# Fall on Backpack: Damage Minimization of Humanoid Robots by Falling on Targeted Body Segments

**Sung-Hee Lee**
School of Information and Communications,
Gwangju Institute of Science and Technology,
Gwangju, South Korea 500-712
e-mail: shl@gist.ac.kr

**Ambarish Goswami**[1]
Honda Research Institute USA,
Mountain View, CA 94043
e-mail: agoswami@honda-ri.com

*Safety and robustness will become critical issues when humanoid robots start sharing human environments in the future. In physically interactive human environments, a catastrophic fall is a major threat to the safety and smooth operation of humanoid robots. It is, therefore, imperative that humanoid robots be equipped with a comprehensive fall management strategy. This paper deals with the problem of reducing the impact damage to a robot associated with a fall. A common approach is to employ damage-resistant design and apply impact-absorbing material to robot limbs, such as the backpack and knee, that are particularly prone to fall related impacts. In this paper, we select the backpack to be the most preferred body segment to experience an impact. We proceed to propose a control strategy that attempts to reorient the robot during the fall such that it impacts the ground with its backpack. We show that the robot can fall on the backpack even when it starts falling sideways. This is achieved by generating and redistributing angular momentum among the robot limbs through dynamic coupling. The planning and control algorithms for a fall are demonstrated in simulation.* [DOI: 10.1115/1.4006783]

## 1 Introduction

As humanoid robots start to migrate from controlled laboratory environments to free and physically interactive surroundings containing objects and people, the issues of safety and robust operation of these robots will demand major attention from the research community. A catastrophic fall is one of the gravest threats to the safety and security of these robots and their surroundings. Yet, despite the stringent control that is imposed on every humanoid movement, fall remains an uncontrolled, understudied, and basically overlooked aspect of humanoid technology.

Because of its complex dynamics and lack of well-defined theoretical tools, one is tempted to completely ignore the treatment of a humanoid fall. This, however, does nothing to reduce the chances of falling, but rather makes the effects of falling unpredictable and potentially more harmful. In a comparable situation involving automobiles, we have learned that crash studies can significantly improve the "crashworthiness" of a car by increasing the safety of both car occupants and outside pedestrians. Inspired by this, we deliberately focus our attention to the phenomenon of humanoid falling and attempt to develop a *fall controller* to deal with this undesired and catastrophic event.

Let us clarify that a fall controller complements, and does not replace or compete with, either a balance controller or a step controller for push-recovery [1]. Postural balance controllers try to maintain balance against disturbances through joint torque control and postural movements. Step controllers are essentially extended balance controllers, which deal with external disturbances of magnitudes larger than what the postural balance controller can handle. In this case, the robot must take a step in order to regain balance. The fall controller is activated only when both the postural balance controller and the step controller are deemed insufficient for the disturbance, and the robot is expected to face a guaranteed fall. The objective of the fall controller is not to regain the balance, because it is irreversibly lost, but to fall in the safest manner.

We have made the following assumptions in formulating our controller:

(1) All motors stay active during the entire motion of the robot.
(2) The robot can perfectly sense its states, including its global position and all the joint angles.
(3) The floor has enough friction to ensure no slip.

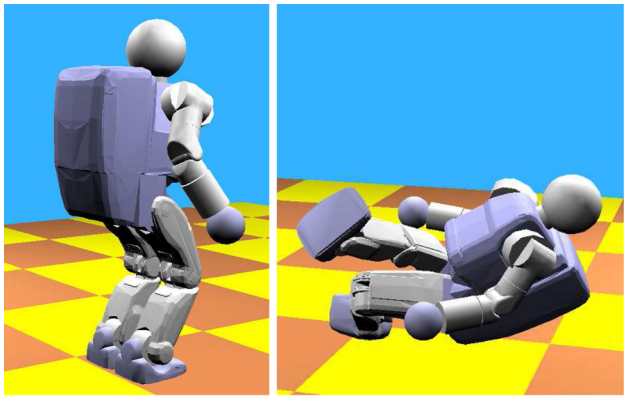
During an accidental fall the humanoid controller may have two primary, and distinctly different, objectives: (a) self-damage minimization and (b) minimization of damage to other objects or people. If during the fall the robot can hit nearby objects or persons, its primary objective would be to prevent this from happening. The robot may try to achieve this by means of changing its default fall direction, as we have reported before [2,3]. If, however, the fall occurs in an open space, a self-damage minimization strategy can attempt to reduce the harmful effects of the ground impact.

