REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
13-02-2014	Final Report	21-Feb-2011 - 20-Aug-2015

4. TITLE AND SUBTITLE
Final Report: Achieving Maximum Mobility and Manipulation Using
Human-like Compliant Behavior and Behavior Libraries

5a. CONTRACT NUMBER
W911NF-11-1-0098

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
0620BK

6. AUTHORS
Christopher G. Atkeson

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES
Carnegie Mellon University
5000 Forbes Avenue

Pittsburgh, PA 15213 -3589

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
59650-MS-DRP.4

12. DISTRIBUTION AVAILIBILITY STATEMENT
Approved for Public Release; Distribution Unlimited

13. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT
We implemented balance, walking, and push recovery on our humanoid robot. We developed a new approach to efficient robust policy design. We developed efficient algorithms to calculate first and second order gradients of the cost of a control law with respect to its parameters, to speed up policy optimization. This approach achieves robustness by simultaneously designing one control law for multiple models with potentially different model structures, which represent model uncertainty and unmodeled dynamics. We developed a footstep planning approach that takes into account robot dynamics. We showed that optimal stepping trajectories and trajectory cost

15. SUBJECT TERMS
robotics control

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Christopher Atkeson
UU	UU	UU	UU		19b. TELEPHONE NUMBER 412-268-5544

Report Title

Final Report: Achieving Maximum Mobility and Manipulation Using Human-like Compliant Behavior and Behavior Libraries

ABSTRACT

We implemented balance, walking, and push recovery on our humanoid robot. We developed a new approach to efficient robust policy design. We developed efficient algorithms to calculate first and second order gradients of the cost of a control law with respect to its parameters, to speed up policy optimization. This approach achieves robustness by simultaneously designing one control law for multiple models with potentially different model structures, which represent model uncertainty and unmodeled dynamics. We developed a footstep planning approach that takes into account robot dynamics. We showed that optimal stepping trajectories and trajectory cost for a walking biped robot can be encoded as a simple function of initial state and footstep sequence.

Enter List of papers submitted or published that acknowledge ARO support from the start of the project to the date of this printing. List the papers, including journal references, in the following categories:

(a) Papers published in peer-reviewed journals (N/A for none)

Received Paper

TOTAL:

Number of Papers published in peer-reviewed journals:

(b) Papers published in non-peer-reviewed journals (N/A for none)

Received Paper

TOTAL:

Number of Papers published in non peer-reviewed journals:

(c) Presentations

Number of Presentations: 0.00

Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received	Paper

TOTAL:

Number of Non Peer-Reviewed Conference Proceeding publications (other than abstracts):

Peer-Reviewed Conference Proceeding publications (other than abstracts):

Received	Paper	
08/27/2012	1.00	Christopher G. Atkeson, Eric C. Whitman. Control of Instantaneously Coupled Systems applied to humanoid walking, 2010 10th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2010). 06-DEC-10, Nashville, TN, USA. : ,
08/27/2012	2.00	Benjamin J. Stephens, Christopher G. Atkeson. Push Recovery by stepping for humanoid robots with force controlled joints, 2010 10th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2010). 06-DEC-10, Nashville, TN, USA. : ,
08/27/2012	3.00	B J Stephens, C G Atkeson. Dynamic Balance Force Control for compliant humanoid robots, 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2010). 18-OCT-10, Taipei. : ,
TOTAL:	**3**	

Number of Peer-Reviewed Conference Proceeding publications (other than abstracts):

(d) Manuscripts

Received	Paper

TOTAL:

Number of Manuscripts:

Books

Received Book

TOTAL:

Received Book Chapter

TOTAL:

Patents Submitted

Patents Awarded

Awards

Graduate Students

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Names of Post Doctorates

NAME	PERCENT_SUPPORTED
FTE Equivalent:	
Total Number:	

Names of Faculty Supported

NAME	PERCENT_SUPPORTED

FTE Equivalent:
Total Number:

Names of Under Graduate students supported

NAME	PERCENT_SUPPORTED

FTE Equivalent:
Total Number:

Student Metrics

This section only applies to graduating undergraduates supported by this agreement in this reporting period

The number of undergraduates funded by this agreement who graduated during this period:	0.00
The number of undergraduates funded by this agreement who graduated during this period with a degree in science, mathematics, engineering, or technology fields:.....	0.00
The number of undergraduates funded by your agreement who graduated during this period and will continue to pursue a graduate or Ph.D. degree in science, mathematics, engineering, or technology fields:.....	0.00
Number of graduating undergraduates who achieved a 3.5 GPA to 4.0 (4.0 max scale):.....	0.00
Number of graduating undergraduates funded by a DoD funded Center of Excellence grant for Education, Research and Engineering:.....	0.00
The number of undergraduates funded by your agreement who graduated during this period and intend to work for the Department of Defense	0.00
The number of undergraduates funded by your agreement who graduated during this period and will receive scholarships or fellowships for further studies in science, mathematics, engineering or technology fields:	0.00

Names of Personnel receiving masters degrees

NAME

Total Number:

Names of personnel receiving PHDs

NAME
Eric Whitman

Total Number: **1**

Names of other research staff

NAME	PERCENT_SUPPORTED

FTE Equivalent:
Total Number:

Sub Contractors (DD882)

Inventions (DD882)

Scientific Progress

See Attachment

Technology Transfer

Final Report: Achieving Maximum Mobility and Manipulation Using Human-like Compliant Behavior and Behavior Libraries

Chris Atkeson, CMU

CMU-Atkeson Team
W911NF1110098
CMU 25122.1.1130132
ARO PROPOSAL NO.: 59650-MS-DRP
BAA number: 10-65
Track: C: Control Methods
Technical area: Mobility and Manipulation
Lead organization: Carnegie Mellon University
Technical Point of Contact: Christopher Atkeson
Robotics Institute, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA, 15213-3815
phone: 412-681-8354; fax: 412-268-6436; cga@cmu.edu
Administrative Point of Contact: Kristen Jackson
Office of Sponsored Programs, Carnegie Mellon University
Robotics Institute, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA, 15213-3890
phone: 412-268-9527; fax: 412-268-6279; kristenr@andrew.cmu.edu

Contents

1 Statement Of The Problem Studied

Our research addressed how robots can achieve high performance levels. Major challenges include the computational cost of planning high performance behavior, and modeling errors and unmodeled dynamics. We used a variety of approaches based on optimal control and libraries of learned or planned behavior to enable robots to achieve skilled performance.

In our test domain, legged locomotion through rough terrain, we use a hierarchy of abstracted models and planning horizons to both consider a wide range of options and generate robust plans. In our hierarchical planning approach, the most abstract model ignores dynamics and chooses the path of a point (typically the robot center of mass (COM)) across a 2D cost map. This high level plan would have a long planning horizon, planning at least to the perception horizon and possibly further using any available map or building plan information. Many groups including us have used Dynamic Programming to implement search at this level [13]. The next level down searches for footsteps compatible with the COM path [10, 28, 29], typically using A* to search at this level, with the value function generated by the higher level Dynamic Programming serving as the heuristic function. This plan would only go as far as robot perception could see the terrain ahead. The next level down would search for a COM trajectory (now including velocity) and orientation trajectory that was compatible with the footstep plan [22], typically planning as far as the footstep plan reached. Footsteps are locally adjusted during this search process. It is at this level that approaches like Preview Control using the Linear Inverted Pendulum Model (LIPM) can be applied. Other useful but simplified models include decoupled dynamics derived by considering leg motion in the sagittal and coronal planes [26, 27], a 3D flywheel representing the upper body [21], and total angular momentum [20]. This plan would also only go as far as robot perception could see the terrain ahead. The bottom level generates a complete and fully coupled trajectory for the entire robot consistent with a full model of the robot dynamics. This plan would typically be computed for one or just a few steps. Obstacle avoidance would be implemented at each level. If a level was unable to find a feasible plan due to obstacles, or could only find high cost plans, it would provide the failed plan information to the higher level so that the higher level could generate a different plan that avoided the failure. For example, if a COM or full body trajectory planner failed, it would "poison" relevant footsteps and rerun the footstep planner to force it to choose a different footstep sequence. Current approaches that do consider the full dynamics of the robot typically only plan ahead for a short time interval using techniques like Quadratic Programming, due to the computational cost of planning with the full robot dynamics [11, 24].

It has become clear that unmodeled dynamics are a major issue. Unmodeled dynamics can be due to the presence of a torso, arms, head, or a jointed or flexible spine. Even if these body parts are well-controlled with reasonable damping, their presence can drive other otherwise well-controlled subsystems unstable or degrade performance. Combining simultaneous behaviors across the body creates unmodeled dynamics for each subsystem. Other sources of unmodeled dynamics include liquids in rigid or soft containers, non-rigidly attached (slung or tied) loads, and towed loads. Ground dynamics and other contact dynamics (objects, people, and other robots) contribute unmodeled dynamics, as does structural compliance. In biology, muscles, soft tissues, and flexible joints are a major source of unmodeled dynamics. As a field we tend to exacerbate the problem by ignoring actuator dynamics, ignoring state estimation dynamics, and by using simple or ap-

proximate dynamics models. Electric motors have dynamics due to inductance and transmission dynamics. Hydraulic systems have valve and oil flow dynamics. Series elastic actuators have the dynamics of the resulting spring mass system. In control, it has been known that the dynamics of a state estimator may drive an otherwise robust control design (for example an LQR design) unstable, and Loop Transfer Recovery Techniques (LTR) were developed to deal with this. In terms of simplified models, center of mass models, LIPM models, planar models, and decoupled models such as Raibert's 3 part running control and our own decoupled walking control are common.

We are developed an approach that deals with unmodeled dynamics, based on gradient-based policy optimization simultaneously using multiple models. We get efficiency from analytic gradients, and robustness from designing with multiple models. This is an ideological switch from using value iteration style optimal control design to policy optimization. We have done several simulation studies that show the new approach is promising.

2 Humanoid Robot Implementation

Ben Stephens (graduated PhD student) led an effort to implement our control approaches on a hydraulic humanoid robot, the Sarcos Primus System [24]. He focused on the problem of controlling push recovery for full-body force-controlled humanoid robots. The small base of support limits the forces and torques that can be generated to recover balance. For large pushes, a change of support can be achieved by stepping but cannot be performed instantaneously. A controller needs to reason about future actions and the effects of strict constraints on the available contact forces. The low impedance of the force-controlled joints allows the robot to achieve greater compliance during interaction with other objects and the environment, but results in less stable balance.

2.1 Optimizing The Center of Mass Trajectory

Ben Stephens's thesis demonstrated how a torque controlled robot can respond to large unknown perturbations such as pushes and uneven or unstable ground. His thesis demonstrated the use of simple models to approximate the dynamics and simplify the design of reactive balance controllers. These simple models define distinct balance recovery strategies and improve state estimation. Push Recovery Model Predictive Control (PR-MPC), an optimization-based reactive balance controller that considers future actions and constraints using a simple COM model, is presented. This controller outputs feasible controls which are realized by Dynamic Balance Force Control (DBFC), a force controller that produces full body joint torques. Push recovery, walking and other force-based tasks were tested both in simulation and in experiments on the Sarcos Primus hydraulic humanoid robot. Specific contributions include the following. 1) Push recovery strategy decision surfaces determined analytically by simple models: Using simple models of the robot dynamics and allowable reaction forces, the stability of certain push recovery strategies were described analytically and used to determine which strategies to use based on the current state of the robot. 2) Push Recovery Model Predictive Control: Model predictive control was performed using simple models, a special objective function and carefully chosen constraints to compute desired forces and footstep locations that account for future actions and constraints. This controller is run continuously online

to reactively maintain balance, recover from pushes, and perform dynamic walking. 3) Dynamic Balance Force Control: Full body joint torques was computed using simple models of the allowable contact and task forces and used to perform a variety of force-based tasks. 4) State estimation with modeling error: Constructing Kalman filters by augmenting simple models with different types of modeling error results in improved COM state estimation which is important for control. 5) Implementations on a hydraulic humanoid robot: The hydraulic humanoid robot is operated in force-control mode by first determining desired joint torques based on a rigid body dynamics model and then converting to a valve command using an independent joint level force-feedback controller. Implementations presenting in the thesis include standing balance, lifting heavy objects, push recovery using hip strategy and stepping, dynamic walking and dancing in place using human motion capture.

