Proceedings of the 2005 IEEE
International Conference on Robotics and Automation
Barcelona, Spain, April 2005

# A Running Controller of Humanoid Biped HRP-2LR

Shuuji Kajita*, Takashi Nagasaki†, Kenji Kaneko*, Kazuhito Yokoi* and Kazuo Tanie*†
*National Institute of Advanced Industrial Science and Technology(AIST)
Tsukuba, Ibaraki 305-8568, Japan, Email: s.kajita@aist.go.jp
† University of Tsukuba, Email: t-nagasaki@aist.go.jp

***Abstract*— This article explains a control system, which stabilizes running biped robot HRP-2LR. The robot uses prescribed running pattern calculated by resolved momentum control, and a running controller stabilizes the system against disturbances. The running controller consists of posture stabilization, inverted pendulum stabilization, contact torque control, impact absorbing control, foot vertical force control and torque distribution control. Applying the proposed controller, HRP-2LR could successfully run with average speed of 0.16(m/s) repeating flight phase of 0.06 (s) and support phase of 0.3 (s).**

***Index Terms*— Biped locomotion, Humanoid, Running, Jogging, Stabilization**

## I. Introduction

Research on humanoid robots is currently one of the most exciting topics in the field of robotics and there exist many projects [1]–[6]. Most of them focus on biped walking as an important subject and have already demonstrated reliable dynamic biped walking. Watching those successful demonstrations, one can ask a natural question, "Can we build a humanoid that can run?"

We believe this is worthwhile as a technical challenge for the following reasons. First, studying robot running will add new functions of mobility to humanoid robots. For example, jumping over large obstacles or a crevasse in the ground might be realized by a derivative of running control. Second, studying extreme situations will give us insights to improve the hardware itself. Current robots are too fragile to operate in any environment. Even when the robot operates at low speed, we must treat them carefully. We hope to overcome this fragility in the process of developing a running humanoid.

Running robots have been intensively studied by Raibert and his colleagues [7]. Their famous hopping robots driven by pneumatic and hydraulic actuators performed various actions including somersaults [8]. Using a similar control strategy, Hodgins simulated a running human in the computer graphics [9]. Ahmadi and Buehler studied running monopods from a standpoint of energy efficiency [10].

All of those robots have a spring mechanism to retrieve kinetic energy during running cycles. It is obvious that these springs help running but they might prevent ordinary humanoid activities such as walking, carrying objects and so on. Since our intention is to add a running function to a versatile humanoid robot, we started with a mechanism without springs. A similar approach is taken by Gienger et al. [1], Nagasaka et al. [11] and Chevallereau et al. [12]. [1]

In this paper, we explain our running controller for humanoid biped HRP-2LR. In Section II, the hardware of HRP-2LR is briefly explained. Section III outlines the running controller, and its details is explained in Section IV and Section V. The experimental result is shown in Section VI. We conclude this paper and address our future plans in Section VII.

## II. Humanoid biped HRP-2LR

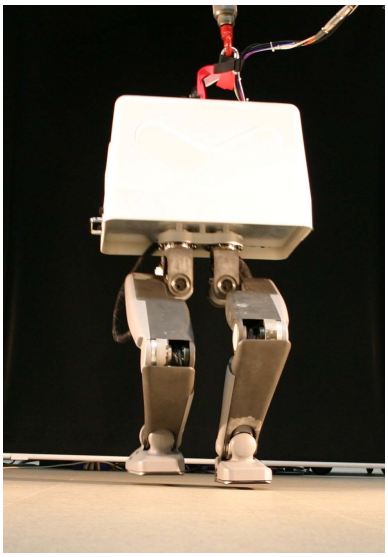

Fig. 1. HRP-2LR running

Fig.1 shows running HRP-2LR, which is a 12 DOF biped robot whose legs have humanoid configuration. Its total weight is 31.0 (kg) and the height is 1.27 (m). The body contains a 3-axes acceleration sensor, three gyro sensors, twelve servo drivers and a CPU board (Pentium III, 933 (MHz)). Each foot is equipped with a 6-axes force sensor and rubber bushing which protects the sensor and robot from the touchdown impact. For detailed specifications of HRP-2LR, see [14].

[1]On December 15, 2004, Honda announced that they developed the next-generation ASIMO which can run at 3[km/h] [13]. So far, its technical details are not yet disclosed and we cannot tell whether ASIMO is equipped with spring mechanisms or not.

0-7803-8914-X/05/$20.00 ©2005 IEEE.