One practical approach to reducing impact related damages is to employ better design and apply impact-absorbing materials to robot parts that are frequently prone to fall-related impacts. One can identify a number of body segments, such as the knee, hip, hand, head, etc., that are more likely to contact the floor in case of a fall. For many humanoid robots, the backpack is a segment that, due to its design and 3D profile, would come into frequent contact with the ground if the robot were to fall. The backpack, which is an integral part of the humanoid trunk for most robots, can be properly designed to prevent the effect of the impact propagating to other, more fragile, components. By virtue of its special design, the backpack, akin to an automobile bumper, becomes better-equipped to face an impact compared to other parts. The challenge then is to ensure that the backpack is the part of choice, more than other parts of the robot, with which the robot ought to touch the ground during a fall. This precisely is the objective of the control strategy proposed in this paper (see Fig. 1).

This is a difficult task because the underactuation of the falling robot prevents the controller from achieving a desired orientation of a body part using only kinematic relation with other body parts.

[1]Corresponding author.

Contributed by the Design Engineering Division of ASME for publication in the Journal of Computational and Nonlinear Dynamics. Manuscript received August 30, 2011; final manuscript received April 18, 2012; published online July 23, 2012. Assoc. Editor: Arend L. Schwab.



(a) (b)

**Fig. 1 Backpack is rigidly attached to the back of the trunk (left), and we aim to control a falling robot to touch down with a specific body part, in our case the backpack (right)**

Rather, the movement of a body part can only be understood and controlled through dynamic coupling involving other body parts.

When the ground reaction force (GRF) and the gravity are the only external forces applied to the robot, the angular momentum about the *lean line*, the line connecting the center of pressure (CoP) and the center of mass (CoM), of a falling robot is conserved because all external forces have zero moment-arms from the lean line if we assume that the support foot and the ground make a point contact and the rotational friction between the foot and the ground is zero.[2]

This behavior is similar to that of a rotating top where the angular momentum about the spin axis is conserved. Falling humanoids are not expected to attain angular velocity of too large magnitude about the spin axis; thus, we ignore the precession effect that can be observed in a rapidly spinning top. Therefore, a change of angular velocity of the trunk about the lean line can only be achieved by rotating other body parts in the opposite direction. On the other hand, due to the ground contact, the angular momentum in the other directions is not preserved and can be changed.

In this paper, we present an inverse dynamics-based control algorithm that controls the joint motion and the GRF to achieve the desired touchdown orientation of the trunk at touchdown time using momentum-based dynamic coupling of the links.

Our general motivation behind this work is to cause a robot to fall on *any* targeted body segment through the use of dynamic coupling between different limb segments. As a proof of concept, in this work we have chosen to guide the robot to fall on its backpack. The backpack of most robots is expected to contain a shock resistive internal frame and is a prominent protruding feature. Other fall strategies may include falling on the hip, knee, etc.

Time is a premium during the occurrence of a fall. Theoretical analysis of single rigid body models of human-sized humanoid robots indicate that a fall from the vertical upright stationary configuration due to a mild push takes only about 800–900 ms. In real situations, the time to fall may be even shorter, and there is no opportunity for elaborate planning or time-consuming control. Yet, simulation and experimental results indicate that meaningful modification to the default fall behavior can be imparted in this brief time window to minimize damage to the robot or to the environment.

Reported work in the area of the humanoid fall is relatively rare. A few recent papers reported on the damage minimization aspect of a humanoid fall. In their body of work Fujiwara et al. [4–9] proposed martial arts type motion for damage reduction, computed optimal falling motions using minimum impact and angular momentum, and fabricated special hardware for fall damage study. Ishida et al. employed servo loop gain shift to reduce shock due to a fall [10]. Ogata et al. proposed online planning techniques for the trajectory of the center of mass of a falling robot to reduce impact [11,12], and del Solar et al. manually tuned the falling motion of soccer robots [13]. Fall damage minimization is obviously of natural interest in human biomechanics [14–16] among the ample research to analyze human locomotion, e.g., Ref. [17].