2.2 Inverse Dynamics Based on Quadratic Programming

Material from this section is discussed more fully in Chapter 5 and 7 of [24].

This section describes the current implementation of the full-body controller on our Sarcos humanoid robot. The biped dynamics equations can be augmented with additional virtual constraints, or objectives. The resulting set of equations can be solved for the joint torques that can realize those objectives. When many objectives are added, the system of equations may become over-constrained, and can be solved using weighted least-squares. The weights have to be adjusted to describe the relative importance of the objectives. This weight-tuning is often done manually. The objectives used in the weighted-objective inverse dynamics include linear and angular momentum regulation, torso angle regulation, torque minimization, and desired COP control.

$$\begin{bmatrix} \mathbf{M} & -\mathbf{S} & -\mathbf{J}^T \\ \mathbf{J} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{D}_1 \\ \mathbf{0} & \mathbf{0} & \mathbf{D}_2 \\ \mathbf{J_T} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{D_C} \end{bmatrix} \begin{pmatrix} \ddot{\mathbf{q}} \\ \tau \\ \mathbf{F} \end{pmatrix} = \begin{pmatrix} -\mathbf{N} \\ \dot{\mathbf{P}}_{\text{des}} - \dot{\mathbf{J}}\dot{\mathbf{q}} \\ m\ddot{\mathbf{C}}_{\text{des}} + \mathbf{F}_g \\ \ddot{\mathbf{H}}_{\text{des}} \\ \mathbf{K_{p\theta}}(\theta_{\text{des}} - \theta) - \mathbf{K_{d\theta}}\dot{\theta} \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix} \tag{1}$$

where $\mathbf{J_T}$ is the Jacobian associated with the torso angle, $\mathbf{K_{p\theta}}$ and $\mathbf{K_{d\theta}}$ are torso gains, and $\mathbf{D_C}$ relates the contact forces to the COP such that

$$\mathbf{D_C} = \begin{bmatrix} 0 & 0 & P_{LX} - COP_{X-des} & 0 & -1 & 0 & 0 & 0 & P_{RX} - COP_{X-des} & 0 & -1 & 0 \\ 0 & 0 & P_{LY} - COP_{Y-des} & 1 & 0 & 0 & 0 & 0 & P_{RY} - COP_{Y-des} & 1 & 0 & 0 \end{bmatrix} \tag{2}$$

In order to solve Eq. (1), the objectives are weighted by multiplying each side of the equation by a matrix $\mathbf{W_I D}$ that is diagonal in the objective weights such that

$$\mathbf{W_I D} = \text{diag}\left(\left[\mathbf{w_{DYN}}^T, \mathbf{w_{CON}}^T, \mathbf{w_{COM}}^T, \mathbf{w_{ANG}}^T, \mathbf{w_{HIP}}^T, \mathbf{w_{TOR}}^T, \mathbf{w_{COP}}^T\right]\right) \tag{3}$$

Each of these weight vectors correspond to the objective rows in Eq. (1). Example values are given in Table 1. The dynamics and constraints are given the highest weight. When controlling a

Weight	Value
$\mathbf{w_{DYN}}$	1.0
$\mathbf{w_{CON}}$	1.0
$\mathbf{w_{COM}}$	$[0, 0, 1.0e^{-2}]$
$\mathbf{w_{ANG}}$	$1.0e^{-4}$
$\mathbf{w_{HIP}}$	$1.0e^{-2}$
$\mathbf{w_{TOR}}$	$1.0e^{-4}$
$\mathbf{w_{COP}}$	$1.0e^{-2}$

Table 1: Objective weight values for weighted objective inverse dynamics control.

desired COP, the x and y components of the COM objective are given zero weights. The angular momentum objective can be given a low weight to allow the optimization to sacrifice angular momentum regulation in order to achieve better COM control.

Given an equation of the form $\mathbf{WAx} = \mathbf{Wb}$, the unknown vector $\mathbf{x}$ can be solved using quadratic programming. QP is used because of constraints on the elements of $\mathbf{x}$ such as COP constraints and torque limits. This QP optimization is run at every time step, the torques are extracted from $\mathbf{x}$ and are applied to the robot. Generally, low gain PD controls are also added to these torques to stabilize the system and bias towards a reference pose.

3 Fast Optimization of Bipedal Walking

In this section we describe our work on abstract models and fast optimization of the COM trajectory of bipedal walking using time varying linearized models. In planning stepping patterns over rough terrain, we need to estimate costs of arbitrary stepping patterns. We generalize Preview Control to linear dynamic models that include angular momentum of the torso and swing and stance legs. and use it to predict the costs of arbitrary stepping patterns. Our key idea is to use a second order gradient-based trajectory optimization method, Differential Dynamic Programming (DDP), so that optimal robot trajectories can be found and movement timing adjusted based on the Hessian (2nd derivative) of the cost with respect to the trajectory. Given linear dynamic models and quadratic cost models, this approach finds the optimal trajectory after two backward and two forward passes along the candidate trajectory. The approach can also be applied to nonlinear dynamics and more complex nonlinear cost functions, with more passes along the trajectory typically required and the risk of a local optimum. Our approach can take advantage of simpler nonlinearities such as bilinear dynamics and low order polynomial costs and optimize those more quickly. Our approach has its roots in Dynamic Programming, and the computation of a value function and its derivatives is a key component.

3.1 Dynamic Programming

Dynamic Programming provides a way to find globally optimal control laws (policies), $\mathbf{u} = \mathbf{u}(\mathbf{x})$, which give the appropriate action $\mathbf{u}$ for any state $\mathbf{x}$ [8, 9]. Dynamic Programming takes as input a one step cost function $L(\mathbf{x}, \mathbf{u})$ (a.k.a. "reward" or "loss") function and the dynamics of the problem to be optimized (expressed in discrete time): $\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k)$. We use discrete time in this paper, with k as a time index. It is equally easy to use continuous time.

One approach to Dynamic Programming is to approximate the value function $V(\mathbf{x})$ (the optimal total future cost from each state $V(\mathbf{x}) = \min_{\mathbf{u}_k} \sum_{k=0}^{\infty} L(\mathbf{x}_k, \mathbf{u}_k)$), by repeatedly solving the Bellman equation $V(\mathbf{x}) = \min_{\mathbf{u}}(L(\mathbf{x}, \mathbf{u}) + V(\mathbf{f}(\mathbf{x}, \mathbf{u})))$ at sampled states $\mathbf{x}_j$ until the value function estimates have converged. Typically the value function and control law are represented on a regular grid. Some type of interpolation is used to approximate these functions within each grid cell. If each dimension of the state and action is represented with a resolution R, and the dimensionality of the state is d_x and that of the action is d_u, the computational cost of the conventional approach is proportional to $R^{d_x} \times R^{d_u}$ and the memory cost is proportional to R^{d_x}. This is known as the Curse of Dimensionality [8].

Our group has proposed several Dynamic Programming approaches that produce globally optimized value functions that can be used in a value function-based planning approach [2, 18, 25, 17, 3, 6, 5, 7, 26, 23, 27].

3.2 Differential Dynamic Programming

To avoid the curse of dimensionality while still solving linear and nonlinear problems efficiently, we use Differential Dynamic Programming [12, 14, 2, 4], which is a trajectory optimization approach which applies Dynamic Programming along a trajectory. This approach can find globally optimal trajectories for problems with time-varying linear dynamics and quadratic costs, and rapidly converges to locally optimal trajectories for problems with nonlinear dynamics or costs. Globally optimal solutions for higher order dynamics and costs, such as bilinear or quadratic dynamics and quartic costs, can also be found using a small number of "passes" through the trajectory using higher order versions of DDP. This approach modifies (and complements) existing approximate Dynamic Programming approaches in a numbers of ways: 1) We approximate the value function and policy using many local models (quadratic for the value function, linear for the policy) along the trajectory. 2) We use trajectory optimization to directly optimize the sequence of commands $\mathbf{u}_{0,N-1}$ and the corresponding states $\mathbf{x}_{1,N}$. 3) Refined local models of the value function and policy (control law) are created as a byproduct of our trajectory optimization process.

3.3 Local models of the value function and policy

We represent value functions and policies using Taylor series approximations at each time point. At each sampled state $\mathbf{x}^p$ the local quadratic model for the value function is:

$$V^p(\mathbf{x}) = V_0^p + \mathbf{V}_{\mathbf{x}}^p \hat{\mathbf{x}} + \frac{1}{2} \hat{\mathbf{x}}^{\mathrm{T}} \mathbf{V}_{\mathbf{xx}}^p \hat{\mathbf{x}} \tag{4}$$

where $\hat{\mathbf{x}} = \mathbf{x} - \mathbf{x}^p$ is the vector from the sampled state $\mathbf{x}^p$ to the query $\mathbf{x}$, V_0^p is the constant term, $\mathbf{V}_\mathbf{x}^p$ is the first derivative with respect to state at $\mathbf{x}^p$, and $\mathbf{V}_{\mathbf{xx}}^p$ is the second spatial derivative at $\mathbf{x}^p$. The local linear model for the policy is:

$$\mathbf{u}^p(\mathbf{x}) = \mathbf{u}_0^p - \mathbf{K}^p \hat{\mathbf{x}} \tag{5}$$

where $\mathbf{u}_0^p$ is the constant term, and $\mathbf{K}^p$ is the first derivative of the local policy with respect to state at $\mathbf{x}^p$ and also the gain matrix for a local linear controller. V_0, $\mathbf{V}_\mathbf{x}$, $\mathbf{V}_{\mathbf{xx}}$, and $\mathbf{K}$ are stored with each sampled state.

