# Secondary Action in Robot Motion

Michael J. Gielniak, C. Karen Liu, and Andrea L. Thomaz

***Abstract*— Secondary action, a concept borrowed from character animation, improves the animation realism by augmenting natural, passive motion to primary action. We use dynamic simulation to induce three techniques of secondary motion for robot hardware, which exploit actuation passivity to overcome hardware constraints and change the dynamic perception of the robot and its motion characteristics. Results of secondary motion due to internal and external forces are presented including discussion on how to choose the appropriate technique for a particular application.**

## I. INTRODUCTION

Our work is in the realm of social robots, which interact with people as part of their functional goal (e.g., service robots in homes, schools, or hospitals). We believe that machines designed to interact with everyday people should utilize behaviors and conventions that are socially relevant to the humans with whom they interact. This is supported by Nass' findings that effectiveness and engagement in an interaction increase when a machine exhibits human-like traits [1]. Work in conversational agents also supports the notion that natural motion of the character improves the signal-to-noise ratio in communication with a human [2]. Thus, our goal is to develop principles for generating robot motion for human-robot interaction that is "natural", "believable", or "life-like."

The fields of traditional and computer animation offer insight on producing natural motion for robots. 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 [3]. These twelve principles transcend the medium of application. 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 [4].

The larger goal of our research is to uncover methods for applying the principles of animation to improve the appearance of robot motion. Since it is widely accepted in animation that these principles imbue a character with "life" or "naturalness," it is a reasonable assumption that they can bring the same qualities to robot motion.

In this paper we focus on applying one of the twelve principles to robot motion generation: *secondary action*, which is loosely defined as the direct result of the primary, or task-oriented, action. Sometimes secondary action is considered to be the subtle details of a motion that go unnoticed if they exist, but become noticeable when they are lacking [3]. From the standpoint of Newtonian physics, secondary action 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.

We present three techniques that create natural, compliant, secondary action for humanoid robots by overcoming the hardware and software constraints that normally damp or mask secondary motion. Our approach exploits 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. Our method provides two key advantages. First, by simulating secondary action virtually, we can manipulate the visual perception of physical properties of a robot, such as making a heavy robot appear light-weight or a highly actuated robot appear compliant to perturbations. Second, our method produces 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.

## II. SECONDARY MOTION IN ROBOTICS

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 action that can noticeably result from natural physics. Mechanical limitations, such as large mass, damp transients of motion [5]. 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. Thus, secondary motion must be induced while maintaining a finite, non-zero level of active control.

M.J. Gielniak is with Department of Electrical & Computer Engineering, Georgia Institute of Technology, 777 Atlantic Dr. NW, Atlanta, GA, 30332 USA (mgielniak3@mail.gatech.edu)

C.K. Liu and A. L. Thomaz are with the School of Interactive Computing, Georgia Institute of Technology, 85 5th St. NE, TSRB 317, Atlanta, GA, 30308 USA ((karenliu,athomaz)@cc.gatech.edu)

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 [6]. If hardware and control are fixed, then to overcome this challenge, we must adapt the hardware motion to minimize exhibition of the artificial control system response (e.g. by using input command shaping) or obviate the constraint of constant control gains.

In this paper, we exploit simulation to create virtual secondary action. 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 a vision system, the simulation responds to virtual objects with characteristics different from the real world to produce different hardware responses. For example, a light box in the world can be simulated to appear heavy, and when the robot hardware lifts the box, the secondary motion non-verbally communicates different information about the world, objects in the world, or robot capabilities.

## III. Related Work

We are not aware of prior work which integrates the specific principle of secondary action into robot motion. The bulk of similar work exists in the field of computer animation where dynamic simulations are pervasive, such as producing responsive motion to external forces [7], [8], [9].

The most closely related work in computer animation is presented by Ye and Liu, which uses constrained optimization to satisfy equations of motion for "near-unactuated" degrees-of-freedom (DOFs) to create stylistic responsive motion from upper-body external perturbations [10]. Since both near-unactuated DOFs and the DOFs which are most influenced by secondary action are the less controlled DOFs for the agent, we adapt Ye and Liu's technique to identify near-unactuated DOFs. However, our algorithm calculates secondary action for internal and external forces via simulated control forces without the complexity of a constrained optimization framework.