Directly controlling the robot during the fall in an attempt to make ground touchdown with a specified body part, our approach clearly distinguishes itself from most prior research.

Although our ultimate objective is to implement fall strategies on full-sized humanoid robot hardware, there are practical issues that are to be first resolved before we are able to do so. Humanoid robots are complex, fragile, and expensive systems, and the effect of a fall can be so catastrophic, even in a controlled experiment, that one may never want to undertake a hardware test involving a deliberately inflicted fall. Although we believe that research needs to be done to address issues surrounding the occurrence of a humanoid fall, the barrier to hardware test is high. Consequently, for the study of this catastrophic and rare event our dependence on simulation has been strong.

Our approach is to first formulate control strategies in a high-fidelity dynamic simulation environment containing accurate physical parameters of the robot. A fall controller is developed and evaluated in this environment through numerous simulations. Next we will implement this fall controller, which has been refined and updated in the simulator, on the robot hardware. We have started this process with a "table-top" humanoid robot (Aldebaran NAO), which, due to its small size, is likely to suffer less damage.

## 2 Fall Control Framework

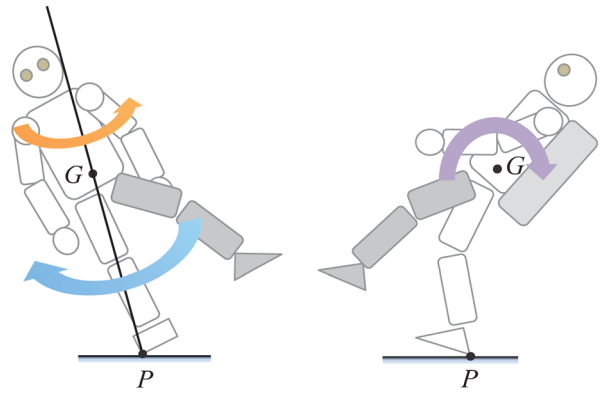
The problem addressed in this paper is quite unique in many respects. It is instructive to analyze these special characteristics and understand their implications. Our control objective can be divided into two parts: the first is to ensure that all other body parts, except the feet, are above the ground during touchdown, and the second is to reorient the trunk so that the backpack faces the ground at touchdown, even though we do not know the exact time of touchdown.

The first objective implies that the backpack is the first to touch the ground and is mainly meant to keep the arms out of the way. To deal with this we take the simple step of maintaining or returning the arms to their default pose of keeping the hands naturally positioned in front of the pelvis. This way the arms will be located above the ground during the touchdown.

In order to reorient the trunk so that the robot touches the ground with the backpack, we may specify the desired world-frame orientation of the backpack at the instant of touchdown and try to achieve it purely through kinematic control of the links. However, since the falling (tipping) robot is underactuated, a certain rotation of the backpack may result in a different rotation in another segment, completely modifying the trunk orientation in complex and nonintuitive ways. In addition, we do not know the precise length of time that the robot takes to fall to the ground.

Therefore, it is essential to take into account the dynamic coupling among the body parts of the falling robot: by moving other parts of the robot, we attempt to indirectly control the trunk orientation. The arms of our robot are relatively lightweight, and their movement does not affect the whole body dynamics appreciably. Therefore, we will use the legs, which are relatively heavier, for this purpose. In typical sideways falls, where our fall strategy will be the most valuable, one foot of the robot will be in contact with the floor while the other will be free. Let us call the corresponding legs the support leg and the swing leg, respectively. We will use the inertial effect of the rotating swing leg to indirectly control the trunk orientation. We do not use the support leg for this purpose.

[2]Our fall controller tries to make a point contact rather than an edge contact between the foot and the ground. The assumption on zero rotational friction simplifies the dynamic analysis of the falling robot, but the effect of the friction might not be negligible in reality.