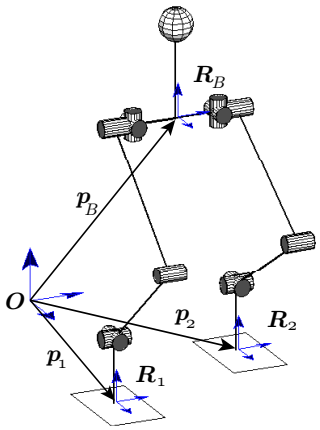


Fig. 2. Reference frames for running pattern generation

TABLE I

SPECIFICATIONS OF HRP-2LR

| | | |
|---|---|---|
| Size | 6D.O.F/Leg(Hip:3 Knee:1 Ankle:2) | |
| | Upper leg length: | 300 [mm] |
| | Lower leg length: | 300 [mm] |
| | Ankle-sole height: | 93 [mm] |
| | Length between hip joints: | 120 [mm] |
| | Toe-heel length: | 170 [mm] |
| Weight | Legs:8.6 [kg/leg]×2 [legs] = | 17.2 (kg) |
| | Controller: | 7.0 (kg) |
| | Body structure | 6.8 (kg) |
| | Total: | 31.0 (kg) |

Fig.2 and Table I illustrates the basic configuration of HRP-2LR. The body translation is represented by a 3D vector $\boldsymbol{p}_B$, which indicates a midpoint of the hip joints with respect to the ground fixed origin $O$. The body posture is represented by a $3 \times 3$ matrix $\boldsymbol{R}_B$. Also, $\boldsymbol{p}_i, \boldsymbol{R}_i (i = 1, 2)$ are defined for the position and orientation of the right $(i = 1)$ and left $(i = 2)$ feet.

## III. RUNNING CONTROLLER

### A. Outline

Fig.3 illustrates the entire running control system including the pattern generation. The designed total linear momentum $\mathcal{P}^d$, total angular momentum $\mathcal{L}^d$ and the foot motion $\boldsymbol{v}_i^d, \boldsymbol{\omega}_i^d \ \ (i = 1, 2)$ are processed by Resolved Momentum Control (RMC) [16] and we obtain a running pattern consists of the following parameters.

| | |
|---|---|
| $\boldsymbol{q}^d$ | Joint angles $\boldsymbol{q}^d \equiv [\boldsymbol{q}_{leg_1}^T \boldsymbol{q}_{leg_2}^T]^T$ |
| $\phi^d, \theta^d, \psi^d$ | Body posture ($\boldsymbol{R}_B^d$ in roll-pitch-yaw) |
| $\boldsymbol{\omega}_B^d$ | Body angular velocity |
| $\boldsymbol{f}_i^d, \boldsymbol{\tau}_i^d$ | Foot force and moment |
| *phase* | Running phase (support leg) (Right,Left,Double,Flight) |

Although RMC calculates the absolute body position $\boldsymbol{p}_B^d$, HRP-2LR cannot sense the corresponding data due to the lack of absolute position sensors (ex. GPS). Therefore $\boldsymbol{p}_B^d$ is not sent to the running controller, but to a "trash-can".

The running patterns are fed to *running controller* where the target joint angles $\boldsymbol{q}^d$ are modified so that the robot can continue running under disturbances. For this purpose, the running controller uses foot force sensor information and body posture estimated by Kalman filter which handles gyros and acceleration sensors.

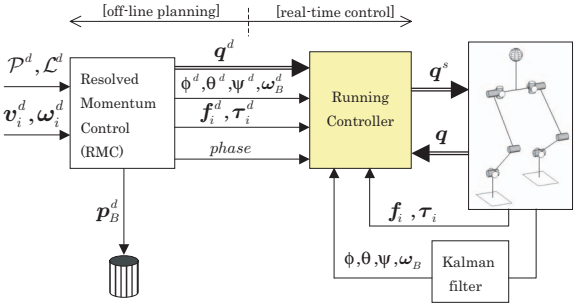


Fig. 3. Outline of HRP-2LR control system

### B. Model-based framework

For each servo cycle, the running controller calculates the reference feet configuration by forward kinematics.

$$[\boldsymbol{p}_1^d, \boldsymbol{R}_1^d, \boldsymbol{p}_2^d, \boldsymbol{R}_2^d] = FK(\boldsymbol{p}_B^{in}, \boldsymbol{R}_B^d, \boldsymbol{q}^d), \quad \boldsymbol{R}_B^d \equiv \boldsymbol{R}_{rpy}(\phi^d, \theta^d, \psi^d) \tag{1}$$

where $FK()$ is a forward kinematics for both feet, $\boldsymbol{p}_B^{in}$ is the nominal body position, $\boldsymbol{R}_{rpy}(\cdot)$ calculates a rotation matrix corresponding to given roll-pitch-yaw angles.

Since HRP-2LR cannot sense its absolute body position, we simply set the nominal body position $\boldsymbol{p}_B^{in}$ to be the origin.

$$\boldsymbol{p}_B^{in} = \boldsymbol{O}$$

The same procedure calculates the current (real) feet configuration.

$$[\boldsymbol{p}_1, \boldsymbol{R}_1, \boldsymbol{p}_2, \boldsymbol{R}_2] = FK(\boldsymbol{p}_B^{in}, \boldsymbol{R}_B, \boldsymbol{q}), \quad \boldsymbol{R}_B \equiv \boldsymbol{R}_{rpy}(\phi, \theta, \psi), \tag{2}$$

where $\phi, \theta, \psi$ are the current body posture (roll, pitch, yaw) of the robot estimated by the Kalman filter.

The *stabilizing algorithms* uses these feet configurations and calculate a new configuration for servo control (see Fig.4).