Other robotics researchers realize the importance of bridging the gap between animation and robotics. The vast software repertoire generated by the animation community can be re-used to assist in producing natural-looking robot action [11]. End-effector velocity and jerk (the time derivative of acceleration) can act as quantitative metrics to compare motions produced by animation-inspired control to a recorded human to establish naturalness [12]. Our technique similarly borrows insights from the computer animation community. Unlike previous methods, we introduce a novel technique that integrates physics-based character animation with robotic control in an online fashion.

Existing methods for creating natural motion with secondary action are expensive and time consuming. These methods tend to rely on using databases of natural motion trajectories either from human motion capture data [13], [14] or created by a professional animator [15]. Our goal on the other hand, is to develop mechanisms to automatically generate natural motion for a robot. In this case any algorithm or animator can generate a primary or functional action for the robot to perform, and our techniques can automatically generate secondary motion for that action.

## IV. Secondary Motion Techniques

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 reduce actuation either in magnitude or dimension. The three proposed methods achieve this goal.

*1) Simulation-In-The-Loop (SIL):* The first technique determines the secondary motion that would exist in passive actuators by simulating frictionless motors unimpeded by rotor inertia and commanding the hardware to the simulated positions. SIL requires selection of the passive actuators, and we find best results work with extremity DOFs.

*2) Feedforward+Feedback Control (F&F):* 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. Our technique simulates secondary action using F&F control and commands the robot to track the simulated trajectory. In our experience with PID control on our robot hardware, simulated proportional motor feedback gains should be reduced less than 0.5% of actual hardware gains and noticeable secondary action will result.

*3) Eigenphysics+Low Gain Feedback:* Based on Ye and Liu's method for animating responsive motion [10], we 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 action. Instead of determining these under-actuated components by heuristics and hand-tuning their physical parameters, we apply eigenanalysis on the primary motion to define a new set of coordinates, ranked by the level of joint actuation in the primary motion.

Since torque is the mechanism for actuation, secondary action would logically occur in DOFs that have low variance in actuation (i.e. near-unactuated coordinates). To identify the actuation spectrum ordered by DOF for a given motion, a singular-value-decomposition is performed on the covariance matrix in equation 1 to obtain an M x M matrix of eigenvectors.

$$SVD((\tau - \mu)^T (\tau - \mu)) \tag{1}$$

where,
M = number of DOFs
$\mu_M$ = mean torque of DOF M for the entire trajectory

$\mu_1$ = mean torque of DOF 1 for the entire trajectory
T = number of equidistant time samples in the original trajectory
$\mu$ = M x T stacked matrix, where each column is $[\mu_1 ... \mu_M]^T$
$\tau$ = 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, 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.

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 2, the command torque trajectory that includes secondary action is known.

$$U^T U(\tau - \mu) + \mu \tag{2}$$

where,
P = number of eigenvectors in the mostly actuated set
$U$ = P x M column stacked matrix of mostly actuated eigenvectors

Since the dimensions of motion where active body control does not play an important role are represented by principal components with lower eigenvalues, eigenphysics can produce the secondary motion from internal forces and torques without significantly modifying the primary motion [10]. Then, eigenphysics is coupled with the very low gain feedback to recover from drift and external forces.

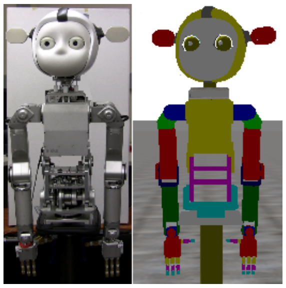

Fig. 1. Hardware (left) and simulation (right) of SIMON.

## V. Research Platform

Our robot platform is an upper-torso humanoid robot we call SIMON (Fig. 1). It has sixteen controllable DOFs on the body. The hand and head DOFs remained unactuated to produce the results in this paper. 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. However, in simulation the robot gains vary according to the techniques discussed in Sec. IV. The normal consequence of high gains is stiff robot response to external perturbations, forcing the simulation to augment motion in real-time for compliance and secondary response.

