Robust Low Torque Biped Walking Using Differential Dynamic Programming With a Minimax Criterion

J. Morimoto†⋆ and C. Atkeson⋆
†Human Information Science Laboratories, ATR International, Department 3
2-2-2 Hikaridai Seika-cho Soraku-gun, Kyoto, JAPAN 619-0288
xmorimo@atr.co.jp
⋆The Robotics Institute, Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA, USA 15213
xmorimo@cs.cmu.edu, cga@cmu.edu

ABSTRACT

We developed a control policy design method for robust low torque biped walking by using differential dynamic programming with a minimax criterion. As an example, we applied our method to a simulated five link biped robot. The results show lower joint torques from the optimal control policy compared to a hand-tuned PD servo controller. Results also show that the simulated biped robot can successfully walk with unknown disturbances that cause controllers generated by standard differential dynamic programming and the hand-tuned PD servo to fail. Future work will implement these controllers on a robot we are currently developing.

Recent humanoid robots using Zero Moment Point (ZMP) control strategies have demonstrated impressive biped walking [7, 11, 8]. However, robots using ZMP control are often neither robust nor energy efficient and can generate large joint torques. McGeer [9] demonstrated that passive dynamic walking was possible. His robots walked down a slight incline without applying any torques at the joints. Recently, several studies have explored how to generate energy efficient biped walking [1, 10]. Many studies using optimization methods [4, 2] focus on finding optimal biped walk trajectories, but do not provide control laws to cope with disturbances. Our strategy is to use differential dynamic programming [3, 6], an optimization method, to find both a low torque biped walk and a policy or control law to handle deviations from the nominal trajectory. We use a minimax reward to insure the policy is robust.

1 BIPED ROBOT MODEL

In this paper, we use a simulated five link biped robot (Fig. 1) to explore our approach. Kinematic and dynamic parameters of the simulated robot are chosen to match those of a biped robot we are currently developing (Fig. 2) and which we will use to further explore our approach. Height and total weight of the robot are about 0.4 [m] and 2.0 [kg] respectively. Table 1 shows the parameters of the robot model.

Table 1: Physical parameters of the robot model

	link1	link2	link3	link4	link5
mass [kg]	0.05	0.43	1.0	0.43	0.05
length [m]	0.2	0.2	0.01	0.2	0.2
inertia [kg·m $\times 10^{-4}$]	1.75	4.29	4.33	4.29	1.75

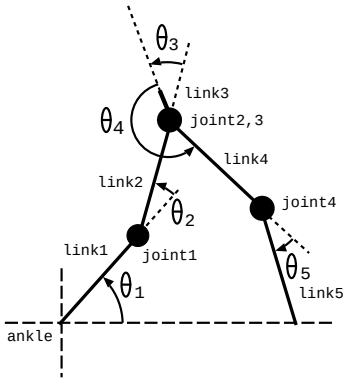

Figure 1: Five link robot model

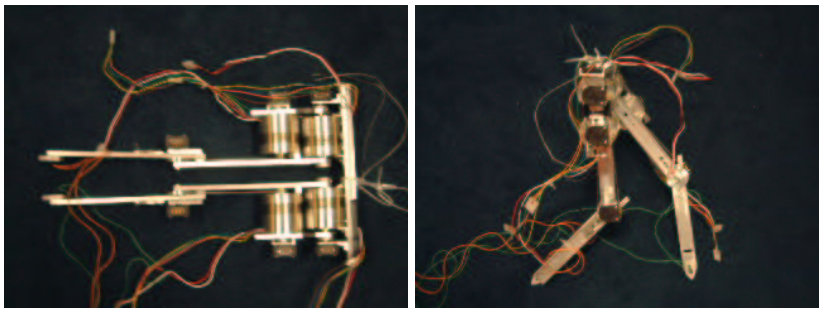

Figure 2: Real robot

We can represent the forward dynamics of the biped robot as

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{b}(\mathbf{x})\mathbf{u}, \tag{1}$$

where $\mathbf{x} = \{\theta_1, \ldots, \theta_5, \dot{\theta}_1, \ldots, \dot{\theta}_5\}$ denotes the input state vector, $\mathbf{u} = \{\tau_1, \ldots, \tau_4\}$ denotes the control command (each torque τ_i is applied to joint i (Fig. 1)). In the minimax optimization case, we explicitly represent the existence of the disturbance as