These local models are created using Differential Dynamic Programming (DDP) [12, 14, 2, 4]. This local trajectory optimization process is similar to linear quadratic regulator design in that a value function and policy is produced. In DDP, value function and policy models are produced at each point along a trajectory. Suppose at a time step i we have 1) a local second order Taylor series approximation of the optimal value function:

$$V^i(\mathbf{x}) = V_0^i + \mathbf{V}_\mathbf{x}^i \hat{\mathbf{x}} + \frac{1}{2} \hat{\mathbf{x}}^\mathrm{T} \mathbf{V}_{\mathbf{xx}}^i \hat{\mathbf{x}} \tag{6}$$

where $\hat{\mathbf{x}} = \mathbf{x} - \mathbf{x}^i$. 2) a local second order Taylor series approximation of the robot dynamics ($\mathbf{f}_x^i$ and $\mathbf{f}_u^i$ correspond to $\mathbf{A}$ and $\mathbf{B}$ of the linear plant model used in linear quadratic regulator (LQR) design):

$$\mathbf{f}^i(\mathbf{x}, \mathbf{u}) = \mathbf{f}_0^i + \mathbf{f}_x^i \hat{\mathbf{x}} + \mathbf{f}_u^i \hat{\mathbf{u}} + \frac{1}{2} \hat{\mathbf{x}}^\mathrm{T} \mathbf{f}_{xx}^i \hat{\mathbf{x}} + \hat{\mathbf{x}}^\mathrm{T} \mathbf{f}_{xu}^i \hat{\mathbf{u}} + \frac{1}{2} \hat{\mathbf{u}}^\mathrm{T} \mathbf{f}_{uu}^i \hat{\mathbf{u}} \tag{7}$$

where $\hat{\mathbf{u}} = \mathbf{u} - \mathbf{u}^i$, and 3) a local second order Taylor series approximation of the one step cost, which is often known analytically ($\mathbf{L}_{\mathbf{xx}}$ and $\mathbf{L}_{\mathbf{uu}}$ correspond to $\mathbf{Q}$ and $\mathbf{R}$ of LQR design):

$$L^i(\mathbf{x}, \mathbf{u}) = L_0^i + \mathbf{L}_\mathbf{x}^i \hat{\mathbf{x}} + \mathbf{L}_\mathbf{u}^i \hat{\mathbf{u}} + \frac{1}{2} \hat{\mathbf{x}}^\mathrm{T} \mathbf{L}_{\mathbf{xx}}^i \hat{\mathbf{x}} + \hat{\mathbf{x}}^\mathrm{T} \mathbf{L}_{\mathbf{xu}}^i \hat{\mathbf{u}} + \frac{1}{2} \hat{\mathbf{u}}^\mathrm{T} \mathbf{L}_{\mathbf{uu}}^i \hat{\mathbf{u}} \tag{8}$$

Given a trajectory, one can integrate the value function and its first and second spatial derivatives backwards in time to compute an improved value function and policy. We utilize the "Q function" notation from reinforcement learning: $Q(\mathbf{x}, \mathbf{u}) = L(\mathbf{x}, \mathbf{u}) + V(\mathbf{f}(\mathbf{x}, \mathbf{u}))$. The backward sweep takes the following form (in discrete time):

$$\mathbf{Q}_\mathbf{x}^i = \mathbf{L}_\mathbf{x}^i + \mathbf{V}_\mathbf{x}^i \mathbf{f}_x^i; \quad \mathbf{Q}_\mathbf{u}^i = \mathbf{L}_\mathbf{u}^i + \mathbf{V}_\mathbf{x}^i \mathbf{f}_u^i \tag{9}$$

$$\mathbf{Q}_{\mathbf{xx}}^i = \mathbf{L}_{\mathbf{xx}}^i + \mathbf{V}_\mathbf{x}^i \mathbf{f}_{xx}^i + (\mathbf{f}_x^i)^\mathrm{T} \mathbf{V}_{\mathbf{xx}}^i \mathbf{f}_x^i \tag{10}$$

$$\mathbf{Q}_{\mathbf{ux}}^i = \mathbf{L}_{\mathbf{ux}}^i + \mathbf{V}_\mathbf{x}^i \mathbf{f}_{ux}^i + (\mathbf{f}_u^i)^\mathrm{T} \mathbf{V}_{\mathbf{xx}}^i \mathbf{f}_x^i \tag{11}$$

$$\mathbf{Q}_{\mathbf{uu}}^i = \mathbf{L}_{\mathbf{uu}}^i + \mathbf{V}_\mathbf{x}^i \mathbf{f}_{uu}^i + (\mathbf{f}_u^i)^\mathrm{T} \mathbf{V}_{\mathbf{xx}}^i \mathbf{f}_u^i \tag{12}$$

$$\Delta \mathbf{u}^i = (\mathbf{Q}_{\mathbf{uu}}^i)^{-1} \mathbf{Q}_\mathbf{u}^i; \quad \mathbf{K}^i = (\mathbf{Q}_{\mathbf{uu}}^i)^{-1} \mathbf{Q}_{\mathbf{ux}}^i \tag{13}$$

$$\mathbf{V}_\mathbf{x}^{i-1} = \mathbf{Q}_\mathbf{x}^i - \mathbf{Q}_\mathbf{u}^i \mathbf{K}^i; \quad \mathbf{V}_{\mathbf{xx}}^{i-1} = \mathbf{Q}_{\mathbf{xx}}^i - \mathbf{Q}_{\mathbf{xu}}^i \mathbf{K}^i \tag{14}$$

where subscripts indicate derivatives and superscripts indicate the trajectory index. After the backward sweep, forward integration can be used to update the trajectory itself:

$$\mathbf{u}_{new}^i = \mathbf{u}^i - \Delta \mathbf{u}^i - \mathbf{K}^i (\mathbf{x}_{new}^i - \mathbf{x}^i) \tag{15}$$

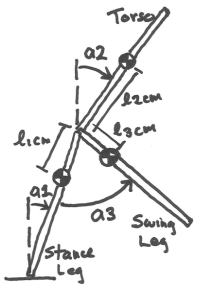


Figure 1: Example System.

We note that the cost of this approach grows at most cubically rather than exponentially with respect to the dimensionality of the state. When the dynamics are linear and the cost function is quadratic in the states and actions, DDP finds the globally optimal solution after a backward sweep to compute the second order terms such as $\mathbf{V_{xx}}$ and $\mathbf{K}$, a forwards trajectory simulation, a backward sweep to compute the first order terms such as $\mathbf{V_x}$ and $\Delta\mathbf{u}$, and a second forwards trajectory simulation. The initial backward sweep can be performed before any footstep constraints are known, as the ZMP locations only affect $\mathbf{V_x}$ and $\Delta\mathbf{u}$.

3.4 Example

We will show how our approach can be applied to a linear 3D model that includes angular momentum of the legs and torso. Figure 1 shows the system, with a torso and two legs. For this simple example the models in the sagittal and coronal directions are the same. In each direction the first element of the configuration $a1$ is the stance ankle angle. The second element of the configuration $a2$ is the angle of the torso with respect to vertical. The third element of the configuration $a3$ is the angle of the swing leg with respect to the stance leg. The stance leg has mass $m1 = 10.9kg$, moment of inertia $I1 = 0.57kgm^2$ about the center of mass (COM), and length $l1 = 0.77m$. The torso has mass $m2 = 65kg$ and moment of inertia $I2 = 4.3kgm^2$ about the COM, and the swing leg has mass $m3 = 12.6kg$ and moment of inertia $I3 = 0.85kgm^2$ about the COM. The locations of the centers of mass of each link relative to the hip are given by $l1cm = 0.37m$, $l2cm = 0.4m$, and $l3cm = 0.43m$. The linearized dynamics in both the sagittal and coronal planes are given by:

$$\ddot{\mathbf{q}} = \mathbf{M}^{-1}(\mathbf{G}\mathbf{q} + \boldsymbol{\tau}) \tag{16}$$

with $\mathbf{q} = (a1, a2, a3)$. The elements of $\mathbf{M}$ are given by: $m_{11} = I1 + I3 + m1 * (l1 - l1cm)^2 + m2 * l1^2 + m3 * (l1 - l3cm)^2$, $m_{12} = m_{21} = m2 * l1 * l2cm$, $m_{13} = m_{31} = -I3 + m3 * l3cm * (l1 - l3cm)$, $m_{22} = I2 + m2 * l2cm^2$, $m_{23} = m_{32} = 0$, and $m_{33} = I3 + m3 * l3cm^2$, and the elements of $\mathbf{G}$ are given by: $g_{11} = g * ((l1 - l1cm) * m1 + l1 * m2 + (l1 - l3cm) * m3)$, $g_{12} = 0$, $g_{13} = g * l3cm * m3$, $g_{21} = 0$, $g_{22} = g * l2cm * m2$, $g_{23} = 0$, $g_{31} = g * l3cm * m3$, $g_{32} = 0$, and $g_{33} = g * l3cm * m3$. The Coriolis and centripetal forces do not appear in a linearization about zero velocity. This results in a linearized model with state $\mathbf{x} = (a1, \dot{a1}, a2, \dot{a2}, a3, \dot{a3})$ and action $\mathbf{u} = (\tau_1, \tau_2, \tau_3)$, which includes the torques on the stance leg at the stance ankle, the torso at the hip, and the swing leg at the hip.

$$\mathbf{x}_{k+1} = \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k \tag{17}$$

$$\mathbf{A} = \begin{pmatrix} 1.0018 & 0.01 & -0.0011 & 0 & 0.0002 & 0 \\ 0.3565 & 1.0 & -0.2211 & 0 & 0.0444 & 0 \\ -0.0024 & 0 & 1.0024 & 0.01 & -0.0003 & 0 \\ -0.4862 & 0 & 0.4746 & 1.0 & -0.0605 & 0 \\ 0.0003 & 0 & 0.0003 & 0 & 0.9991 & 0.01 \\ 0.0551 & 0 & 0.0691 & 0 & -0.1805 & 1.0 \end{pmatrix} \tag{18}$$

$$\mathbf{B} = 0.001 * \begin{pmatrix} 0.0032 & -0.0043 & -0.0010 \\ 0.6355 & -0.8667 & -0.1987 \\ -0.0043 & 0.0093 & 0.0014 \\ -0.8667 & 1.8608 & 0.2710 \\ -0.0010 & 0.0014 & 0.0160 \\ -0.1987 & 0.2710 & 3.1944 \end{pmatrix} \tag{19}$$

In this example we only model a standing phase (one second) followed by single support phases with no intervening double support phase. The discussion section describes how double support phases can be added. The timing and location of each foot step is specified, and the goal is to find a trajectory of the system that minimizes the optimization criteria in each direction:

$$cost = \sum (\mathbf{x}^{\mathrm{T}} \mathbf{Q} \mathbf{x} + \mathbf{u}^{\mathrm{T}} \mathbf{R} \mathbf{u})/2 \tag{20}$$

where $\mathbf{Q} = diag(0.01, 0.01, 0.01, 0.01, 1, 0.01)$ and $\mathbf{R} = diag(1, 0.01, 0.01)$. Because the system is unstable in balance with respect to the center of pressure (COP) or zero moment point (ZMP), the effect of this cost function is to drive the system to balance over the specified foot locations. Penalty functions are used to drive the swing leg to the next foot location with zero velocity at touchdown, implementing this requirement as soft constraints. $\mathbf{V_{xx}}$ is initialized to the quadratic cost function generated by a steady state linear quadratic regulator (dlqr(A,B,Q,R) in Matlab).

At this point the second order quantities and feedback gains can be computed backwards in time:

$$\mathbf{Q}^i_{\mathbf{xx}} = \mathbf{L}^i_{\mathbf{xx}} + (\mathbf{f}^i_x)^{\mathrm{T}} \mathbf{V}^i_{\mathbf{xx}} \mathbf{f}^i_x \tag{21}$$

$$\mathbf{Q}^i_{\mathbf{ux}} = \mathbf{L}^i_{\mathbf{ux}} + (\mathbf{f}^i_u)^{\mathrm{T}} \mathbf{V}^i_{\mathbf{xx}} \mathbf{f}^i_x \tag{22}$$

$$\mathbf{Q}^i_{\mathbf{uu}} = \mathbf{L}^i_{\mathbf{uu}} + (\mathbf{f}^i_u)^{\mathrm{T}} \mathbf{V}^i_{\mathbf{xx}} \mathbf{f}^i_u \tag{23}$$

$$\mathbf{K}^i = (\mathbf{Q}^i_{\mathbf{uu}})^{-1} \mathbf{Q}^i_{\mathbf{ux}} \tag{24}$$

$$\mathbf{V}^{i-1}_{\mathbf{xx}} = \mathbf{Q}^i_{\mathbf{xx}} - \mathbf{Q}^i_{\mathbf{xu}} \mathbf{K}^i \tag{25}$$

Since the system is linear, $\mathbf{f}_{xx}$, $\mathbf{f}_{ux}$, and $\mathbf{f}_{uu}$ are all zero and the second order quantities can be precomputed without knowledge of the actual footstep locations or system trajectory. This computation does require knowledge of the footstep timing in order to enforce the footstep touchdown constraints.

The dynamics are now integrated forward in time from a known initial state, using the optimized feedback gains. The resulting trajectory does use optimal feedback gains, but does not use knowledge of the future (preview control). This forward pass requires knowledge of the footstep locations and timing, as it is creating costs that will be compensated for in the optimization process.