Our dynamic model of the robot hardware, for simulation, is shown in Fig. 1. This model is designed by importing 3-D meshes of the same Solidworks[1] files from which the robot parts were manufactured. Solidworks pre-calculates 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.

The simulation environment that adds physics to the virtual world uses Open Dynamics Engine (ODE), with a time-step corresponding to the actual hardware update rate. With lightweight software and low network delay, bi-directional communication between hardware and software is possible in real-time such that our robot or our simulation can act as either input or output at any given time [16], [17].

The robot hardware is not equipped with force sensors beyond the torque sensors at DOFs. Thus, under external forces, we 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 and apply this to the simulation in real-time. This results in a full simulation loop and hardware read/write cycle in real-time, commonly called hardware-in-the-loop.

## VI. Results

We use two primary motions to exemplify secondary motion using the three techniques above. The first is a tennis swing wherein the right arm is highly actuated over large angle ranges and the left arm holds a static equilibrium pose. Since secondary motion can be induced by internal forces transmitted between body segments, a primary motion with large acceleration in the joint space results in more evident secondary motion. Therefore, we designed a rapid swing motion with large disparity between the right and left arm motion. Further, we leverage the other scalar product in Newton's second law, i.e. mass or moment of inertia, to amplify the simulated secondary motion for hardware command.

To exemplify secondary motion produced from external forces, we push the robot during a common pick-and-place task and generate responsive motion from each of the three techniques. To emphasize the effect of external forces, the velocity and acceleration in this primary motion is kept low to minimize the secondary motion produced via internal forces. We estimate the force magnitude of the push from the robot joint angle sensors, amplify in simulation, and augment the secondary response for the hardware in real-time. Resultant responses may appear larger than expected for such a small push, but this is intended to demonstrate

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

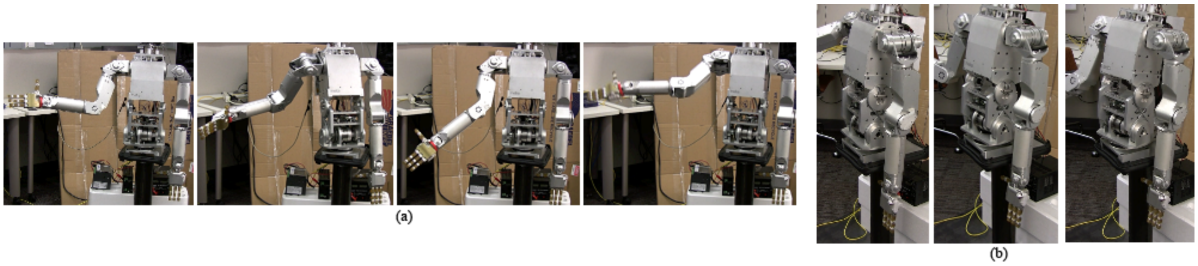

Fig. 2. (a) Key frames of tennis swing motion with entire left arm commanded to zero degrees to exemplify secondary motion in the left arm. (b) Close up of left arm secondary action hardware capability without our techniques.

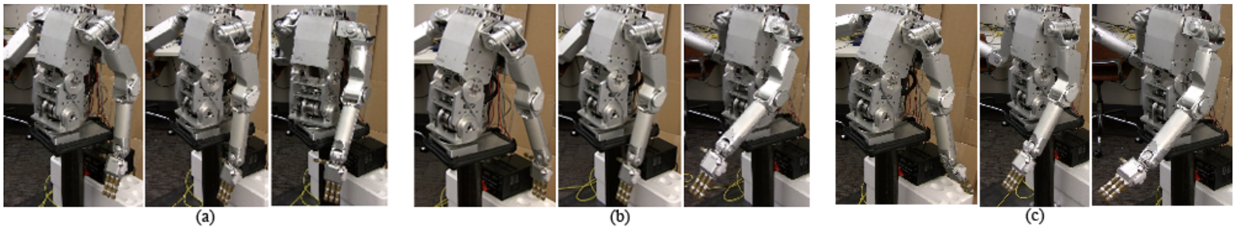

Fig. 3. Tennis swing motion with (a) simulation-in-the-loop, (b) feedforward & feedback, and (c) eigenphysics secondary action techniques.

the techniques in an non-subtle manner without forcefully pushing the robot hardware. Full trajectories with and without secondary motion for both internal and external forces can be viewed on the video accompanying this paper.