Finally, it is transformed into joint angles using inverse kinematics and is sent to the local PD servo controller of HRP-2LR.

$$\boldsymbol{q}^s = IK(\boldsymbol{p}_B^s, \boldsymbol{R}_B^s, \boldsymbol{p}_1^s, \boldsymbol{R}_1^s, \boldsymbol{p}_2^s, \boldsymbol{R}_2^s) \tag{3}$$

where $\boldsymbol{q}^s$ is a servo reference, $IK()$ is an inverse kinematics of HRP-2LR.

This approach should be called model-based framework, since it intensively uses a robot model for forward/inverse kinematics. By this way, we can build a versatile algorithms which are potentially applicable to different types of biped robot.

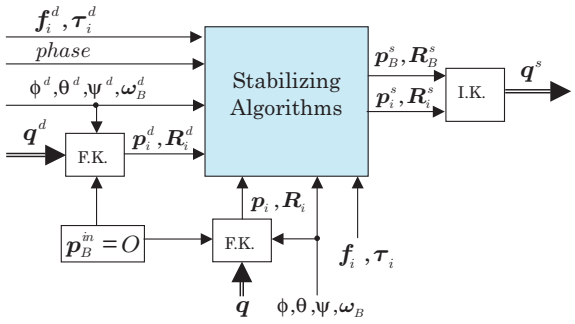


Fig. 4. Architecture of the running controller. F.K.:Forward Kinematics, I.K.:Inverse Kinematics

### C. Objective of stabilizing algorithms

Our goal is to realize

$$\boldsymbol{R}_B = \boldsymbol{R}_B^d \tag{4}$$
$$(\boldsymbol{p}_i, \boldsymbol{R}_i) = (\boldsymbol{p}_i^d, \boldsymbol{R}_i^d) \qquad (i = 1, 2). \tag{5}$$

These conditions must be realized under the following restrictions on foot torque and force.

$$\begin{cases} -x_{toe} f_{i,z} < \tau_{i,y} < -x_{heel} f_{i,z} \\ y_{i,e2} f_{i,z} < \tau_{i,x} < y_{i,e1} f_{i,z}, \end{cases} \tag{6}$$

where $\boldsymbol{f}_i \equiv [f_{i,x}\ f_{i,y}\ f_{i,z}]^T$ and $\boldsymbol{\tau}_i \equiv [\tau_{i,x}\ \tau_{i,y}\ \tau_{i,z}]^T$. $x_{toe}, x_{heel}$ denotes foot length and $y_{i,e1}, y_{i,e2}$ denotes foot width. These inequalities are also known as ZMP conditions which guarantee that the foot does not rotate on the ground.

In addition, there exist another restrictions to guarantee the feet will not slip.

$$\begin{cases} |f_{i,x}|, |f_{i,y}| < \mu f_{i,z} \\ \tau_{i,z} < \mu_z f_{i,z} \end{cases} \tag{7}$$

where $\mu, \mu_z$ are the friction coefficients.

Under large disturbance like a touchdown impact, we must modify the reference trajectory to satisfy the restrictions of eq.(6) and (7). For this purpose, we designed stabilizing algorithms which modify the desired configurations of the body and the feet.

$$(\boldsymbol{p}_B^{in}, \boldsymbol{R}_B^d) \rightarrow (\boldsymbol{p}_B^s, \boldsymbol{R}_B^s) \tag{8}$$
$$(\boldsymbol{p}_i^d, \boldsymbol{R}_i^d) \rightarrow (\boldsymbol{p}_i^s, \boldsymbol{R}_i^s) \qquad (i = 1, 2) \tag{9}$$

The stabilizing algorithms determine these modification. When the disturbances become small, these modification are smoothly canceled towards the original running patten. We designed the algorithms in an ad hoc manner with basic assumption where the robot can be treated as a simple inverted pendulum.

Another stabilizing algorithm using feedback linearization is proposed by Gienger, Löffler and Pfeiffer [1]. The main difference is they modeled a running robot as a torque controlled system while we model it as a position controlled system.

## IV. Stabilizing algorithm I :Geometric part

In this section, we explain the first part of our stabilizing algorithm, which takes care of the robot geometry. By the calculation of (1) and (2), now we have internal representations for a desired configuration and the actual configuration (Fig.5).

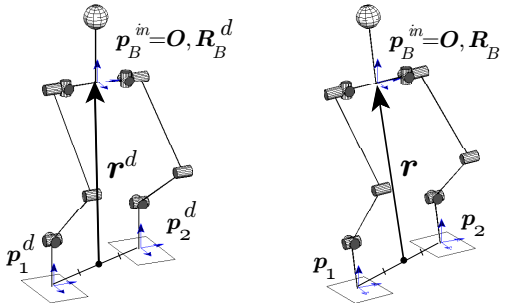


Fig. 5. Robot representation in the stabilizer. (Left) Reference (Right) Sensed state

To represent the inclination of the whole robot, let us define the vector $\boldsymbol{r}$ which directs the body position from the midpoint of the both feet.

$$\boldsymbol{r} \equiv \boldsymbol{p}_B^{in} - (\boldsymbol{p}_1 + \boldsymbol{p}_2)/2 \tag{10}$$
$$\boldsymbol{r}^d \equiv \boldsymbol{p}_B^{in} - (\boldsymbol{p}_1^d + \boldsymbol{p}_2^d)/2 \tag{11}$$

To let $\boldsymbol{r} \rightarrow \boldsymbol{r}^d$ and $\boldsymbol{R}_B \rightarrow \boldsymbol{R}_B^d$, following controllers are designed.

