2007 IEEE International Conference on
Robotics and Automation
Roma, Italy, 10-14 April 2007

# Learning to acquire whole-body humanoid CoM movements to achieve dynamic tasks

Takamitsu Matsubara*†, Jun Morimoto†‡, Jun Nakanishi†‡, Sang-Ho Hyon†‡, Joshua G.Hale†‡, and Gordon Cheng†‡

* Nara Institute of Science and Technology, Ikoma, Nara, Japan
† ATR Computational Neuroscience Laboratories, Kyoto, Japan
‡ Japan Science and Technology Agency, ICORP, Computational Brain Project, 4-1-8 Honcho, Kawaguchi, Saitama, Japan
{takam-m, xmorimo, jun, sangho, jhale, gordon}@atr.jp

*Abstract*— **This paper presents a novel approach to acquire dynamic whole-body movements on humanoid robots focussed on learning a control policy for the Center of Mass. A policy-gradient method is used to acquire a CoM movement as a control policy for achieving a desired dynamic task. A CoM-Jacobian-based redundancy resolution is then used to compute angular velocities for all joints in order to achieve a whole-body movement consistent with the CoM movement acquired through learning. To demonstrate the effectiveness of our method, we apply it in simulation to the learning of a strong punching movement on the Fujitsu humanoid robot, Hoap-2.**

*Index Terms*— **Reinforcement learning, Humanoid robot, Whole-body movement, Policy-gradient method.**

## I. Introduction

Recently, there has been a growing interest in developing humanoid robots and control methods to achieve dynamic whole-body movements on humanoid robots [1], [2], [3], [4]. Since humanoid robots have a similar physical structure to humans, it is expected that humanoid robots will help us in many tasks in our normal living spaces without any additional environment-specific equipment. Especially for the last decade, a number of methods for achieving various tasks on a humanoid robot have been explored, mainly to achieve biped walking and balancing [5], [6], [7], [8]. However, even though a number of real humanoid robots have demonstrated dynamic whole-body movements based on these methods, it is still infeasible to set up humanoid robots in our own living spaces in order to help us because they do not have an ability to adapt to a new environment like humans and animals.

As one of the candidate solutions for granting humanoid robots with such an ability, reinforcement learning (RL) is a promising method compared with other learning frameworks. Reinforcement learning is a framework for improving the control rules of the agent, *i.e.*, the robot, through interaction with the environment according to a trial-and-error paradigm unless using an explicit model of the environment [9], [10]. However, with an increase of dimensionality in state and action space, RL often requires not only a large number of iterations, but also has a large computational cost, especially in the case of learning a complex control policy. Although there have been many attempts to apply reinforcement learning methods for several robots in simulation and real hardware systems in order to acquire a desired movement, most of the robots to which learning has been applied so far have a small number of degrees of freedom (DoFs) and not as many as the 20 to 30 DoFs typically offered by humanoid robots [11], [12], [13], [14], [15]. To the best of our knowledge, only one attempt successfully achieved learning of desired movements on a small humanoid robot [16]. The work was focused on the learning of biped walking [16]. This paper focuses on developing a general approach suitable for learning various dynamic tasks on humanoid robots.

In this paper, we suggest a novel approach for acquiring dynamic whole-body movements on humanoid robots, focused on learning a control policy for the Center of Mass (CoM) in the robot rather than joint trajectories. Humanoid robots cannot however, directly control their own CoM through joint torques because they are not constrained by the ground. Moreover, unreasonable target joint trajectories are infeasible because they make the robot fall over due to dynamic inconsistency. Thus, in order to achieve a desirable CoM movement, it is promising to directly control the ground reaction force [17]. Furthermore, if we keep the Zero Moment Point (ZMP) [18] in the support polygon during CoM control, it may be possible to prevent robots from falling over due to moments on the edges of their feet. A CoM-Jacobian-based redundancy resolution is used to compute angular velocities for all joints to in order to achieve a whole-body movement consistent with the desired CoM movement [7]. The above framework makes the learning problem for the whole-body movement a more reasonable task.

We demonstrate the effectiveness of our proposed framework by applying it to a ball-punching task in numerical simulations using a commercial humanoid robot, Hoap-2.