### A. Secondary Motion from Internal Forces

For each technique we demonstrate the approach with keyframes from the resulting trajectory. These keyframes are not captured at the same time instants in the trajectories. Instead, the photos show expressive moments to characterize the full trajectory produced by each technique.

Fig. 2a shows keyframes of the primary action of a right-armed tennis swing. The left arm in Fig. 2a demonstrates the minute amount of secondary motion normally exhibited when the left arm chain is holding a static zero position during the tennis swing. In the closer view of baseline trajectory (Fig. 2b), at the critical points in the motion, where the right shoulder DOFs change direction of travel, the expected response is a deflection of the torso, which should subsequently cascade to the left arm. Actuator control and the large mass of the robot diminish this response.

In simulation-in-the-loop, all DOFs in the left arm are selected as unactuated in simulation because they do not participate in the primary motion. The simulated response commanded to hardware is shown in (Fig. 3a). Compared to Fig. 2b, the shoulder, elbow, and wrist DOFs deflect and move significantly. SIL produces the smallest noticeable secondary motion of the three techniques, in terms of joint angle ranges, because the torso remains under full virtual gains, which restricts force transmission across the body.

Compared to SIL, the F&F technique in Fig. 3b induces more secondary motion in the torso. For F&F, the large left arm swings pull the torso back and forth under low simulated gains—the effect of Newton's third law. However, this impacts the left arm less than in SIL due to the low gain feedback. In SIL, the torso is under high gains, and exhibits less secondary motion, but that motion impacts the left arm more due to complete passivity of the left arm in simulation. The result is a tradeoff between passivity in the left arm and torso. For SIL, the torso remains stiff and the left arm passive, whereas in F&F both the torso and left arm are at more median amounts of actuation. Also, using F&F, induced secondary motion in the torso DOF that rotates the vertical axis is largest. Since the torso supports the right arm, it must remain actuated in SIL, and therefore cannot exhibit secondary action.

Eigenphysics in Fig. 3c alters the trajectory of all DOFs to create secondary motion everywhere, unlike SIL, which is constrained to secondary motion in the DOFs selected to be virtually unactuated. The projection that is performed by eigenphysics preserves most but not all of the primary motion, so even the primary trajectory is affected by the this technique. In eigenphysics, actuation of the left arm does not depend solely upon low gain feedback as in F&F. The actuation command is calculated directly and the left arm swings forward in a coordinated, controlled, but passive-appearing way, synchronized to movement of the torso and the right arm.

For the tennis swing motion, secondary action is dominantly produced in three DOFs. Table I, which shows variance induced by secondary action in these three left arm DOF axes, illustrates the fundamental noticeable difference in the eigenphysics motion: the left arm DOFs span a larger range of joint angles with eigenphysics than with either SIL or F&F.

### B. Secondary Motion from External Forces

Fig. 4a shows key frames of the box moving trajectory while the robot torso is pushed in the fore/aft direction in the initial frame. While under complete control, without any

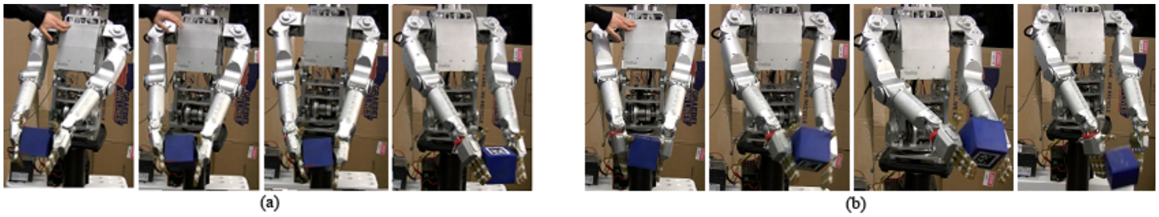

Fig. 4. Pushing the robot during a box-moving trajectory to demonstrate the response (a) without secondary augmentation and (b) using SIL.

