple Simultaneous Constraints

The previous seven experiments have not thoroughly tested the maintenance of multiple internal and external constraints concurrently imposed upon the robot hierarchy in a prioritized fashion. Thus, Experiment 8 was designed to test hypotheses H3 and H4, to quantify constraint error as a function of both priority and kinematic distance between constraints on the robot hierarchy.

Kinematic distance is defined as link-length distances between the force impacting points $x_{robot}(t)$ of two external constraints on the robot body. Recall that all internal constraints have the same priority in my constraint formulation (i.e. always lowest priority), and thus kinematic distance makes no sense with respect to internal constraints. To understand kinematic distance, the robot architecture is treated as a skeleton or "stickman" of single link segments of finite, static length between actuators. The kinematic distance ignores any joints or joint angles in the calculation (i.e. it is time and configuration independent), and it is calculated by measuring the distance between the two $x_{robot}(t)$ points on the skeleton model as measured only along the link lengths. This is similar to the calculation of Manhattan or "city block" distance in 3-D space. However, the nodes for Manhattan distance are substituted by 3-D robot joints or actuator positions in kinematic distance. For example, the kinematic distance between the robot elbow and the sternum is the summation of the distance from the elbow to the shoulder and the distance from the shoulder to the sternum (assuming the shoulder DOF is the only actuator between these two points). Alternatively, the kinematic distance between two points on the robot body can be thought of as the accumulation of distances between all actuators in-between the two $x_{robot}(t)$ constraint points plus each of the distances from the constraint points to the nearest actuator in the hierarchical direction toward the other $x_{robot}(t)$ point on the kinematic hierarchy.

5.1.13.1 Experimental Design

To support H3 and H4, the same five lower priority motions as in all previous experiments were used as the lowest priority constraint. Then for the higher priority constraint, up

to 10 external constraints were imposed. The amount, timing, type, and position on the robot hierarchy varied for all five different lower priority motions. Experiment 8 was performed completely in simulation due to the safety concerns with switching constraints. Constraint error, position (i.e. point of application on the robot body, $x_{robot}(t)$), and priority was recorded.

The simulation was run randomly for two hours. Data recording was automated. Since all data for Experiment 8 was simulated, all different external constraint types were measured and represented in the common units of centimeters by using the distance between the two constrained points $x_{robot}(t)$ and $x_{world}(t)$. These points were easily measured since the simulated Cartesian locations could be directly read off the rigid bodies attached to the augmented model hierarchy in the simulation. Common units of measurement were used so that the conclusions could be drawn from the data that were independent of constraint type.

5.1.13.2 Results

Figure 33 shows a subset of all the data collected in Experiment 8. All the points during the simulation when only two constraints were imposed upon the robot simultaneously were collected. The kinematic distance between these constraints was calculated and the lower priority constraint error is plotted against kinematic distance in Figure 33.

H3 is supported by the trend exhibited in Figure 33, which shows that as the kinematic distance between two external constraints increased, the higher priority constraint had less influence on the lower priority constraint. Therefore, the lower priority constraint was more easily maintained and error in maintaining the second-highest priority constraint decreased. The data for only two constraints in the presence of an lower priority posture constraint was used because the nonlinear interaction between three or more constraints on the hierarchy makes it difficult to isolate the separate influences of the two constraints on the third external constraint.

Figure 34 shows average constraint error versus decreasing constraint priority for up to ten simultaneous external constraints imposed upon the robot. Error bars indicate one

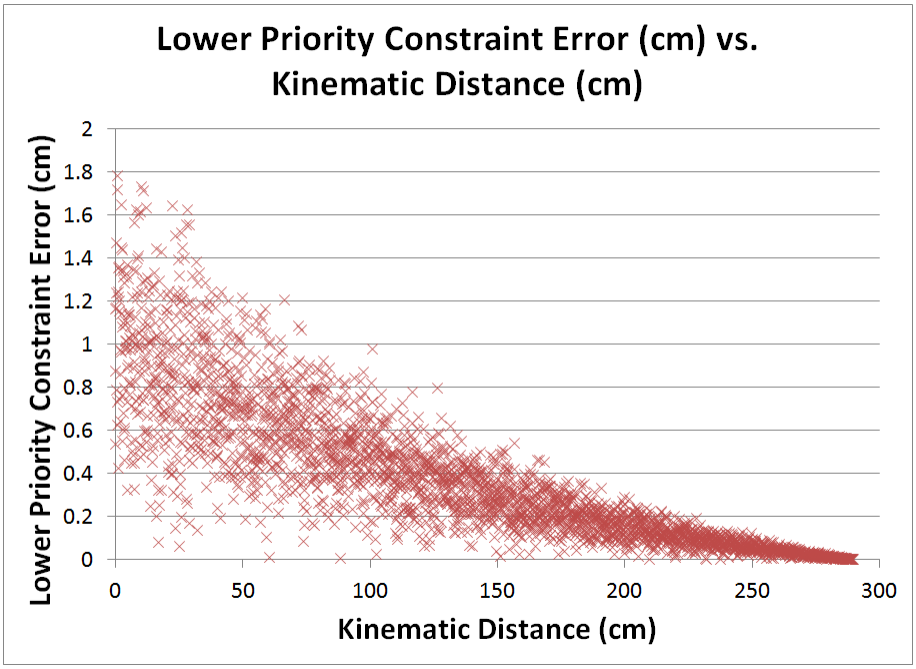

Figure 33: Constraint Error vs. Kinematic Distance. The error in the second-highest priority constraint decreases as kinematic distance between the points of application of the two highest priority constraint increases. Plot shows simulated data for two external constraints and a posture constraint (lowest priority).

standard deviation of the error. Since amount, timing, type, and position of the constraints were varied in Experiment 8, the data in Figure 34 averages across all four types of external constraints and includes situations where anywhere from two to ten constraints were simultaneously imposed on the robot.

Figure 34 supports hypothesis H4, as the clear trend that adding more constraints to the hierarchy affects how well all the constraints can be satisfied. Specifically, for lower priority constraints (i.e. numbers closer to 10 in Figure 34), the error increased due to a smaller remaining null space when subsequent projections were performed. The smaller null space resulted in constraints that were less perfectly satisfied, as indicated by the error trend in Figure 34.

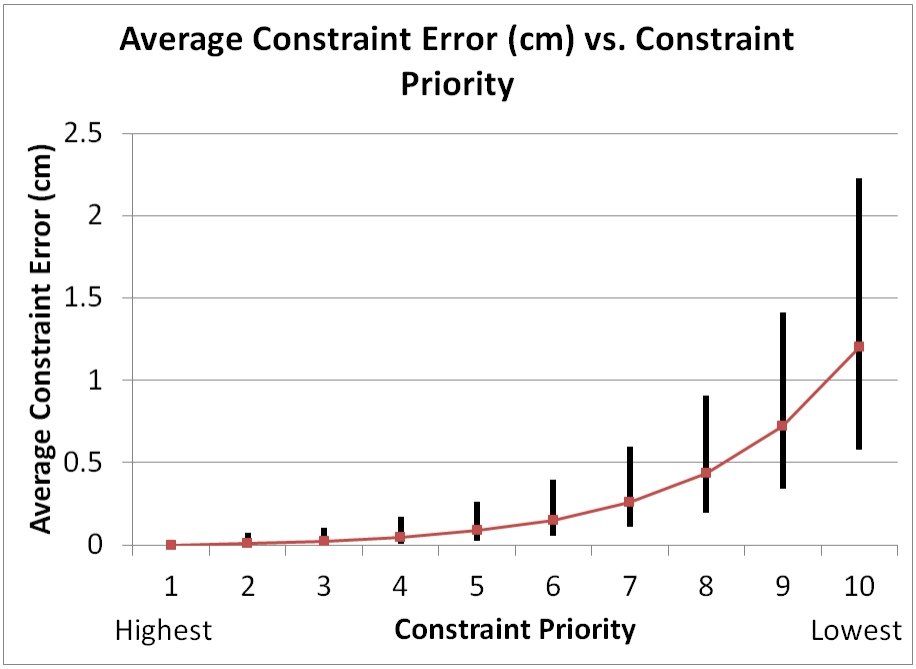

Figure 34: Average Constraint Error vs. Constraint Priority. As constraint priority becomes lower (i.e. higher numbers), average constraint error increases due to a smaller null space (i.e. constraints are satisfied less accurately). Error bars show one standard deviation for all data collected during two hours of simulation.

5.1.14 Discussion

Using the four external constraints, more sophisticated constraints can be developed, similar to the manner in which point and orientation constraints are combined to produce constrained grasping. These complex constraints do not require their own frameworks. Rather, they can be achieved through clever combination of the four external constraints previously developed.

Furthermore, all four basic constraints developed previously use attractive forces to ensure that two points stay together. However, repulsive forces that keep a point on the robot's body apart from a point, angle, etc. have many useful common social robot applications. Three interesting examples of more sophisticated constraints are proximity constraints, collision avoidance (e.g. self or other obstacles), and volume constraints.

A proximity constraint is easily implemented as a point constraint that can freely rotate

about all three Cartesian axes. For example, if the hand needs to stay a finite distance from an object, the point constraint is offset from the hand centroid by a finite extent. Then the point constraint is allowed to freely move on the surface of the sphere of radius equal to the length of the proximity required. If the required constraint is similar to an orbit or satellite constraint, attractive forces pull the world and robot points together. The robot constraint is always on the surface of the sphere directed along the axis collinear with the robot and world constraint point points.

Collision or obstacle avoidance is simply the point constraint framework with repulsive forces. However, care must be taken in assigning the magnitude of the repulsive forces so they influence motion only in the desired vicinity of the obstacle or object.

Volume constraints represent inequality constraints. In a volume constraint, the objective is to disallow a point on the robot's body to penetrate the volume (e.g. the volume might represent a table, which the robot's hand shouldn't penetrate). In all the previously presented constraints, all three Cartesian dimensions were simultaneously constrained. However, a volume constraint represents a situation where only one or two of the three Cartesian dimensions are constrained. In the example of the table and robot end-effector, where the table is represented as a volume constraint, and all the Cartesian coordinate axes (and negative axes) coincide with the normals of a rectangle, which represent the volume constraint not to be penetrated). When the hand reaches the surface of the table, which is represented by a plane, the hand is free to move in two dimensions along the surface of the table. But, penetration of the table would occur if the robot hand moved along the third Cartesian axis in only one direction (i.e. an inequality constraint). For volume constraints, conditionals check for robot motion that would penetrate the volume. When motion would penetrate the volume, the projection triggers forces to nullify robot motion only in that particular direction. In the table and hand example, this would be accomplished by using a point constraint where two of the coordinates are constrained to match the robot hand centroid coordinates when forces at the robot hand are directed toward penetrating the box volume (i.e. the point constraint moves along the table surface, at a finite distance away). To maintain the volume constraint completely, similar inequality

and conditional constraints would be imposed from all six sides of the rectangle volume.