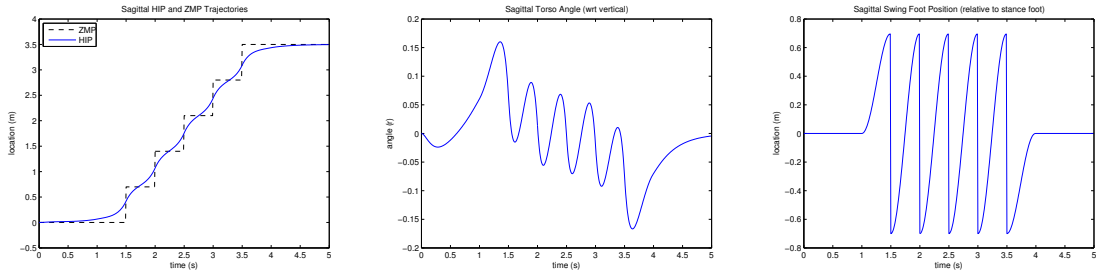


Figure 2: A: HIP and ZMP Sagittal Trajectories, B: Torso Sagittal Trajectory, C: Foot Sagittal Trajectory.

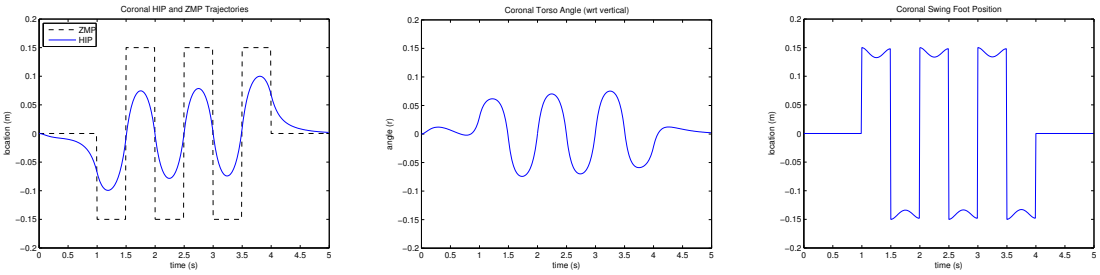


Figure 3: A: HIP and ZMP Coronal Trajectories, B: Torso Coronal Trajectory, C: Foot Coronal Trajectory.

Now the first order quantities are integrated backwards in time, bringing information to the present about future footstep locations and other constraints, and future costs:

$$\mathbf{Q}_{\mathbf{x}}^{i} = \mathbf{L}_{\mathbf{x}}^{i} + \mathbf{V}_{\mathbf{x}}^{i}\mathbf{f}_{x}^{i} \tag{26}$$

$$\mathbf{Q}_{\mathbf{u}}^{i} = \mathbf{L}_{\mathbf{u}}^{i} + \mathbf{V}_{\mathbf{x}}^{i}\mathbf{f}_{u}^{i} \tag{27}$$

$$\Delta\mathbf{u}^{i} = (\mathbf{Q}_{\mathbf{uu}}^{i})^{-1}\mathbf{Q}_{\mathbf{u}}^{i} \tag{28}$$

$$\mathbf{V}_{\mathbf{x}}^{i-1} = \mathbf{Q}_{\mathbf{x}}^{i} - \mathbf{Q}_{\mathbf{u}}^{i}\mathbf{K}^{i} \tag{29}$$

The dynamics are now integrated forward in time a second time from the known initial state, using both the optimized feedback gains and feedforward command $\Delta\mathbf{u}$. The optimal trajectory is the result (Figures 2-3).

3.5 Discussion

Although this is a trajectory optimization approach, it is very fast and can be applied online. The computation costs are comparable to Preview Control [15]. For linear dynamics and quadratic costs, the second order quantities can be precomputed, just as they are in Preview Control. Our approach avoids the need to plan in terms of the third derivative of position variables (jerk), and the need to use integral control to accurately maintain constraints. In the linear dynamics/quadratic

cost case additional iterations are not required to handle complex models, as is done in [19]. Optimal time varying control gains are produced for online feedback control. The value function used by this approach can be used to optimize the timing of footsteps, as described below. Footstep locations can be locally optimized given a quadratic cost function for allowable footstep locations.

Our key idea is to use second order gradient-based trajectory optimization, so that optimal robot trajectories can be found and movement timing adjusted based on the Hessian (2nd derivative) of the cost with respect to the trajectory. Given linear dynamic models and quadratic cost models, this approach finds the optimal trajectory after two backward and two forward passes along the candidate trajectory. The approach can also be applied to nonlinear dynamics and more complex nonlinear cost functions, with more passes along the trajectory typically required and the risk of a local optimum. Our approach can take advantage of simpler nonlinearities such as bilinear or quadratic dynamics and low order polynomial (cubic or quartic) costs and optimize those more quickly.

We have demonstrated how Preview Control can be generalized to handle angular momentum of the torso and legs. In this section we discuss how other aspects of locomotion can be handled.

3.6 Adjusting Footstep Locations and Timing

It is easy in this approach to adjust footstep locations during optimization. The user must specify quadratic cost functions for each footstep location.

Using the value function, we can advance or delay future step timings (choosing j below) based on the predicted trajectory and resulting cost:

$$min_j \left[\sum_{k}^{k+j} L(\mathbf{x}_i, \mathbf{u}_i) \right] + V(\mathbf{x}_{k+j+1}) \tag{30}$$

3.7 Double Support and Using Arm and Body Contact

There are two ways to handle multiple contacts as in double support and when using an arm or having parts of the body in contact with the environment. The first is to simply allocate specified fractions of the total force to each contact. The second is to allow the optimization process to determine an optimal allocation of contact forces and torques. Our approach can allocate forces, and handle constraints on non-negativity of contact forces, in our four pass approach for linear systems, if the constraints do not change from active to inactive status, or vice versa during the optimization. The optimization must repeat backward and forward passes until the status of the constraints remain the same for four passes.

3.8 Yaw Orientation and Estimating the Cost of Turning

Estimating the cost of potentially multiple turns versus sideways steps is an important part of footstep planning. Handling a facing direction and the yaw orientation anisotropy of the body can be handled in several ways. We believe the most significant effect is a strong yaw orientation dependence of the one step cost function, reflecting the need to avoid leg collisions. Less important

orientation effects include weaker actuation and smaller range of leg motion in the sideways direction, at least for many robots if not humans. The need to point sensors in the direction of travel and orient arms can be handled largely by neck and torso twist. Viewing the torso as a cylinder and the pelvis as a point leads to approximately isotropic body dynamics with respect to pelvis and torso yaw orientation.

Yaw orientation dependence of the cost function is likely to be an "even" function due to left/right yaw symmetry, requiring a quadratic dependence on yaw, and an overall quartic cost function. As described previously, we believe this can be handled by either using second order DDP as a nonlinear trajectory optimizer, or using a higher order (fourth order) version of DDP with up to fourth order Taylor series of the value function, dynamics, cost function, and policy. We leave handling yaw orientation as future work.

3.9 Knee Angle and Height Management

Knee behavior becomes important for traversing variable height terrain and taking long steps. A linear model is not of much use for optimizing the knee angle, as minimizing knee force takes advantage of a strong kinematic non-linearity. We believe a low order polynomial approach can be used to model the knee, but leave this for future work. One approach to handle both the stance and swing knee is to generate their desired motion as a function of the overall stance or swing leg angle. It may be the case that correctly optimizing knee behavior needs to take into account the rolling of the foot from heel strike to toe off.

3.10 Other Applications

[16] describe the application of preview control for bracing for a future impact. We could apply our technique to the same problem.

4 Globally Optimal Planning Methods

Eric Whitman (graduated PhD student) led an effort to develop global optimization approaches for our humanoid robot and other complex systems by decomposing these systems into weakly coupled components. He developed an optimal controller for an Instantaneously Coupled System (ICS) which was designed by coordinating multiple lower- dimensional optimal controllers. He augmented subsystems of the ICS with coordination variables, and then used value functions to coordinate the augmented subsystems by managing trade offs of the coordination variables. He presented simulation and robot results on his globally optimal control approach. He applied this method to humanoid walking and presented a controller for a 3D simulation that uses multiple coordinated policies generated using Dynamic Programming. His controller optimizes center of mass motion as well as footstep timing and location, and it can react in real time to perturbations and accumulated modeling error. He presented walking perturbation experiments as well as standing balance results from a force controlled humanoid robot.

5 Inverted Pendulum Models

Taesoo Kwon (Postdoc) explored physical simulation of humans by abstracting human behavior via inverted pendulum/cart models. He has done this in the context of walking, running, and various gymnastic flips and jumps. The simplified inverted pendulum/cart model lacks the degrees of freedom found in human models, so he analyzed a captured reference motion in a preprocessing step and used that information about human running patterns to supplement the balance algorithms provided by the inverted pendulum. At run-time, the controller plans a desired motion at every frame based on the current estimate of the pendulum state and a predicted pendulum trajectory. By tracking this time-varying trajectory, our controller creates a running character that dynamically balances, changes speed and makes turns. The initial controller can be optimized to further improve the motion quality with an objective function that minimizes the difference between a planned desired motion and a simulated motion. He demonstrated the power of his approach by generating running motions at a variety of speeds (3m/s to 5m/s), following a curved path, and in the presence of disturbance forces. In recent results, he has also developed control systems for simple gymnastic flips and jumps.

6 Trajectory Library

Chenggang Liu (visiting PhD student) led an effort to develop our first prototype of a behavior library using trajectory-based optimal control approaches. He explored biped standing balance and walking control using a library of optimal trajectories. These behaviors are formulated as optimal control problems. For walking, Liu took advantage of a parametric trajectory optimization method to find the periodic steady-state trajectory. He then used Differential Dynamic Programming (DDP) to generate a library of optimal trajectories and locally linear models of the optimal control law, which are used to construct a more global control law.

The utility and performance of the proposed method is evaluated using simulated walking control of a planar five-link biped robot. He used the steady-state trajectory and its local models to initialize the trajectory library. He applied horizontal impulsive perturbations of different magnitudes at the hip during walk. If the simulated robot fell down after a perturbation, He added a new trajectory segment to the library. The proposed controller using this trajectory library was evaluated using a variety of perturbations at the hip. The resulting trajectory converges to the periodic steady-state trajectory and the walking speed is maintained after the perturbation. For continuous perturbations, the controller's responses to a continuous forward push shows that the robot walks slightly faster than the desired walking speed when the push is applied. The phase portrait of the motion of one virtual leg shows that the proposed controller drives the robot's trajectory back to the limit cycle of normal walking after the push. The proposed controller was also evaluated using walking on inclines. Simulation results show the proposed controller generated using a level ground still works. The average velocity of the center of mass nearly does not change. Model errors are inevitable during system modeling, so he evaluated the proposed controller using walking control with a model error. The vertical component of the ground reaction force increases but the robot can still walk. The walking speed nearly does not change in this simulation. The controller

drives the robot's trajectory to a different limit cycle because of the model error. An analytical foot-ground contact model (rigid body impact model) is employed for offline controller construction. The resultant controller is evaluated using other foot-ground contact models, such as a spring damper model. For the spring damper contact model, the joint between the stance leg's end and the ground is modeled as a planar joint of three degrees of freedom (DoFs) and the whole system has seven DoFs. The proposed controller generated using a rigid body impact model transfers well to spring damper models of different spring constants. He compared the proposed controller with a trajectory tracking controller with fixed optimized gains. For normal walking, the cost for the trajectory tracking controller is 3.661, while the corresponding cost for the proposed controller was 2.951. For walking in the presence of an impulsive perturbation of 5 Newton-seconds, the cost for the former was 21.679, the corresponding cost for the later was 3.697. He used a parametric trajectory optimization method to solve the periodic steady-state trajectory in regular walking at a specified speed, which is used as an initial trajectory for Differential Dynamic Programming (DDP) to re-optimize and generate local models of the value function and the control law. He also used DDP to generate additional optimal trajectory segments to cover a larger portion of state space. By formulating the optimal control problem with an infinite time horizon, he got time-invariant locally linear models of the optimal control law, which are then used to construct a more global control law for biped walking. The results show lower cost from the proposed controller than a trajectory tracking controller using optimal gains. The simulated planar walking controller based on a trajectory library currently handles quick and continuous pushes, pushes of different sizes, inclines, model perturbations (wrong mass), and ground model variations (ground stiffness).