TABLE I
VARIANCE IN POSITION FOR THREE DOMINANAT LEFT ARM DOFS VS. TECHNIQUE FOR THE TENNIS-SWING MOTION

| DOF | SIL | F&F | Eigen |
|---|---|---|---|
| Shoulder X | 0.061 | 0.069 | 0.099 |
| Elbow X | 0.051 | 0.046 | 0.064 |
| Wrist X | 0.042 | 0.056 | 0.061 |

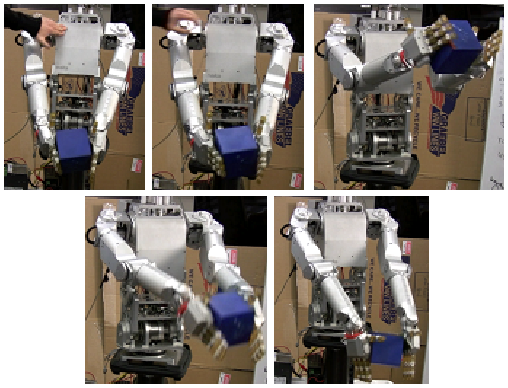

Fig. 5. Pushing the robot during a box-moving trajectory to demonstrate the response using feedforward & feedback for secondary action.

secondary action technique, the torso deflects a maximum of four degrees when perturbed. The effect is very subtle and damped after one oscillation. No other DOFs are noticeably affected by the push without using one of the three secondary action techniques.

For SIL, the elbows and two of the wrist DOFs were chosen to be unactuated in simulation. In Fig. 4b, the noticeable result of push to the torso for SIL is that the elbows curl upward. Since the wrists remain unactuated in the motion and yet are holding a box, the wrists limply clutch the box, while the other arm DOFs supply the force necessary to maintain grasp on the box. When the external push occurs on the robot, the wrists deflect 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 Fig. 4b.

In the F&F technique for the box-moving trajectory (Fig. 5), the push significantly deflects the torso which subsequently creates a force imbalance that cascades through both arms, as the low gains compensate for external perturbations. The effect is most noticeable as the box is lifted much higher in the third frame. The push is performed in a nearly symmetric location and both arms respond in similar ways, yet a slight offset in joint angles exists between the two arms. The difference is insufficient to cause the robot to drop the box. Compared with SIL and no augmentation at all, the secondary action is more pronounced in F&F because the force applied to the torso is not resisted by high actuation gains.

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. For example, comparison of the torso in Figs. 4-6 shows that eigenphysics (Fig. 6) causes the largest deviation from the box-moving trajectory for the torso fore/aft DOF, 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 combines with the actuation inherent from the transformation or from the feedforward term to impact the same DOF in different ways for each of the different techniques.

We cannot claim that one technique produces secondary motion that is the most visually pleasing. Instead, all three techniques are useful. In the next section we discuss the advantages and disadvantages of each technique, which supports identification of situations when each technique would be the appropriate choice to create secondary action.

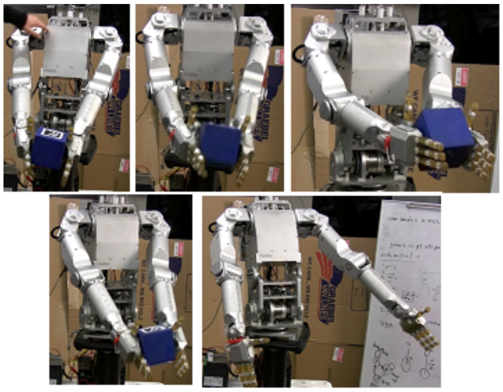

Fig. 6. Pushing the robot during a box-moving trajectory to demonstrate secondary action from external forces using eigenphysics.

## VII. DISCUSSION

Having presented three techniques for producing secondary motion, in this section we summarize the benefits and drawbacks of each from an implementation perspective.

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 action from internal forces.

If an accurate dynamic model of the robot does not exist, then it is difficult to use SIL to generate secondary action. Another consideration when using SIL is deciding which DOFs to choose as unactuated in simulation, which depends on the task. Generally, DOFs near the robot's extremities are best to select for secondary action. However, if for example a wrist DOF is being used, then it is not ideal to have it in the secondary action set. To appropriately select these DOFs for SIL, it is important to understand which ones play an essential role in the primary action. The requirement of simulated passive DOF makes SIL an unfavorable choice for producing secondary motion during manipulation trajectories, for example.