5.1.15 Summary

My contributions in the area of internal and external constraints for social robot applications were to:

1. develop an intelligent application of an existing technique called operational space control (background details in Section 2.1).

2. demonstrate how optimal control can be used to satisfy joint position, velocity, and torque constraints, especially in dynamic constraints that require synchronization.

3. describe how Cartesian and joint-space constraints can be designed as trajectories for use with OSC.

4. develop orientation, point-at, and look-at constraints in the form of point constraints for OSC.

5. show the accuracy of point, orientation, point-at, and look-at constraint and quantify their average and worst-case performance for common social robot gestures.

6. quantify the performance of internal constraints, such as joint position and joint velocity, for common social robot gestures.

7. demonstrate that two external constraints close in proximity on the robot hierarchy produce more error in the secondary constraint than two external constraints applied further apart on the robot.

8. show the effect of a decreasing null space on increasing constraint error as more constraints are added to the robot.

CHAPTER VI

INTEGRATION

6.1 Composition of Algorithms

6.1.1 Introduction

Figure 35: Block diagram composing all algorithms.

All the separate algorithms have been explained, and each can be used independently to perform their separate functionality. However, social robots can benefit from using the composition of all the algorithms simultaneously. One possible combination is shown Figure 35, which is the true vision of my research.

Although, a composition of my algorithms might seem straightforward, consideration must be taken to ensure the appropriate sequence is maintained so that the algorithms don't adversely effect the output of any other preceding algorithms.

Therefore, this integration section is devoted to discussion of appropriate forms of composition for all the algorithms presented herein, to optimize performance and the goals that I am trying to achieve with robot motion: (1) communication, (2) human-likeness, and (3) functional HRI improvement. Also, experiments were conducted to test that interaction between the algorithms does not adversely affect benefits derived by preceding algorithms. Since six independent algorithms exist, there are 720 possible ways to combine all the algorithms in a serial order. Therefore, it is not possible to present experiments that exhaustively test all permutations for all possible benefits. Rather, a few select examples were chosen to demonstrate the power of the algorithms when correctly composed because often, instead of adversely influencing the output of previously applied algorithms,

subsequent algorithms enhance the effects and benefits provided by algorithms applied earlier in the sequence.

6.1.2 Insight

The insight for the proper integration of all six algorithms comes from an in-depth understanding of each of the algorithms and the effects they produce on a motion trajectory. Moreover, each of the algorithms were designed to be complementary, each addressing a deficiency in typical robot motion. Thus, their composition was always envisioned. To a significant extent, their appropriate integration stems from logical cause-effect relationships. For example, I purport that the order of the communicative algorithms is: anticipation, exaggeration, and then secondary motion. Anticipation is known to be preparatory motion; exaggeration emphasizes the primary trajectory; and secondary motion creates the proper effects of the primary trajectory. Hence, a causal order is imposed largely due to the algorithms' roles in the motion. More examples of these relationships will be discussed in Section 6.1.4.

6.1.3 Goal

The goal of the composition of my algorithms is to demonstrate the advantages of different ways of composing the individual parts. I show the performance improvement when certain algorithms are combined properly. In other cases, I demonstrate that benefits are not lost through certain algorithm combinations.

6.1.4 Algorithm

Since each algorithm outputs a trajectory, composition is nothing more than a serial concatenation of the algorithms. However, the caveat is the order in which they should be combined.

Although exceptions exist to the following order that have different advantages in different situations, I present what I believe to be the optimal order for a large number of scenarios. Any individual algorithm can be removed from the sequence and only the benefits of that specific algorithm will be lost to the overall output.

As shown in Figure 35, a motion trajectory should have communication added first. Communication should be added before making the motion more human-like or adding variance because these qualities should influence the communication trajectory that is added also.

To a significant degree, the order of the three communicative algorithms are interchangeable. However, different orders make more sense than others. There are six possible permutations on order of these three algorithms. Here are the advantages and disadvantages of each, using the abbreviations (S = secondary motion, E = exaggeration, and A = anticipation):

- SEA: The effects of secondary motion are diminished (or worse amplified) by the subsequent exaggeration algorithm. Secondary motion won't be appropriate in response to the anticipation motion.
- SAE: Secondary response to the anticipation motion won't be included in the trajectory. True secondary response to the exaggerated dynamics could be corrupted by the exaggeration process.
- ESA: Exaggerated dynamics will have appropriate secondary motion. Secondary motion will be incorrect for the anticipation motion.
- EAS: Secondary response to the anticipation motion will be appropriate. But, the anticipation algorithm might reduce some of the previously-added exaggeration.
- ASE: Secondary response to the anticipation motion will be appropriate. True secondary response to the exaggerated dynamics could be corrupted by the exaggeration process.
- AES: Secondary response to the anticipation motion will be appropriate. Exaggerated dynamics will have appropriate secondary motion.

From the discussion of the advantages and disadvantages of each combination, AES is the combination that has more advantage and fewer disadvantages than any other

combination. Anticipation should occur before exaggeration because anticipation changes the trajectory timing more than any other algorithm. And to appropriately match the secondary motion trajectory to the exaggerated motion trajectory, the secondary motion algorithm should modify the trajectory after exaggeration. Thus, I apply secondary motion last of all possible communicative motion techniques.

When one of these communicative motion algorithms is excluded, it becomes much easier to decide proper order. For example, it doesn't often make sense to use anticipation on motions that don't include a symbol. In these certain cases, the anticipation algorithm can be excluded from the concatenated series.

Following communication, the motion trajectory should be changed to be made more human-like. In theory, it is feasible to exchange the order of the human-like and variance algorithms, but variance is placed as close to last as possible since the output of the variance is an infinite number of motions. This can be an advantage to place earlier, as long as a single variant is carried through the rest of the algorithm because then the most visually pleasing variant or perhaps the most human-like variant can be selected using the algorithm in Section 4.1 as a metric. If the variant-generating algorithm is placed too early and the multiple variants are desired, then the other algorithms must execute many more times than if the variance algorithm was placed in the location shown in Figure 35. The other advantage of placing the variance algorithm with the extended OSC is that internal constraints and variance can be solved together, meaning that the optimal control problem is solved only one time.

External constraints are typically hard constraints that should never be violated. And since the other algorithms modify the trajectory without a guarantee of respecting any external constraints (e.g. human-like, variance), my extended operational space control algorithm is placed last in the sequence.

The final note is regarding concatenating trajectories to create continuous robot motion. The same methodology that was used for non-deterministic transitions (Section 4.2.4.3) is also used to create continuous motions, which is another reason that variance is applied last in the sequence. Furthermore, since transitory motions are known in advance (when

the previous motion trajectory is solved), internal and external constraints are easily applied during these motion sequences too.

6.1.5 Hypotheses

Specifically, I want to address the interaction between my constraint-satisfying, my variance, and my human-like algorithms because the constraint-satisfying algorithm, by design, cancels torques. Intuitively, this can cause adverse effects upon my human-like algorithm, which relies upon a specific coordination between torques. Similarly, canceling torques can logically reduce variance, since less torque means less motion. Thus, I want to address concerns that under normal conditions the resultant variance is still noticeable. Practically, as the number of constraints approaches infinity, the null space disappears, which means that eventually variance will not be noticeable. However, I would like to quantify a typical number of constraints for which variance remains noticeable.

- H1: Constraints reduce motor coordination, but my human-like optimization (Section 4.1) still improves spatial and temporal correspondence in the presence of constraints.

- H2: Internal and external constraints reduce variance, especially closer to the point of application of the constraint. However, I hypothesize that my variance algorithm still produces noticeably different trajectories for typical applications. The phrase "point of application of the constraint" means physical (i.e. spatially proximal to) locations on the robot body where the constraint is applied for *external* constraints, I expect less variance in those nearby DOFs. For *internal* constraints, the phrase "point of application of the constraint" only has temporal meaning (i.e. closer to the time increments when joint position, velocity, or torque are constrained, I hypothesize that less variance can be produced simultaneously with these constraints). The temporal meaning also applies to external constraints.

- H3: Self-collision in human motion occurs infrequently, and therefore, my algorithm will also produce self-colliding motion rarely (including all variants) when the input

motion does not cause self-collision.

The following five experiments were designed to test H1-H3.

6.1.6 Experiment 1: Human-likeness in the Presence of Constraints

6.1.6.1 Experimental Design

Experiment 1 was designed to test H1 and the interaction between the human-like and constraint-satisfying algorithms. In previous experiments, participants were recruited to test whether spatially and temporally optimized motion appeared more human-like than other types of motions. SC and TC were presented as a metric for human-like motion to find a quantitative measure for humans' subjective perception with respect to human-likeness in motion. Thus, participants did not need to be recruited to determine which motions are more human-like. The intuition behind the design of integration Experiment 1 is that if the benefits of lower SC and TC values are preserved in the presence of constraints, then the motion with constraints remains more coordinated (and therefore, more human-like) than motion that excludes the human-like optimization.

Therefore, in Experiment 1, four types of motion were compared:

- OH: Reference condition (i.e. not a test condition) used for comparison of SC and TC of the original human motion against the three test conditions.
- WB: With both the human-like optimization and constraints.
- WH: With only the human-like optimization (without constraints).
- WC: With only the constraint-preserving algorithm (without the human-like optimization).

Human-like Experiment 1 part one from Section 4.1.6.1.1 was extended to include constraints, since this prior experiment included 20 trajectories for which the values of SC and TC had already been computed (i.e. 20 original coordinated (OC) motions). These 20 values for OC motions were the WH motions for Experiment 1. To form the other two groups needed for the analysis in this experiment, the 20 motions from Experiment 1

part one in Section 4.1.6.1.1 that represent the OC motions without coordination (i.e. the original retargeted (OR) motions) had random constraints applied. This formed the WC set of motions. Finally, the WH motions had the same set of random constraints applied to create the WB group of trajectories. The algorithm from Section 4.1.4 was used as a metric to evaluate the human-likeness of each of the groups of trajectories (WB, WH, and WC).

Specifically, since I hypothesized that WH would have the lowest SC and TC values, followed by WB, and then WC, if the results show this order, then I will have provided evidence in support of hypothesis H1.

To strengthen the results, the number of simultaneously applied constraints varied from 1 to 5 for each of the 20 motions to help demonstrate that as number of constraints increases SC and TC decrease. H1 is independent of constraint type, and therefore, Cartesian position, orientation, look-at, point-at, joint position, joint velocity, and joint acceleration (i.e. all types of internal and external constraints) were the types from which random selection was made. Each constraint was applied for a time duration of 5% of the total length of the trajectory. In cases where more than one constraint was simultaneously applied, all constraints were applied during the same time period and for the same duration. Constraint type, duration, number simultaneously applied, and time period of application were identical between the WB and WC groups.