(a) Spinning with swing leg (b) Bending the trunk backward

**Fig. 2 The strategies for falling on the backpack are illustrated. (*a*) By counter-rotating the swing leg, the robot controls the spin of the trunk. (*b*) Additionally, the robot bends the trunk backward to increase the possibility of touching down with the backpack. The points *G* and *P* denote CoM and CoP, respectively.**

One reason is that due to its short moment arm from the lean line (see Fig. 2), the support leg has only a small effect on the rotational dynamics about the lean line. Additionally, the hip joint of the robot has a fairly limited range about the vertical axis, which restricts its ability to generate a large momentum.

If the robot is able to freely rotate about the spin axis, it will be able to fall on the backpack regardless of the initial fall direction. The lean line, described before, plays an important role as the spin axis. Since the gravity and the GRF do not create torques about the spin axis as they both intersect it, the corresponding component of the angular momentum is preserved. This means that the only way to control the spin angle of the trunk is by rotating some other member, such as the swing leg, about the spin axis. However, because the swing leg rotation is constrained by joint limits, we may not attain the desired trunk orientation only through this means. To increase the chance of falling on the backpack, we take the additional heuristic measure of bending the trunk backward as will be detailed later. Figure 2 illustrates these strategies.

Fall control consists of planning and execution steps. When a robot detects an impending inevitable fall, it first plans how to fall. In order to touch down with the backpack, the controller determines the desired rotational velocity of the trunk given the current state of the robot. The planning is performed multiple times during the fall to update the desired velocity while it is falling. At each control time step, the robot determines the joint acceleration and torques to realize the desired falling motion.

We assume that we can either compute or measure the position and orientation of the trunk and its linear and angular velocities as well as the joint angles and velocities using sensors such as gyroscopes, accelerometers, and joint encoders. We also assume that the feet are equipped with force/torque sensors to measure the location of the CoP.

## 3 Planning Fall Control

Since spin control is essential to our fall controller, we will define the *spin frame* and describe the centroidal dynamics of the humanoid with respect to this reference frame. As shown in Fig. 3, the spin frame is located at the CoM. Its axes $\{t, b, s\}$ correspond to frontal, bending, and spin, respectively. The spin axis $s$ coincides with the lean line $GP$. The frontal axis $t$ is given by the intersection of the sagittal plane of the trunk and the plane normal to $s$ (translucent circle). The torso bending direction $b$ is given by $s \times t$.

The centroidal angular momentum $k$, which is the aggregate angular momentum of a humanoid robot projected at its CoM, and

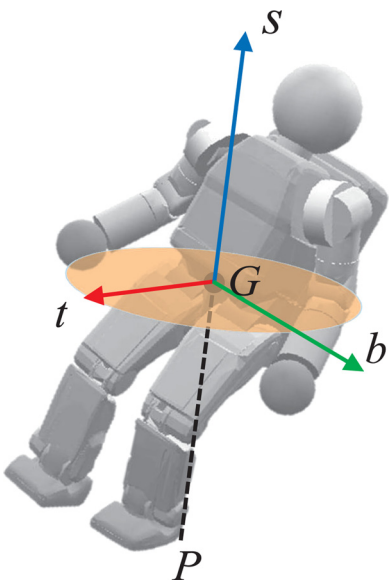



**Fig. 3 The *Spin frame* is located at the CoM and is defined by the axes $\{t, b, s\}$. The spin axis *s* coincides with the lean line $\overrightarrow{PG}$. The frontal axis *t* is the intersection of the sagittal plane of the trunk, and the plane normal to *s* (translucent circle). The bending direction *b* is given by *s*×*t*.**

the generalized velocity vector $\dot{q}$ have the linear relations as follows [18]:

$$k = K\dot{q} \tag{1}$$

where $k$ is a $3 \times 1$ vector and, for a robot with $n$ joints, $\dot{q} \in \mathbb{R}^{n+6}$ is the generalized velocity vector and $K \in \mathbb{R}^{3\times(n+6)}$ transforms the generalized velocity vector to angular momentum. $K$ is called the centroidal momentum matrix if $k$ refers to the angular momentum about CoM [18].