The organization of this paper is as follows. In Section II, we briefly introduce Zero Moment Point and the ZMP equation. Next we describe how we can control the CoM by manipulating the ZMP based on the ZMP equations in Section III. In Section IV, we present the policy-gradient method we use to learn an appropriate control policy for a desired full-body movement on a humanoid robot. In section V, we present a concrete example of the learning system for

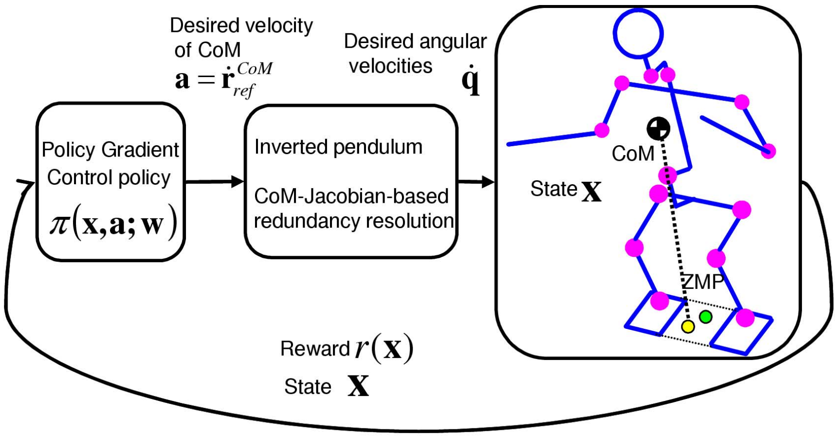
Desired velocity of CoM
$\mathbf{a} = \dot{\mathbf{r}}_{ref}^{CoM}$
Desired angular velocities
$\dot{\mathbf{q}}$
Policy Gradient Control policy
$\pi(\mathbf{x},\mathbf{a};\mathbf{w})$
Inverted pendulum
CoM-Jacobian-based redundancy resolution
CoM
State $\mathbf{X}$
ZMP
Reward $r(\mathbf{x})$
State $\mathbf{X}$

Fig. 1. Approach for learning a desired whole-body movement on a humanoid robot

a ball-punching task on a humanoid robot. In section VI, we describe the results achieved by applying the proposed method in numerical simulations. Finally, we summarize this paper and discuss our future work.

## II. Approach for learning a desired whole-body movement on a humanoid robot

In this section, we briefly describe our suggested approach for learning a desired whole-body movement on a humanoid robot. Figure 1 shows a rough sketch of our suggested approach. The approach focuses on learning a CoM movement suitable for the achievement of the task on a humanoid robot. The CoM is one of the most important features of humanoid robots because it conveniently represents the whole motion of the humanoid robot. Based on this insight, we propose to focus on the learning the CoM in order to achieve a desirable movement on humanoid robot, rather than learning the movement in terms of all joints.

From a motor learning perspective, learning a CoM movement is simpler and easier than learning all joint movements directly. A CoM-Jacobian-based redundancy-resolution technique is used to compute angular velocities for all joints in order to achieve a whole-body movement consistent with the desired CoM movement [7]. We use a weighting matrix in the weighted pseudo-inverse computation which has a significant effect on the movement in joint space. We will focus on generalizing this aspect in forthcoming research. In this work, we use weights manually tuned in advance. The following two sections describe the components of the method: CoM control based on the ZMP, distribution of a CoM movement into joint space, and reinforcement learning of a CoM movement.

## III. CoM controller based on the ZMP equation

As we mentioned in Section I, for the last decade, many humanoid robots have achieved various dynamic tasks such as biped walking and balancing. Most proposed methods are based on the ZMP -an equation representing the dynamics of a humanoid robot's CoM in an approximated manner. This section describes a method for achieving control of the CoM based on the ZMP.

### A. ZMP Compensation control

According to Nagasaka [17] , assuming a mass-concentrated model, the relationship between the moment acting on the ZMP and the objective ZMP is given as

$$\boldsymbol{n}^{ZMP} = \boldsymbol{n}^{OZMP} + (\boldsymbol{r}^{OZMP} - \boldsymbol{r}^{ZMP}) \times \boldsymbol{f}^{CoM}, \quad (1)$$

$$\boldsymbol{n}^{OZMP} = (\boldsymbol{r}^{CoM} - \boldsymbol{r}^{OZMP}) \times \boldsymbol{f}^{CoM}, \quad (2)$$

where $\boldsymbol{n}^{ZMP} \in \boldsymbol{R}^3$ and $\boldsymbol{n}^{OZMP} \in \boldsymbol{R}^3$ are ZMP and objective ZMP moments respectively. $\boldsymbol{r} \in \boldsymbol{R}^3$ is the position vector and $\boldsymbol{f} \in \boldsymbol{R}^3$ is the force acting on the CoM.

