2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)
Beijing, China, November 6-9, 2018

Biped Gait Control based on Spatially Quantized Dynamics

Shuuji Kajita[1], Mehdi Benallegue[1], Rafael Cisneros[1], Takeshi Sakaguchi[1], Shin'ichiro Nakaoka[1], Mitsuharu Morisawa[1], Hiroshi Kaminaga[1], Iori Kumagai[1], Kenji Kaneko[1], Fumio Kanehiro[1]

***Abstract*— We have realized a biped walking control based on the spatially quantized dynamics (SQD) which discretizes a continuous system by a constant unit length along the walk direction. By using SQD, a prescribed sagittal kinematic pattern can be transformed into dynamically consistent robot motion in real-time. The lateral motion is generated by preview control which uses future ZMP predicted by SQD at every control cycle. A successful biped walk of HRP-2Kai with fully stretched knees and long stride was realized by the proposed method.**

I. INTRODUCTION

Model-based walking pattern generation and control based on Zero-Moment Point (ZMP) is a systematic technique, which is sound in terms of engineering [1]. On the other hand, many people criticize that ZMP-based walking is unnatural, as it often leads to a bent knee walk profile with short strides.

This problem was tackled by researchers like Morisawa et al.[2], Li et al.[3], and other researchers [4], [5], [6]. Ogura et al.[7] and Miura et al.[8] have realized good looking human-like biped locomotion by real robots. However, their works seem to rely on special mechanisms like a pelvis roll joint or toe joints to ease the mechanical singularities. Recently, optimization based methods have become popular to achieve more versatile biped gait generation [9], [10], [11], [12]. Although they provided general methods, the resulting gaits were not quite natural yet due to the crouching postures used for singularity avoidance.

Griffin et al. have proposed a trajectory planning based on instantaneous capture point and step duration optimization to solve the above-stated problems [13]. Their insight was that step timings can be used to reduce the required knee bend at the walking. Similarly, we also proposed to use of *timings* for a better gait generation, however, in an unprecedented manner. In our "Spatially Quantized Dynamics (SQD)", we introduced uniform spatial quantization which results in variable time steps. We demonstrated that our SQD based method can realize natural-looking biped walking by computer simulations [14].

In this paper, we describe details of this walking control method, which enabled our humanoid robot HRP-2Kai [15] to walk with fully stretched legs and wide stride (Fig.1).

[1]The authors are with Humanoid Robotics Group of Intelligent Systems Institute, AIST, 1-1-1 Umezono, Tsukuba, Ibaraki 305-8560, Japan `{s.kajita, mehdi.benallegue, rafael.cisneros, hiroshi.kaminaga, iori-kumagai, sakaguchi.t, s.nakaoka, m.morisawa, k.kaneko, f-kanehiro}@aist.go.jp`

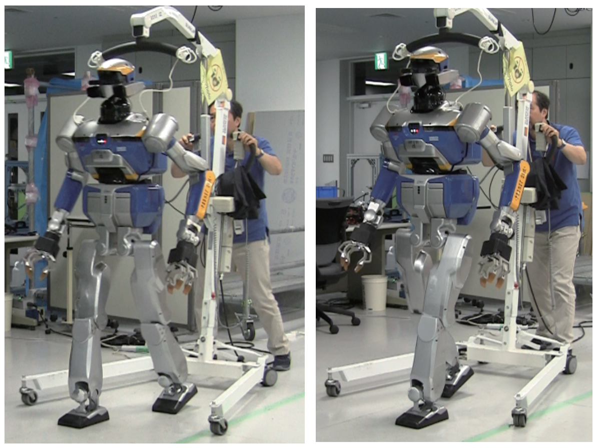
Fig. 1. HRP-2Kai walking with fully stretched knees

II. SAGITTAL CONTROL

In this section, we discuss the sagittal gait generation and control as if the robot is constrained in the sagittal plane. The lateral gait generation and control are discussed in the next section.

A. Kinematic Walking Pattern

As the first step of our gait generation scheme, we prepare a series of robot poses for the desired walk in the sagittal plane. We concentrate on pure kinematics to realize the desired strides with stretched support legs. The swing legs trajectories are designed to start from toe-off and to end at heel-strike.

Figure 2 is an example pattern for three steps of 50cm. The hip and the center of mass (CoM) calculated from this pattern is shown in Fig.3. They move up and down a few centimeters for each step as the consequence of the gait with stretched support legs.

In this phase, we don't take into account of the static balance about the center of mass nor the dynamic balance about the ZMP.