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{b}(\mathbf{x})\mathbf{u} + \mathbf{b}_w(\mathbf{x})\mathbf{w}, \tag{2}$$

where $\mathbf{w} = \{w_0, w_1, w_2, w_3, w_4\}$ denotes the disturbance (w_0 is applied to ankle, and w_i ($i = 1 \ldots 4$) is applied to joint i (Fig. 1)).

2 OPTIMIZATION CRITERION AND METHOD

We use the following objective function, which is designed to reward energy efficiency and enforce periodicity of the trajectory:

$$J_{opt} = \sum_{i=0}^{N-1} L(\mathbf{x}_i, \mathbf{u}_i) + \Phi(\mathbf{x}_0, \mathbf{x}_N) \tag{3}$$

which is applied for half the walking cycle, from one heelstrike to the next heelstrike. This criterion sums the squared deviations from a nominal trajectory, the squared control magnitudes, and the squared deviations from a desired velocity of the center of mass:

$$L(\mathbf{x}_i, \mathbf{u}_i) = (\mathbf{x}_i - \mathbf{x}_i^d)^T Q (\mathbf{x}_i - \mathbf{x}_i^d) + \mathbf{u}_i{}^T R \mathbf{u}_i + (v_i(\mathbf{x}_i) - v_i^d)^T S (v_i(\mathbf{x}_i) - v_i^d), \tag{4}$$

where $\mathbf{x}_i$ is a state vector at the i-th time step, $\mathbf{x}_i^d$ is the nominal state vector at the i-th time step (taken from a trajectory generated by a hand-designed walking controller), v_i denotes the velocity of the center of mass at the i-th time step, v_i^d denotes the desired velocity of the center of mass at the i-th time step, the term $(\mathbf{x}_i - \mathbf{x}_i^d)^T Q(\mathbf{x}_i - \mathbf{x}_i^d)$ encourages the robot to follow the nominal trajectory, the term ${\mathbf{u}_i}^T R\mathbf{u}_i$ discourages using large control outputs, and the term $(v_i(\mathbf{x}_i) - v_i^d)^T S(v_i(\mathbf{x}_i) - v_i^d)$ encourages the robot to achieve the desired velocity.

In addition, penalties on the initial $(\mathbf{x}_0)$ and final $(\mathbf{x}_N)$ states are applied:

$$\Phi(\mathbf{x}_0, \mathbf{x}_N) = F(\mathbf{x}_0) + \Phi_N(\mathbf{x}_0, \mathbf{x}_N). \tag{5}$$

The term $F(\mathbf{x}_0)$ penalizes an initial state where the foot is not on the ground:

$$F(\mathbf{x}_0) = {F_h}^T(\mathbf{x}_0) P_0 F_h(\mathbf{x}_0), \tag{6}$$

where $F_h(\mathbf{x}_0)$ denotes height of the swing foot at the initial state $\mathbf{x}_0$. The term $\Phi_N(\mathbf{x}_0, \mathbf{x}_N)$ is used to help generate periodic trajectories:

$$\Phi_N(\mathbf{x}_0, \mathbf{x}_N) = (\mathbf{x}_N - H(\mathbf{x}_0))^T P_N (\mathbf{x}_N - H(\mathbf{x}_0)), \tag{7}$$

where $\mathbf{x}_N$ denotes the terminal state, $\mathbf{x}_0$ denotes the initial state, and the term $(\mathbf{x}_N - H(\mathbf{x}_0))^T P_N$ $(\mathbf{x}_N - H(\mathbf{x}_0))$ is a measure of terminal control accuracy. A function $H()$ represents the coordinate change caused by the exchange of a support leg and a swing leg, and the velocity change caused by a swing foot touching the ground (Appendix B).