Our trajectory library approach is based on our previous work on trajectory libraries:

1. Use the best available trajectory optimizer (SQP, DDP) to locally optimize each trajectory.
2. Use second order gradient descent (DDP) to compute a second order value function approximation
$$V(\mathbf{x}) = V_0 + V_{\mathbf{x}}\mathbf{x} + 0.5\mathbf{x}^{\mathrm{T}}V_{\mathbf{xx}}\mathbf{x} \tag{31}$$
and optimal gains $\mathbf{K} = \partial\mathbf{u}/\partial\mathbf{x}$ along the trajectory.
3. Check for compatibility of value functions across neighboring trajectories to reduce the effect of local minima and policy discontinuities.
4. $V(\mathbf{x})$ can be used as terminal penalty for receding horizon or model predictive control (MPC).

7 Editing The Library

Kwang Won Sok (visiting PhD student) has explored momentum based editing techniques for trajectory library data. He presented an integrated framework for interactive editing of the momentum and external forces in a trajectory. Allowing user control of the momentum and forces provides a powerful and intuitive editing tool for dynamic motions. To make a higher jump, for example, the

user simply increases the linear momentum in the vertical direction, while our system automatically calculates a motion that maintains both the same landing position and physical plausibility. Our key insight is using trajectory optimization based on normalized dynamics to simultaneously propagate momentum and force space changes. He demonstrated the approach with edits of long sequences of dynamic actions, including kicks, jumps, and spins.

8 Using Simple Functions to Represent Plan Databases

We showed that optimal stepping trajectories and trajectory cost for a walking biped robot can be encoded as a simple function of initial state and footstep sequence. Given an initial state and the sequence of foot placements on uneven terrain, our quadratic function provides a reliable estimate of the state trajectory and the required effort for the optimal walking motion. In order to find this encoding, we built a database of optimal walking trajectories for a 3D humanoid model by sampling the input space (initial state and footstep sequence) and solving a physically-based trajectory optimization problem for each sample. Then, the function coefficients were obtained by fitting to the data using least squares. The performance of the proposed method is evaluated by comparing the function values with other optimal walking motion data generated with different footstep samples. As an application, we use the quadratic function to calculate the effort cost used in finding an optimal footstep sequence with an A* algorithm. Our study shows that a simple function can be effectively used to encode optimal walking and this provides us a fast alternative to optimizing walking dynamics of a full body model online.

9 Footstep Planning

We explored footstep planning taking into account dynamics. Most cost functions for footstep planning in the literature are designed based on only terrain information. The risk and effort necessary to achieve a pattern of footsteps is ignored or taken into account heuristically. As a first step in considering risk and effort in footstep planning, we developed cost functions modeled on those that reflect metabolic cost in human gait. This allowed us to rapidly evaluate proposed footstep patterns. We had to invent a cost function for the energy cost of turning. Our energy cost and terrain cost are combined to obtain an optimal step planning sequence using A* search.

10 Policy Optimization

We explored a policy optimization approach to designing behavior controllers. We developed a new approach to efficient robust policy design. We developed efficient algorithms to calculate first and second order gradients of the cost of a control law with respect to its parameters, to speed up policy optimization. We have also developed ways to accurately take derivatives of the dynamics of rigid body systems, and found a formulation for rigid body dynamics that further supports efficient derivatives as well as symbolic generation of the dynamics equation and reduction of the

computational cost by manually refining the code. This approach achieves robustness by simultaneously designing one control law for multiple models with potentially different model structures, which represent model uncertainty and unmodeled dynamics. Providing explicit examples of possible unmodeled dynamics during the control design process is easier for the designer and is more effective than providing simulated perturbations to increase robustness, as is currently done in machine learning. The approach supports the design of deterministic nonlinear and time varying controllers for both deterministic and stochastic nonlinear and time varying systems, including policies with internal state such as observers or other state estimators. We highlight the benefit of control laws made up of collections of simple policies where only one component policy is active at a time. Controller optimization and learning is particularly fast and effective in this situation because derivatives are decoupled.

10.1 Taking Derivatives

Taking derivatives of the dynamics is a key component of policy optimization. It is also useful in creating state estimators based on the Kalman filter, using implicit integration in simulation, performing trajectory optimization using first and second order gradient techniques like Differential Dynamic Programming, using dual control techniques to design controllers for stochastic systems, performing sensitivity analysis, and using constrained inverse dynamics.

When Lagrangian dynamics are used to eliminate constraint forces, dynamics for rigid body systems have the form:

$$\ddot{\mathbf{q}} = \mathbf{M}^{-1}(\mathbf{q}) \left(\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{G}(\mathbf{q}) + \mathcal{J}^{\mathrm{T}}(\mathbf{q}) \mathbf{u} \right) \tag{32}$$

$\mathbf{q}$ is a generalized coordinate, $\dot{\mathbf{q}}$ is a generalized velocity, and $\ddot{\mathbf{q}}$ is a generalized acceleration. $\mathbf{M}(\mathbf{q})$ is the inertia matrix, which depends on the current configuration. $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ are the Coriolis and centripetal terms, which depend on the configuration and quadratically on the velocity. $\mathbf{G}(\mathbf{q})$ are the gravity terms, which depend on the configuration. $\mathbf{u}$ are actuator forces mapped to the generalized forces by a configuration dependent Jacobian $\mathcal{J}(\mathbf{q})$.

The gradient with respect to a configuration variable $\mathbf{q}_i$ is:

$$\begin{aligned} \frac{\partial \ddot{\mathbf{q}}}{\partial \mathbf{q}_i} &= \frac{\partial \mathbf{M}^{-1}(\mathbf{q})}{\partial \mathbf{q}_i} \left(\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{G}(\mathbf{q}) + \mathcal{J}^{\mathrm{T}}(\mathbf{q}) \mathbf{u} \right) \\ &\quad + \mathbf{M}^{-1}(\mathbf{q}) \left(\frac{\partial \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})}{\partial \mathbf{q}_i} + \frac{\partial \mathbf{G}(\mathbf{q})}{\partial \mathbf{q}_i} + \frac{\partial \mathcal{J}^{\mathrm{T}}(\mathbf{q})}{\partial \mathbf{q}_i} \mathbf{u} \right) \end{aligned} \tag{33}$$

We note that

$$\frac{\partial \mathbf{M}^{-1}(\mathbf{q})}{\partial \mathbf{q}_i} = -\mathbf{M}^{-1}(\mathbf{q}) \frac{\partial \mathbf{M}(\mathbf{q})}{\partial \mathbf{q}_i} \mathbf{M}^{-1}(\mathbf{q}) \tag{34}$$

and that $\frac{\partial \mathbf{M}^{-1}(\mathbf{q})}{\partial \mathbf{q}_i}$, $\mathbf{M}^{-1}(\mathbf{q})$, and $\frac{\partial \mathbf{M}(\mathbf{q})}{\partial \mathbf{q}_i}$ are symmetric matrices, since $\mathbf{M}(\mathbf{q})$ is symmetric.

The gradient with respect to a velocity variable $\dot{\mathbf{q}}_i$ is:

$$\frac{\partial \ddot{\mathbf{q}}}{\partial \dot{\mathbf{q}}_i} = \mathbf{M}^{-1}(\mathbf{q}) \left(\frac{\partial \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}_i} \right) \tag{35}$$

The second derivative with respect to two configuration variables $\mathbf{q}_i$ and $\mathbf{q}_j$ is:

$$\begin{aligned}\frac{\partial^2 \ddot{\mathbf{q}}}{\partial \mathbf{q}_i \partial \mathbf{q}_j} &= \frac{\partial^2 \mathbf{M}^{-1}(\mathbf{q})}{\partial \mathbf{q}_i \partial \mathbf{q}_j}\left(\mathbf{C}(\mathbf{q},\dot{\mathbf{q}}) + \mathbf{G}(\mathbf{q}) + \mathcal{J}^{\mathrm{T}}(\mathbf{q})\mathbf{u}\right) \\ &+ \frac{\partial \mathbf{M}^{-1}(\mathbf{q})}{\partial \mathbf{q}_i}\left(\frac{\partial \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})}{\partial \mathbf{q}_j} + \frac{\partial \mathbf{G}(\mathbf{q})}{\partial \mathbf{q}_j} + \frac{\partial \mathcal{J}^{\mathrm{T}}(\mathbf{q})}{\partial \mathbf{q}_j}\mathbf{u}\right) \\ &+ \frac{\partial \mathbf{M}^{-1}(\mathbf{q})}{\partial \mathbf{q}_j}\left(\frac{\partial \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})}{\partial \mathbf{q}_i} + \frac{\partial \mathbf{G}(\mathbf{q})}{\partial \mathbf{q}_i} + \frac{\partial \mathcal{J}^{\mathrm{T}}(\mathbf{q})}{\partial \mathbf{q}_i}\mathbf{u}\right) \\ &+ \mathbf{M}^{-1}(\mathbf{q})\left(\frac{\partial^2 \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})}{\partial \mathbf{q}_i \partial \mathbf{q}_j} + \frac{\partial^2 \mathbf{G}(\mathbf{q})}{\partial \mathbf{q}_i \partial \mathbf{q}_j} + \frac{\partial^2 \mathcal{J}^{\mathrm{T}}(\mathbf{q})}{\partial \mathbf{q}_i \partial \mathbf{q}_j}\mathbf{u}\right) \end{aligned} \tag{36}$$

We note that

$$\mathbf{E}1 = \mathbf{M}^{-1}(\mathbf{q})\frac{\partial \mathbf{M}(\mathbf{q})}{\partial \mathbf{q}_i}\frac{\partial \mathbf{M}^{-1}(\mathbf{q})}{\partial \mathbf{q}_j} \tag{37}$$

$$\frac{\partial^2 \mathbf{M}^{-1}(\mathbf{q})}{\partial \mathbf{q}_i \partial \mathbf{q}_j} = -\mathbf{E}1^{\mathrm{T}} - \mathbf{M}^{-1}(\mathbf{q})\frac{\partial^2 \mathbf{M}(\mathbf{q})}{\partial \mathbf{q}_i \partial \mathbf{q}_j}\mathbf{M}^{-1}(\mathbf{q}) - \mathbf{E}1 \tag{38}$$

and that $\frac{\partial^2 \mathbf{M}^{-1}(\mathbf{q})}{\partial \mathbf{q}_i \partial \mathbf{q}_j}$ and $\frac{\partial^2 \mathbf{M}(\mathbf{q})}{\partial \mathbf{q}_i \partial \mathbf{q}_j}$ are symmetric matrices.