### A. Body posture control

Body posture controller calculates additional body rotation so that its roll and pitch inclination $(\phi, \theta)$ matches the specified value $(\phi^d, \theta^d)$ for the running.[2] The control law is as follows.

$$\Delta\ddot{\phi} = k_B(\phi^d - \phi) + d_B(\omega_{Bx}^d - \omega_{Bx}) \tag{12}$$
$$\Delta\ddot{\theta} = k_B(\theta^d - \theta) + d_B(\omega_{By}^d - \omega_{By}) \tag{13}$$

where $k_B, d_B$ are feedback gains, $\omega_{Bx}, \omega_{By}$ are angular velocities measured by gyros and $\omega_{Bx}^d, \omega_{By}^d$ are the reference angular velocities.

By integrating (12) and (13) twice, we obtain additional body rotation angle $\Delta\phi, \Delta\theta$, which determines reference body orientation.

$$\boldsymbol{R}_B^s = \Delta\boldsymbol{R}_B \boldsymbol{R}_B^d \tag{14}$$
$$\Delta\boldsymbol{R}_B \equiv \boldsymbol{R}_{rpy}(\Delta\phi, \Delta\theta, 0). \tag{15}$$

To let the body rotate around its center of mass (CoM), the origin of the base frame is also changed.

$$\boldsymbol{p}_B^s = \boldsymbol{p}_B^{in} + \boldsymbol{R}_B^d \bar{\boldsymbol{c}} - \boldsymbol{R}_B^s \bar{\boldsymbol{c}}, \tag{16}$$

where $\bar{\boldsymbol{c}}$ is the CoM of the body presented in the base frame.

The calculated body configuration $(\boldsymbol{p}_B^s, \boldsymbol{R}_B^s)$ is translated into joint angles by inverse kinematics (I.K.) of Fig.4.

[2]We neglect yaw orientation $\psi$ in our current implementation.

As the result, the robot adjusts its body posture by using all leg joints.

### B. Linear inverted pendulum control

To correct the inclination of the whole robot, let us define the error vector.

$$\begin{aligned} \Delta \boldsymbol{r} &= \boldsymbol{r} - \boldsymbol{r}^d \\ &\equiv [\Delta x,\ \Delta y,\ \Delta z] \end{aligned} \tag{17}$$

$$\tag{18}$$

We assume the error vector $\Delta \boldsymbol{r}$ behaves as a 3D linear inverted pendulum (3D-LIP) [17] while the robot is on the ground. Because, both of $\boldsymbol{r}$ and $\boldsymbol{r}^d$ approximately point the center of mass and the small error vector might be ruled by linear dynamics.

$$\Delta \ddot{x} = \frac{g}{z_c}\Delta x + \frac{1}{M z_c}\tau_y \tag{19}$$

$$\Delta \ddot{y} = \frac{g}{z_c}\Delta y - \frac{1}{M z_c}\tau_x \tag{20}$$

where $g$ is gravity acceleration, $z_c$ is the nominal height of the CoM and $M$ is the total mass. $\tau_y, \tau_x$ are the pitch/roll torque which act from the floor to the robot.

3D-LIP dynamics allows us to design a simple linear feedback law.

$$\bar{\tau}_y = -k_x \Delta x - d_x \Delta \dot{x}, \tag{21}$$

$$\bar{\tau}_x = k_y \Delta y + d_y \Delta \dot{y} \tag{22}$$

$\bar{\tau}_x, \bar{\tau}_y$ are sent to the torque control system described in the next section.

## V. Stabilizing algorithm II: Force/torque handling

The second part of the stabilizing algorithm realizes the forces and torques which were calculated in the last section and specified by the running pattern.

### A. 2DOF torque controller

Since each foot of HRP-2LR is equipped with rubber bush, it can generate force/torque proportional to its translation/rotation while the foot is contacting on the ground [2].

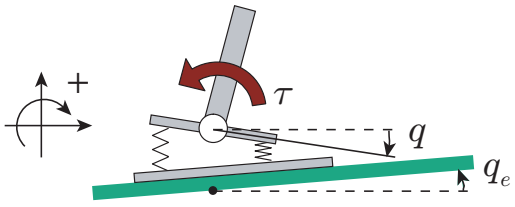


Fig. 6. Generation of floor reaction torque

Fig.6 illustrates the relationship between floor reaction torque and foot rotation. The floor reaction torque $\tau$ is generated as

$$\tau = -k_e(q - q_e), \tag{23}$$

where $k_e$ is the spring constant of the foot mechanism and rubber bush, $q$ is the amount of the foot rotation and $q_e$ is the rotation of ground. We assume $q_e$ to evaluate the backdrivability of the system.