6.1.6.2 Results

Table 58 demonstrates a distinct trend for constraints. As more constraints were added to a particular motion in both the WB and WC groups, SC and TC increased, which means that motion became less coordinated. Although the trend was more apparent with more constraints added to the motion, the WH group, which is unconstrained motion that is optimized according to the SC and TC metrics had values closest the original human performances for 16 of the 20 motions (spatial) and 14 of 20 motions (temporal). Considering only the two data groups that were optimized according to my human-like metric, the SC values were closest to the original human trajectories in 20 of 20 motions, and the TC values were closest in 18 of 20 motions.

Table 58: SC and TC values for three data groups (WH, WB, WC) as calculated by the respective parts of Equation 38. Motions grouped between double horizontal lines according to number of simultaneously applied constraints (i.e. first four motions have one constraint, second four have two simultaneously applied constraints, etc.). OH = original human performance; WH = with only the human-like algorithm applied; WB = with both the human-like and constraint preserving algorithms; WC = with only the constraint-preserving algorithm.

	SC				TC			
Motion	OH	WH	WB	WC	OH	WH	WB	WC
Move Object	0.189	0.198	0.204	0.212	0.165	0.154	0.158	0.162
Look Around	0.088	0.087	0.089	0.092	0.066	0.061	0.063	0.065
Sneeze	0.068	0.069	0.071	0.072	0.082	0.079	0.083	0.086
Take Cover	0.167	0.163	0.165	0.171	0.126	0.128	0.131	0.136
Shrug	0.178	0.183	0.190	0.204	0.151	0.148	0.151	0.155
One-hand Bow	0.148	0.145	0.155	0.167	0.120	0.116	0.127	0.128
Worship	0.208	0.204	0.209	0.214	0.190	0.187	0.189	0.200
Shucks	0.145	0.151	0.156	0.163	0.124	0.122	0.131	0.133
Beckon	0.092	0.091	0.098	0.109	0.125	0.129	0.131	0.148
Presentation	0.143	0.139	0.146	0.147	0.104	0.108	0.111	0.116
Call/Yell	0.158	0.155	0.157	0.163	0.156	0.148	0.162	0.179
Throw	0.127	0.135	0.137	0.152	0.122	0.128	0.147	0.162
Wave	0.108	0.103	0.117	0.139	0.140	0.141	0.159	0.187
Two-hand Bow	0.166	0.171	0.183	0.218	0.143	0.140	0.154	0.168
Stick 'em Up	0.137	0.135	0.151	0.159	0.101	0.119	0.138	0.145
Clap	0.175	0.170	0.196	0.217	0.127	0.127	0.143	0.157
Scan	0.105	0.108	0.113	0.120	0.094	0.096	0.097	0.112
Air Guitar	0.150	0.163	0.169	0.172	0.121	0.124	0.137	0.151
Bird	0.182	0.195	0.196	0.243	0.165	0.157	0.175	0.206
Cradle	0.150	0.143	0.149	0.153	0.150	0.151	0.158	0.175

Furthermore, of the three data groups that were tested (WH, WB, and WC), WH was the most coordinated motion of all three, as seen by the lowest SC and TC values in Table 58. This data supports H1 and the conclusion that concatenating the constraint-preserving algorithm in series after the human-like algorithm does not corrupt the benefits added by human-like optimization by canceling torque using a null-space projection.

6.1.7 Experiment 2: Variance with External Constraints

In Chapter 5 a thorough set of experiments were performed to measure the performance of constraints when combined with each other, but it was also important to test that the projection into the null-space to satisfy these constraints does not adversely affect the variance induced by my algorithm. To support H2, Experiment 2 was designed to analyze variance in the presence of multiple constraints.

6.1.7.1 Experimental Design

Specifically, one point constraint and one orientation constraint were applied to the robot's left hand for the entire duration of a shrug trajectory to constrain the hand so that a cup of water did not spill. Deviation in the left arm was then analyzed with respect to the original motion (Equation 45) and average inter-variant variance (Equation 46) for two conditions: with and without constraints. Comparable numbers for these two conditions on a per-DOF basis would support the null hypothesis that constraint projection does not adversely affect variance. This analysis was specifically performed on the arm closest to the constraint because part of hypothesis H2 suggests that variance decreases closer to the constrained location on the architecture.

6.1.7.2 Results

To measure task-aware variance in the presence of constraints, the variance of the left arm DOFs provided more insight into how constraints affect the task space because the left arm was closer to the constraint in the task. I first applied operational space control to constrain the original motion, and then I used the original motion torques as the mean to compute the variance (Equation 45). The results are shown in Table 59.

Table 59: Average variance (standard deviation) of joint angles for all left arm DOFs (in square degrees). Deviation from original motion and inter-variant variance (deviation between generated variants) given for conditions of constrained and unconstrained motion. Averages taken using the first 12 motions generated by the variance algorithm for a shrug ("I don't know") gesture while constrained with the left hand holding a cup.

	Unconstrained		Constrained	
DOF	w.r.t. orig	between variants	w.r.t. orig	between variants
shoulder X	312 (241)	396 (197)	116 (55)	354 (121)
shoulder Z	305 (190)	213 (209)	271 (123)	133 (58)
shoulder Y	314 (139)	457 (212)	298 (286)	599 (342)
elbow X	602 (229)	954 (799)	581 (119)	846 (837)
wrist Y	442 (251)	941 (642)	560 (175)	1132 (899)
wrist X	279 (281)	260 (317)	22 (16)	189 (119)
wrist Z	171 (192)	194 (236)	44 (37)	124 (63)

As expected, physical constraints reduce variance overall, as shown in Table 59. This was especially true closer to the actual constrained DOF, which for the cup constraint occurred in the wrist for the x and y degrees-of-freedom. This effect was also evident in the calculations that reference the other variants. However, average variance between variants was still large for the left arm DOFs.

By comparing data for constrained and unconstrained, the trend was that constrained deviation was smaller when further from the wrist (i.e. point of constraint application). The constrained numbers approach unconstrained variance further from the wrist, which indicates the impact of constraints upon variance. From my analysis I concluded that my variance algorithm also produces variance in constrained motion, provided that a null space for the constraints exists. To a certain extent, H2 is supported, although the data clearly indicates that variance is limited near the point of application of a constraint. As more constraints are added, less variance can be expected.

6.1.8 Experiment 3: Variance with Internal Constraints

In Experiment 2, I provided support for H2 for external constraints. In Experiment 3, I will provide evidence in support of H2 for internal constraints.

When my variance algorithm uses an exemplar that is already human-like, the cost

function in the optimal control solution helps to maintain naturalness. This unique framework provides a simple way to maintain internal constraints (e.g. joint position, velocity, torque) by modulating the cost function weights over the motion timing. For example, synchronization to the reference (i.e. input) motion can be produced from internal constraints by enforcing velocity constraints at specific time instants during the trajectory. I demonstrated this combination of algorithms using a common robot task: timing gestures to speech.

Although there are many ways to define the task of synchronizing motion to speech, I defined this task through ensuring that the emphasized word in a phrase is spoken at the very same instant in time as the zero velocity point in the accompanying gesture. I demonstrated this with an "I don't know" gesture. My goal in Experiment 3 was to prevent the added variance from disrupting the synchronized points so that the same speech can be triggered to synchronize the gesture variants. This goal is possible with my algorithm because salient time instants in the variants remain deterministic.

6.1.8.1 Experimental Design

For comparison, I implemented a basic version of style-based inverse kinematics, a computer animation technique which builds a low-dimensional model from human input motions, and interpolates the space of motions to generate new motions. The output motions produced through this approach can be realistic because they are based on human input data. However, interpolation does not guarantee that the task features such as zero velocity points will align in different exemplars unless preprocessing of the input data occurs to warp the timing of all motions, which I did not perform.

My variance algorithm can maintain the synchronized gestures by setting high cost for deviation at the moments of emphasized words. Unlike interpolation, my variance algorithm only requires one representative exemplar in order to create an infinite number of variants, which removes the tedious process of warping and aligning a large number of motions.

In Section 4.2.9, I showed that no statistical difference (in terms of human-likeness)

exists between style-based inverse kinematics and the variants produced by my variance algorithm. By adding internal constraints, I wanted to test the coupled effects between variance and internal constraints, to ensure that variance was still significantly noticeable and that internal constraints were still satisfied when solved together with variance. Since my variance algorithm is task-aware, internal constraints should remain satisfied in the presence of variance.

An "I don't know" motion was chosen for Experiment 3 since the single zero velocity inflection point (i.e. when the robot arms hit the peak of the gesture) provides a clear timing point to synchronize to speech. 1,000 gesture variants of this motion with and without internal constraints were generated and three analyses were performed: deviation, velocity, and timing. (1) Deviation: average deviation from the original motion (Equation 45) and average inter-variant variance (Equation 46) were calculated with internal constraints to determine if adding internal constraints limits variance. (2) Velocity: at each constrained time increment, zero average velocity was expected. Thus, to measure how well the constraint magnitudes were maintained in the presence of variance, average velocity was calculated for these points over all variants. (3) Timing: to measure how well the timing of the constraints was maintained in the presence of variance, average difference between the constrained time and actual time of the occurrence of the zero velocity point in each gesture was calculated. For (1), (2), and (3), for comparison, these same measures were calculated for 1,000 gesture variants without internal constraints.

6.1.8.2 Results

The optimal way to determine success of synchronizing speech and gesture is to actually view the algorithms working in combination, where speech and gesture are synchronized according to emphasized words. For example, "I don't know" versus "I don't know". With my variance algorithm, the gesture can be varied autonomously, but the timing of the zero-velocity point still remains deterministic and predictable, thereby allowing synchronization of speech and varied motion simultaneously.

Table 60: Average variance (standard deviation) of joint angles for all left arm DOFs (in square degrees). Deviation from original motion and inter-variant variance (deviation between generated variants) given for conditions of constrained and unconstrained motion. Averages taken using the first 1000 motions generated by the variance algorithm for a shrug ("I don't know") gesture. Averages also taken over entire motion time length. For the constrained condition, the internal constraint is a zero velocity inflection point.

	Unconstrained		Constrained	
DOF	w.r.t. orig	between variants	w.r.t. orig	between variants
shoulder X	1047 (663)	2745 (1549)	884 (541)	2300 (1113)
shoulder Z	991 (343)	701 (544)	854 (387)	645 (478)
shoulder Y	1409 (931)	2011 (1629)	1297 (877)	1607 (1472)
elbow X	5487 (1406)	2817 (522)	5351 (1284)	1927 (379)
wrist Y	5562 (1442)	2342 (794)	5369 (1327)	2231 (656)
wrist X	1376 (389)	477 (228)	1263 (422)	437 (202)
wrist Z	1270 (391)	424 (294)	1214 (503)	411 (255)

6.1.8.2.1 Noticeable Variance

Using the first 1,000 variants generated from the "I don't know" motion, Equation 45 produced an average deviation from the original motion and between variants on a per-DOF basis as shown in Table 60. Although the analysis was conducted for all degrees-of-freedom, only the left arm data is shown in Table 60 because the original input motion was symmetric for both arms. Therefore, one arm was sufficient to demonstrate the produced variance. Paired t-tests on each of these two distributions (per DOF) produced a p-value greater than 0.05 for all 16 controllable body DOFs, which means that on a per-DOF basis, each of these distribution pairs are not statistically different. The p-values for all distribution pairs on a per-DOF basis exceeded 0.05 for both deviation from original and inter-variant variance. In Sections 4.2.6 to 4.2.8, I established that my variance algorithm produces noticeable variants without constraints. From the first analysis of Experiment 3, my variance algorithm produces variants *with internal constraints* that are not statistically different than the noticeable variants that are produced *without* constraints.