The second derivative with respect to a configuration variable $\mathbf{q}_i$ and a velocity variable $\dot{\mathbf{q}}_j$ is:

$$\frac{\partial^2 \ddot{\mathbf{q}}}{\partial \mathbf{q}_i \partial \dot{\mathbf{q}}_j} = \frac{\partial \mathbf{M}^{-1}(\mathbf{q})}{\partial \mathbf{q}_i}\left(\frac{\partial \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}_j}\right) + \mathbf{M}^{-1}(\mathbf{q})\left(\frac{\partial^2 \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})}{\partial \mathbf{q}_i \partial \dot{\mathbf{q}}_j}\right) \tag{39}$$

The second derivative with respect to two velocity variables $\dot{\mathbf{q}}_i$ and $\dot{\mathbf{q}}_j$ is:

$$\frac{\partial^2 \ddot{\mathbf{q}}}{\partial \dot{\mathbf{q}}_i \partial \dot{\mathbf{q}}_j} = \mathbf{M}^{-1}(\mathbf{q})\left(\frac{\partial^2 \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}_i \partial \dot{\mathbf{q}}_j}\right) \tag{40}$$

10.2 A Dynamics Formulation That Best Supports Symbolic Dynamics

We are building on a dynamics formulation that makes deriving dynamics equations easier and best supports symbolic dynamics and efficient derivatives, the Natural Orthogonal Complement Approach developed by Angeles and colleagues [1]. This approach handles constrained dynamics, unlike Newton-Euler dynamics. A key aspect of this approach is that only derivatives of a kinematic quantity, a Jacobian, are necessary. Lagrangian dynamics are not as effective for symbolic dynamics, as derivatives of energy (a more complex quantity) are required as well as generation of many terms which are later canceled.

We will describe this dynamics formulation applied to planar systems (since handling 3D rotations complicates the notation). The masses and moments of inertia of each of n bodies can be put

along the diagonal of a matrix $\mathcal{M}$:

$$\mathcal{M} = \begin{pmatrix} m_1 & 0 & \cdots & & & & \\ 0 & m_1 & 0 & \cdots & & & \\ 0 & 0 & I_1 & 0 & \cdots & & \\ \vdots & \vdots & & & & \vdots & \vdots \\ & & \cdots & 0 & m_n & 0 & 0 \\ & & & \cdots & 0 & m_n & 0 \\ & & & & \cdots & 0 & I_n \end{pmatrix} \tag{41}$$

A corresponding vector expresses the motion of each of the n bodies in inertial coordinates:

$$\mathcal{X} = \left(\dot{x}_1, \dot{y}_1, \dot{\theta}_1, \cdots, \dot{x}_n, \dot{y}_n, \dot{\theta}_n\right)^{\mathrm{T}} \tag{42}$$

We can express the velocities in inertial coordinates as a Jacobian matrix times the velocities in generalized coordinates:

$$\mathcal{X} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}} \tag{43}$$

The kinetic energy can be expressed as:

$$\mathrm{K} = \frac{1}{2}\mathcal{X}^{\mathrm{T}}\mathcal{M}\mathcal{X} = \frac{1}{2}\dot{\mathbf{q}}^{\mathrm{T}}\mathbf{J}^{\mathrm{T}}(\mathbf{q})\mathcal{M}\mathbf{J}(\mathbf{q})\dot{\mathbf{q}} \tag{44}$$

The potential energy for a system in which the Y axis is vertical is the sum of the potential energies of the bodies: $\mathrm{P} = \sum_b m_b G y_b$. The y_b can be expressed as a nominal value $y0_b$ plus the effect of the appropriate Jacobian $\mathbf{J}^{y_b}(\mathbf{q})$, so the potential energy is also expressed in terms of the same Jacobians:

$$\mathrm{P} = \sum_b m_b G(\mathbf{J}^{y_b}(\mathbf{q})(\mathbf{q} - \mathbf{q}_0) + y0_b) \tag{45}$$

In Lagrangian dynamics, the contribution of the kinetic energy to the dynamics equation for the ith generalized coordinate is given by

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \mathrm{K}}{\partial \dot{\mathbf{q}}_i}\right) - \frac{\partial \mathrm{K}}{\partial \mathbf{q}_i} \tag{46}$$

The contribution of the potential energy is

$$\left.\frac{\partial \mathrm{P}}{\partial \mathbf{q}_i}\right|_{\mathbf{q}=\mathbf{q}_0} = \sum_b m_b G \frac{\partial \mathbf{J}^{y_b}(\mathbf{q})}{\partial \mathbf{q}_i} \tag{47}$$

Given that:

$$\mathbf{E}2 = \mathbf{J}^{\mathrm{T}}(\mathbf{q})\mathcal{M}\dot{\mathbf{J}}(\mathbf{q}) \tag{48}$$

The first term in the kinetic energy equation above is

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial \mathrm{K}}{\partial \dot{\mathbf{q}}_i}\right) = \mathbf{J}^{\mathrm{T}}(\mathbf{q})\mathcal{M}\mathbf{J}(\mathbf{q})\ddot{\mathbf{q}} + (\mathbf{E}2 + \mathbf{E}2^{\mathrm{T}})\dot{\mathbf{q}} \tag{49}$$

We note that

$$\dot{\mathbf{J}}(\mathbf{q}) = \sum_i \left(\frac{\partial \mathbf{J}(\mathbf{q})}{\mathbf{q}_i} \dot{\mathbf{q}}_i \right) \tag{50}$$

With

$$\mathbf{E}3(i) = \mathbf{J}^{\mathrm{T}}(\mathbf{q}) \mathcal{M} \frac{\partial \mathbf{J}(\mathbf{q})}{\mathbf{q}_i} \tag{51}$$

then

$$\mathbf{E}2 = \sum_l \mathbf{E}3(l) \dot{\mathbf{q}}_l \tag{52}$$

The dynamics equation for the ith generalized coordinate is given by:

$$\mathcal{F}_i = \left[\mathbf{J}^{\mathrm{T}}(\mathbf{q}) \mathcal{M} \mathbf{J}(\mathbf{q}) \ddot{\mathbf{q}} + (\mathbf{E}2 + \mathbf{E}2^{\mathrm{T}}) \dot{\mathbf{q}} \right]_i - \frac{1}{2} \dot{\mathbf{q}}^{\mathrm{T}} \left(\mathbf{E}3(i) + \mathbf{E}3^{\mathrm{T}}(i) \right) \dot{\mathbf{q}} + \frac{\partial \mathrm{P}}{\partial \mathbf{q}_i} \tag{53}$$

where $\mathcal{F}_i$ is the generalized force corresponding to the ith generalized coordinate.

For planar systems, much of the Lagrangian dynamics cancel:

$$0 = \left[\mathbf{E}2^{\mathrm{T}} \dot{\mathbf{q}} \right]_i - \frac{1}{2} \dot{\mathbf{q}}^{\mathrm{T}} \left(\mathbf{E}3(i) + \mathbf{E}3^{\mathrm{T}}(i) \right) \dot{\mathbf{q}} \tag{54}$$

and the dynamics equation for the ith generalized coordinate is given by:

$$\mathcal{F}_i = \left[\mathbf{J}^{\mathrm{T}}(\mathbf{q}) \mathcal{M} \mathbf{J}(\mathbf{q}) \ddot{\mathbf{q}} + \mathbf{J}^{\mathrm{T}}(\mathbf{q}) \mathcal{M} \dot{\mathbf{J}}(\mathbf{q}) \dot{\mathbf{q}} \right]_i + \frac{\partial \mathrm{P}}{\partial \mathbf{q}_i} \tag{55}$$

10.3 Derivatives Of Our Dynamics Approach

We will show how our dynamics formulation supports taking derivatives. For the planar example:

$$\mathbf{M}(\mathbf{q}) = \mathbf{J}^{\mathrm{T}}(\mathbf{q}) \mathcal{M} \mathbf{J}(\mathbf{q}) \tag{56}$$

$$\mathbf{C}_i(\mathbf{q}, \dot{\mathbf{q}}) = \left[\mathbf{J}^{\mathrm{T}}(\mathbf{q}) \mathcal{M} \dot{\mathbf{J}}(\mathbf{q}) \dot{\mathbf{q}} \right]_i = \left[\sum_l \dot{\mathbf{q}}_l \mathbf{J}^{\mathrm{T}}(\mathbf{q}) \mathcal{M} \frac{\partial \mathbf{J}(\mathbf{q})}{\partial \mathbf{q}_l} \dot{\mathbf{q}} \right]_i \tag{57}$$

$$\mathbf{G}_i(\mathbf{q}) = \frac{\partial \mathrm{P}}{\partial \mathbf{q}_i} \tag{58}$$

The following derivatives are now straightforward:

$$\frac{\partial \mathbf{M}(\mathbf{q})}{\partial \mathbf{q}_j} = \mathbf{E}3(j) + \mathbf{E}3^{\mathrm{T}}(j) \tag{59}$$

$$\mathbf{E}4(j,k) = \mathbf{J}^{\mathrm{T}}(\mathbf{q}) \mathcal{M} \frac{\partial^2 \mathbf{J}(\mathbf{q})}{\partial \mathbf{q}_j \partial \mathbf{q}_k} \tag{60}$$

$$\mathbf{E}5(j,k) = \left(\frac{\partial \mathbf{J}(\mathbf{q})}{\partial \mathbf{q}_j} \right)^{\mathrm{T}} \mathcal{M} \frac{\partial \mathbf{J}(\mathbf{q})}{\partial \mathbf{q}_k} \tag{61}$$

$$\frac{\partial^2 \mathbf{M}(\mathbf{q})}{\partial \mathbf{q}_j \partial \mathbf{q}_k} = \mathbf{E}4(j,k) + \mathbf{E}4^{\mathrm{T}}(j,k) + \mathbf{E}5(j,k) + \mathbf{E}5^{\mathrm{T}}(j,k) \tag{62}$$

$$\frac{\partial \mathbf{C}_i(\mathbf{q},\dot{\mathbf{q}})}{\partial \mathbf{q}_j} = \left[\sum_l \dot{\mathbf{q}}_l \left(\mathbf{E}4(j,l) + \mathbf{E}5(j,l) \right) \dot{\mathbf{q}} \right]_i \tag{63}$$

$$\frac{\partial \mathbf{C}_i(\mathbf{q},\dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}_j} = \left[\mathbf{E}3(j)\dot{\mathbf{q}} + \sum_l \dot{\mathbf{q}}_l \mathbf{E}3(l)\mathbf{i}(j) \right]_i \tag{64}$$

$\mathbf{i}(j)$ is a vector that is all zeros except for a 1 in the jth position.

$$\mathbf{E}6(j,k,l) = \left(\frac{\partial \mathbf{J}(\mathbf{q})}{\partial \mathbf{q}_j} \right)^{\mathrm{T}} \mathcal{M} \frac{\partial^2 \mathbf{J}(\mathbf{q})}{\partial \mathbf{q}_k \partial \mathbf{q}_l} \tag{65}$$

$$\mathbf{E}7(j,k,l) = \mathbf{J}^{\mathrm{T}}(\mathbf{q}) \mathcal{M} \frac{\partial^3 \mathbf{J}(\mathbf{q})}{\partial \mathbf{q}_j \partial \mathbf{q}_k \partial \mathbf{q}_l} \tag{66}$$

$$\frac{\partial^2 \mathbf{C}_i(\mathbf{q},\dot{\mathbf{q}})}{\partial \mathbf{q}_j \partial \mathbf{q}_k} = \left[\sum_l \dot{\mathbf{q}}_l \left(\mathbf{E}6(j,l,k) + \mathbf{E}6^{\mathrm{T}}(l,j,k) + \mathbf{E}6(k,j,l) + \mathbf{E}7(j,k,l) \right) \dot{\mathbf{q}} \right]_i \tag{67}$$

$$\frac{\partial^2 \mathbf{C}_i(\mathbf{q},\dot{\mathbf{q}})}{\partial \mathbf{q}_j \partial \dot{\mathbf{q}}_k} = \left[\left(\mathbf{E}4(j,k) + \mathbf{E}5(j,k) \right) \dot{\mathbf{q}} + \sum_l \dot{\mathbf{q}}_l \left(\mathbf{E}4(j,l) + \mathbf{E}5(j,l) \right) \mathbf{i}(k) \right]_i \tag{68}$$

$$\frac{\partial^2 \mathbf{C}_i(\mathbf{q},\dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}_j \partial \dot{\mathbf{q}}_k} = \left[\mathbf{E}3(j)\mathbf{i}(k) + \mathbf{E}3(k)\mathbf{i}(j) \right]_i \tag{69}$$