Dynamic programming provides a methodology to develop planners and controllers for nonlinear systems. However, general dynamic programming is computationally intractable. We use differential dynamic programming (DDP) which is a second order local trajectory optimization method to generate locally optimal plans and local models of the value function [3, 6]. This method also gives us a local policy or feedback controller to correct errors from the planned trajectory.

We introduce a robust DDP method realized by adding a minimax term to the criterion (Appendix A). We use a modified objective function:

$$J_{minmax} = J_{opt} - \sum_{i=0}^{N-1} {\mathbf{w}_i}^T G \mathbf{w}_i, \tag{8}$$

where $\mathbf{w}_i$ denotes a disturbance vector at the i-th time step, and the term ${\mathbf{w}_i}^T G\mathbf{w}_i$ rewards coping with large disturbances. This explicit representation of the disturbance $\mathbf{w}$ provides the robustness for the controller.

2.1 Neighboring Extremal Method

Differential dynamic programming finds a locally optimal trajectory $\mathbf{x}_i^{opt}$ and the corresponding control trajectory $\mathbf{u}_i^{opt}$. When we apply our control algorithm to a real robot, we usually need a feedback controller to cope with unknown disturbances or modeling errors. Fortunately, DDP provides us a local policy along the optimized trajectory:

$$\mathbf{u}^{opt}(\mathbf{x}_i, i) = \mathbf{u}_i^{opt} + \mathbf{K}_i(\mathbf{x}_i - \mathbf{x}_i^{opt}), \tag{9}$$

where $\mathbf{K}_i$ is a time dependent gain matrix given by taking the derivative of the optimal policy with respect to the state [3, 6].

3 RESULTS

We compare the optimized controller with a hand-tuned PD servo controller, which also is the source of the initial and nominal trajectories in the optimization process. We set the parameters for the optimization process as $Q = 0.25\mathbf{I}_{10}$, $R = 3.0\mathbf{I}_4$, $S = 0.3\mathbf{I}_1$, desired velocity $v^d = 0.4$[m/s] in equation (4), $P_0 = 1000000.0\mathbf{I}_1$ in equation (6), and $P_N = diag\{$10000.0, 10000.0, 10000.0, 10000.0, 10000.0, 10.0, 10.0, 10.0, 5.0, 2.5$\}$ in equation (7), where $\mathbf{I}_N$ denotes N dimensional identity matrix. Each parameter is set to acquire the best results in terms of both the robustness and the energy efficiency.

Results in table 2 show that the controller generated by the optimization process did halve the cost of the trajectory, as compared to that of the original hand-tuned PD servo controller. Note that we defined the control cost as $\sum_{i=0}^{N-1} ||\mathbf{u}_i||^2$, where $\mathbf{u}_i$ is the control output (torque) vector at i-th time step.

Table 2: One step control cost (average over 100 steps)

	DDP	PD servo
control cost $[(N \cdot m)^2]$	11.7	24.8

To test robustness, we assume that there is unknown viscous friction at each joint:

$$\tau_j^d = -\mu_j \dot{\theta}_j \quad (j = 1, \ldots, 4), \tag{10}$$

where μ_j denotes the viscous friction coefficient at joint j, and an unknown disturbing torque at the ankle:

$$\tau_{ankle}^d = \tau_c, \tag{11}$$

where τ_c denotes a constant disturbing torque. We considered three disturbance conditions as 1) viscous friction 2) constant ankle torque 3) friction and ankle torque.

We used two levels of disturbances in the simulation, with the higher level being about 3 times larger than the base level (Table 3).

Table 3: Parameters of the disturbance

	μ_2,μ_3 (hip joints)	μ_1,μ_4 (knee joints)	τ_c (ankle)
base	0.01	0.04	-0.003
large	0.03	0.15	-0.01

All methods could handle the base level disturbances. Both the standard and the minimax DDP generated much less control cost than the hand-tuned PD servo controller (Table 4). However, because the minimax DDP is more conservative in taking advantage of the plant dynamics it has a slightly higher control cost than the standard DDP. We also found that the friction disturbance increased the control cost, as would be expected.