To extract the motion of the link of interest, the trunk in our case, we divide $\dot{q} = (\omega, v, \dot{\theta})$ into the angular and linear velocities of the trunk $(\omega, v)$ and the joint velocity vector $\dot{\theta} \in \mathbb{R}^n$. Then, Eq. (1) is expanded as

$$k = K_w\omega + K_v v + K_\theta\dot{\theta} \tag{2}$$

where $K_w$, $K_v$, and $K_\theta$ are sub-matrices of $K$ corresponding to $\omega, v$, and $\dot{\theta}$, respectively.

If we express $k$, $\omega$, and $v$ with respect to the spin frame (or, more specifically, a fixed frame that coincides with the spin frame instantaneously), the linear term vanishes because the spin frame is located at the CoM and the linear velocity of the trunk $v$ does not affect $k$,

$$k = K_w\omega + K_\theta\dot{\theta} \tag{3}$$

where $K_w$ is the locked rotational inertia of the robot, given by a symmetric positive definite matrix. Later, in the planning step we will pay attention to the spin component $k_s$ of $k = (k_t, k_b, k_s)$.

**3.1 Touchdown Time and Trunk Orientation Estimation.** In order to control the orientation of the backpack so that it hits the ground first, it is important to know when the robot will collide with the ground. Although the exact time of touchdown depends on the movement of the robot, we can get a rough estimate by simulating a reduced model of the robot. For faster computation, we assume that all the joints of the robot are locked, such that the entire robot is treated as a single rigid body, and simulate the motion of the rigid body until it collides with the ground.

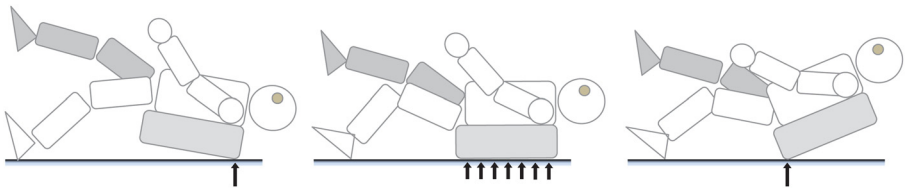


**Fig. 4 Various configurations of the backpack when a robot successfully touches down with the backpack. Flat contact (middle) may reduce impact at collision, but it may not be achieved in practice because the backpack may have a curved contour. We do not care which point of the backpack collides with the ground.**

Specifically, the rigid body is set to rotate about the robot CoP with the initial angular velocity being the same as that of the robot trunk. The spin component of the angular velocity is denoted as $\omega_{s,0}$. We perform the dynamic simulation to track the orientation of the rigid body until the height of CoM falls below a threshold $h_t$, approximating the collision between the robot and the ground, and estimate the duration of fall $t_f$. We set $h_t$ to about one third of the height of the CoM of the robot standing with the default pose.

Also, we measure the orientation of the backpack at touchdown. This would be the backpack orientation if the robot did not take any action, i.e., it locked all its joints during fall. This is the starting point of our fall control.

**3.2 Desired Trunk Angular Velocity Estimation.** There could be an optimal collision point on the backpack that minimizes the damage depending on the material property and the particular shape of a backpack, but its consideration is out of the scope of this work. Instead, we assume that the backpack is well-designed overall such that it can effectively block shock propagation to other body parts regardless of the specific location of the collision point. Then, so long as the backpack is the first segment to touch the ground, we do not care about the specific collision point on the backpack that touches the ground. In other words, any sagittal plane orientation of the trunk during the ground contact is taken as acceptable, as explained in Fig. 4. We, therefore, try to orient the vector $n$, which is normal to the backpack and points backward from the trunk (Fig. 6), in such a way that the plane made by $GP$ and $n$ is normal to the ground. This would ensure that the backpack is facing the ground regardless of the actual orientation of $n$ in the sagittal plane. See Fig. 5 for a schematic of this configuration.