The sample size of 1,000 used in this analysis is large, and as sample sizes increase, sample means of distributions become more accurate with respect to the true mean. For

the particular "I don't know" motion used in this analysis, the mean value of deviation for each DOF was slightly lower in the constrained case, which supports H2 in that variance was slightly limited by adding an internal constraint.

The mean was lower slightly for constrained variance because when solving the optimization a constraint at one point affects the solution for other points in the temporal vicinity of the internally constrained point. The cost function in the optimization is biased against producing discontinuities in the trajectory (unless discontinuities exist in the original motion). Thus, even a single internally constrained point limits variance through biasing the trajectory in the temporal vicinity toward the constrained position, velocity, or torque. Since deviation is calculated based on joint position variance, joint position constraints will impact the variance to the most significant extent of all the possible internal constraints.

Comparing integration Experiments 2 and 3, external constraints more severely restricted variance in more degrees-of-freedom. However, this was strongly dependent upon the location of the external constraints. When the constraints were placed closer to the end of the kinematic chain, less per-DOF variance was produced in DOFs between the root and constraint location, since more of these parent DOFs were affected by the external constraint.

6.1.8.2.2 Constraint Magnitude

For each of the seven DOFs in each arm, there existed a single velocity constraint at the point when the arms reach the apex of the gesture. The average error in velocity magnitude from 1,000 variants with and without constraints are displayed in Table 61 as a function of arm DOF.

Since an internal velocity constraint is a "soft" constraint, it is not satisfied perfectly on the hardware. However, the distribution of constrained DOFs were closer to zero than the unconstrained case (i.e. lower error and lower standard deviation) for velocity magnitude. Comparing unconstrained and constrained data for each DOF, all 14 of these distribution pairs were statistically significant ($p<0.05$), which means that the magnitude

Table 61: Average error in velocity magnitude (standard deviation x1000) for all arm DOFs (in degrees per second x1000) taken over the first 1,000 generated variants with and without an internal velocity constraint for a shrug ("I don't know") gesture. For the constrained condition, the internal constraint is a zero velocity inflection point.

	Left		Right	
DOF	Unconstrained	Constrained	Unconstrained	Constrained
shoulder X	379 (154)	101 (32)	365 (143)	130 (29)
shoulder Z	424 (187)	180 (46)	390 (178)	154 (43)
shoulder Y	409 (192)	118 (55)	359 (193)	167 (61)
elbow X	388 (178)	187 (71)	420 (144)	192 (44)
wrist Y	321 (166)	142 (84)	364 (113)	131 (52)
wrist X	365 (208)	176 (79)	384 (169)	173 (49)
wrist Z	346 (155)	188 (60)	356 (142)	140 (54)

of the zero velocity points were significantly closer to constrained value when the constraint algorithm was used to accompany the variance algorithm. In short, it is proof that the constraint algorithm performs its job well, despite being solved simultaneously with variance. This supports H2, since the zero velocity points are more deterministic when constrained in the presence of variance.

Compare the results in Table 61 for my algorithm against the results for style based inverse kinematics, shown in Table 62. The units in Table 62 are not scaled by 1000. The style-based inverse kinematics has no method to maintain internal constraints, and therefore, as expected, the velocity errors in Table 62 are significantly larger than the velocity error presented in Table 61. Even comparing the unconstrained condition with my algorithm against style-based IK, my algorithm maintained all 14 degrees-of-freedom to a statistically smaller error ($p<0.05$) when compared to style-based IK. The result of this unpredictable velocity inflection point was that using style-based IK it was difficult to synchronize the gestures to speech because timing of the inflection point is not deterministic.

6.1.8.2.3 Constraint Timing

Instead of measuring velocity magnitude error, the temporal error was measured to determine how well the constraint is satisfied. This measure for constraint error makes more

Table 62: Average error in velocity magnitude (standard deviation) for all arm DOFs (in degrees per second) taken over the first 1000 generated variants for a shrug ("I don't know") gesture. Only data without a constraint is presented since style-based IK is not capable of handling constraints.

DOF	Left Arm	Right Arm
shoulder X	1.46 (1.62)	1.81 (1.45)
shoulder Z	1.17 (1.64)	1.44 (1.51)
shoulder Y	0.89 (1.37)	1.01 (0.60)
elbow X	1.91 (0.46)	1.56 (0.79)
wrist Y	1.04 (0.93)	1.79 (2.01)
wrist X	0.71 (0.58)	0.96 (1.82)
wrist Z	0.88 (0.72)	0.42 (0.59)

sense from the perspective of achieving task-based goals because the goal is to know that the constrained timing is more deterministic and repeatable. The average error in velocity timing from 1,000 variants with and without constraints are displayed in Table 63 as a function of arm DOF.

Table 63: Average error in velocity timing (standard deviation) for all arm DOFs (in milliseconds) taken over the first 1,000 variants with and without an internal velocity constraint for a shrug ("I don't know") gesture. For the constrained condition, the internal constraint is a zero velocity inflection point.

	Left		Right	
DOF	Unconstrained	Constrained	Unconstrained	Constrained
shoulder X	472 (348)	14 (26)	490 (373)	22 (15)
shoulder Z	489 (327)	19 (32)	508 (327)	21 (19)
shoulder Y	456 (336)	21 (21)	521 (398)	16 (24)
elbow X	602 (321)	17 (15)	578 (309)	13 (25)
wrist Y	498 (384)	14 (26)	471 (313)	15 (27)
wrist X	441 (375)	11 (24)	452 (295)	17 (26)
wrist Z	486 (366)	17 (29)	499 (401)	14 (30)

Comparing unconstrained and constrained data for each DOF, all 14 of these distribution pairs were statistically significant ($p<0.05$), which means that the timing of the zero velocity points were significantly closer to constrained value when the constraint algorithm is used to accompany the variance algorithm. The timing of the zero velocity

points was deterministic when internal constraints were included, which enabled better timing of gesture and speech.

To produce a task-based measure of success for these zero velocity points, the timing was compared to style-based inverse kinematics to demonstrate the ability of my algorithm to maintain synchronization. 50 variants were generated with my variance and internal constraints algorithms and with style-based inverse kinematics. The average timing difference between the emphasized word and apex of the gesture was recorded for all 50 generated variants. When timing of the apex of the gesture is deterministic, the appropriate time to wait between playing speech and moving is known.

For style-based IK, the algorithm does not include a method for determining the timing of the apex of the gesture, since variants are produced by blending two or more gestures. Thus, two different strategies were attempted for style-based IK: random and weight-based. In the weight-based strategy, the timing of the resultant variant apex was assumed to occur at $t_1 * w_1 + t_2 * w_2 + ... + t_N * w_N$, where t_i represents the timing of gesture i and w_i represents the weight used to blend motion i for the final gesture.

Averaging over all DOFs, the timing difference in style-based IK was 1.823 (0.147, standard deviation) seconds between the apex of the gesture and the emphasized word in the speech for the random strategy, and 1.149 seconds (0.227 standard deviation). For my variance algorithm with constraints, the average time difference was 0.017 seconds (0.012 seconds, standard deviation). Thus, synchronizing gesture and speech was more accurate because timing remained deterministic with my approach.

6.1.9 Experiment 4: Variance in Internal and External Constraints

In integration Experiments 2 and 3, I supported H2 for external and internal constraints separately. In Experiment 4, I will provide evidence in support of H2 for the combination of both types of constraints, and I demonstrate variance in the presence of internal and external constraints with the example of a synchronized dancing task.

Although there are many ways to define the task of dancing, I defined it as moving the body rhythmically to the same beat as a reference motion that is timed according to

the music (e.g. the dance partner's motion can be observed while dancing and used as a reference).

The specific task goal was to maintain the same joint velocity as the input motion at the moments when the joint velocity is zero. This is an example of internal constraints used to define a task. In the resultant combination of two algorithms (i.e. internal constraints and variance), the output variants synchronize with the original dancing motion. Additionally, I included eye gaze a part of the dancing task, imposed as a time-varying look-at constraint, that is enforced by my algorithm in Chapter 5.

6.1.9.1 Experimental Design

I concatenated two simple dance moves together to use as the input motion. In the first primitive motion, one arm moved up as the other moved down and the robot turned to the side. The other motion was a similar motion, except all DOFs moved in the opposite direction. Two representative poses from these primitives are shown in Figure 36. When concatenated, they created a cyclic, repetitive dance sequence. For my task, the internal constraints that I desire to maintain were the zero velocity points, when one motion primitive transitions to the other. These constraint points occurred only twice during each cycle of the dance.

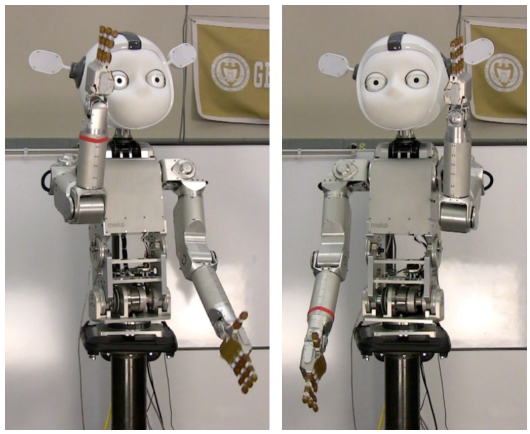

Figure 36: Representative poses of the two concatenated dancing input motions.

To measure variance, Equation 45 was used to determine deviation from the original motions, and 46 was used to compute deviation from the other dance cycles (i.e. inter-variant variance). The test conditions included the following groups:

- OV: Only the variance algorithm was used to produce typical variance in the absence of constraints.
- VI: The variance algorithm with internal synchronization constraints was used to find typical variance in the presence of only internal constraints.
- VB: Operational space control was used on the VI motion to satisfy a look-at task.

In previous experiments with more than one algorithm combined, the combination of algorithms was a serial concatenation. For Experiment 4, the internal constraint and variance were combined into a single optimal control formulation. Thus, the influence of *both* algorithms upon each other needed to be tested. I needed to test the ability to maintain the task, which was defined by how closely the output velocity points match the constraint value at the appropriate discrete time instants. For the dancing task, the constraints were tested over a series of 100 cycles per experimental condition to measure average deviation of the velocity trajectory inflection points from zero (i.e. the constrained value).