Note that we used same parameters as we used in previous experiment for both the standard DDP and the minimax DDP (i.e. Q,R,S,v^d,P_0,P_N). For the minimax DDP, we set the parameter for the disturbance reward in equation (8) as $G = diag\{$5.0, 20.0, 20.0, 20.0, 20.0$\}$ (G with smaller elements generates more conservative but robust trajectories).

Only the minimax DDP control design could cope with the higher level of disturbances with the friction disturbance. Figure 3 shows trajectories for the three different methods with the friction and the ankle torque disturbances. Both the robot with the standard DDP and the hand-tuned PD servo controller fell down before achieving 50 steps. The bottom of figure 3 shows successful 50 steps of the robot with the minimax DDP. Table 5 shows the number of steps before the robot fell down. We terminated a trial when the robot achieved 100 steps. We found that the failed trials were mainly caused by the friction disturbance.

Table 4: One step control cost [$(N \cdot m)^2$] with the base setting (averaged over 100 steps)

	standard DDP	minimax DDP	PD servo
1)friction	14.3	15.8	26.3
2)ankle torque	11.7	12.7	25.0
3)friction & ankle torque	14.4	16.5	26.5

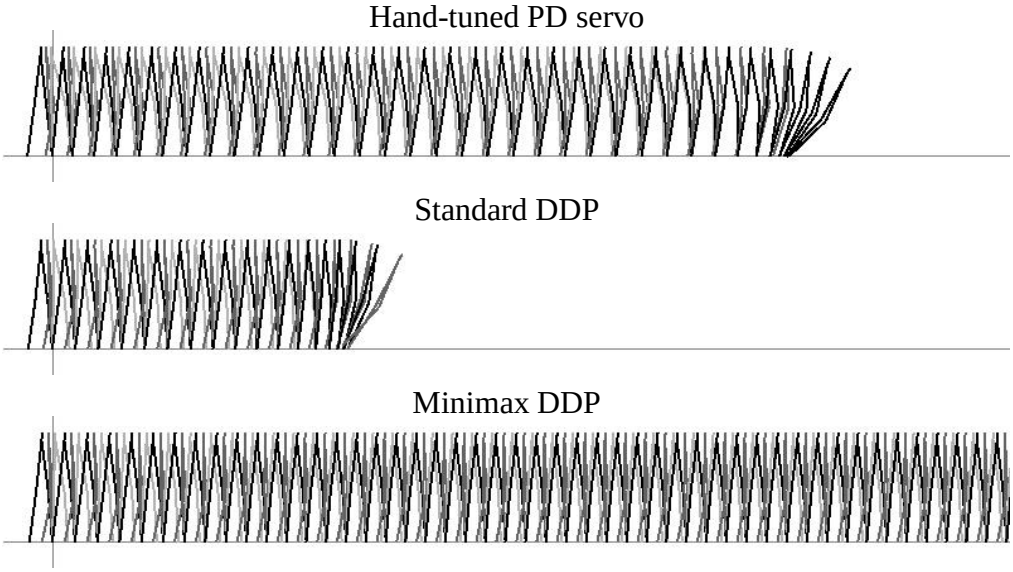

Figure 3: Biped walk trajectories with the three different methods

Table 5: Number of steps with large disturbances

	standard DDP	minimax DDP	PD servo
1)friction	25	100	47
2)ankle torque	100	100	100
3)friction & ankle torque	16	100	33

4 DISCUSSION

In this study, we developed an optimization method to generate biped walking trajectories by using Differential Dynamic Programming (DDP). Both standard DDP and minimax DDP gen-

erated low torque biped trajectories. We showed that minimax DDP had more robustness than the controller provided by standard DDP and the hand-tuned PD servo. DDP provides a feedback controller which is important in coping with unknown disturbances and modeling errors. However, as shown in equation (9), the feedback controller depended on time, and development of a time independent feedback controller is a future goal.