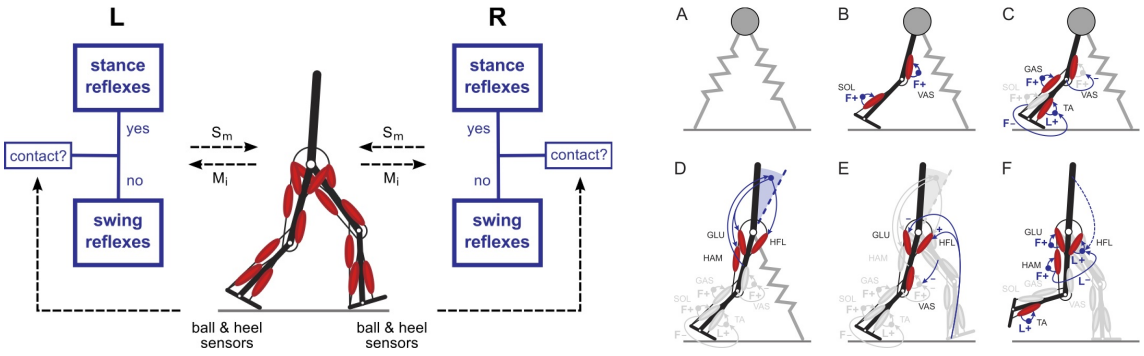
$$\frac{\partial \mathbf{G}_i(\mathbf{q})}{\partial \mathbf{q}_j} = \frac{\partial^2 \mathrm{P}}{\partial \mathbf{q}_i \partial \mathbf{q}_j} \tag{70}$$

$$\frac{\partial^2 \mathbf{G}_i(\mathbf{q})}{\partial \mathbf{q}_j \partial \mathbf{q}_k} = \frac{\partial^3 \mathrm{P}}{\partial \mathbf{q}_i \partial \mathbf{q}_j \partial \mathbf{q}_k} \tag{71}$$

In general, the only derivatives that actually need to be computed are the first, second, and third derivatives of the Jacobian with respect to configuration generalized coordinates. All other derivatives are computed using the chain rule (matrix multiplication). Forming and taking derivatives of dynamics equations now boils down to manipulating the Jacobian matrix (a purely kinematic construct).

11 Neuromuscular Model

We extended a previous neuromuscular model of human locomotion. This forward-dynamic model represents the human musculoskeletal system as a planar, seven segment system modeling the trunk as well as the thighs, shanks and feet (Fig. 4a). The segments are connected by revolute joints, which model hip, knee and ankle. The joints are actuated by seven Hill-type muscle models



(a) foot extension (b) neuromuscular model implementation

Figure 4: Details of existing neuro-musculo-skeletal model of human locomotion. (Left) Each leg is actuated by seven Hill-type muscle models that are controlled by muscle reflexes. The model does not use desired joint paths; it generates its motor control signals S_m entirely from sensory feedback. (Right) The model's sensorimotor control is synthesized element by element by translating a bipedal spring-mass model into an articulated one.

per leg, representing major leg muscles or muscle groups active in human walking (soleus, SOL; gastrocnemius, GAS; tibialis anterior, TA; vastii group, VAS; hamstring group, HAM; gluteus maximus, GLU; and grouped hip flexors, HFL; Fig. 4a). The muscle models consist of contractile elements, who take motor control signals S_m from 0% to 100% as input, combined with series and parallel elasticities. Each muscle's force translates into a joint torque contribution $\tau_{m,j} = F_m r_m(\varphi_j)$ at the joints j it spans, using variable moment arms $r_m(\varphi_j)$ that mimic the physiological moment arms observed for these muscles and joints.

The muscle stimulations are generated by the model's sensorimotor control. The control consists of separate stance and swing phase reflexes which are based on sensory signals measuring the muscle state (mostly homonymous, positive force or length feedbacks marked by F+ and L+ in Fig. 4b). To reflect neural transport delays, these signals are time-delayed, as well as multiplied by a gain, and fed back into sum blocks that represent alpha motor neurons and produce muscle stimulations. The sensory feedback pathways that are used in the model have been synthesized element by element by translating a bipedal spring-mass model into an articulated one, and encoding compliant leg behavior and other principles of legged dynamics and control into muscle reflexes control (A-F in Fig. 4b).

We extended this model by adding a detailed foot to increase the model's potential for predicting human control strategies for balance, tripping, slipping and rough ground adaptations (Fig. 6). The foot represents the main dynamical components of the human foot, including the longitudinal foot arch with metatarsal and metatarsal-phalangeal joints, as well as the plantar fasciae and biarticular muscle actuators flexing and extending the toe. With F+ control of the toe flexor muscle and L+ control of the toe extensor, the foot model seamlessly integrates into the existing walking model, and generates steady-state kinematics, kinetics and muscle activities that compare to human patterns (Fig. 7).

We implemented an optimization algorithm for key control parameters of the neuromuscular

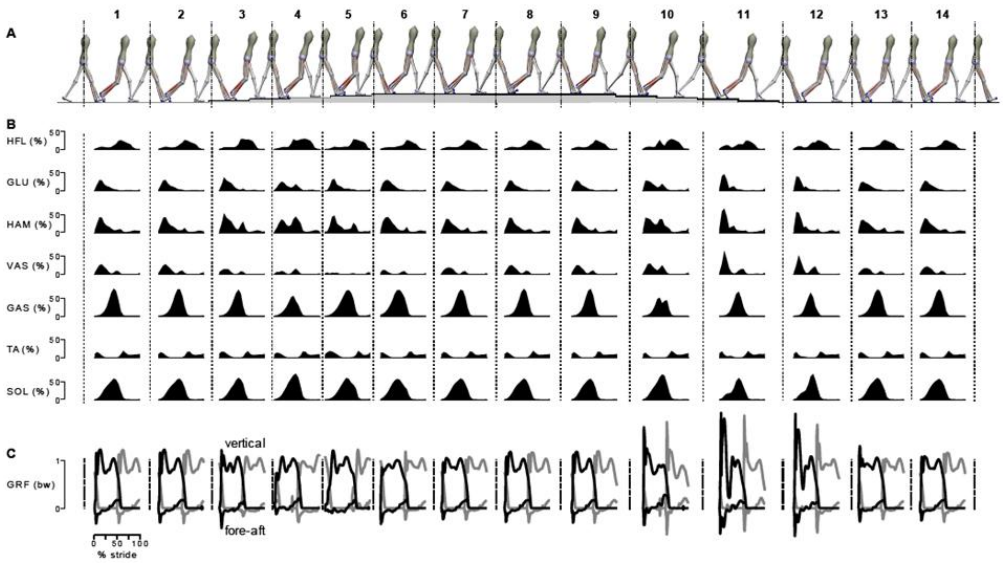
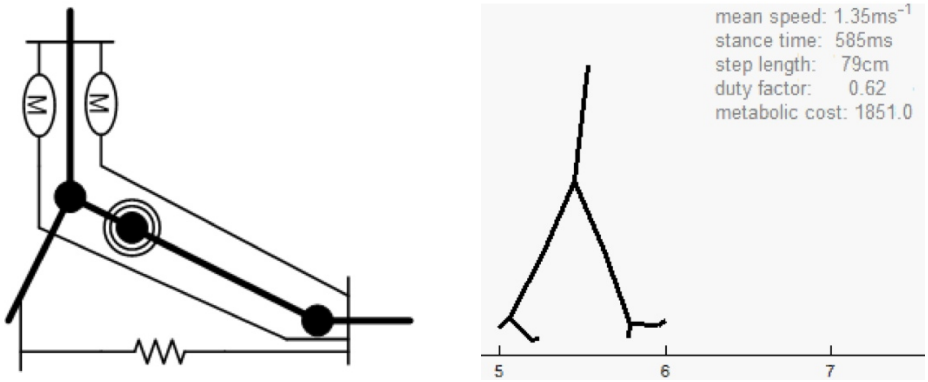


Figure 5: Terrain adaptation of neuromuscular model with predicted muscle activation outputs.



(a) foot extension (b) neuromuscular model implementation

Figure 6: Foot extension of neuromuscular model.

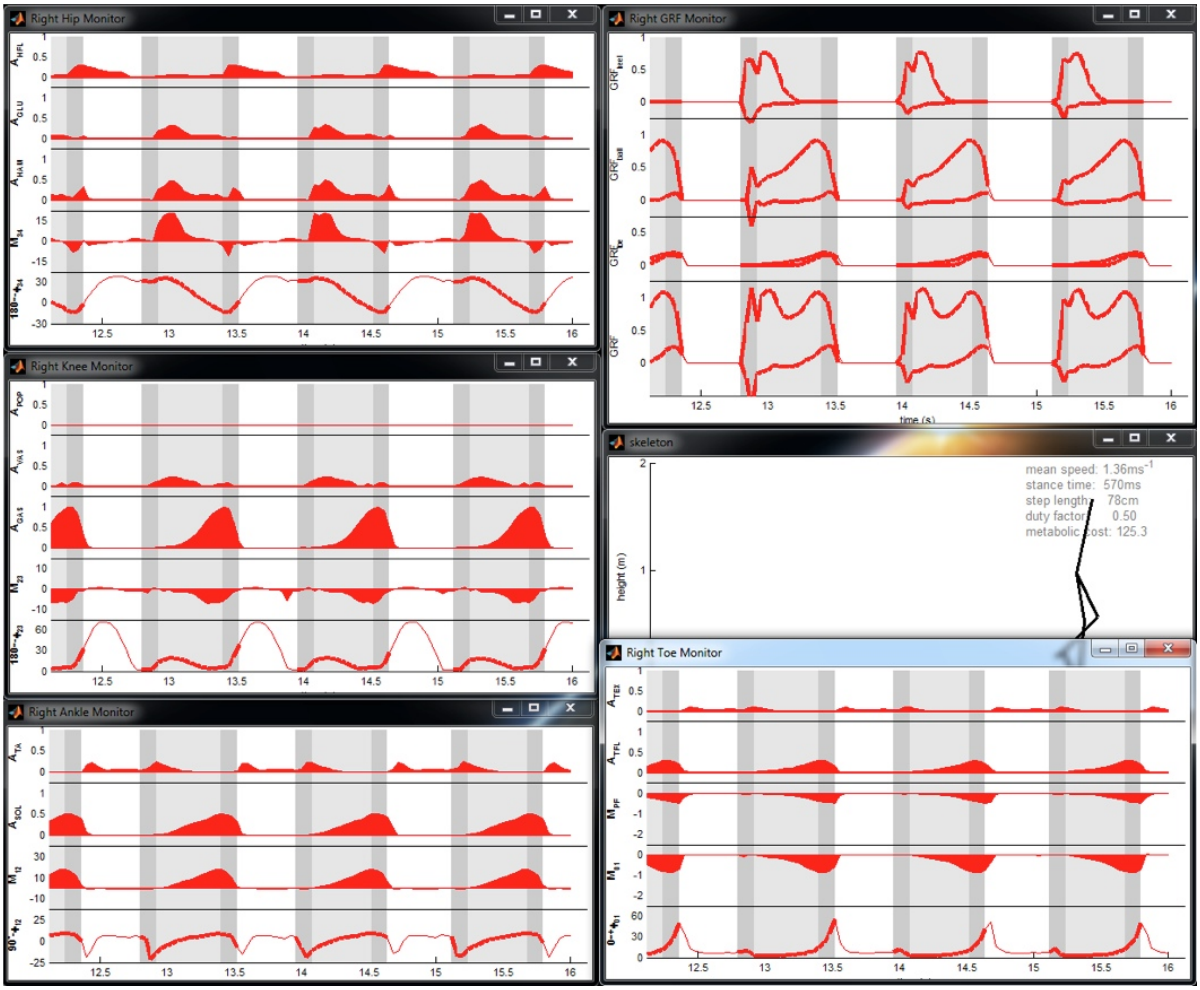


Figure 7: Model monitors and human model walking. The model behavior is shown in five panels detailing the joint kinematics (φ) and net torques (M) as well as the leg muscle activations (A) for hip, knee, ankle and toe, allowing for detailed experimental comparison with human subject walking.

reflexes that uses the covariance matrix adaptation evolutionary strategy (CMA-ES). Our implementation of the CMA-ES algorithm samples a population of N solutions from the space of control parameters and initial conditions according to a multivariate normal distribution, evaluates these solutions based on a cost function, and uses the M best solutions to reshape the covariance matrix of the normal distribution. This procedure repeats until the overall best solution does not change for G generations. Typical values are N=15, M=13, and G=50.