6.1.9.2 Results

The best way to verify the results is to see the varied dancing as it changes velocity in sync with the primitives from which it was generated, when both are running simultaneously. Although synchronization cannot be shown from static images, a sequence of a few poses can provide some idea of the variance. Contrast the varied poses in Figure 37 against the input motions shown in Figure 36 to see the variance induced.

Systematically, I calculated deviation from the original motion and inter-variant variance, as I had for previous experiments to support that the variance algorithm still works in the presence of internal and external constraints. The analysis was performed on a single arm since the arms are physically symmetric on the robot and the original dancing

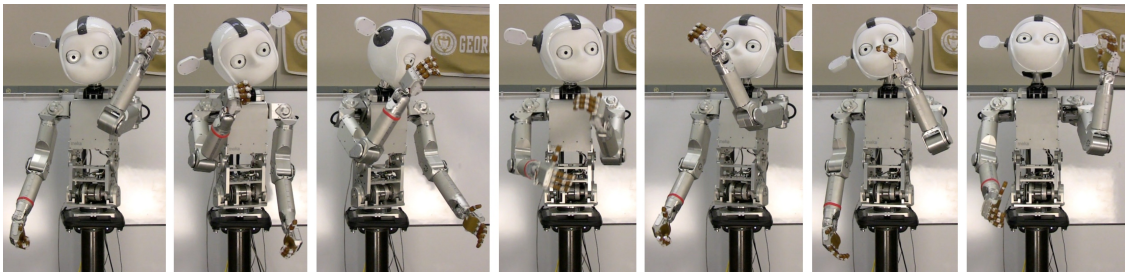

Figure 37: Representative poses from the varied dancing produced by TAV. These poses are taken at the time-dependent feature points during the task.

trajectories were out of phase by one-half of a cycle for the left and right arms. Therefore, it sufficed to analyze variance only on a single arm.

Comparing the first column (OV) with the second column (VI) in Tables 64 and 65, the variance decreased slightly in the left arm DOFs when the internal velocity constraint was added. For paired t-tests, 6 of 7 left arm DOFs and 0 of 7 eye chain DOFs between OV and VI were statistically significant for deviation with respect to the original motion. This result is consistent with H2 because the constrained velocity points during each dance cycle restrict the amount of realizable variance. The eye chain DOFs were not significantly affected by the constrained arm velocity points since they are a part of a different kinematic chain.

Similarly, comparing columns VI and VB, the left arm deviation from the input motion did not change much when the external constraint was added (1 of 7 DOFs statistically significant) because the left arm and eye chains are mostly independent. The external look-at constraint decreased variance in the DOFs that comprise the eye chain (6 of 7 statistically significant for VI vs. VB).

Inter-variant variance in Table 65 provided similar evidence to support H2 because when more constraints are added, the total potential for variance on an agent decreases (due to a smaller null space). Thus, inter-variant variance should be largest for OV, and VB should have the smallest average inter-variant variance. With one constraint (i.e. VI), 12 of 14 DOFs were statistically significant ($p<0.05$), and the two that failed the t-tests were

Table 64: Average deviation from the original motion (standard deviation) as calculated by Equation 45 for all left arm DOFs and the eye chain (in square degrees) taken over the first 100 dance cycles (i.e. variants) for three test conditions: (1) OV: without constraints, (2) VI: with only internal velocity constraints for synchronization, and (3) VB: VI with an external look-at constraint.

DOF	OV	VI	VB
shoulder X	1376 (901)	1107 (888)	1238 (865)
shoulder Z	145 (132)	92 (141)	121 (127)
shoulder Y	211 (178)	168 (182)	179 (165)
elbow X	773 (602)	690 (570)	688 (564)
wrist Y	824 (578)	704 (554)	672 (523)
wrist X	125 (80)	103 (102)	99 (71)
wrist Z	213 (145)	181 (122)	170 (117)
neck X	54 (40)	57 (47)	45 (28)
neck Y	49 (42)	42 (46)	33 (29)
top neck X	61 (59)	66 (64)	51 (51)
neck Z	116 (92)	121 (101)	104 (66)
l. eye Y	35 (21)	39 (22)	31 (29)
r. eye Y	37 (25)	41 (40)	30 (38)
eye X	121 (88)	118 (100)	105 (73)

Table 65: Average deviation between variants (standard deviation) as calculated by Equation 46 for all left arm DOFs and the eye chain (in square degrees) taken over the first 100 dance cycles (i.e. variants) for three test conditions: (1) without constraints (OV), (2) with only internal velocity constraints for synchronization (VI), and (3) VI with an external look-at constraint (VB).

DOF	OV	VI	VB
shoulder X	3506 (1253)	2489 (569)	1991 (532)
shoulder Z	755 (274)	612 (171)	489 (142)
shoulder Y	496 (168)	356 (125)	284 (108)
elbow X	962 (281)	774 (204)	622 (143)
wrist Y	923 (360)	743 (301)	595 (275)
wrist X	361 (181)	147 (79)	132 (81)
wrist Z	358 (174)	176 (122)	155 (106)
neck X	287 (132)	246 (127)	110 (75)
neck Y	623 (211)	568 (199)	491 (172)
top neck X	371 (215)	337 (204)	288 (174)
neck Z	385 (194)	322 (186)	231 (160)
l. eye Y	117 (68)	109 (73)	74 (46)
r. eye Y	110 (65)	90 (81)	71 (50)
eye X	106 (63)	88 (59)	72 (48)

in the eye chain where the internal constraint had less influence. Adding the second constraint and comparing inter-variant variance, 12 of 14 DOFs were statistically significant ($p<0.05$), and the two degrees-of-freedom that were not statistically different between one and two constraints are in the arm chain, where the added eye constraint had less impact. Variance decreased near the constraint because constraints have a more local effect (i.e. largely isolated to a single chain) as hypothesized by H2.

The preceding analysis for dancing has only demonstrated the effects of constraints on variance. The following analysis tests the effects of variance on constraints. The robot was run for 100 cycles of dancing with the variance algorithm. The sequence without variance would provide a single number of zero under the ideal conditions, and since the measurement error is likely extremely close to the measurement tolerance, the dance sequence without variance was not executed. The static value of zero was used as the reference for comparison. If the results show high average timing deviation, I can conclude that variance corrupts the ability to maintain internal constraints for this specific dancing task. This is a highly improbable result because the internal constraints are solved simultaneously with the variance, (i.e. they are taking both taken into account in the optimization and the optimal result for each should be achieved for both).

On a DOF-by-DOF basis, the experimental results are shown in Table 66. Based on the small deviation between the velocity inflection points in the varied dance motions, I concluded that variance does not corrupt the ability to maintain internal constraints.

The final analysis for Experiment 4 was meant to ensure that the look-at (i.e. external constraint) was not disrupted by either variance or the internal constraint. External constraints are always the last algorithm executed and always occupy higher priority than internal constraints. Based on data presented previously for the respective individual algorithms, there is no reason to expect that external constraints are affected by any other algorithm. For the sake of completeness, I conducted the analysis.

Two test conditions were required for the analysis: OE (only the external constraint)

Table 66: Average error in velocity timing (standard deviation) for all arm DOFs (in milliseconds) taken over the first 100 cycles with variance for a dancing task. The internal constraint is a zero velocity inflection point.

DOF	with variance
shoulder X	15 (24)
shoulder Z	21 (29)
shoulder Y	17 (20)
elbow X	16 (19)
wrist Y	12 (25)
wrist X	13 (26)
wrist Z	11 (23)

and VB (variance and both internal and external constraints). Statistical significance between the data for these groups for the dancing task will support the unexpected conclusion that variance or the internal constraint corrupts the extended operational space control algorithm's ability to maintain external constraints.

Over the duration of the entire dancing task the OE error was accumulated into a distribution for the look-at constraint error, which was 1.98 (2.31) millimeters on average for the left eye and 2.00 (2.14) millimeters for the right eye. Comparatively, when data for 100 variants that included the internal constraints for synchronizing velocity was accumulated, the distributions were 2.17 (2.14) millimeters for the left eye and 2.09 (1.99) for the right eye. Paired t-tests were not statistically significant ($p<0.05$) for these two data sets, which means that external constraint error is not statistically different when the motion includes internal constraints and variance.

6.1.10 Experiment 5: Self-Collision

Experiment 5 was designed to test H3 to characterize how frequently my algorithm and variants generated by it produce self-colliding motion. I do not hypothesize that the trajectory modulation that occurs as a result of my algorithm will never produce self-colliding trajectories. It produces human-like motion, and since humans occasionally collide due to their motion (e.g. biting their own lip, stubbing their toe) an algorithm

that produces human-like motion well should also include these rare accidents. A more important research question is what is the most appropriate human-like motion response after a collision. I will discuss this topic more in future work (Chapter 7). For now, I wanted to understand the severity of the self-collision problem for my current algorithm implementation when used without a collision detection and avoidance algorithm.

6.1.10.1 Experimental Design

To support H3, that my algorithm produces self-colliding motion from collision-free input motion very rarely, I used 19 separate collision-free motions as the input to my algorithm. I randomly applied constraints (internal and external) while transitioning between these 19 motions and I counted the number of separate occurrences of self-collision in the continuous output motion. Motion was continuous throughout the experiment. The next motion in the sequence was selected randomly and non-deterministic transitions (Section 4.2.4.3) were used to vary the starting point of the next motion. Collisions were counted during the entire experiment, including constraint and motion transition times.

The 19 motions used were: reach out, place, grab, backoff, retract arm, choo-choo, beckon, handoff, shrug, wave (two versions), turn head, one-handed bow, sneeze, throw, cradle, air guitar, bird, and 'I don't know.'

Experiment 3 was performed in the same dynamic simulation used previously so that the model was physically accurate with respect to the volume occupied by the real hardware and so that even the smallest collisions could be detected. A collision detection algorithm was used to identify when two rigid bodies representing the robot body parts penetrated each other, thus causing a self-collision event. This algorithm was used for measurement and detection only. It did not affect any motion trajectory.

6.1.10.2 Results

Three measurements were taken for Experiment 3 to help quantify which parts of my algorithm induce the self-collisions. The first measurement was taken after the human-like algorithm. All 19 motions were input to my algorithm and the output motion after the human-like optimization was simulated. None of the 19 collision-free input trajectories

experienced self-collision after the stage of the spatiotemporal optimization.

The second measurement was taken after the variance algorithm. Measurement two contained no constraints and did not include continuous motion transitioning between variants. 500 variants of each of the 19 motions were generated and simulated to detect self-collisions. Of the 9,500 (19 x 500) trajectories generated after my variance algorithm only 1 experienced self-collision (0.01%).