APPENDIX

A Calculation of The Minimax DDP

Here, we show the update rule of minimax DDP. Minimax DDP can be derived as an extension of standard DDP [3, 6]. The difference is that the proposed method has an additional disturbance variable $\mathbf{w}$ to explicitly represent the existence of disturbances. This representation of the disturbance provides the robustness for optimized trajectories and policies.

The total return R by using control output $\mathbf{u}_i$ and disturbance $\mathbf{w}_i$ at the state $\mathbf{x}_i$ is given by

$$\begin{aligned} R(\mathbf{x}_i, \mathbf{u}_i, \mathbf{w}_i) &= L(\mathbf{x}_i, \mathbf{u}_i, \mathbf{w}_i) + \sum_{j=i+1}^{N-1} L(\mathbf{x}_j, \mathbf{u}_j, \mathbf{w}_j) + \Phi(\mathbf{x}_0, \mathbf{x}_N) \\ &= L(\mathbf{x}_i, \mathbf{u}_i, \mathbf{w}_i) + V(\mathbf{x}_{i+1}), \end{aligned} \tag{12}$$

where the value function V is defined as

$$V(\mathbf{x}_i) = \min_{\mathbf{u}_i} \max_{\mathbf{w}_i} \sum_{j=i}^{N-1} L(\mathbf{x}_j, \mathbf{u}_j, \mathbf{w}_j) + \Phi(\mathbf{x}_0, \mathbf{x}_N), \tag{13}$$

and the reward L is defined as

$$L(\mathbf{x}_i, \mathbf{u}_i, \mathbf{w}_i) = (\mathbf{x}_i - \mathbf{x}_i^d)^T Q (\mathbf{x}_i - \mathbf{x}_i^d) + \mathbf{u}_i{}^T R \mathbf{u}_i - \mathbf{w}_i{}^T G \mathbf{w}_i + (v_i(\mathbf{x}_i) - v_i^d)^T S (v_i(\mathbf{x}_i) - v_i^d). \tag{14}$$

Meaning of each parameter in equation (14) is described in section 2.

Then, we expand the return R to second order in terms of $\delta\mathbf{u}$, $\delta\mathbf{w}$ and $\delta\mathbf{x}$ about the nominal solution:

$$\begin{aligned} R(\mathbf{x}_i, \mathbf{u}_i, \mathbf{w}_i) &= Z(i) + Z_x(i)\delta\mathbf{x}_i + Z_u(i)\delta\mathbf{u}_i + Z_w(i)\delta\mathbf{w}_i \\ &+ [\delta\mathbf{x}_i^T \delta\mathbf{u}_i^T \delta\mathbf{w}_i^T] \begin{bmatrix} Z_{xx}(i) & Z_{xu}(i) & Z_{xw}(i) \\ Z_{ux}(i) & Z_{uu}(i) & Z_{uw}(i) \\ Z_{wx}(i) & Z_{wu}(i) & Z_{ww}(i) \end{bmatrix} \begin{bmatrix} \delta\mathbf{x}_i \\ \delta\mathbf{u}_i \\ \delta\mathbf{w}_i \end{bmatrix}, \end{aligned} \tag{15}$$

where $Z(i) = L(\mathbf{x}_i, \mathbf{u}_i, \mathbf{w}_i) + V(\mathbf{x}_{i+1})$. Here, $\delta\mathbf{u}_i$ and $\delta\mathbf{w}_i$ must be chosen to minimize and maximize the return $R(\mathbf{x}_i, \mathbf{u}_i, \mathbf{w}_i)$ respectively, i.e.,

$$\begin{aligned} \delta\mathbf{u}_i &= Z_{uu}^{-1}(i)[Z_{ux}(i)\delta\mathbf{x}_i + Z_{uw}(i)\delta\mathbf{w}_i + Z_u(i)] \\ \delta\mathbf{w}_i &= Z_{ww}^{-1}(i)[Z_{wx}(i)\delta\mathbf{x}_i + Z_{wu}(i)\delta\mathbf{u}_i + Z_w(i)]. \end{aligned} \tag{16}$$