From the definition of the ZMP: the point such that horizontal components of the moment acting at the point are zero, we can derive a control law to compensate the ZMP to the objective ZMP by kinematically manipulating the CoM as follows:

$$\begin{aligned}\Delta r_{x,i+1}^{CoM} &= K(r_x^{ZMP} - r_x^{OZMP}) + \Delta r_{x,i}^{CoM} \\ &+(\Delta r_{x,i}^{CoM} - \Delta r_{x,i-1}^{CoM}) + K\Delta r_{x,i}^{CoM},\end{aligned} \quad (3)$$

$$\begin{aligned}\Delta r_{y,i+1}^{CoM} &= K(r_y^{ZMP} - r_y^{OZMP}) + \Delta r_{y,i}^{CoM} \\ &+(\Delta r_{y,i}^{CoM} - \Delta r_{y,i-1}^{CoM}) + K\Delta r_{y,i}^{CoM},\end{aligned} \quad (4)$$

where $K = f_{z,i}^{CoM}\Delta t^2/(r_{z,i}^{CoM} - r_{z,i}^{OZMP})$ and $\Delta t$ is a discrete time step. $\Delta r$ is the deviation of position during $\Delta t$. It is straightforward to determine that the desired velocity of the CoM is given as

$$\dot{r}_x^{CoM} = \Delta r_{x,i+1}^{CoM}/dt, \quad (5)$$

$$\dot{r}_y^{CoM} = \Delta r_{y,i+1}^{CoM}/dt. \quad (6)$$

Under such control, the robot can be regarded as an inverted pendulum with its supporting point at the objective ZMP.

### B. Calculating the reference ZMP according to the inverted-pendulum model

As we mentioned above, since the horizontal components of the moment on the ZMP are zero, the mass-concentrated model of the humanoid robot can be regarded as an inverted pendulum. Based on this analogy, we can apply a simple PID controller to control CoM by manipulating the ZMP as described in [7].

The dynamics of the mass-concentrated model approximately linearized around an equilibrium point are given as

$$\ddot{r}_x^{CoM} = \omega^2 (r_x^{CoM} - r_x^{ZMP}), \tag{7}$$

$$\ddot{r}_y^{CoM} = \omega^2 (r_Y^{CoM} - r_y^{ZMP}) \tag{8}$$

where, $\omega = \sqrt{\frac{\ddot{r}_z^{CoM}+g}{r_z^{CoM}-r_z^{ZMP}}}$. The dynamics equations above represent the horizontal movement of the CoM. Due to the symmetry of the $x$ and $y$ components, we are able to focus on the $x$ component in following derivation without loss of generality. By differentiating Equation (7) and ignoring the change in $\omega$, the following equation may be derived.

$$\dddot{r}_x^{CoM} = \omega^2 (\dot{r}_x^{CoM} - \dot{r}_x^{ZMP}). \tag{9}$$

In order to control the CoM $r_x^{CoM}$ with the reference $r_{xref}^{CoM}$ as a target, we can apply following simple controller.

$$\dot{r}_x^{ZMP} = (K_P + \frac{K_I}{s} + K_D s)(\dot{r}_{xref}^{CoM} - \dot{r}_x^{CoM}). \tag{10}$$

$$\dot{r}_{xref}^{CoM} = K_C (r_{xref}^{CoM} - r_x^{CoM}). \tag{11}$$

$K_P$, $K_I$, $K_D$ and $K_C$ are gains. By the final-value theorem, it may easily be proven that $r_x^{CoM}$ converges to $r_{xref}^{CoM}$ with appropriate settings for the gains.

By integrating the two components presented in this section, the CoM can be controlled by manipulating the ZMP. In the next section, we describe a CoM-Jacobian-based redundancy-resolution technique used to achieve a whole-body movement consistent with the desired CoM movement.

### C. Distributing the CoM movement into joint space

In the previous section, we presented a ZMP manipulation method based on the inverted-pendulum model in order to control the CoM according to the desired trajectory. In this section, we present a CoM-Jacobian-based redundancy-resolution technique used to achieve a whole-body movement consistent with the desired CoM movement [7]. We also present the CoM controller used in our framework which is based on the CoM Jacobian.

*1) Whole body motion generation for balancing:* Sugihara *et al.* [7] proposed the concept of, and a calculation method for the CoM Jacobian which relates the velocity of the CoM with the angular velocities of all joints as

$$\dot{\boldsymbol{r}}^{CoM} = \boldsymbol{J}_C(\boldsymbol{q})\dot{\boldsymbol{q}}. \tag{12}$$

$\boldsymbol{J}_C(\boldsymbol{q}) \in \boldsymbol{R}^{3\times n}$ is the CoM Jacobian. $n$ is the the number of DoFs in the robot. By using the CoM Jacobian and the weighted pseudo-inverse calculation, we can distribute the CoM velocity to the angular velocities of all the joints according to a sum-squared minimization of all the joint angular velocities as follows:

$$\dot{\boldsymbol{q}} = \boldsymbol{J}_C^+ \dot{\boldsymbol{r}}^{CoM} + (\boldsymbol{I} - \boldsymbol{J}_C^+ \boldsymbol{J}_C)\boldsymbol{k}, \tag{13}$$

where,

$$\boldsymbol{J}_C^+ = \boldsymbol{W}^{-1}\boldsymbol{J}_C^T(\boldsymbol{J}_C\boldsymbol{W}^{-1}\boldsymbol{J}_C^T)^{-1}, \tag{14}$$

$\boldsymbol{W} = diag\{w_i\}(i = 1, \cdots, n)$ and $\boldsymbol{k} \in \boldsymbol{R}^n$ is an arbitrary vector. $I \in \boldsymbol{R}^{n\times n}$ is the identity matrix. The above redundancy-resolution technique with a weighting matrix determines a whole-body motion consistent with the desired CoM movements at the velocity level.