There are many advantages to F&F in creating secondary action. 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 action. 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 because the feedforward torques are often predicted through 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).

Eigenphysics is the most computationally costly of the three techniques, but we 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 action in all DOF, which makes it an optimal choice for motions such as manipulation where the primary trajectory requires minimal interruption!

## VIII. Future Work

Our literature survey uncovered no widely accepted metric for motion quality. Thus, we are currently investigating a quantitative metric to evaluate motion quality. Such a metric would be useful to measure amount of improvement for robot motion with respect to the desired criteria of naturalness. We hypothesize that spatiotemporal correspondence of distributed actuators (i.e. motor coordination) is a metric for natural motion. Research experiments are ongoing to prove our metric.

## IX. Conclusion

We have explored the concept of secondary action because we believe the same principles used to generate life-like and natural motion in animation can apply to robots. We presented three techniques for producing secondary motion for robot hardware: simulation-in-the-loop, feedforward & feedback, and eigenphysics. We demonstrated resulting motions for internal and external forces, and illustrated the trade-offs between the three techniques.

## References

[1] C. Nass, J. Steuer, and E.R. Tauber, "Computers are social actors," in *1994 Proc. SIGCHI Conf. on Human Factors in Computing Systems*, Boston, MA, pp. 72-78, Apr. 1994.

[2] J. Cassell, "Embodied Conversational Interface Agents," *Communications of the ACM*, vol. 43, no. 4, pp. 70-78, Apr. 2000.

[3] F. Thomas, and O. Johnston, *Disney Animation: The Illusion of Life*, New York: Abbeville Press, 1981.

[4] J. Lasseter, "Principles of traditional animation applied to 3D computer animation," *Computer Graphics*, vol. 21, no. 4, pp. 35-44, July 1987.

[5] N.S. Pollard, J.K. Hodgins, M.J. Riley, and C.G. Atkeson, "Adapting human motion for the control of a humanoid robot," *Proc. Intl. Conf. Robotics and Automation*, pp.1390-1397, May 2002.

[6] M.T. Rosenstein. "Learning to exploit dynamics for robot motor coordination," Ph.D. thesis, Univ. of Mass. Amherst, May 2003.

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

[8] T. Komura, E.S.L. Ho, and R.W.H. Ho, "Animating reactive motions using momentum-based inverse kinematics," *Journal Computer Animation and Virtual Worlds*, vol. 16, no. 3-4, pp. 213-223, 2005.

[9] K. Yin, M.B. Cline, and D.K. Pai, "Motion perturbation based on simple neuromotor control models," *Proc. Pacific Graphics*, pp. 445-449, 2003.

[10] Y. Ye, and C.K. Liu, "Animating responsive characters with dynamic constraints in near-unactuated coordinates," *ACM Trans. On Graphics*, vol. 27, no. 5, article 112 , Dec. 2008.

[11] X. Shusong, and Z. Huanyun, "From character animation to robot motion," *Sixth Intl. Conf. on Industrial Informatics INDIN 2008*, pp. 57-62, Daejeon, July 2008.

[12] M.J. Mataric, V.B. Zordan, and M.M. Williamson, "Making complex articulated agents dance," *Autonomous Agents and Multi-Agent Systems*, vol. 2, no.1, pp. 23-43, Mar.1999.

[13] L. Kovar, M. Gleicher, and F. Pighin, "Motion Graphs" *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 473-482, July 2002.

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

[15] C. Rose, B. Bodenheimer, and M.F. Cohen, "Verbs and Adverbs: Multidimensional Motion Interpolation Using Radial Basis Functions" it IEEE Computer Graphics and Applications vol. 18, no. 5, pp. 32-40, September 1998.

[16] J.K. Hahn, "Realistic animation of rigid bodies," *Proc. SIGGRAPH '88 in Computer Graphics*, vol. 22, no 4, pp. 299-208, Aug. 1988.

[17] D. Baraff, "Analytical methods for dynamic simulation of non-penetrating rigid bodies," *Proc. SIGGRAPH '89 in Computer Graphics*, vol. 23, no. 3, Boston, MA, July 1989.