Measurement three is the most realistic measurement because it includes continuous motion, transitions between the 19 different motions (and variants) randomly, random application of constraints, nondeterministic transitions, constraint transients, simultaneously applied constraints, and full algorithm functionality. The simulation ran for 24 continuous hours. Each of the 19 input motions or a variant of that motion was executed at least 613 times. During the 24 hour period, 23,300 motion transitions, 40,005 constraints (at least 5,715 of each constraint type), and 80,010 constraint transitions occurred. There were 35,467 instances of simultaneously applied constraints (2-10 constraints simultaneously applied).

Over the 24 hour period, the 19 input motions and full functionality of the algorithm resulted in a total of 3 self-collisions. Maximum penetration depth of one body part into the other was 6.73 millimeters (0.26 inches).

While these results are not bad, I believe that typical humans do not collide with themselves three times each day. Improvement is still necessary to reach human-like performance with respect to collisions.

When using the algorithm in my development environment (instead of real-time on the robot), I simulate the next variant dynamically to check for self-collision while executing the current motion. If the next motion will cause self-collision, I generate a new variant instead. This provides me with extra collision protection because the odds of two self-colliding motions occurring in series is infinitesimal. According to the data from Experiment 5, it does not occur even once in a 24-hour period. For real-time performance on the hardware, I use a bounding box collision detection strategy and stop the hardware when motions cause the rigid bodies to come too close to self-collision in the accompanying

simulation. Future work is necessary to define a more human-like strategy than simply stopping the hardware when collisions are imminent.

6.1.11 Discussion

Throughout this thesis an effort is made to provide systematic ways to define the algorithms and parameters for consistent performance of my algorithms. However, someone with more experience using these algorithms can cleverly exploit the few available parameters to achieve "better" results, where "better" is defined by the person using the algorithm. For example, assume that human-likeness of the output motion is the most important criteria for a particular robot or application. The flexibility remaining in my algorithm can be harnessed to make the resultant motion more human-like. Strategically placed internal (i.e. waypoint) constraints, weight tuning on optimizations, appropriately selected alpha values for exaggeration, or diminishing variance torques can result in more human-like motion.

Whether this is seen as an advantage or disadvantage, some flexibility remains in most of the individual algorithms that can be used to enhance performance beyond what is described in this thesis. In short, the output of this algorithm still partially depends on the implementation and definition of constraints. Control is relinquished to craft the constraints in a manner that enhances motion, which also leaves the potential for problematic output when constraints are improperly defined. For example, if an constraint is specified far from the given trajectory, the algorithm will attempt to optimize the motion, for mistaken constraints too. Thus, care must be taken in constraint definition and application. At that point, the algorithm is only as intelligent as the person defining its constraints.

With respect to real-time performance, the fundamental underlying assumption remains that if the motion and internal constraints are known in advance, my algorithm can produce motion in real-time to satisfy them. External constraints are implemented as projections so they can be defined instantaneously, if desired. However, as mentioned in previous sections, avoiding extreme torque transients is desirable so as to not damage hardware. The fundamental assumption is not unrealistic, since in reality, humans need

to know where an object is before they reach for it. Thus, *a priori*, they know constraint location and motion, which are my two assumptions.

One possible disadvantage is that the output is unknown until it is executed, which leaves a lot of ambiguity since the variance algorithm can produce a wide range of variants. A wide range of variants is an advantage, and if more subtle variants are required, then the variance torques can be proportionately scaled to ensure that deviation from the original remains smaller than that presented in this thesis.

One major advantage of my algorithm is that it saves huge amounts of data storage space and data collection time. One exemplar of a motion can be used to replace the entire database, and it takes a fraction of the time to collect one exemplar, when compared to what it takes to collect, preprocess, and synthesize a database.

6.1.12 Summary

I presented three algorithms for adding communication to robot motion. I supported these algorithms as communication mechanisms by presenting data that shows that they increase recognition accuracy, decrease recognition time, allow humans to better fill in missing context, increase observer prediction of state parameters from robot motion, and improve motion appearance. I discovered a metric for the synthesis and measurement of human-like motion, and I developed a method to produce an infinite number of variants from one motion exemplar. I extended operational space control to encompass beyond just kinematic constraints, and I composed all the algorithms to demonstrate composite performance.

In closing, social robots will benefit from using my algorithm because their motion will engage human partners more by increasing the perceived entertainment value and demonstrating more emphatic motion. This leads to prolonged interaction time. If a social robot is learning from its human partner, prolonged interaction time may allow the robot perform more different examples or new tasks with its partner, thereby learning more than if the interaction ended earlier. By using my algorithm to generate motion, coordination between human and social robots increases due to increased state transparency, which

allows human partners clearer understanding of internal robot state and increased time to react. Fluidity of interaction increases because there is greater redundancy in information transfer, and communicative signals exhibit less error during transmission between collaborative partners.

CHAPTER VII

FUTURE WORK & CONCLUSION

7.1 Future Work

This thesis began by scoping out a small portion of a larger problem that could yield successful results and significant contribution in the course of a 5-year Ph.D. program. Thus, many interesting avenues of research went unexplored, left for future work. Each finding led to more hypotheses, and positive results fueled ideas and other possible directions to take my research. In the end, I have a very long list of ideas that I still want to pursue in the vein of making social robots move more like human beings, in order to communicate with humans using their motion. Due to space limitations, I cannot discuss all possible ideas that I have. However, I would like to discuss a few of these ideas. The majority of my directions for future work arise when I consider relaxing the assumptions made at the outset of this thesis or eliminating constraints imposed upon me by research conditions.

7.1.1 Other Plug-And-Play Algorithms

The algorithm to synthesize human-like motion was always designed with the intent that it would be customizable to meet the motion needs of all social robots. Algorithms could be swapped or removed when not needed without disrupting the operation of other algorithms. Thus, the most obvious extension to my work is to create additional communicative motion algorithms and add them to my pipeline.

As mentioned in numerous locations throughout this thesis, the three communication algorithms were inspired by principles of animation set forth by Thomas and Johnston [41]. There are more than just three principles of animation that are widely accepted to imbue cartoon characters with life and help them communicate with their audiences. Thus logically, these other principles might be adapted to benefit human-robot interaction. Specifically, I have hypotheses about a subset of five more of the principles of animation and what they might add to HRI. These principles are: slow in and out; timing; follow

through; overlapping action; and squash and stretch.

However, adding other principles doesn't change the underlying contribution of my thesis. I presented three methods of generating communicative motion and demonstrated their contributions for HRI. Adding others would only serve as additional support for this contribution. I believe that exploration of other communicative motion algorithms is important future work because these algorithms have the potential to add other benefits to HRI.

7.1.2 Constraints & Self-Collision

The algorithm was designed with a "plug-and-play" mentality, which does not only apply to the communicative motion generating algorithms. The controlled, experimental environment of the research lab where I perform user studies cannot compare to the real-world, where social robots will be deployed. Thus, constraints will exhibit rapid dynamics, and assumptions made about a semi-static environment (i.e. the world doesn't change for the duration it takes me to execute one motion) will no longer hold true. To satisfy these rapidly changing environmental constraints, my constraint algorithms will have to be adapted to guarantee that both internal and external constraints can be satisfied. The remainder of the algorithms do not require adjustment. For now, as long as objects don't move in the duration it takes to reach them and the world does not change at the discrete points in time when the next motion begins execution, all constraints can be satisfied.

An important consideration for robot motion is self-collision. I could implement a collision detection or collision avoidance algorithm. However, I do not believe that a human-like robot never collides with itself because humans sometimes collide with themselves (e.g. accidentally biting a lip while chewing). I believe that the occurrence of collision events is infrequent. Therefore, a human-like robot motion generation algorithm will also create motion that causes collisions infrequently. Thus, I propose that collision is a separate research question: what is the appropriate human-like response when a human body collides with itself or an object in the world? To answer this question is another year or two of research, and thus, not achievable in the time-frame of my Ph.D. thesis.

However, I am very interested in studying and implementing an algorithm that would be added to my pipeline to generate this response. Most likely it would be placed before or after constraints, since I see many similarities between the two. In fact, my extended version of OSC is capable of handling self-collision in its present form, but it is resource intensive to do so.

7.1.3 Hardware Limitations

A constraint imposed upon me based upon my research environment was that I had access to only one humanoid social robot for the majority of my work. I pursued some work in computer animation to attempt to circumvent this limitation of my work, but it is often unfair to compare results in the virtual and real worlds, since robots that operate in the real world are far more constrained. Therefore, my work to date has only been thoroughly tested on a single hardware platform, Simon (Section 2.5). Simon provides two main limitations: (1) it does not achieve the specifications of real human motion in terms of torque limits, joint velocity, and degrees-of freedom (e.g. no translatable DOFs), and (2) it is humanoid in form, which does not allow me to test operation of my algorithm on non-humanoid robots.

7.1.3.1 High Velocity

The first limitation of Simon (i.e. non-human-like DOF specifications) is not as significant of a disadvantage, since results with this limitation at least partially removed can be seen in simulation. However, I believe that to develop robot motion that is human-like, this goal must be in mind from the outset of the project, even before the robot is designed and built. Unfortunately, Simon's actuators were not designed with the torque and velocity limits of a typical human. The response of the control system is not adequate to imitate a human response without introducing artificial dynamics, since the robot was not designed this way. Thus, these constraints affected the development of my algorithms, influencing how I generate human-like motion with Simon. The advantage is that Simon is typical of most humanoid social robots, which makes my algorithm applicable across a wide breath of platforms. The disadvantage is that I cannot demonstrate on real hardware the true

power and capability of producing human-like motion until I have a platform that was designed with hardware to handle my research problem. In fact, I hypothesize that some of my findings will change when a robot can move more quickly. For example, anticipation in cartoon animation is used to prepare the audience for a subsequent fast motion, but since Simon could not move quickly, anticipation manifests as very direct motion on social robots, rather than a preparatory action like in cartoons. I hypothesize that if Simon could move more quickly, anticipation as a preparatory action would be much more useful. Since Simon moves slowly now, the preparatory actions are more direct than they are in cartoons.

Having a robot that was designed specifically for my research has another advantage. As a robot's structure begins to approach the human form in terms of kinematics and dynamics, the DOF correspondence problem becomes less problematic. Human motion capture data retargeting will work optimally when there is no kinematic or dynamic difference between the robot and human. However, motion capture trajectories still do not provide variance for social robots, which means that my algorithms are still needed. The advantage is that for my research, having a humanoid robot, which has a structure identical to human kinematics and dynamics, will allow me to make better comparisons between my results and a technique that also produces excellent results. Currently, when I make comparisons to retargeted motions, I must either show the retargeted motions on a model that looks differently than Simon (e.g. different kinematic or dynamic structure) so the motion looks good, or I make comparisons to retargeted motions displayed on Simon's architecture, which look poor due to high data loss in the retargeting process. If I had a robot with human dynamics and kinematics, my comparisons between my algorithm and retargeted motion would be more fair.