As a parameter to represent this walking pattern, we take the horizontal position of the hip joint and discretize it with a uniform grid size of Δx. We use index variable k to represent a node of the grid. The hip position at the k-th node is given as

$$x_k = \Delta x \cdot k \ \ (k = 0 \cdots N), \tag{1}$$

where N is determined by the total walking distance D as $N = \lfloor D/\Delta x \rfloor$. In this example, we used $\Delta x = 0.001$m for

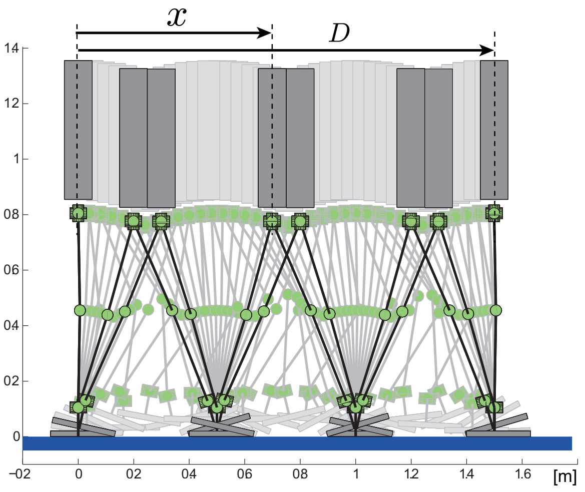

Fig. 2. Kinematic walking pattern

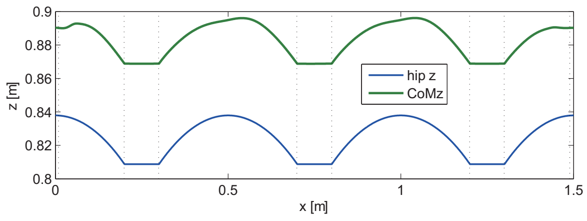

Fig. 3. Hip and CoM heights

the walking pattern of $D = 1.5$m, thus we have $N = 1500$ and 1501 poses in total.

A kinematic walking pattern is represented by the following series of vectors:

$$\boldsymbol{q}_k^* = \boldsymbol{q}(x_k) \ \ (k = 0 \cdots N), \tag{2}$$

where $\boldsymbol{q}(x_k)$ is a pose vector which consists of the joint angles when the hip is at x_k.

Note that this kind of spatial gait prescription can be observed in prior research like the Hybrid Zero-Dynamics approach [16], [17] and the PDAC framework [18]. The originality of our approach exists in the treatment of dynamics as explained in the next subsection.

B. Spatially Quantized Dynamics

We expect that the robot dynamics can be approximated by linear inverted pendulum model (LIPM)[19], [20] such that the hip horizontal position x corresponds to the center of mass (CoM) location.

$$\begin{aligned} \ddot{x} &= \omega^2(x - p), \\ \omega &:= \sqrt{g/z}, \end{aligned} \tag{3}$$

where z is the average height of CoM of the walking pattern, p is ZMP and g is gravity acceleration. Note that we can also use the real CoM height, which varies in time as in Fig.3. Nevertheless, we treated it as an averaged constant value because of its simplicity and effectiveness confirmed by our experiments.

Unlike an ordinary discretization with unit time, we quantize this system along the uniform spatial grid of (1). Let us look at the state transition from the k-th node to the $(k+1)$-th node of the grid,

$$(t_k, x_k, v_k) \rightarrow (t_{k+1}, x_{k+1}, v_{k+1}).$$

Since we already know the position x, let's take a look at the transition of time. The time varies from t_k to t_{k+1} depending on the speed v_k as

$$\begin{aligned} t_{k+1} &= t_k + \Delta t_k, \\ \Delta t_k &:= \frac{\Delta x}{v_k}. \end{aligned} \tag{4}$$

Note that the above equation requires that $v_k \neq 0$. This is obvious because the CoM would take infinite time to move from x_k to x_{k+1} if its speed is zero ($v_k = 0$). To avoid this divide-by-zero problem, we introduce a lower bound on the speed which can be regarded as being almost stationary. We therefore determined the lowest speed as $\epsilon = 0.005$m/s and used it as the initial/terminal speed for simulation and control. At this lowest speed, the CoM takes $\Delta t = 0.2$s to move $\Delta x = 1$mm, which is the distance between one node and its neighbor.

The change of speed from k to $k + 1$ is given as

$$v_{k+1} = v_k + \ddot{x}\Delta t_k. \tag{5}$$

By substituting the variable time step Δt_k (4) and the acceleration of LIPM (3), we obtain

$$v_{k+1} = v_k + \omega^2(x_k - p_k)\frac{\Delta x}{v_k}. \tag{6}$$

This equation represents the Spatially Quantized Dynamics of LIPM. It is nonlinear since v_k appears in the denominator of the second term of the equation's right hand side, thus we can not apply ordinal control theories.

C. ZMP optimization

Suppose that we have a robot following the Spatially Quantized Dynamics (SQD) of (6) and that should realize the commanded value for its speed and ZMP as v^{cmd} and p^{cmd}. We define a cost function to minimize as

$$J := (v_{k+1} - v^{cmd})^2 + \beta(p_k - p^{cmd})^2, \tag{7}$$

where β is a weight to specify the importance of the ZMP error.

We can solve this optimization problem by substituting the SQD (6) into this cost function and by zeroing its partial derivative about p_k. The ZMP which minimizes the cost J is obtained as

$$\begin{aligned} p_k &= \frac{\beta p^{cmd} - A(B - v^{cmd})}{\beta + A^2} \\ A &:= -\frac{\Delta x}{v_k}\omega^2 \ , B := \frac{\Delta x}{v_k}\omega^2 x_k + v_k. \end{aligned} \tag{8}$$

Let us call this a *Local Optimization*, as it locally optimizes the speed and the ZMP between two adjacent nodes. Figure 4 shows the ZMP and speed calculated by using (6) and (8).