*2) CoM controller in the double support case:* We composed the following controller for the CoM assuming both feet are contacting the ground:

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^+\dot{\boldsymbol{r}} + (\boldsymbol{I} - \boldsymbol{J}^+\boldsymbol{J})\boldsymbol{k}, \tag{15}$$

where, $\dot{\boldsymbol{r}} \in \boldsymbol{R}^6 = [\dot{\boldsymbol{r}}_C - \dot{\boldsymbol{r}}_{rl}, \dot{\boldsymbol{r}}_C - \dot{\boldsymbol{r}}_{ll}]^T$ and $\boldsymbol{J}(\boldsymbol{q}) \in \boldsymbol{R}^{6\times n} = [\boldsymbol{J}_C(\boldsymbol{q}) - \boldsymbol{J}_{rl}(\boldsymbol{q}), \boldsymbol{J}_C(\boldsymbol{q}) - \boldsymbol{J}_{ll}(\boldsymbol{q})]^T$. $\boldsymbol{k} \in \boldsymbol{R}^6$ is an arbitrary vector. The variables are defined in Figure 2.

The desired $\dot{\boldsymbol{r}}$ to control the CoM according to the desired trajectory is given by Equations (5), (6) and (10).

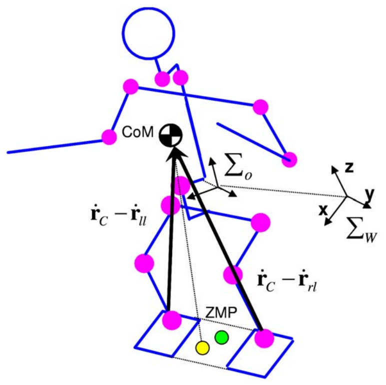
CoM
$\Sigma_o$
z
y
x
$\Sigma_W$
$\dot{\mathbf{r}}_C - \dot{\mathbf{r}}_{ll}$
$\dot{\mathbf{r}}_C - \dot{\mathbf{r}}_{rl}$
ZMP

Fig. 2. The definition of the variables.

## IV. Reinforcement learning for CoM movement

In this section, we present a reinforcement learning method used for the proposed learning framework. Specifically, we use a policy-gradient method for learning CoM motion. The policy-gradient method is a kind of reinforcement learning method which maximizes the average reward with respect to parameters controlling action rules known as the *policy* [19], [11], [20]. Compared with most standard value-function-based reinforcement learning methods, this type of method has particular features suited to robotic applications. Firstly, the policy-gradient method is applicable to Partially Observable Markov Decision Processes [21]. It is almost impossible to consider all possible states of the robot because even if it has a complete set of sensors there will be a degree of noise. It is also possible to consider a partial set of states as input for a reinforcement learning system. Secondly, the policy-gradient method is a stochastic gradient-descent method. The policy can therefore be improved upon every update. In this section, we briefly describe a framework for reinforcement learning with the policy-gradient method.

## A. Reinforcement Learning with a policy-gradient method

Assuming a Markov Decision Process, the average reward, discounted cumulative reward and value functions are defined as

$$\begin{aligned}
\eta(\boldsymbol{\theta}) &= \lim_{T\to\infty} \mathbf{E}\left[\frac{1}{T}\sum_{t=0}^{T} r(\mathbf{x}_t)\right], && (16)\\
\eta_\beta(\boldsymbol{\theta}) &= \lim_{T\to\infty} \mathbf{E}\left[\frac{1}{T}\sum_{t=0}^{T} \beta r(\mathbf{x}_t)\right], && (17)\\
V_\beta^\pi(\mathbf{x}) &= \mathbf{E}\left[\sum_{k=0}^{\infty} \beta r_{t+k+1}\middle|\mathbf{x}_t=\mathbf{x}\right], && (18)\\
Q_\beta^\pi(\mathbf{x},a) &= \mathbf{E}\left[\sum_{k=0}^{\infty} \beta r_{t+k+1}\middle|\mathbf{x}_t=\mathbf{x}, a_t=a\right]. && (19)
\end{aligned}$$

$\eta(\boldsymbol{\theta})$ is the average reward and $\eta_\beta(\boldsymbol{\theta})$ is the discounted cumulative reward. $V_\beta^\pi(\mathbf{x})$ and $Q_\beta^\pi(\mathbf{x},a)$ are state-value function and action-value function, respectively [9]. $\mathbf{x}$ is the state, $a$ is the action and $\boldsymbol{\theta}$ is the parameters of the stochastic policy. $\beta$ is a discounting factor. The goal of reinforcement learning here is to maximize the average reward.