To identify the control parameters for human-like walking solutions at different speeds, we used

$$J = 10(\dot{x}_{avg} - \dot{x}_{tgt}) + COT \tag{72}$$

as the cost function in the optimization where $\dot{x}_{tgt}$ is a target walking speed, and $\dot{x}_{avg}$ and COT are the average walking speed and the cost of transport in steady walking. For humans, reported values of the COT are 3.3–3.6 $\mathrm{Jkg^{-1}m^{-1}}$ for walking at the preferred speed of $1.2\mathrm{ms^{-1}}$. We assumed that an individual simulation had achieved steady walking as soon as the leg joint positions do not change significantly relative to the touch-down position between subsequent heel-strikes.

For the three subsequent steady-state gait cycles, we computed the average cost of transport, $COT = E_M/(m\,x_d)$, where m is the body mass, x_d is the distance travelled, and E_M is the total metabolic energy. We estimated E_M as the sum of metabolic energies expended by all muscles, using the muscle energy model by Umberger et al. (2003). To test the sensitivity of the optimization results, we compared three different model configurations that differ in the actual foot model that is implemented.

For the three neuromuscular models with the baseline, human, and passive ankle-foot configurations, the optimization minimizes the COT while trying to maintain the target speed x_{tgt}. At a typical human walking speed of $1.2\mathrm{ms^{-1}}$, all three configurations show similar COT of about 3.6 $\mathrm{Jkg^{-1}m^{-1}}$, which matches human data.

For slower speeds, the cost substantially rise again in accordance with human data. For faster speeds, humans show increasing COT as well. We find this increase only for the model with the human-like foot. The difference could be related either to an opportunity to construct legged systems more efficient than humans, or to the optimization getting stuck in a local minimum in this preliminary study.

The control parameters that had the most consistent changes with speed in the three model configurations are as follows: The reference lean Θ_{ref} for trunk balance control shows a clear increase with speed for all models (a). Note that for the human model, the value Θ_{ref} largely diverges from the other two models at slow speeds, because the optimization found a marching gait that is unlike 'normal' walking. For the swing leg motion, the F+ gain of the GLU and the length offset $l_{off,HAM}$ of the HAM (for the L- suppression of the HFL) show a clear trend consistent among all three models (b). The F+ gain prevents excessive forward swings while $l_{off,HAM}$ defines the onset of the shank rotation passively overtaking the thigh to straighten the swing leg. A larger value delays that onset indicating larger strides, which are contained by a more aggressive GLU. Finally, for the stance control, the F+ gains of SOL and GAS show consistent increase with speed for the baseline and passive configurations, suggesting a clear contribution to forward propulsion (c). This trend does not show for the model with the human foot, which may be related to the marching gait identified at very slow speeds and the added propulsion that TFL provides.

Our initial results indicate that we have a sufficiently fast algorithm implementation that delivers model optimizations in realistic time frames. This optimization has indicated several control parameters that show consistent changes with speed.

To obtain more human-like walking solutions, we have added a "pain" term in the cost function of our optimization that penalizes unphysiological joint angles. This has resulted in human-like walking solutions from very slow to fast walking (0.8ms^{-1} to 1.8ms^{-1}). In addition, we have investigated speed increments of 0.2ms^{-1}. This increased resolution led to nine key control parameters of walking at different speeds, which cluster around three dynamic categories. The first is trunk forward lean. It correlates nearly linearly with speed as larger trunk lean balances the effects of gravitational forces, impact forces, and the larger forces generated by hip extensors that provide forward propulsion. The second category is the prevention of knee overextension which is induced by inertial coupling from the increased ankle push-off and from the increased hip extensor torques. The third category is the accelerated swing motion at higher speeds, which is generated by increased hip flexion torques in double support, automatically entraining faster knee flexion again due to inertial coupling. With a second optimization step, we have identified speed transition controllers that attract the model into the previously identified steady walking solutions at different speeds.

We seek to achieve these large speed transitions robustly in uneven terrain. Toward this goal, we have recently included rough terrain in our speed optimization. For walking speeds from 0.8ms^{-1} to 1.8ms^{-1}, we have identified controls that tolerate shallow random ground ($\Delta y = 0.7cm \pm 0.4cm$ mean$\pm$s.d. with max step of +1.2cm and min step of -1.8cm). With the same method, we currently obtain local feedback controls for walking at 1.3ms^{-1} in more challenging terrain ($\Delta y = 2.6cm \pm 2cm$ mean$\pm$s.d. with max step of +7.7cm and min step of -6.9cm).

We have integrated into the neuromuscular model a hip controller that seeks to place the leg into explicit targets. The targets are derived from the bipedal linear inverted pendulum model, an extension of the linear inverted pendulum model which does not require predefined swing leg times. The explicit placement control of the hip flexor resulted in auto-adaptation from slow to normal speed walking (0.8ms^{-1} to 1.4ms^{-1}) without changes in other control parameters (Fig. 1). Larger speed changes still require changes of the nine key control parameters identified in the previous quarter, mainly due to the undesired shank dynamics.

We developed a 3D dynamic contact model and integrated it into our simulation environment. We introduced the pelvis width but locked the roll DOF at the hips. Applying the previous planar reflex control to the 3D model, we identified the roll stability about the ankle and the yaw moment induced by the swing-leg as main issues; without additional biomechanical design or control the model fell after few steps. To resolve these issues we studied the influence of a three-contact foot on yaw stability and of adding passive roll motion at the ankle on roll stability. Using these two extensions, our model achieves weakly stable walking at 1.3ms^{-1}.

12 Summary Of The Most Important Results

- We implemented balance, walking, and push recovery on our humanoid robot.

- We developed a two level optimization-based controller for walking: optimization of the COM trajectory followed by inverse dynamics using quadratic programming.
- We developed a globally optimal approach to high level control of bipedal walking based on dynamic programming applied to weakly coupled subsystems.
- We developed a prototype behavior library, and a method to edit the library entries.
- We discovered that it was possible to approximate the library using simple functions.
- We developed a way to improve behaviors using policy optimization.
- We developed an approach to footstep planning that takes into account robot dynamics.
- We improved our neuromuscular model of human walking.

References

[1] J. Angeles. *Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms*. Springer, New York, NY, 1997.

[2] C. G. Atkeson. Using local trajectory optimizers to speed up global optimization in dynamic programming. In Jack D. Cowan, Gerald Tesauro, and Joshua Alspector, editors, *Advances in Neural Information Processing Systems*, volume 6, pages 663–670. Morgan Kaufmann Publishers, Inc., 1994.

[3] C. G. Atkeson. Randomly sampling actions in dynamic programming. In *IEEE International Symposium on Approximate Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 185–192, 2007.

[4] C. G. Atkeson and J. Morimoto. Non-parametric representation of a policies and value functions: A trajectory based approach. In *Advances in Neural Information Processing Systems 15*. MIT Press, 2003.

[5] C. G. Atkeson and B. Stephens. Multiple balance strategies from one optimization criterion. In *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2007.

[6] C. G. Atkeson and B. Stephens. Random sampling of states in dynamic programming. In *Neural Information Processing Systems (NIPS) Conference*, 2007.

[7] C. G. Atkeson and B. Stephens. Random sampling of states in dynamic programming. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 38(4):924–929, 2008.

[8] R. Bellman. *Dynamic Programming*. reprinted by Dover (2003), 1957.

[9] D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 1995.

[10] J. Chestnutt, K. Nishiwaki, J.J. Kuffner, and S. Kagami. An adaptive action model for legged navigation planning. IEEE/RAS Int. Conf. on Humanoid Robotics (Humanoids'07), 2007.

[11] H. Diedam, D. Dimitrov, P. B. Wieber, K. Mombaur, and M. Diehl. Online walking gait generation with adaptive foot positioning through linear model predictive control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1121–1126, 2008.

[12] P. Dyer and S. R. McReynolds. *The Computation and Theory of Optimal Control*. Academic Press, New York, NY, 1970.

[13] Special Issue. International Journal of Robotics Research, 30(2), 2011.

[14] D. H. Jacobson and D. Q. Mayne. *Differential Dynamic Programming*. Elsevier, New York, NY, 1970.

[15] S Kajita, F Kanehiro, K Kaneko, K Fujiwara, K Harada, K Yokoi, and H Hirukawa. Biped walking pattern generation by using preview control of zero-moment point. In *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, pages 1620–1626, Taipei, Taiwan, September 2003.

[16] S. Kanzaki, K. Okada, and M. Inaba. Bracing behavior in humanoid through preview control of impact disturbance. In *Proceedings of the IEEE-RAS International Conference on Humanoid Robots*, 2005.

[17] T. Mandersloot, M. Wisse, and C.G. Atkeson. Controlling velocity in bipedal walking: A dynamic programming approach. In *Proceedings of the 6th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2006.

[18] J. Morimoto, G. Zeglin, and C. G. Atkeson. Minmax differential dynamic programming: Application to a biped walking robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2003.

[19] K. Nishiwaki and S. Kagami. Online walking control system for humanoids with short cycle pattern generation. *IJRR*, 28(6):729–742, 2009.

[20] J. Park and Y. Youm. General ZMP preview control for bipedal walking. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2007.

[21] J. Pratt, J. Carff, and S. Drakunov. Capture point: A step toward humanoid push recovery. In *in 6th IEEE-RAS International Conference on Humanoid Robots*, pages 200–207, 2006.

[22] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa. Chomp: Gradient optimization techniques for efficient motion planning. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, 2009.

[23] S. Schaal and C. G. Atkeson. Learning control for robotics. *IEEE Robotics & Automation Magazine*, 17(2):20–29, 2010.

[24] B. Stephens. *Push Recovery Control for Force-Controlled Humanoid Robots*. PhD thesis, Carnegie Mellon University, 2011.

[25] M. Stilman, C. G. Atkeson, J. J. Kuffner, and G. Zeglin. Dynamic programming in reduced dimensional spaces: Dynamic planning for robust biped locomotion. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2399–2404, 2005.

[26] E. Whitman and C. G. Atkeson. Control of a walking biped using a combination of simple policies. In *International Conference on Humanoid Robots*, pages 520–527, 2009.

[27] E. Whitman and C. G. Atkeson. Control of instantaneously coupled systems applied to humanoid walking. In *International Conference on Humanoid Robots*, pages 210–217, 2010.

[28] M. Zucker, J. A. Bagnell, C. G. Atkeson, and J. Kuffner. An optimization approach to rough terrain locomotion. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3589–3595, 2010.

[29] M. Zucker, N. D. Ratliff, M. Stolle, J. E. Chestnutt, J. A. Bagnell, C. G. Atkeson, and J. Kuffner. Optimization and learning for rough terrain legged locomotion. *International Journal of Robotic Research*, 30(2):175–191, 2011.