To design a torque controller, we take an input as the speed $\omega \equiv \dot{q}$. The torque generation process in frequency domain is written as

$$\begin{aligned} \tau &= -\frac{k_e}{s}(\omega - \omega_e) \\ &\equiv P\,(\omega - \omega_e), \end{aligned}$$

where $\omega_e \equiv \dot{q}_e$ is the speed of the ground, and the process model is determined as $P \equiv -k_e/s$.

We designed a two degrees of freedom (2DOF) controller which has direct feedback of output filtered by $C_A$ as well as conventional feedback of error filtered by $C_B$ (see Fig.7).

$$\omega = -C_A\,\tau + C_B(\tau^d - \tau) \tag{24}$$

$C_A$ and $C_B$ are calculated as follows [18].

$$C_A = \frac{1}{P}\frac{Q}{1-Q} \tag{25}$$

$$C_B = \frac{G_r}{1-G_r}\frac{1}{P}\frac{1}{1-Q} \tag{26}$$

where $G_r$ is a desired transfer function from the target value $\tau^d$ to the output $\tau$, and $Q$ is a free parameter. With this choice the transfer function from the target to the output $G_{\tau^d \to \tau}$ becomes

$$G_{\tau^d \to \tau} = G_r.$$

In addition, the transfer function from the disturbance to the output $G_{\omega_e \to \tau}$ becomes

$$G_{\omega_e \to \tau} = (1-Q)(1-G_r)P.$$

Therefore, the free parameter $Q$ determines the system behavior against the disturbance, i.e. backdrivability.

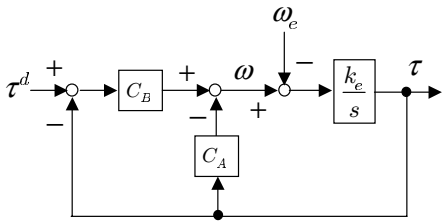


Fig. 7. 2DOF torque controller

The complete form of our torque controller is represented as

$$\begin{aligned} q &= \int -C_A\,\tau + C_B(\tau^d - \tau)dt \\ &\equiv TwoDof(\tau, \tau^d). \end{aligned} \tag{27}$$

The controller adjust a foot rotation $q$ such that measured floor reaction toqrue $\tau$ becomes specified value $\tau^d$.

In our experiments, the feedback filters $C_A, C_B$ are calculated by (25),(26) with following parameters.

$$G_r = Q = \frac{1}{0.04s+1}, \text{ (for foot torque control)}$$
$$G_r = Q = \frac{1}{0.07s+1}. \text{ (for vertical force distribution)}$$

### B. Foot torque control

To control the torque acting on each foot from the ground, we uses the 2DOF controller to get proper foot rotation angle $\Delta\phi_i, \Delta\theta_i$.

$$\Delta\phi_i = TwoDof(\tau_{ix}, \tau^s_{ix}) \quad (28)$$
$$\Delta\theta_i = TwoDof(\tau_{iy}, \tau^s_{iy}) \quad (29)$$

where $\tau_{ix}, \tau_{iy}$ are roll and pitch torque components measured by the foot torque sensors and $\tau^s_{ix}, \tau^s_{iy}$ are the corresponding references.

The calculated angles determines the reference feet orientations.

$$\boldsymbol{R}^s_i = \Delta\boldsymbol{R}_i\boldsymbol{R}^d_i \quad (30)$$
$$\Delta\boldsymbol{R}_i \equiv \boldsymbol{R}_{rpy}(\Delta\phi_i, \Delta\theta_i, 0) \quad (31)$$

### C. Foot vertical force control

To control vertical force from the ground, we uses two different types of controller. The first one is to absorb the huge impact force at the landing. When excessive vertical force was detected, it shorten the legs.

$$\Delta\dot{z}_a = \sigma(f_z)v_a + (1-\sigma(f_z))(-\omega_a\Delta z_a) \quad (32)$$
$$f_z \equiv f_{1z} + f_{2z},$$

where $\Delta z_a$ is the modification of foot height. When the total vertical force $f_z$ exceeds the threshold of 410(N), the first term of (32) becomes active and the feet is lifted with a speed of $v_a$ (shorten the legs). When the total vertical force becomes smaller than the threshold of 400(N), the second term returns $\Delta z_a$ to be zero with a time constant determined by $\omega_a$. The control mode is switched by the function $\sigma(f_z)$(Fig.8), which smoothly changes between 0 and 1 to avoid chattering. We determined a proper switching function which effectevely absorb impacts by simulations and experiments.

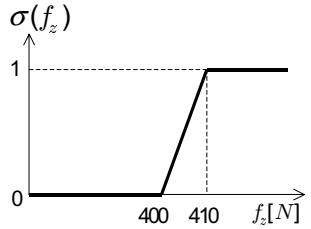


Fig. 8. Switching function for impact absorber

The second controller works when the robot is in double support phase and it adjusts the vertical force distribution between the right and the left feet. It modifies the difference of foot elevation $\Delta z_b$ by using 2DOF controller.

$$\Delta z_b = TwoDof(f_{2z} - f_{1z}, \Delta f^s_z), \quad (33)$$

where $f_{2z} - f_{1z}$ is the difference of vertical forces and $\Delta f^s_z$ is its reference value.