If we can calculate the gradient of $\eta(\boldsymbol{\theta})$ with respect to policy parameters $\boldsymbol{\theta}$, it is possible to search for a (sub)optimal policy in policy-parameter space by updating the parameters to $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha\nabla\eta(\boldsymbol{\theta})$. $\nabla\eta(\boldsymbol{\theta})$ is the gradient of $\eta(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$. Various derivations and algorithms have been proposed in order to estimate the gradient based on sampling, though interaction with the environment. According to Kimura and Kobayashi [22], the gradient is given by

$$\begin{aligned}
\nabla\eta &= (1-\beta)\nabla\eta_\beta && (20)\\
&= (1-\beta)\int d(\mathbf{x})\int \pi(a,\mathbf{x})\Big[\nabla\log d(\mathbf{x})\\
&\quad +\frac{1}{1-\beta}\nabla\log\pi(a,\mathbf{x})\Big]Q_\beta^\pi(\mathbf{x},a)\,da\,d\mathbf{x} && (21)\\
&= \int d(\mathbf{x})\int \pi(a,\mathbf{x})\Big[(1-\beta)\nabla\log d(\mathbf{x})\\
&\quad +\nabla\log\pi(a,\mathbf{x})\Big]\left\{Q_\beta^\pi(\mathbf{x},a)-V_\beta^\pi(\mathbf{x})\right\}da\,d\mathbf{x} && (22)\\
&= \lim_{T\to\infty,\beta\to 1}\frac{1}{T}\sum_{t=0}^{T}\nabla\log\pi(a_t,\mathbf{x}_t)\sum_{s=t}^{T}\beta^{s-t}\delta(\mathbf{x}_s,a_s)\\
&= \lim_{T\to\infty,\beta\to 1}\frac{1}{T}\sum_{t=0}^{T}\delta(\mathbf{x}_t,a_t)\sum_{s=0}^{t}\beta^{t-s}\nabla\log\pi(a_s,\mathbf{x}_s). && (23)
\end{aligned}$$

Where, $\pi(\mathbf{x},a;\boldsymbol{\theta}) = P(a|\mathbf{x};\boldsymbol{\theta})$ is the stochastic policy, which maps a state $\mathbf{x}$ to an action $a$ stochastically. $\nabla\pi(\mathbf{x},a;\boldsymbol{\theta})$ means the deviation of $\pi(\mathbf{x},a;\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$. $d(\mathbf{x})$ is the stationary distribution of $\mathbf{x}$. $\delta(\mathbf{x},a)$ is TD-error defined as $\delta(\mathbf{x}_t,a_t) = r(\mathbf{x}_t) + \beta\int p(\mathbf{x}_{t+1}|\mathbf{x}_t,a_t)V_\beta^\pi(\mathbf{x}_{t+1})d\mathbf{x}_t - V_\beta^\pi(\mathbf{x}_t)$. Equation (20) is presented in [23] as a Theorem.1, and Equation (21) is derived in [24]. The derivation of Equation (22) is based on $\int\nabla\pi(\mathbf{x},a)V_\beta^\pi(\mathbf{x})da = 0$. If we neglect $V_\beta^\pi(\mathbf{x})$, the algorithm is exactly same as the GPOMDP algorithm developed in [20]. As pointed out in [20], the discounting factor $\beta$ controls a bias-variance trade-off in the policy-gradient estimated by sampling.

In fact, we update the policy parameters according to the following rule: $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha D_t\delta(\mathbf{x}_t,a_t)$, where, $D$ is updated by $D_t = \beta D_{t-1} + \nabla\log\pi(\mathbf{x}_t,a_t)$. However, to gain TD-error $\delta(\mathbf{x}_t,a_t)$, we need the state-value function $V_\beta^\pi(\mathbf{x})$. In this paper, we simultaneously approximate it by using function approximator $\hat{V}_\beta^\pi(\mathbf{x};\mathbf{w})$ and a simple TD-learning method presented as $\mathbf{w} = \mathbf{w} + \alpha\delta_t\frac{\partial\hat{V}_\beta^\pi(\mathbf{x};\mathbf{w})}{\partial\mathbf{w}}$. TD-error $\delta(\mathbf{x}_t,a_t)$ is then approximately calculated by $\delta(\mathbf{x}_t,a_t) = r(x_t) + \beta\hat{V}_\beta^\pi(\mathbf{x}_{t+1}) - \hat{V}_\beta^\pi(\mathbf{x}_t)$. Note that $\beta$ should be satisfy $0 \le \beta < 1$ in order to prevent the state-value function from diverging.

## V. Application to learning of a dynamic task: ball-punching

In order to demonstrate the effectiveness of our proposed learning framework for learning whole-body movements with a humanoid robot, we applied it to the learning of a dynamic ball-punching motion. The goal was to make the ball-punching stronger though the learning process focused on the CoM motion. This section describes the implementation of a punching motion and the learning process.