Given the estimated normal vector $n$ at the time of touchdown, we determine the spin angle $\phi$ that is required to rotate $n$ about the spin direction to the desired vector $n'$ (Fig. 5). Then $\frac{\phi}{t_f}$ represents the "additional spin velocity" that the robot should have had to fall on its backpack. Therefore, the desired spin velocity $\omega_{s,d}$ is determined as

$$\omega_{s,d} = \omega_{s,0} + \frac{\phi}{t_f} \tag{4}$$

where $\omega_{s,0}$ is the initial value of the spin component of angular velocity of the trunk at the preview stage, which remains constant during the simulation, and $t_f$ is the estimated duration of fall that was computed previously. However, $\omega_{s,d}$ may not be physically attainable due to limits on joint angles and velocities of the swing leg. Therefore, we need to determine its *admissible* value. To this end, we use the third row of the momentum relation of Eq. (3).

$$k_s = K_{w(3,1)}\omega_t + K_{w(3,2)}\omega_b + K_{w(3,3)}\omega_s + K_\theta\dot{\theta} \tag{5}$$

Recall that the spin component of the angular momentum $k_s$ is conserved because no external forces can induce torques about the spin axis. To simplify Eq. (5), we will make the following approximations: (1) The contributions of frontal and bending components of angular velocity to $k_s$ are negligible, i.e., $K_{w(3,1)}\omega_t \approx 0$, $K_{w(3,2)}\omega_b \approx 0$, and (2) only the joint velocities of the hip and knee joints of the swing leg make appreciable contribution to the spin. Then Eq. (5) is approximated as

$$k_s \approx K_{w(3,3)}\omega_s + K_{\text{swg}}\dot{\theta}_{\text{swg}} \tag{6}$$

where $\dot{\theta}_{\text{swg}}$ is the joint velocity vector of the swing leg. Then the possible range of $\omega_s$ is determined from the constraints on joint angles and velocities of the swing leg.

$$\omega_s^l \le \omega_s \le \omega_s^u \tag{7}$$

where

$$\omega_s^l = \min \frac{k_s - K_{\text{swg}}\dot{\theta}_{\text{swg}}}{K_{w(3,3)}} \tag{8}$$

$$\omega_s^u = \max \frac{k_s - K_{\text{swg}}\dot{\theta}_{\text{swg}}}{K_{w(3,3)}} \tag{9}$$

The extrema of $\dot{\theta}_{\text{swg}}$ of each joint are determined from the lower and upper limits on joint angle and velocity as well as the current joint angle. For example, the maximum velocity that a joint $\dot{\theta}_i$ can have is set to be $\min(\dot{\theta}_i^u, (\theta_i^u - \theta_i)/t_f)$ where $\theta_i^u$ and $\dot{\theta}_i^u$ are the upper limits for joint angle and velocity. If the desired spin velocity is within the range, the admissible spin velocity $\omega_{s,a}$ can be set to the desired value. Otherwise, it is clipped to either $\omega_s^l$ or $\omega_s^u$.

Next, we determine the desired bending velocity. If the desired spin velocity is admissible, the robot will be able to touchdown with the backpack without rotating the trunk backward. If the difference between the desired value and the admissible value of the spin velocity is large, the robot would have to rotate backward to increase the chance of falling on the backpack as illustrated in Fig. 6. Using this rationale, we use a heuristic method to determine

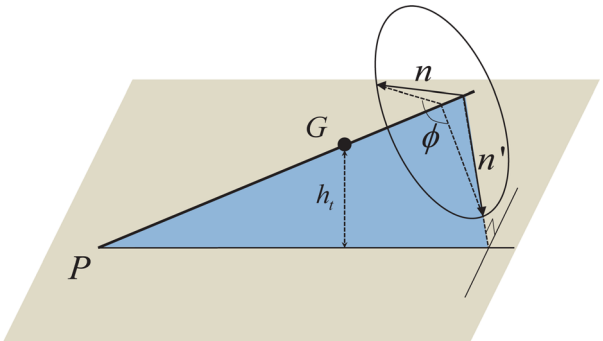



**Fig. 5 A falling robot is modeled as a single rigid body by locking all the joints and is simulated until its CoM drops down to a certain height $h_t$, then the time to fall $t_f$ and the vector $n$, the normal direction pointing backward from the backpack, are measured. Note that $n$ is not necessarily orthogonal to the lean line $GP$ (see Fig. 6 for $n$). The direction of $n'$ is the desired value of $n$ determined by rotating $n$ about the lean line such that the plane made by $GP$ and $n'$ is orthogonal to the ground plane.**