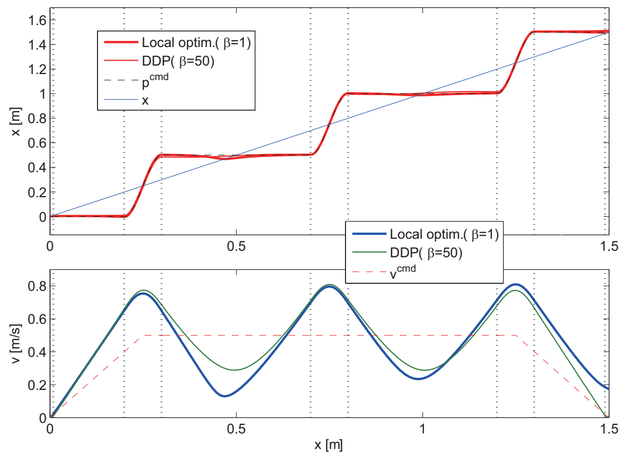

Fig. 4. ZMP and speed obtained by optimization

In this example, we specified the commanded ZMP p^{cmd} for three 50cm steps (broken line in the upper graph of Fig.4). For the commanded speed v^{cmd}, a trapezoidal pattern was specified so that it increases from the lowest speed ϵ to 0.5m/s within a half-step, keeps constant in the following two steps, and decreases to ϵ at the final half-step (broken line in the lower graph of Fig.4). We used the weight $\beta = 1$ for this example. The ZMP obtained by the local optimization is plotted as a bold red line which follows well the commanded ZMP. The speed obtained by the local optimization is plotted by a bold blue line in the bottom of Fig.4, which waves about the reference speed (broken line). This waving speed pattern is typically observed in dynamic biped walking. At the end of walking ($x = 1.5$m), however, the speed still remains 0.2m/s meaning the robot does not stop. This occurs because (7) considers only local state information.

A feasible walking pattern can be obtained by an optimization procedure, considering a global cost function as follows

$$\begin{aligned} \underset{p_k}{\text{minimize}} \quad & J_G := \sum_{k=0}^{N}(v_k - v_k^{cmd})^2 + \beta(p_k - p_k^{cmd})^2 \\ \text{subject to} \quad & v_{k+1} = v_k + \omega^2(x_k - p_k)\frac{\Delta x}{v_k}, \\ & \|v_k\| > \epsilon. \end{aligned} \tag{9}$$

The cost function J_G is a sum of the local costs (7), from start to stop of the walking pattern.

To solve this non-linear optimization problem, we used Differential Dynamic Programming (DDP) by Tassa, Mansard and Todorov [21]. The result is shown by thin red line (ZMP) and thin green line (speed) of Fig.4. We can observe a smooth speed pattern which gets stopped at the end of the walking.

We decided to create an offline walking pattern using DDP. Our walking pattern consists of the reference ZMP, the reference speed, and the kinematic walking pattern (2) as

$$\{p_k^*,\ v_k^*,\ \boldsymbol{q}_k^*\}\ (k = 0 \cdots N), \tag{10}$$

where p_k^* and v_k^* are the ZMP and the speed generated by DDP optimization, respectively.

Let us call this data set a "Spatial Walking Pattern." Our robot uses this spatial walking pattern prepared in advance and uses local optimization (8) to accommodate the pattern to the physical robot state in real-time. This is explained in the next section.

D. Space to time conversion algorithm

For our robot HRP-2Kai, we used a constant control cycle of $T = 2$ms, such that target joint angles must be calculated at every cycle [15]. Contrarily, the SQD time step Δt changes by the walking speed. At low speed, we have $T < \Delta t$ where several time steps T passes within one Δt. At high speed, we have $\Delta t < T$ where multiple SQD steps passes within one control cycle T. To handle this non-trivial problem, we developed an algorithm which is presented by the pseudo code of Algorithm 1.

Algorithm 1: SQD gait control

```
Data:
    current time $t_{now}$
    robot state $v^{sens}$, $p^{sens}$
    spatial pattern $v_k^*$, $p_k^*$, $\boldsymbol{q}_k^*$ $(k = 0 \cdots N)$
Result: $\boldsymbol{q}$
1  if first time then
2      $k = 0$;
3      $t_0 = t_{now}$;
4      $v_0 = \epsilon$;
5  end
6  while true do
7      $x_k = \Delta x \cdot k$;
8      $v^{cmd} = v_k^* + K_v(v^{sens} - v_k^*)$;
9      $p^{cmd} = p_k^* + K_{zmp}(p^{sens} - p_k^*)$;
10     $A = -(\Delta x / v_k)\omega^2$;
11     $B = (\Delta x / v_k)\omega^2 x_k + v_k$;
12     $p_k = \frac{\beta p^{cmd} - A(B - v^{cmd})}{\beta + A^2}$;
13     $a = \omega^2(x_k - p_k)$;
14     $\Delta t = \Delta x / v_k$;
15     $v_{k+1} = \max(v_k + a \cdot \Delta t, \epsilon)$;
16     $t_{k+1} = t_k + \Delta t$;
17     if $t_{k+1} \geq t_{now}$ then
18         break;
19     else
20         $k = \min(k+1, N)$ ;
21     end
22 end
23 $h = (t_{now} - t_k)/\Delta t$;
24 $\boldsymbol{q} = (1-h)\boldsymbol{q}_k^* + h\boldsymbol{q}_{k+1}^*$;
```

This algorithm is executed once for every control cycle T. Therefore, the timer variable t_{now} gets incremented by T in every cycle. The algorithm takes t_{now} as its input, as well as the measured robot state v^{sens} (actual robot speed), p^{sens} (actual ZMP), and the spatial walking pattern. Its output is a vector of the joint angles $\boldsymbol{q}$ for the current cycle.