### A. Punching motion projected onto the null-space of the CoM controller

A punching motion was straightforwardly implemented by tracking a target trajectory in task space. In this study, we achieve the tracking control in the null-space of the CoM controller by introducing the following vector as the arbitrary vector in Equation (15):

$$\boldsymbol{k} = \tilde{\boldsymbol{J}}_{ra}^{+}(\dot{\boldsymbol{r}}_{ra} - \boldsymbol{J}_{ra}\boldsymbol{J}^{+}\dot{\boldsymbol{r}}), \qquad (24)$$

where, $\boldsymbol{J}_{ra} \in \boldsymbol{R}^{3\times n}$ is the Jacobian relating the right hand velocity in task space $\dot{\boldsymbol{r}}_{ra}$ with $\dot{\boldsymbol{q}}$ as $\dot{\boldsymbol{r}}_{ra} = \boldsymbol{J}_{ra}\dot{\boldsymbol{q}}$, and $\tilde{\boldsymbol{J}}_{ra}^{+} = \boldsymbol{J}_{ra}(\boldsymbol{I} - \boldsymbol{J}^{+}\boldsymbol{J})$. Introducing this vector yields target tracking with the right hand in the null-space of the CoM controller [25].

### B. Learning the CoM movement for the dynamic punch

*1) The Gaussian policy and function approximator for the state-value function:* We implemented the following Gaussian policy as a stochastic policy for controlling the CoM.

$$\pi(\mathbf{x},a;\boldsymbol{\theta}) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(\frac{-(a-\mu(\mathbf{x};\boldsymbol{\theta}))^2}{2\sigma^2}\right), \qquad (25)$$

where $\mu(\mathbf{x};\boldsymbol{\theta}) = \boldsymbol{\theta}^T\boldsymbol{\phi}(\mathbf{x})$. $\mathbf{x}$ is the state, $a$ is the action. We located the Gaussian basis functions $\boldsymbol{\phi}(\mathbf{x})$ on a grid with even intervals in each dimension of the observation space as in [10], [15]. The function approximator for the state-value function is also modeled as $\hat{V}_\beta^\pi(\mathbf{x}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x})$

*2) The reward function:* The purpose of the ball-punching task is to make the punching as strong as possible. We designed the reward function according to this objective as

$$r = (t - 2.5)\sqrt{\boldsymbol{v}_b^T \boldsymbol{v}_b} \tag{26}$$

because the velocity of the ball $\boldsymbol{v}_b$ hit by punching is proportional to the momentum of the ball. The term associated with time $t$ is incorporated in the reward function in order to avoid local-minima motions which involve the robot falling forwards and disregard the timing of the punch. The value $2.5$ is a bias to achieve distribution of the reward to positive and negative. The negative reward $-5$ is given when the both feet leave the ground in order to avoid acquiring a punching motion with jumping.

## VI. Simulation settings and results

We applied the proposed learning framework to the acquisition of a strong punching movement on a Fujitsu Hoap-2 humanoid robot (see Figure 3) in numerical simulation. As a first step, in this simulation, we focused on learning a policy controlling x-axis component of the CoM, *i.e.*, the output from the policy is the target velocity of the x-axis component of CoM $\dot{r}^{CoM}_{xref}$. The state-space was simply defined as $\mathbf{x} = \left(r^{CoM}_x, t\right)$. We then allocated 100 $(= 10 \times 10)$ basis functions $\boldsymbol{\phi}(\mathbf{x})$ in the state-space $(-1.0 < r^{CoM}_x < 0.0, 0.5 < t < 4.0)$ to represent the mean of the policy $\mu(\mathbf{x})$. The ball was modeled as a simple point mass (0.1kg) and the impact calculation model was a spring-damper model. A spring-damper model was also used to model the floor. The integration time-step for the robot was 0.2ms, and the time interval for learning was 50ms.

For the CoM and right-arm controllers, it is required to set the weighting matrix suitable for this task to appropriately achieve a whole-body motion. In order to avoid using the DoFs in the right arm (which are used for the punching motion) for the CoM controller, the weights in right arm are set to smaller (0.01) than other joints (1.0) in the CoM controller described in Equation (13). For the right-arm controller described in Equation (24), to achieve a punching motion mainly using the right arm, we set the weights in body joint larger (3.0) than other joints (1.0). The target trajectory for the right-arm controller in order to achieve a punching motion was designed as $r_{ra_x\_ref} = p\sin\left(2\pi f\left(t-3.5\right)\right) + q \; (t \geq 3.5)$, and we set the parameters so that the amplitude $p = -0.03$m, the bias $q = 0.21$m and the frequency $f = 1.5$ Hz by considering the Hoap2's physical model. During $0 < t < 3.5$, $r_{ra_x\_ref}$ is the constant $p$.