7.1.3.2 Non-Humanoid Platforms

My thesis is focused on the benefits that human-like motion can give humanoid robots in HRI. This was due to the fact that (1) I only had access to a humanoid robot; (2) humanoid robots are the most frequent type of robots used in HRI, and (3) humanoid form is the optimal platform for generating human-like motion (i.e. if I can't demonstrate results

on a humanoid robot, other platforms would only prove more difficult). However, non-humanoid robots also interact with people (e.g. most existing assembly robots in factories are not humanoid), and it is an interesting research problem to investigate if my results apply to robots that are not anthropomorphic. In other words, as the DOF correspondence problem between the human and robot worsens, what benefits for HRI are maintained? For the benefits that are lost, I would like to investigate why they are lost and if it is possible to retrieve these benefits through modifications to the algorithms. For example, factory workers can take advantage of knowing that a robot DOF is fatigued just by looking at its motion. However, if the factory robot is not humanoid in form, do robot state parameters still manifest in its motion? There is nothing in my previous work to suggest otherwise; however, it is a research question that I am interested in investigating.

7.2 Conclusion

My contribution in this thesis has been to demonstrate that by generating communicative, human-like motion, social robots can improve collaborative interaction with human partners. Communicative, human-like robot motion allows human partners to recognize robot motions earlier and more accurately, providing additional response time; furthermore it can direct gaze and allow human partners to better remember details of the interaction; by communicating missing context and state parameters, transparency is increased, which enables human partners to more easily synchronize to robot actions. My algorithm allows for a subset of degrees-of-freedom in a trajectory to be actuated and the appropriate full body response to that primary motion will be produced automatically. The human-like motion that is produced by the algorithm is shown to preserve motion intent, while producing an infinite number of motion variants, to allow social robots flexibility in movement. Four basic external constraints and three internal constraints allow robots to constrain body parts in either Cartesian or joint space to interact with their environment. Basic constraints can be combined so that social robots can satisfy more complex constraints. Error is quantified for multiple simultaneous constraints based upon priority and proximity on the kinematic hierarchy. The algorithms are combined to produce a single algorithm with

all the functionality and benefits of each of the algorithms, which requires only a single exemplar and constraints as inputs (both of which can be observed). The output of the algorithm is communicative, human-like, task-aware, varied motion that obeys constraints.

REFERENCES

[1] J. Cassell, *Gesture, speech, and sign*, L. Messing and R. Campbell, Eds. Oxford University Press, 1999.

[2] J. Cassell *et al.*, "Animated conversation: Rule-based generation of facial expression, gesture and spoken intonation for multiple conversational agents." in *Proceedings of the 21st Annual International Conference on Computer Graphics and Interactive Techniques*, 1994, pp. 413–420.

[3] C. Busso *et al.*, "Analysis of emotion recognition using facial expressions, speech, and multimodal information," in *Proceedings of the 6th International Conference on Multimodal Interfaces*, 2004, pp. 205–211.

[4] H. Wallbott and K. Scherer, "Cues and channels in emotion recognition," *Journal of Personality and Social Psychology*, vol. 51, no. 4, pp. 690–699, 1986.

[5] D. Zelter, "Motor control techniques for figure animation," *IEEE Computer Graphics and Applications*, pp. 53–59, November 1982.

[6] M. Riley *et al.*, "Methods for motion generation and interaction with a humanoid robot: Case studies of dancing and catching." *AAAI and CMU Workshop on Interactive Robotics and Entertainment*, April 2000.

[7] S. Kopp and I. Wachsmuth, "A knowledge-based approach for lifelike gesture animation." in *ECAI 2000 - Proceedings of the 14th European Conference on Artificial Intelligence*. IOS Press, 2000, pp. 663–667.

[8] B. Duffy, "Anthropomorphism and the social robot," *Robotics and Autonomous Systems*, vol. 42, no. 3-4, pp. 177–190, 2003.

[9] G. Hoffman, "Ensemble: Fluency and embodiment for robots acting with humans," Ph.D. dissertation, Massachusetts Institute of Technology (MIT), 2007, available from http://web.media.mit.edu/ guy/publications/HoffmanPhDThesis.pdf. Accessed Thursday January 15, 2009. 9:24 p.m.

[10] ——, "Robotic partners' bodies and minds: An embodied approach to fluid human-robot collaboration," *Fifth International Workshop on Cognitive Robotics*, July 2006.

[11] N. Pollard *et al.*, "Adapting human motion for the control of a humanoid robot." in *Proceedings of International Conference on Robotics and Automation*, 2002, pp. 1390–1397.

[12] V. Zordan and J. Hodgins, "Tracking and modifying upper-body human motion data with dynamic Simulation." *Computer Animation and Simulation.*, pp. 13–22, 1999.

[13] E. Westervelt *et al.*, "Experimental validation of a framework for the design of controllers that induce stable walking in planar bipeds," *International Journal of Robotics Research*, June 2004.

[14] K. Yamane *et al.*, "Human motion database with a binary tree and node transition graphs," in *Robotics Science and Systems*, 2005.

[15] S. Kim *et al.*, "Human-like arm motion generation for humanoid robots using motion capture database," in *IROS*, 2006.

[16] T. Asfour *et al.*, "Programming of manipulation tasks of the humanoid robot ARMAR," *The 9th Intl. Conf. on Advanced Robotics (ICAR 99)*, pp. 107–112, October 1999.

[17] R. Featherstone and D. Orin, "Robot dynamics: Equations and algorithms," *IEEE Intl. Conf. Robotics and Automation*, pp. 826–834, 2000.

[18] J. Kuffner *et al.*, "Motion planning for humanoid robots," *International Journal of Robotics Research*, vol. 15, pp. 365–374, 2005.

[19] S. Petti and T. Fraichard, "Safe motion planning in dynamic environments," in *Intelligent Robots and Systems*, 2005.

[20] J. Baltes and Y. Lin, *Path-tracking control of non-holonomic car-like robots using reinforcement learning*, E. P. Manuela Veloso and H. Kitano, Eds. Springer, 2000.

[21] M. Walker and C. H. Messom, "A comparison of genetic programming and genetic algorithms for auto-tuning mobile robot motion control," in *The First IEEE International Workshop on Electronic Design, Test, and Applications*, 2002.

[22] C. Zimmer, "Early signifiers." *Discover.*, May 1995.

[23] J. Hirth, "Concept for behavior generation for the humanoid robot head ROMAN based on habits of interaction." *Humanoid Robots, 2007 7th IEEE-RAS International Conference.*, November 29-December 1, Pittsburgh, USA 2007.

[24] D. Baldwin and J. Baird, "Discerning intentions in dynamic human action." *Trends in Cognitive Sciences.*, vol. 5, no. 4, pp. 171–178, 2001.

[25] D. Dennett, *The Intentional Stance.* MIT Press, 1987, ch. 3: Three kinds of intentional psychology.

[26] B. Malle *et al.*, Eds., *Intentions and intentionality.* MIT Press, 2001.

[27] B. DePaulo, "Nonverbal behavior and self-presentation." *Psychological Bulletin.*, vol. 111, pp. 203–243, 1992.

[28] J. Gnepp, "Children's social sensitivity: Inferring emotions from conflicting cues," *Developmental Psychology*, vol. 19, no. 6, pp. 805–814, November 1983.

[29] J. Cassell, "Embodied conversational agents: Representation and intelligence in user interfaces." *AI Magazine.*, vol. 22, no. 4, pp. 67–84, 2001.

[30] ——, "Embodied conversational interface agents," *Communications of the ACM*, vol. 43, no. 4, pp. 70–78, April 2000.

[31] K. Thorisson, "Communicative humanoids: A computational model of psychosocial dialogue skills." Ph.D. dissertation, Massachusetts Institute of Technology, September 1996.

[32] T. Sheridan, "Eight ultimate challenges of humanrobot communication," in *International Workshop on Robots and Human Communication*, 1997.

[33] T. Fong *et al.*, "A survey of socially interactive robots," *Robotics and Autonomous Systems*, vol. 42, 2003.

[34] T. Beth *et al.*, "Characteristics in human motion – From acquisition to analysis," *IEEE International Conference on Humanoid Robots.*, pp. 56–75, October 2003.

[35] S. Xing and I. Chen, "Design expressive behaviors for robot puppets." in *Proceedings of the 7th International Conf. Control, Automation, Robotics, Vision*, 2002, pp. 378–383.

[36] J. Mezger *et al.*, "Trajectory synthesis by hierarchical spatio–temporal correspondence: Comparison of different methods." in *Proceedings of the 2nd Symposium on Applied Perception in Graphics and Visualization*, vol. 95, 2005, pp. 25–32.

[37] V. Potkonjak *et al.*, "Modeling robot 'psycho-physical' state and reactions - A new option in human-robot communication. Part 1: Concept and background." *Journal of Intelligent and Robotic Systems.*, vol. 35, pp. 339–352, 2002.

[38] T. Fukuda *et al.*, "How far away is 'artificial man?'," *IEEE Robotics Automat. Mag.*, pp. 66–73, March 2001.

[39] P. Reitsma and N. Pollard, "Perceptual metrics for character animation: Sensitivity to errors in ballistic motion," *ACM Trans. Graph*, vol. 22, no. 3, pp. 537–542, 2003.

[40] K. Amaya *et al.*, "Emotion from motion." *Graphics Interface*, pp. 222–229, 1996.

[41] F. Thomas and O. Johnston, *Disney animation: The illusion of life*. Abbeville Press, 1981.

[42] J. Lasseter, "Principles of traditional animation applied to 3-D computer animation." *Computer Graphics*, vol. 21, no. 4, pp. 35–44, July 1987.

[43] L. Sentis and O. Khatib, "Synthesis of whole-body behaviors through hierarchical control of behavioral primitives." *International Journal of Humanoid Robotics.*, vol. 2, no. 4, pp. 505–518, Dec 2005.

[44] ——, "A whole-body control framework for humanoids operating in human environments." *IEEE Intl. Conf. on Robotics and Automation (ICRA).*, pp. 2641–2648, May 2006.

[45] O. Khatib *et al.*, "Whole-body dynamic behavior and control of human-like robots." *International Journal of Humanoid Robotics.*, vol. 1, no. 1, pp. 29–43, 2004.

[46] O. Khatib, "Dynamic control of manipulators in operational spaces." in *Proceedings Sixth World Congress on Theory of Machines and Mechanisms.*, 1983, pp. 1128–1131.

[47] ——, "A unified approach for motion and force control of robot manipulators: The operational space formulation." *International Journal of Robotics Research.*, vol. 3, no. 1, pp. 43–53, 1987.

[48] E. Todorov, *Bayesian brain.*, K. Doya, Ed. MIT Press, 2006.

[49] E. Todorov and M. Jordan, "Optimal feedback control as a theory of motor coordination," *Nature Neuroscience, (online 28)*, vol. 5, no. 11, pp. 1226–1235, October 2002.

[50] A. Kolmogorov, "Entropy per unit time as a metric invariant of automorphisms," *Doklady Akademii Nauk SSSR*, vol. 124, pp. 754–755, 1959.