In this algorithm, lines 1 through 5 initialize the parameters only at the very first execution. For example, line 2 zeros the index k to point at the beginning of the spatial walking pattern.

Lines 6 through 22 are the essential part to update SQD. Line 7 calculates the reference CoM position form the index k. Lines 8 and 9 generate the command values for the local optimization from the measured robot state.

$$v^{cmd} = v_k^* + K_v(v^{sens} - v_k^*) \quad (11)$$
$$p^{cmd} = p_k^* + K_{zmp}(p^{sens} - p_k^*) \quad (12)$$

where K_v, K_{zmp} are feedback gains for the speed and the ZMP. Depending on these gains, the real robot state v^{sens}, p^{sens} can affect the SQD dynamics by changing the instantaneous command value for the local optimization. For example, a large K_{zmp} induces a speed change depending on the ZMP error, which effectively reduces the foot landing vibration.

Lines 10 through 12 calculate the optimal ZMP by (8). Following lines 13 and 14 correspond to the SQD update using it. Line 15 uses a 'max' function to prevent that the updated speed falls below the minimum speed ϵ. The next node time is obtained in the next line 16. Line 17 checks if the next node time is at the future of t_{now} or not, and if yes, it quits the loop at line 18 and jumps to line 23. If the node time is still in the past, the algorithm increases the index k and goes back to line 7 to repeat the SQD calculation.

Line 23 calculates the parameter h which indicates the relationship between t_{now} and the latest node time by

$$h = (t_{now} - t_k)/\Delta t. \quad (13)$$

It is guaranteed that $0 < h \le 1$ since following relationship is satisfied by lines 17 through 21:

$$t_k < t_{now} \le t_k + \Delta t = t_{k+1} \quad (14)$$

Finally, line 24 generates the current joint angles by interpolating the adjacent node data of the spatial walking pattern.

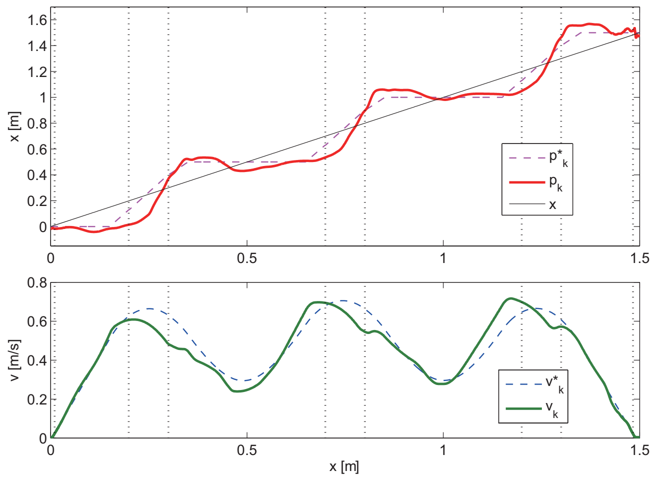

Fig. 5. ZMP and speed by SQD optimization $K_{zmp} = -1.5, K_v = -0.03$

Figure 5 shows the ZMP and the speed obtained by this algorithm from the experiment of Fig.1. The broken line in the upper graph is the ZMP of the spatial walking pattern and the bold line is the actual ZMP. The broken line in the lower graph is the speed v^* and the bold line is the actual speed. We can observe a large deviation of the walking speed and the ZMP from the walking pattern, whereas the errors vanish at the end of the walking.

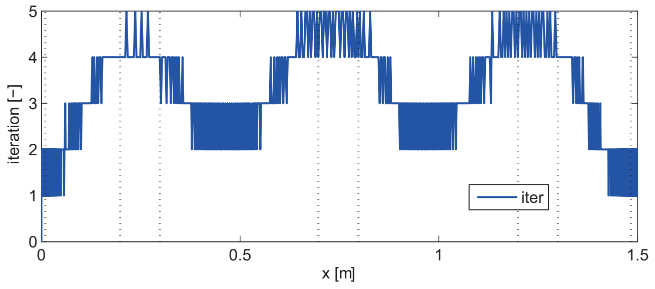

Fig. 6. SQD iteration

Figure 6 indicates the number of iterations of line 6 through 22 of Algorithm 1 executed in the walking control. Notice that the number of the SQD iterations is once or twice at the slow walk speed around $x = 0$m, and $x = 1.5$m, whereas it increases to five times at the high walk speed around $x = 0.25$m, $x = 0.75$m, and $x = 1.25$m. Hence our method has a drawback which requires heavier calculation at higher walking speed. Nevertheless, it does not become a serious problem since each iteration is very light as shown in Algorithm 1.