Figure 4 shows the reward at each episode according to the policy-gradient method. The curve means that the (sub)optimal punching motion with maximal reward was acquired in around 3000 episodes. Figure 5 is an acquired policy for controlling x-axis component of CoM, and Figure 6 presents a whole-body punching motion acquired with the control policy.

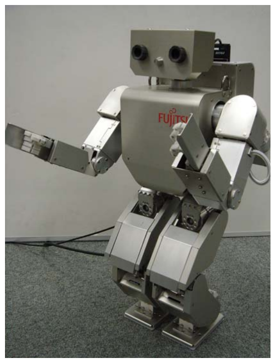

Fig. 3. Fujitsu humanoid robot Hoap-2. 6DoF for the legs, 4DoF for the arms and 1DoF for the waist. The total weight is about 7kg, and the height is about 0.4m.

The punching motion with keeping the CoM at the initial point yielded the ball momentum about 0.037 kgm/sec. The acquired punching motion without any probabilistic factors made the ball momentum about 0.085 kgm/sec in the average (the standard deviation was 0.005) , which means the ball momentum generated by the learned policy was about 2.3 times larger than initial performance.

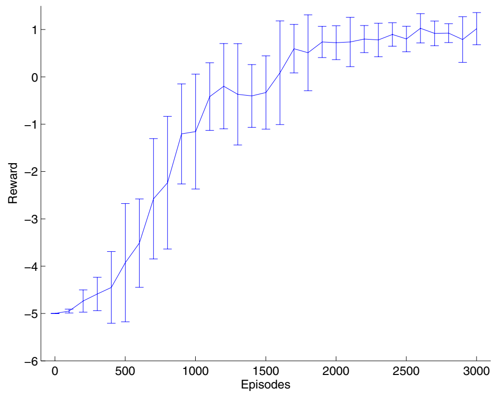

Fig. 4. The acquired reward at each episodes. The learning curve was averaged by 5 experiments and smoothed out by taking a 50-moving average.

## VII. Discussion

This paper presented an approach for acquiring dynamic whole-body movements on humanoid robots focussed on learning a control policy for the Center of Mass. We applied the framework to the learning of a dynamic ball-punching motion on Hoap-2 model in numerical simulations. As a result, we demonstrated that it is possible to acquire dynamic punching motions though learning using our approach. We are currently working on implementing the acquired control policy on the real humanoid robot (in Figure 3). The proposed learning framework will be applied to the humanoid robot in real environment in the near future.

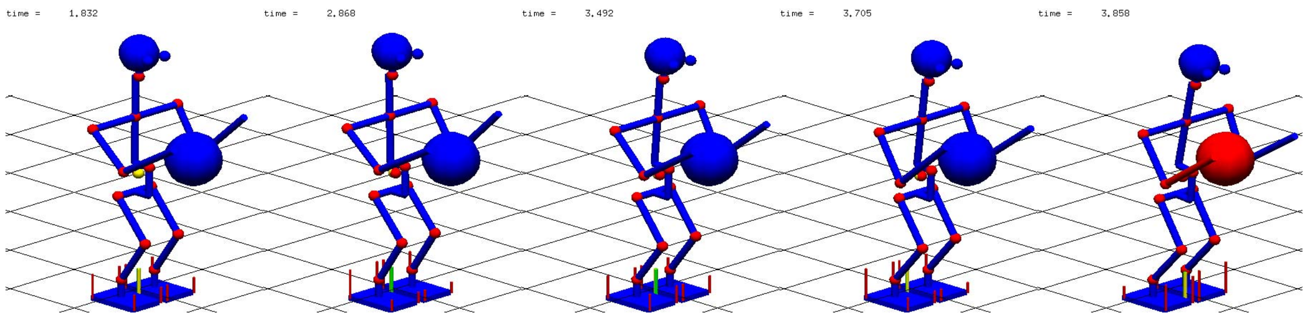

Fig. 6. An acquired whole-body punching movement. The snapshots are corresponding in 1.832 sec, 2.868 sec, 3.492 sec, 3.705 sec and 3.858 sec, respectively. The red bar on the foot of the robot means the ground reaction force.

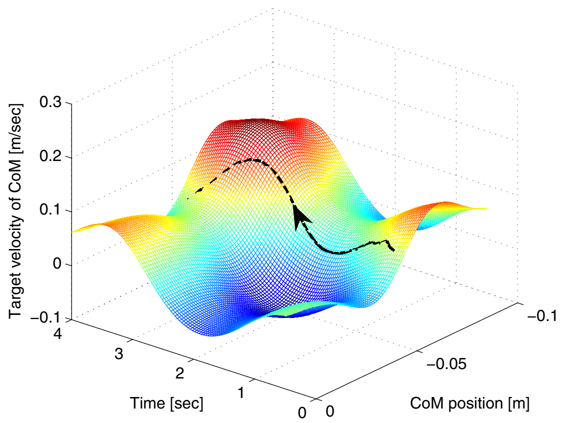

Fig. 5. An acquired control policy for axis component of the CoM. The dash-line is the trajectory corresponding to the punching motion in Figure 6

## References

[1] K. Hirai, M. Hirose, Y. Haikawa, and T. Takenaka, "The development of honda humanoid robot," in *IEEE International Conference on Robotics and Automation*, 1998, pp. 1321–1326.