Finally, the foot elevation of each foot is determined as follows.

$$p^s_{1z} = p^d_{1z} + \Delta z_a + \Delta z_b/2 \quad (34)$$
$$p^s_{2z} = p^d_{2z} + \Delta z_a - \Delta z_b/2, \quad (35)$$

### D. Realization of LIPM stabilizing torque

The stabilizing torques for the 3D-LIPM in section IV-B is distributed depending on the phase specified by the running pattern.

*1) Single support phase:* When the running pattern specifies a single support phase, the 3D-LIPM stabilizing torque is simply fed to the torque controller of the supporting foot. Therefore, if the right foot is in support,

$$\tau^s_{1x} = \bar{\tau}_x, \ \tau^s_{1y} = \bar{\tau}_y \quad (36)$$
$$\tau^s_{2x} = \tau^s_{2y} = 0 \quad (37)$$

Similarly, if the left foot is in support,

$$\tau^s_{1x} = \tau^s_{1y} = 0 \quad (38)$$
$$\tau^s_{2x} = \bar{\tau}_x, \ \tau^s_{2y} = \bar{\tau}_y \quad (39)$$

In addition, the vertical force reference is specified for an unexpected contact of swing foot.

$$\Delta f^s_z = f^d_{2z} - f^d_{1z} \quad (40)$$

*2) Double support phase:* When the running pattern specifies a double support phase, we must carefully distribute the 3D-LIPM stabilizing torque to both feet.

First of all, the stabilizing torque $\bar{\tau}_x$ is effectively generated by the vertical force distribution (Fig. 9). At the same time, $\Delta f^s_z$ must be limited by the total vertical force to let both of the feet keep in contact with the ground.

$$\Delta f^s_z = trim(2\bar{\tau}_x/w + \Delta f^d_z, -f^d_z, f^d_z) \quad (41)$$
$$\Delta f^d_z \equiv f^d_{2z} - f^d_{1z}$$
$$f^d_z \equiv f^d_{1z} + f^d_{2z}$$

where $w$ is the distance between right and left feet, $trim()$ is the function to trim the first argument between the second and the third arguments

To properly distribute stabilizing torque to both feet, we used a parameter $\gamma$ defined as

$$\gamma \equiv (f^d_z - \Delta f^s_z)/2f^d_z,$$

where $\gamma \in [0,1]$ is 1 at right support and 0 at left support.

The foot torque references are determined as

$$\tau^s_{1x} = \gamma(\bar{\tau}_x - w(\Delta f^s_z - \Delta f^d_z)/2) \quad (42)$$
$$\tau^s_{1y} = \gamma\bar{\tau}_y, \quad (43)$$
$$\tau^s_{2x} = (1-\gamma)(\bar{\tau}_x - w(\Delta f^s_z - \Delta f^d_z)/2) \quad (44)$$
$$\tau^s_{2y} = (1-\gamma)\bar{\tau}_y. \quad (45)$$

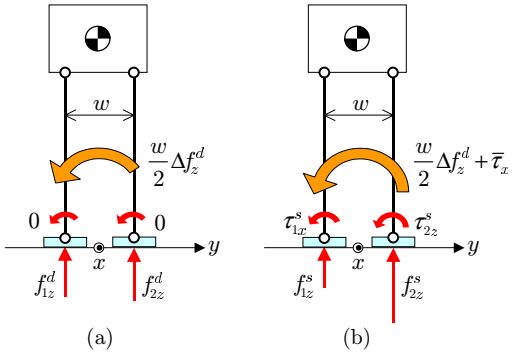


Fig. 9. Lateral forces and torques at double support (a) running pattern (b) modified by stabilizing algorithm

## VI. Running Experiment

For running experiment, we designed a pattern with support duration of 0.3(s), flight duration of 0.06(s), and speed of 0.25(m/s) (0.9(km/h)). The robot lost balance and turned over in the open loop running experiment without use of the proposed controller. By applying the running controller, HRP-2LR could successfully run. However the actual running speed of the robot was 0.16(m/s) (0.58(km/s)) due to slips between the robot's sole and the ground. Fig.10 shows a running experiment of HRP-2LR. We can recognize the flight phase at time T+0.36(s).

The upper graph of Fig.11 shows the errors in the body posture control described in section IV-A. The maximum absolute errors are 5(deg) in pitch (thin line) and 3(deg) in roll (bold line).

The middle graph of Fig.11 shows the errors in the linear inverted pendulum control described in section IV-B. We can observe the errors are 3(cm) at maximum in both directions. The controller specifies the recovery torques to fix it.

The bottom graph of Fig.11 shows the torques measured by the foot torque sensors.

The upper graph of Fig.12 shows the vertical reaction force in the running experiment. We can confirm the existence of flight phase where both of the vertical force becomes zero. Immediately after each flight phase, the robot suffers huge impacts of approximately 1000(N), which is more than three times of the robot's weight. The vertical dotted lines indicates the phase transitions specified by the running pattern. Comparing with the force data, we can see the actual running motion has a delay of 20 to 30 (ms) from the running pattern.