E. Control result in the sagittal plane

Using the algorithm of the former subsection, our humanoid robot HRP-2Kai could successfully realize a walk with fully stretched knees, heel contact, and toe off.

Figure 7 shows the experimental data along the sagittal plane. It shows the CoM (bold black line), the ZMP (bold red line), and the support polygon (shaded area) in vertical dotted lines indicating the touchdown and liftoff. We can confirm that the ZMP is kept well inside of the support polygon, while the CoM sometimes goes outside of it, which indicates a dynamic walking with a certain degree of stability margin.

III. Lateral Control

A. Previewing lateral ZMP

In our control algorithm, each gait period varies due to the SQD optimization which is affected by the CoM speed and the ZMP of the controlled robot. To adapt this variable gait cycle, it is necessary to generate a dynamically consistent lateral CoM motion in real-time.

For this purpose, the lateral ZMP data p_y^* were added to the spatial walking pattern as shown in Fig.8 taking the sagittal CoM x as a parameter. From this information, our controller generates a *Previewed ZMP* which is used to generate the lateral CoM motion, as explained later. The pseudo code to

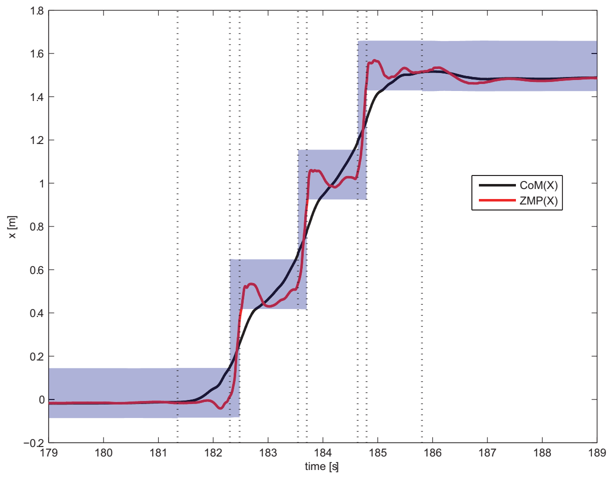

Fig. 7. CoM and ZMP in the walking direction

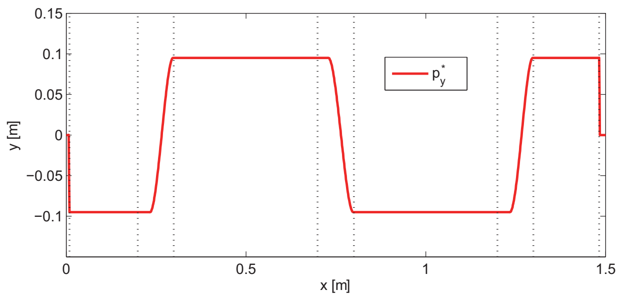

Fig. 8. Lateral ZMP of spatial walking pattern

calculate the previewed ZMP in every control cycle T is shown in Algorithm 2.

```
Algorithm 2: Generate previewed ZMP
  Data:
      current time $t_{now}$
      current node $k$
      preview horizon $T^{prev}$
      spatial pattern $v_k^*,\ p_{x,k}^*,\ p_{y,k}^*\ (k = 0 \cdots N)$
  Result: $p_y^{prev}$
1 $s = k$;
2 while $t_k < t_{now} + T^{prev}$ do
3     $x_s = \Delta x \cdot s$;
4     $A = -(\Delta x / v_s)\omega^2$;
5     $B = (\Delta x / v_s)\omega^2 x_s + v_s$;
6     $p_s = \frac{\beta p_{x,s}^* - A(B - v_s^*)}{\beta + A^2}$;
7     $a = \omega^2 (x_s - p_s)$;
8     $\Delta t = \Delta x / v_s$;
9     $v_{s+1} = \max(v_s + a \cdot \Delta t, \epsilon)$;
10    $t_{s+1} = t_s + \Delta t$;
11    $s = s + 1$;
12 end
13 $L = \lfloor T_{prev}/T \rfloor$;
14 $p_{y,k:k+L}^{prev} = \mathit{Interpolate}(t_{k:s}, p_{y,k:s}^*, T, L)$;
```

The algorithm takes the current time t_{now}, the current node number k, the preview horizon T^{prev}, and the spatial walking pattern as its input. Line 1 sets the current node number to s which will increase in the following loop. Line 2 through 12 repeatedly calculate the SQD dynamics towards the future until the node time reaches $t_{now} + T^{prev}$. At the end of this loop, we obtain vectors $p_{y,\cdot}$ and $t_{\cdot}$ from k to s. In line 14, these vectors of variable time step are converted into a vector of constant time step T. Note that we used the following notation to simplify the pseudo code,

$$\begin{aligned} p_{y,k:s} &:= [p_{y,k}, p_{y,k+1}, \cdots, p_{y,s}], \\ t_{k:s} &:= [t_k, t_{k+1}, \cdots, t_s], \\ p_{y,k:k+L}^{prev} &:= \left[p_{y,k}^{prev}, p_{y,k+1}^{prev}, \cdots, p_{y,k+L}^{prev}\right]. \end{aligned}$$

The function *Interpolate*() takes the vector of node time $t_{k:s}$, node ZMP $p_{y,k:s}$, unit time T, and the preview data length L, then returns a previewed ZMP vector $p_{y,k:k+L}^{prev}$.