[2] Y. Kuroki, T. Ishida, J. Yamaguchi, M. ujita, and T. Doi, "A small biped entertainment robot," in *IEEE-RAS International Conference on Humanoid Robots*, 2001, pp. 181–186.

[3] J. Morimoto, G. Endo, J. Nakanishi, S. Hyon, G. Cheng, D. Bentivegna, and C. Atkeson, "Modulation of simple sinusoidal patterns by a coupled oscillator model for biped walking," in *IEEE International Conference on Robotics and Automation*, 2006, pp. 1579–1584.

[4] S. Hyon and G. Cheng, "Passivity-based whole-body motion control for humanoids: Gravity compensation, balancing and walking," in *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, 2006, pp. 4915–4922.

[5] K. Nagasaka, H. Inoue, and M. Inaba, "Dynamic walking pattern generation for a humanoid robot based on optimal gradient method," in *IEEE International Conference on Systems, Man and Cybernetics*, 1998, pp. 908–913.

[6] S. Kagami, F. Kanehiro, Y. Tamiya, M. Inaba, and H. Inoue, "Autobalancer: An online dynamic balance compensation scheme for humanoid robots," in *Algorithmic and Computational Robotics: New Directions*, B. R. Donald, K. Lynch, and D. Rus, Eds. A K Peters, Ltd., 2001, pp. 329–340.

[7] T. Sugihara and Y. Nakamura, "Whole-body cooperative balancing of humanoid robot using cog jacobian," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2002, pp. 2575–2580.

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

[9] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[10] K. Doya, "Reinforcement learning in continuous time and space," *Neural Computation*, vol. 12, pp. 219–245, 2000.

[11] H. Kimura, K. Miyazaki, and S. Kobayashi, "Reinforcement Learning in POMDPs with Function Approximation," in *Proceedings of the 14th International Conference on Machine Learning*, 1997, pp. 152–160.

[12] H. Kimura, T. Yamashita, and S. Kobayashi, " Reinforcement Learning of Walking Behavior for a Four-Legged Robot," in *Proceedings of the IEEE Conference on Decision and Control*, 2001, pp. 411–416.

[13] J. Morimoto and K. Doya, "Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning," *Robotics and Autonomous Systems*, vol. 36, pp. 37–51, 2001.

[14] R. Tedrake, T. W. Zhang, and H. S. Seung, "Stochastic Policy Gradient Reinforcement Learning on a Simple 3D Biped," in *Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, 2004, pp. 2849–2854.

[15] T. Matsubara, J. Morimoto, J. Nakanishi, M. Sato, and K. Doya, "Learning sensory feedback to cpg for biped locomotion with policy gradient," in *IEEE International Conference on Robotics and Automation*, 2005, pp. 4175–4180.

[16] G. Endo, J. Morimoto, T. Matsubara, J. Nakanishi, and G. Cheng, "Learning cpg sensory feedback with policy gradient for biped locomotion for a full body humanoid," in *The Twentieth National Conference on Artificail Intelligence (AAAI-05)*, 2005, pp. 1267–1273.

[17] K. Nagasaka, "The whole-body motion generation of humanoid robot using dynamics filter (in japanese)," *PhD thesis, Tokyo univ, Japan*, 2000.

[18] M. Vukobratović and B. Borovac, "Zero-moment point - thirty five years of its life," *International Journal of Humanoid Robotics*, 2004.

[19] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine Learning*, vol. 8, pp. 229–256, 1992.

[20] J. Baxter and P. L. Bartlett, "Infinite-horizon policy-gradient estimation," *Journal of Artificial Intelligence Research*, vol. 15, pp. 319–350, 2001.

[21] D. A. Aberdeen, "Policy-gradient algorithms for partially observable markov decision processes, ph.d thesis," *Australian National University*, 2003.

[22] H. Kimura and S. Kobayashi, "An analysis of actor/critic algorithms using eligibility traces: Reinforcement learning with imperfect value function," *Internal Conference on Machine Learning*, pp. 278–286, 1998.

[23] J. Baxter and P. L. Bartlett, "Direct gradient-based reinforcement learning: I. gradient estimation algorithms," *Technical Report, Computer Sciences Lab, Australian National University*, 1999.

[24] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," *Advances in Neural Information Processing Systems*, vol. 12, pp. 1057–1063, 2000.

[25] T. Yoshikawa, *Foundations of robotics: analysis and control*. MIT Press, 1990.