By solving (16), we can derive both $\delta\mathbf{u}_i$ and $\delta\mathbf{w}_i$. After updating the control output $\mathbf{u}_i$ and the disturbance $\mathbf{w}_i$ with derived $\delta\mathbf{u}_i$ and $\delta\mathbf{w}_i$, the value function $V(\mathbf{x}_i)$, first order derivative $V_x(\mathbf{x}_i)$, and second order derivative $V_{xx}(\mathbf{x}_i)$ are given as

$$\begin{aligned} V(\mathbf{x}_i) &= V(\mathbf{x}_{i+1}) - Z_u(i)Z_{uu}^{-1}(i)Z_u(i) - Z_w(i)Z_{ww}^{-1}(i)Z_w(i) \\ V_x(\mathbf{x}_i) &= Z_x(i) - Z_u(i)Z_{uu}^{-1}(i)Z_{ux}(i) - Z_w(i)Z_{ww}^{-1}(i)Z_{wx}(i) \\ V_{xx}(\mathbf{x}_i) &= Z_{xx}(i) - Z_{xu}(i)Z_{uu}^{-1}(i)Z_{ux}(i) - Z_{xw}(i)Z_{ww}^{-1}(i)Z_{wx}(i). \end{aligned} \tag{17}$$

B Ground Contact Model

The function $H()$ in equation (7) includes the mapping (velocity change) caused by the ground contact. To derive the first derivative of the value function $V_x(\mathbf{x}_N)$ and the second derivative $V_{xx}(\mathbf{x}_N)$, where $\mathbf{x}_N$ denotes the terminal state, the function $H()$ should be analytical. Then, we used an analytical ground contact model[5]:

$$\dot{\boldsymbol{\theta}}^{+} - \dot{\boldsymbol{\theta}}^{-} = M^{-1}(\boldsymbol{\theta})D(\boldsymbol{\theta})\mathbf{f}\Delta t, \tag{18}$$

where $\boldsymbol{\theta}$ denotes joint angles of the robot, $\dot{\boldsymbol{\theta}}^{-}$ denotes angular velocities before ground contact, $\dot{\boldsymbol{\theta}}^{+}$ denotes angular velocities after ground contact, M denotes inertia matrix, D denotes Jacobian matrix which converts the ground contact force $\mathbf{f}$ to the torque at each joint, and Δt denotes time step of the simulation.

REFERENCES

[1] F. Asano, M. Yamakita, and K. Furuta. Virtual passive dynamic walking and energy-based control laws. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2000.

[2] C. Chevallerau and Y. Aoustin. Optimal running trajectories for a biped. In *2nd International Conference on Climbing and Walking Robots*, pages 559–570, 1999.

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

[4] M. Hardt, J. Helton, and K. Kreuts-Delgado. Optimal biped walking with a complete dynamical model. In *Proceedings of the 38th IEEE Conference on Decision and Control*, pages 2999–3004, 1999.

[5] Y. Hurmuzlu and D. B. Marghitu. Rigid body collisions of planar kinematic chains with multiple contact points. *International Journal of Robotics Research*, 13(1):82–92, 1994.

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

[7] S. Kagami, K. Nishiwaki, J. J. Kuffner, Y. Kuniyoshi, M. Inaba, and H. Inoue. Design and implementation of software research platform for humanoid robotics:H7. In *International Conference on Humanoid Robots*, pages 253–258, 2001.

[8] Y. Kuroki, T. Ishida, J. Yamaguchi, M. Fujita, and T. Doi. A small biped entertainment robot. In *International Conference on Humanoid Robots*, pages 181–186, 2001.

[9] T. McGeer. Passive dynamic walking. *International Journal of Robotics Research*, 9(2):62–82, 1990.

[10] S. Miyakoshi, G. Cheng, and Y. Kuniyoshi. Transferring human biped walking function to a machine -towards the realization of a biped bike-. In *4th International Conference on Climbing and Walking Robots*, pages 763–770, 2001.

[11] K. Yokoi, F. Kanehiro, K. Kaneko, K. Fujiwara, S. Kajita, and H. Hirukawa. A honda humanoid robot controlled by aist software. In *International Conference on Humanoid Robots*, pages 259–264, 2001.