Figure 9 shows the ZMP generated by this algorithm. We chose $T^{prev} = 1.6$s as the preview horizon, thus a vector of previewed ZMP for future in 1.6s is generated at every control cycle.

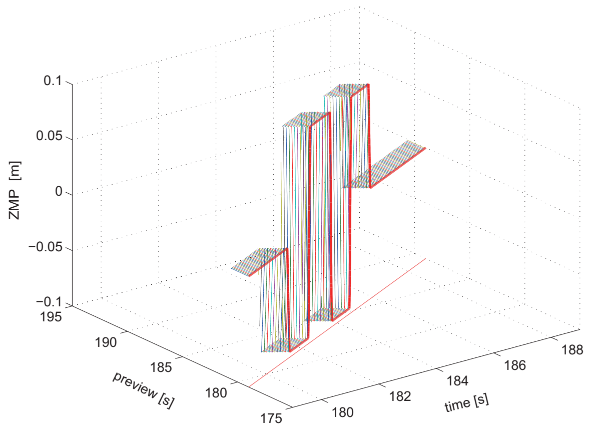

Fig. 9. Previewed ZMP p_y^{prev}

B. Lateral motion generation

Using the previewed ZMP, we calculate the corresponding CoM by the following steps. First, we define a state vector which consists of the lateral CoM's position, velocity, and acceleration.

$$\boldsymbol{y}_k \equiv [\ y(kT) \quad \dot{y}(kT) \quad \ddot{y}(kT)\ ]^T \tag{15}$$

Then we take the jerk (time derivative of acceleration) as its input.

$$u_k = \dddot{y}(kT) \tag{16}$$

Assuming a cart-table model [22], a discrete system dynamics which takes a jerk input and a ZMP output is represented as

$$\boldsymbol{y}_{k+1} = \boldsymbol{A}\boldsymbol{y}_k + \boldsymbol{b}u_k, \tag{17}$$

$$p_{y,k} = \boldsymbol{c}\boldsymbol{y}_k, \tag{18}$$

where

$$\begin{aligned} \boldsymbol{A} &\equiv \begin{bmatrix} 1 & T & T^2/2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix}, \quad \boldsymbol{b} \equiv \begin{bmatrix} T^3/6 \\ T^2/2 \\ T \end{bmatrix}, \\ \boldsymbol{c} &\equiv [\ 1 \quad 0 \quad -z/g\ ]. \end{aligned}$$

This dynamics generate the desired lateral CoM motion under the following control by using a previewed ZMP:

$$u_k = -K\boldsymbol{y}_k - K_s e_k^{sum} + \sum_{j=1}^{L} g_j p_{y,k+j}^{prev}, \tag{19}$$

$$e_{k+1}^{sum} = e_k^{sum} + p_{y,k}^{prev} - p_{y,k}, \tag{20}$$

where K, K_s, and g_j are the state feedback gain, the gains for the ZMP error, and the preview gain. The ZMP error is accumulated on e_k^{sum} by (20).

We can calculate the control gains K, K_s, and g_j by using the LQR framework. Its details are explained in our former works [22], [23].

C. Experimental result

Figure 10 shows the generated lateral CoM and ZMP during the experiment of Fig.1. It plots the CoM (black bold line) and the ZMP (red bold line) showing stable lateral walking motion. This lateral CoM motion is merged with the sagittal walking pattern generated in section II-D.

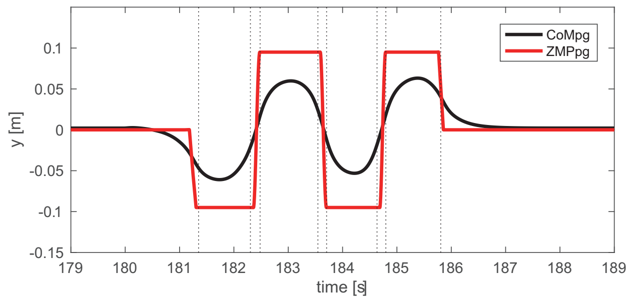

Fig. 10. CoM and ZMP in the lateral direction

Figure 11 shows the vertical ground reaction force during the experiment. We can recognize a regular support force exchange except for the vibration after every touch down. This vibration occurred because of the touchdown impact caused by the body up-down motion as a result of the fully stretched knee support.

The lateral CoM and ZMP during the three steps are shown in Fig.12. We can observe the ZMP trajectory shown in red bold line moving in the support polygon (blue gray area) with sufficient safety margin. The CoM trajectory was also well controlled contributing the steady and successful leg exchanges.

Figure 13 shows the snapthots taken from the video of the walking experiment. The lateral motion described in this section was marged with the sagittal pattern (Fig.2) to generate 3D motion. Nevertheless, the sagittal pattern was well preserved, and we can observe a biped gaint with fully stretched legs as expected.