[51] M. Prokopenko *et al.*, "Measuring spatiotemporal coordination in a modular robotic system," in *Proceedings of Artificial Life X*, L. Rocha *et al.*, Eds., 2006.

[52] ——, "Evolving spatiotemporal coordination in a modular robotic system," in *From animals to animats 9: 9th International Conference on the Simulation of Adaptive Behavior (SAB 2006), Rome, Italy*, S. Nolfi *et al.*, Eds., vol. 4095 of Lecture Notes in Computer Science. Springer Verlag, September 2006, pp. 558–569.

[53] O. Jenkins and M. Mataric, "A spatiotemporal extension to isomap nonlinear dimension reduction," University of Southern California Center for Robotics and Embedded Systems, Tech. Rep. CRES-04-003, 2004.

[54] P. Grassberger and I. Procaccia, "Estimation of the kolmogorov entropy from a chaotic signal," *Phys. Review A*, vol. 28, no. 4, pp. 2591–2593, 1983.

[55] J. Hahn, "Realistic animation of rigid bodies," in *Proceedings of SIGGRAPH'88. Computer Graphics*, vol. 22, no. 4, August 1988, pp. 299–308.

[56] D. Baraff, "Analytical methods for dynamic simulation of non-penetrating rigid bodies," *Proc. SIGGRAPH '89 in Computer Graphics*, vol. 23, no. 3, July 1989, Boston, MA.

[57] C. Nass *et al.*, "Computers are social actors," *CHI94, Human Factors in Computing Systems*, pp. 72–78, 1994.

[58] M. Gielniak *et al.*, "Anticipation in robot motion," *ROMAN*, 2011.

[59] J. Kim *et al.*, *Anticipation effect generation for character animation*. Springer-Verlag Berlin Heidelberg, 2006, pp. 639–646, CGI 2006, LNCS 4035.

[60] A. Nijholt *et al.*, *Verbal and Nonverbal Features of Human-Human and Human-Machine Interaction*. Springer-Verlag, 2008.

[61] L. Kovar *et al.*, "Motion graphs." in *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, July 2002, pp. 473–482.

[62] M. Mizuguchi *et al.*, "Data driven motion transitions for interactive games," *Eurographics 2001. Short Presentations*, 2001.

[63] D. Swaminathan *et al.*, *Computer Music Modeling and Retrieval. Sense of Sounds*. Springer-Verlag, 2008.

[64] M. Gielniak *et al.*, "Enhancing interaction through exaggerated motion synthesis," *HRI*, 2012.

[65] C. Howard and A. Holcombe, "Unexpected changes in direction of motion attract attention," *Attention, Perception, and Psychophysics*, vol. 72, no. 8, 2010.

[66] A. Bur *et al.*, "Dynamic visual attention: Motion direction versus motion magnitude," *Proceedings of the SPIE 20th Annual Symposium*, vol. 6806, January 2008.

[67] Y. Ye and C. Liu, "Animating responsive characters with dynamic constraints in near-unactuated coordinates," *ACM Trans. Graph.*, vol. 27, no. 5, December 2008.

[68] M. Gielniak *et al.*, "Secondary action in robot motion," *ROMAN*, 2010.

[69] ——, "Generalized motion through adaptation of velocity profiles," *ROMAN*, 2010.

[70] B. Mutlu *et al.*, "A storytelling robot: Modeling and evaluation of human-like gaze behavior," *Proc. of the IEEE-RAS Conference on Humanoid Robots (Humanoids'06)*, 2006.

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

[72] J. Laczko *et al.*, "Components of the end-effector jerk during voluntary arm movements," *Journal Applied Biomechanics*, vol. 16, pp. 14–25, 2000.

[73] M. Rosenstein, "Learning to exploit dynamics for robot motor coordination," Ph.D. dissertation, University of Massachusetts – Amherst, May 2003.

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

[75] C. Rose *et al.*, "Verbs and adverbs: Multidimensional motion interpolation using radial basis functions." *IEEE Computer Graphics and Applications.*, vol. 18, no. 5, pp. 32–40, September 1998.

[76] I. Gelfand *et al.*, *Models of the structural functional organization of certain biological systems*. Cambridge: MIT Press, 1971.

[77] Z. Li, "Functional degrees-of-freedom," *Motor Control*, vol. 10, pp. 301–310, 2006.

[78] M. Gielniak *et al.*, "Spatiotemporal correspondence as a metric for human-like robot motion," *HRI*, 2011.

[79] P. Reitsma and N. Pollard, "Perceptual metrics for character animation: Sensitivity to errors in ballistic motion," *ACM Trans. Graph*, vol. 22, no. 3, pp. 537–542, 2003.

[80] D. Wu, "Ad hoc meta-analysis of perceived error in physically simulated characters," Game Developers Convention. Microsoft Game Studios, Presentation title: Animation with momentum: Interactive performances with style and substance, 2009.

[81] L. Ren *et al.*, "A data-driven approach to quantifying natural human motion," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 1090–1097, 2005.

[82] D. Gavrila, "The visual analysis of human movement: A survey," *CVIU*, pp. 82–98, 1999.

[83] T. Moeslund and E. Granum, "A survey of computer vision-based human motion capture," *CVIU*, pp. 231–268, 2001.

[84] J. Aggarwal and Q. Cai, "Human motion analysis: A review," *CVIU*, pp. 428–440, 1999.

[85] J. Lee *et al.*, "Interactive control of avatars animated with human motion data." in *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, 2002, pp. 491–500.

[86] Y. Song *et al.*, "Unsupervised learning of human motion," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25, no. 7, pp. 814–827, July 2003.

[87] D. Oziem *et al.*, "Combining sampling and autoregression for motion synthesis," in *Proceedings of the Computer Graphics International Conference*, June 2004, pp. 510–513.

[88] K. Yamane *et al.*, "Synthesizing animations of human manipulation tasks," *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 532–539, 2004.

[89] T. Asfour *et al.*, "ARMAR-III: A humanoid platform for perception-action integration," *2nd International Workshop on Human-Centered Robotic Systems (HCRS'06)*, 2006.

[90] ——, "The humanoid robot ARMAR: Design and control," *1st IEEE-RAS International Conference on Humanoid Robots (HUMANOIDS 2000)*, September 2000.

[91] E. Westervelt and J. Grizzle, "Design of asymptotically stable walking for a 5-link planar biped walker via optimization," *ICRA*, May 2002.

[92] K. Harada *et al.*, "Natural motion generation for humanoid robots," in *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 2006, pp. 833–839.

[93] T. Flash and N. Hogan, "The coordination of arm movements: An experimentally confirmed mathematical model," *The Journal of Neuroscience*, vol. 5, no. 7, pp. 1688–1703, July 1985.

[94] K. Kondo, "Inverse kinematics of a human arm," Stanford University, Tech. Rep. CS-TR-94-1508, 1994.

[95] R. Tomovic *et al.*, "A strategy for grasp synthesis with multifingered robot hands," in *Proceedings IEEE Int. Conf. on Robotics and Automation*, 1987, pp. 83–89.

[96] T. Asfour *et al.*, "Control of ARMAR for the realization of anthropomorphic motion patterns," *Second Inter. Conf. on Humanoid Robots (HUMANOIDS 2001)*, pp. 22–24, November 2001.

[97] N. A. Bernstein, *The Coordination and Regulation of Movements*. Oxford, UK: Pergamon Press, 1967.

[98] M. Giese and T. Poggio, "Morphable models for the analysis and synthesis of complex motion pattern," *Intl. Journal of Computer Vision*, February 2000.

[99] S. Salvador and P. Chan, "Toward accurate dynamic time warping in linear time and space," *Intelligent Data Analysis*, 2007.

[100] J. Yang *et al.*, "Capturing and analyzing of human motion for designing humanoid motion," in *International Conference on Information Acquisition*, June/July 2005, pp. 332–337.

[101] B. Lay *et al.*, "Practice effects on coordination and control, metabolic energy expenditure, and muscle activation," *Human Movement Science*, vol. 21, pp. 807–830, 2002.

[102] M. Gielniak *et al.*, "Task-aware variations in robot motion," *ICRA*, 2011.

[103] M. Mancini *et al.*, "Non-verbal behaviors expressivity and their representation," LINC-Paragrape, IUT of Montreuil, University of Paris VIII, PF-star Technical Report 3, 2004.

[104] A. Sloman and M. Croucher, "Why robots will have emotions," in *Proceedings International Joint Conference on Artificial Intelligence, IJCAI*, June 1981.

[105] L. Y. Chang, S. S. Srinivasa, and N. S. Pollard, "Planning pre-grasp manipulation for transport tasks," *Proceedings of the IEEE International Conference on Robotics and Automation*, May 2010.

[106] S. LaValle and J. Kuffner, "Randomized kinodynamic planning," *Int'l. Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[107] T. Asfour and R. Dillmann, "Human-like motion of a humanoid robot arm based on a closed-form solution of the inverse kinematics problem," *IEEE/RSJ Intern. Conf. on Intelligent Robots and Systems (IROS 2003)*, pp. 1407–1412, October 2003.

[108] W. Ma *et al.*, "Modeling style and variation in human motion," *The ACM SIGGRAPH / Eurographics Symposium on Computer Animation (SCA 2010)*, 2010.

[109] K. Perlin, "Real time responsive animation with personality," *IEEE Transactions on Visualization and Computer Graphics*, vol. I, no. 1, 1995.

[110] A. Egges, T. Molet, and N. Magnenat-Thalmann, "Personalised real-time idle motion synthesis," in *Pacific Graphics*, 2004, pp. 121–130.

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

[112] A. Witkin and Z. Popović, "Motion warping," in *SIGGRAPH*, Aug. 1995.

[113] C. K. Liu, A. Hertzmann, and Z. Popović, "Learning physics-based motion style with nonlinear inverse optimization," *ACM Trans. on Graphics (SIGGRAPH)*, vol. 24, no. 3, pp. 1071–1081, 2005.

[114] F. L. Lewis and V. L. Syrmos, *Optimal Control*. Wiley-Interscience, 1995.

[115] K. Growchow *et al.*, "Style-based inverse kinematics," *ACM Trans. on Graphics (SIGGRAPH)*, vol. 23, no. 3, pp. 522–531, 2004.

[116] J. Scholz and G. Schoner, "The uncontrolled manifold concept: Identifying control variables for a functional task," *Experimental Brain Research*, no. 126, pp. 289–306, 1999.

[117] J. Yen *et al.*, "Joint-level kinetic redundancy is exploited to control limb-level forces during human hopping," *Experimental Brain Research*, vol. 196, no. 3, pp. 439–451, 2009.

[118] O. Khatib, "Inertial properties in robotic manipulation: An object-level framework," *International Journal of Robotics Research*, vol. 14, no. 1, pp. 19–36, February 1995.

[119] ——, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal of Robotics and Automation*, vol. 3, no. 1, pp. 43–53, February 1987.