The lower graph of Fig.12 shows the modifiers of foot elevation for vertical force control described in section V-C. The thin line shows $\Delta z_a$ for impact absorbing which works every landing. The bold line shows $\Delta z_b$ for vertical force distribution which works at double support phases occurs in the beginning and the finishing of the run.

For comparison, let us see the data of stepping at a place stabilized by using our running controller (Fig.13). The upper graph shows that the vertical force is acting on each or both feet at all time. Since the robot did not *jump*, the

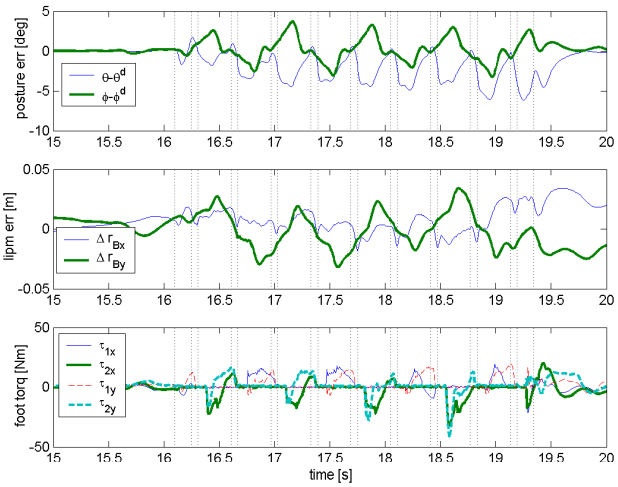


Fig. 11. Error in posture and horizontal displacement

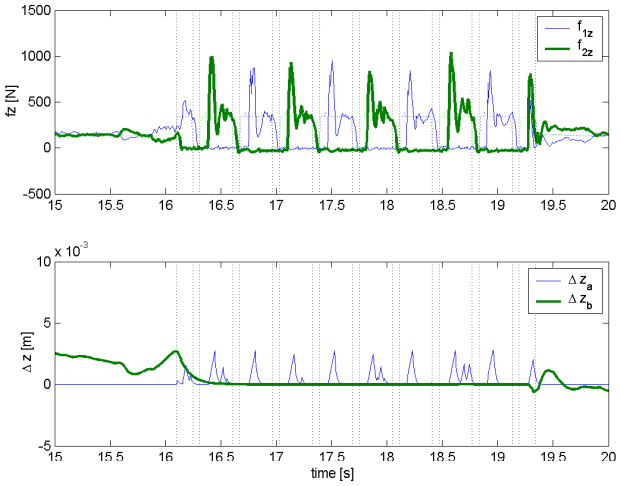


Fig. 12. Vertical force, foot updown control at running

maximum force is about 500 (N) that is half of the running. From the lower graph, we can observe the impact absorbing control $(\Delta z_a)$ works fewer times but the force distribution control $(\Delta z_b)$ works at every double support phase.

## VII. Conclusion

In this article, we explained a control system, which stabilizes running of a humanoid biped HRP-2LR. The running was realized by using prescribed pattern and stabilized by running controller. The running controller consists of body posture control, inverted pendulum stabilization, contact torque control, impact absorbing control, foot vertical force control and torque distribution control. Applying the proposed controller, HRP-2LR could successfully run with average speed of 0.16(m/s) repeating 0.06 (s) flights and 0.3 (s) supports.

The realization of faster running and on-line running pattern generation will be our future work.

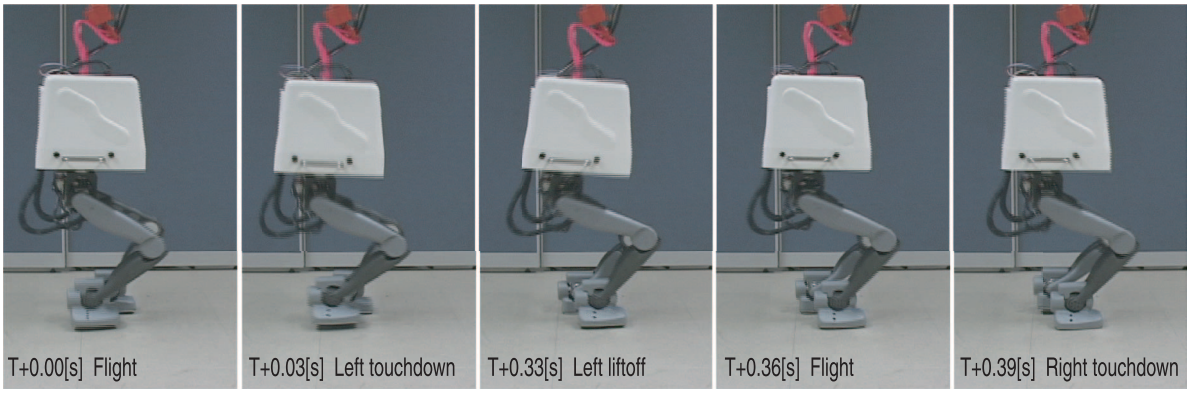


Fig. 10. Running experiment of HRP-2LR. The robot is running from left to right.