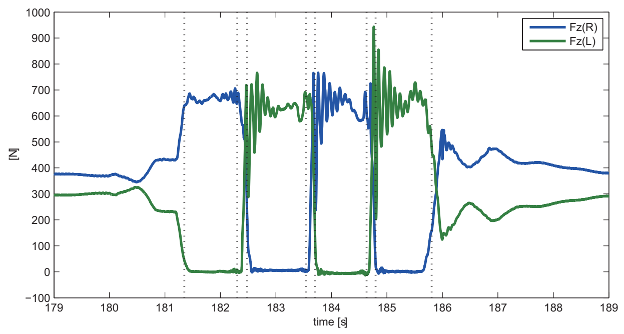

Fig. 11. Vertical ground reaction force of the experiment

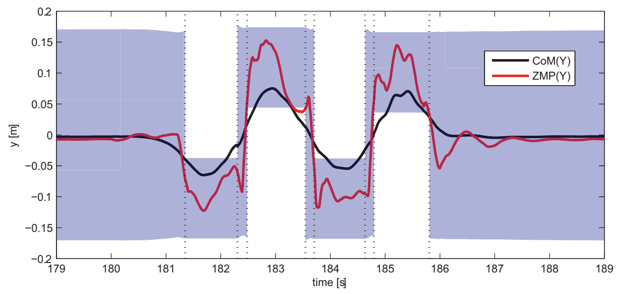

Fig. 12. Lateral CoM and ZMP of the experiment

To realize this stable walking, we have also developed a stabilizer which can overcome the mechanical singularity at double support with fully stretched knees. It will be reported in our next paper.

IV. CONCLUSIONS

We have implemented a gait generator from the spatial walking pattern, the lateral pattern generator, and the ZMP based stabilizer for the onboard computer of HRP-2Kai (Intel Core i7, 3.1GHz), and it stably ran at every 2ms.

There exist, at least two issues to be answered. Firstly, it is still unclear what is the real benefit of SQD compared with conventional methods. One possible answer is that a parameter to represent a task progress should be taken from *the task space* rather than the uniform time. As the classical example, one should recall the solving process of the brachistochrone curve, anyway, we need further consideration.

Secondly, the feedback control based on local optimization in Section II.C and D lacks solid theoretical background. Although its effectiveness is empirically shown, the feedback gains are hand tuned by simulations and experiments. We must appropriately associate our SQD based feedback to the conventional control theory.

Despite of the above problems, the essence of SQD approach is simple and it actually works. We believe that our approach will rewrite the future of the biped robot control.

REFERENCES

[1] M.Vukobratović and J.Stepanenko, "On the Stability of Anthropomorphic Systems," *Mathematical Biosciences*, vol. 15, pp. 1–37, 1972.

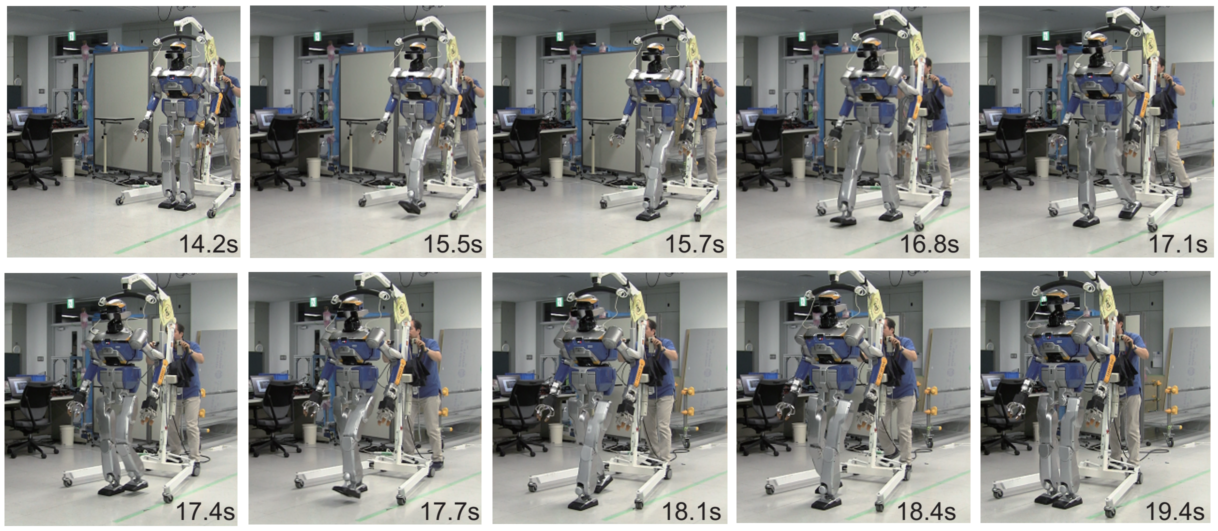

Fig. 13. Snapthots from the experiment

[2] M. Morisawa, S. Kajita, K. Kaneko, K. Harada, F. Kanehiro, K. Fujiwara, and H. Hirukawa, "Pattern Generation of Biped Walking Constrained on Parametric Surface," in *Proceedings of IEEE International Conference on Robotics and Automation*, 2005, pp. 2416–2421.

[3] Z. Li, N.G. Tsagarikis, D.G. Caldwell, and B. Vanderborght, "Trajectory generation of straightened knee walking for humanoid robot iCub," in *Proceedings of International Conference on Control, Automation, Robotics and Vision*, 2010, pp. 2355–2360.

[4] R. Kurazume, S. Tanaka, M. Yamashita, T. Hasegawa, and K. Yoneda, "Straight Legged Walking of a Biped Robot," in *Proceedings of IEEE International Conference on Intelligent Robots and Systems*, 2005, pp. 3695–3191.

[5] A. Sekiguchi, Y. Atobe, K. Kameta, Y. Tsumaki, and D.N. Nenchev, "A Walking Pattern Generator around Singularity," in *Proceedings of IEEE-RAS International Conference on Humanoid Robots*, 2007, pp. 270–275.

[6] M-S. Kim, I. Kim, S. Park, and J.H. Oh, "Realization of Stretch-legged Walking of the Humanod Robot," in *Proceedings of IEEE-RAS International Conference on Humanoid Robots*, 2005, pp. 118–124.

[7] Y.Ogura, K.Shimomura, H.Kondo, A.Morishima, T.Okubo, S.Momoki, H.Lim, and A.Takanishi, "Human-like walking with knee stretched, heel-contact and toe-off motion by a humanoid robot," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, pp. 3976–3981.

[8] K.Miura, M.Morisawa, F.Kanehiro, S.Kajita, K.Kaneko, and K.Yokoi, in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 4428–4435.

[9] P.-B. Wieber, "Trajectory free linear model predictive control for stable walking in the presence of strong perturbations," in *Proceedings of IEEE-RAS International Conference on Humanoid Robots*, 2006, pp. 137–142.

[10] S.Feng, X.Xinjilefu, W.Huang, and C.G.Atkeson, "3D Walking Based on Online Optimization," in *Proceedings of IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2013, pp. 21–27.

[11] S. Kuindersma, R. Deits, M. Fallon, A. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake, "Optimization-based locootion planning, estimation, and control design for the atlas humanoid robot," *Autonomous Robots*, vol. 40, no. 3, pp. 429–455, 2016.

[12] J. Carpentier, S. Tonneau, M. Naveau, O. Stasse, and N. Mansard, "A Versatile and Efficient Pattern Generator for Generalized Legged Locomotion," in *Proceedings of IEEE International Conference on Robotics and Automation*, 2016, pp. 3555–3561.

[13] R. Griffin, S. Bertrand, G. Wiedebach, A. Leonessa, and J. Pratt, "Capture Point Trajectories for Reduced Knee Bend using Step Time Optimization," in *Proceedings of IEEE-RAS International Conference on Humanoid Robotics*, 2017, pp. 25–30.

[14] S.Kajita, M.Benallegue, R.Cisneros, T.Sakaguchi, S.Nakaoka, M.Morisawa, K.Kaneko, and F.Kanehiro, "Biped Walking Pattern Generation based on Spatially Quantized Dynamics," in *Proceedings of IEEE International Conference on Humanoid Robots*, 2017, pp. 599–605.

[15] K. Kaneko, M. Morisawa, S. Kajita, S. Nakaoka, T. Sakaguchi, R. Cisneros, and F. Kanehiro, "Humanoid Robot HRP-2Kai –Improvement of HRP-2 towards Disaster Response Tasks –," in *Proceedings of IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, November 2015, pp. 132–139.

[16] E.R. Westervelt, J.W. Grizzle, and D.E. Koditschek, "Hybrid Zero Dynamics of Planar Biped Walkers," *IEEE Transactions on Automatic Control*, vol. 48, no. 1, pp. 42–56, January 2003.

[17] E.R.Westervelt, J.W.Grizzle, C.Chevallereau, J.H.Choi, and B.Morris, *Feedback Control of Dynamic Bipedal Robot Locomoation*. CRC Press, 2007.

[18] M. Doi, Y. Hasegawa, and T. Fukuda, "Passive Dynamic Autonomous Control of Bipedal Walking," in *Proceedings of IEEE/RSJ International Conference on Humanoid Robots*, 2004, pp. 811–829.

[19] S.Kajita, O.Matsumoto, and M.Saigo, "Real-time 3d walking pattern generation for a biped robot with telescopic legs," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2001, pp. 2299–2306.

[20] T.Sugihara, Y.Nakamura, and H.Inoue, "Realtime humanoid motion generation through zmp manipulation based on inverted pendulum control," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2002, pp. 1404–1409.

[21] Y.Tassa, N.Mansard, and E.Todorov, "Control-Limited Differential Dynamic Programming," in *Proceedings of IEEE International Conference on Robotics and Automation*, May 2014.

[22] S.Kajita, F.Kanehiro, K.Fujiwara, K.Harada, K.Yokoi, and H.Hirukawa, "Biped Walking Pattern Generation by using Preview Control of Zero-Moment Point," in *Proceedings of IEEE International Conference on Robotics and Automation(ICRA)*, 2003, pp. 1620–1626.

[23] S.Kajita, H.Hirukawa, K.Harada, and K.Yokoi, *Introduction to humanoid robotics*. Springer, 2014.