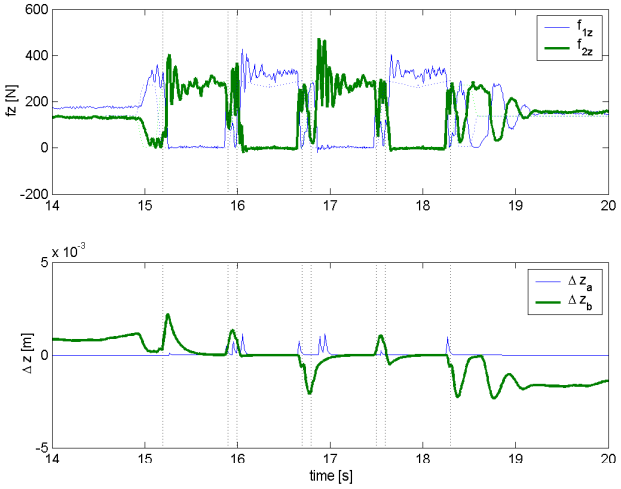


Fig. 13. Vertical force, foot up-down control at stepping

## ACKNOWLEDGMENT

We thank people of Kawada Industries, Inc. especially Jiro Sakurai, Toshikazu Kawasaki and Takakatsu Isozumi for their excellent technical support. We also thank Hirohisa Hirukawa, Fumio Kanehiro, Kiyoshi Fujiwara, Kensuke Harada, Hajime Saito and Mitsuharu Morisawa of Humanoid Research Group, AIST for their useful advice.

## REFERENCES

[1] Gienger, M., Löffler, F. and Pfeiffer, F., "Toward the Design of a Biped Jogging Robot," Proc. of the 2001 ICRA, pp.4140–4145, 2001.
[2] Hirai, K., Hirose, M., Haikawa, Y. and Takenaka, T., "The Development of Honda Humanoid Robot," Proc. of the 1998 ICRA, pp.1321–1326, 1998.
[3] Inoue, H., Tachi, S., Nakamura, Y., Hirai, K., et.al, "Overview of Humanoid Robotics Project of METI," Proc. Int. Symp. Robotics, pp.1478–1482, 2001.
[4] Nishiwaki, K., Sugihara, T., Kagami, S., Kanehiro, F., Inaba, M., and Inoue, H., "Design and Development of Research Platform for Perception-Action Integration in Humanoid Robot: H6," Proc. Int. Conference on Intelligent Robots and Systems, pp.1559–1564, 2000.
[5] Yamaguchi, J., Soga, E., Inoue, S. and Takanishi, A., "Development of a Bipedal Humanoid Robot – Control Method of Whole Body Cooperative Dynamic Biped Walking –," Proc. of the 1999 ICRA, pp.368–374, 1999.
[6] Kim, J., Oh, J., "Walking Control of the Humanoid Platform KHR-1 based on Torque Feedback Control," Proc. of the 2004 ICRA, pp.623–628, 2004.
[7] Raibert, M., *Legged Robots that Balance*, Cambridge, MA, MIT Press, 1986.
[8] Playter, Robert R. and Raibert, Marc H., "Control of a Biped Somersault in 3D," Proc. of IFToMM-jc International Symposium on Theory of Machines and Mechanisms (in Nagoya, Japan), pp.669–674, 1992.
[9] Hodgins, J. K.,"Three-Dimensional Human Running," Proc. of the 1996 ICRA, pp.3271-3277, 1996.
[10] Ahmadi, M. and Buehler, M., "The ARL Monopod II Running Robot: Control and Energetics," Proc. of the 1999 ICRA, pp.1689-1694, 1999.
[11] Nagasaka, K., Kuroki, Y., Suzuki, S., Itoh, Y., Yamaguchi, J., "Integrated Motion Control for Walking, Jumping and Running on a Small Bipedal Entertainment Robot," Proc. of the 2004 ICRA, pp.3189–3194, 2004.
[12] C. Chevallereau, E.R. Westervelt, and J.W. Grizzle, Asymptotically Stable Running for a Five-Link, Four-Actuator, Planar Bipedal Robot, Pre-print, paper submitted to IJRR. Paper, 2004.
[13] http://world.honda.com/ASIMO
[14] Kajita, S. et al.,"A Hop towards Running Humanoid Biped," Proc. of the 2004 ICRA, pp.629–635, 2004.
[15] Kaneko, K. et al.,"Design of Advanced Leg Module for Humanoid Robotics Project of METI," Proc. of the 2002 ICRA, pp.38–45, 2002.
[16] Kajita, S., Kanehiro, F., Kaneko, K., Fujiwara, K., Harada, K., Yokoi, K. and Hirukawa, H., "Resolved Momentum Control: Humanoid Motion Planning based on the Linear and Angular Momentum," Proc. of the 2003 IROS, pp.1644–1650, 2003.
[17] Kajita, S., Matsumoto, O. and Saigo, M., "Real-time 3D walking pattern generation for a biped robot with telescopic legs," Proc. of the 2001 ICRA, pp.2299-2306, 2001.
[18] Hori, Y. and Ohnishi, K., Ohyou-Seigyo Kougaku (Applied Control Engineering), Maruzen, 1998. (in Japanese)