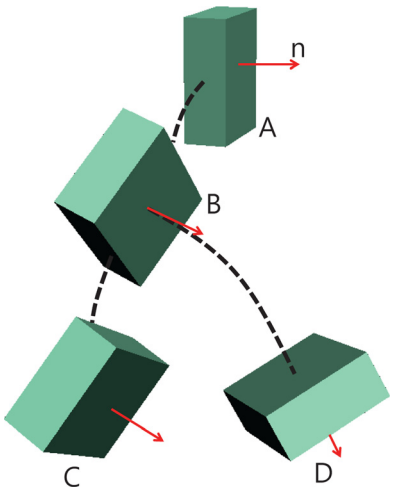



**Fig. 6 The figure illustrates how the bending strategy helps falling on the backpack. The boxes represent the configurations of the backpack at different instances during the fall: A: when a robot is standing upright. B: when the fall controller is activated. C: when the robot hits the ground while only the spinning strategy is employed. D: when the robot hits the ground while both the spinning and bending strategies are employed. The vector *n* represents the normal direction of the backpack, pointing backward. Including the bending strategy makes *n* more orthogonal to the ground plane.**

the backward-bending velocity to be proportional to the difference of the $\omega_{s,a}$ and $\omega_{s,d}$,

$$\omega_b = -c|\omega_{s,a} - \omega_{s,d}| \tag{10}$$

where $c$ is a scaling factor. Following some trial cases, we set $c = 3$. Altogether, the desired angular velocity of the trunk is set as follows:

$$\omega_d = (0, \omega_b, \omega_{s,a}) \tag{11}$$

The first component of $\omega_d$ is set to zero to reduce the rotational component, which is not particularly related with our control strategy.

## 4 Executing Fall Control

Given the desired angular velocity in Eq. (11) of the trunk computed in the planning step, the fall controller controls the joint torques to realize the desired angular velocity.

To this end, we first derive the governing equations that relate the angular acceleration of the trunk, joint accelerations, and GRF. From this relation, we determine the desired joint accelerations and GRF that will drive the robot to have the desired angular velocity of the trunk, followed by computing necessary joint torques to realize the desired joint accelerations.

**4.1 Governing Equations.** The relation between GRF at the contact point and the rate of change of net spatial momenta are as follows:

$$\dot{k} = K\ddot{q} + \dot{K}\dot{q} = GP \times f \tag{12}$$

$$\dot{l} = L\ddot{q} + \dot{L}\dot{q} = f + mg \tag{13}$$

where $f$ is GRF, $l$ is linear momentum, $m$ is the total robot mass, and $g$ is the gravity vector. Recall that $GP$ is the lean line, which connects the CoM and the CoP. Equation (12) is obtained by differentiating Eq. (1). The matrix $L \in \mathbb{R}^{3\times(n+6)}$ transforms the generalized velocity vector to linear momentum, i.e., $l = L\dot{q}$.

Additionally, if the robot is to keep contact with the ground (i.e., the CoP does not move), the following kinematic constraint must be maintained:

$$\dot{P} = J\dot{q} = 0 \tag{14}$$

where $J = dP/dq$. We will use its time differentiation,

$$0 = J\ddot{q} + \dot{J}\dot{q} \tag{15}$$

As in Eq. (2), let us split $\ddot{q}$ into the angular and linear accelerations of the base frame $(\dot{\omega}, \dot{\upsilon})$ and the joint accelerations $\ddot{\theta}$. Then Eqs. (12), (13), and (15) are rewritten, respectively, as

$$K_w\dot{\omega} + K_v\dot{\upsilon} + K_\theta\ddot{\theta} + \dot{K}\dot{q} = GP \times f \tag{16}$$

$$L_w\dot{\omega} + L_v\dot{\upsilon} + L_\theta\ddot{\theta} + \dot{L}\dot{q} = f + mg \tag{17}$$

$$J_w\dot{\omega} + J_v\dot{\upsilon} + J_\theta\ddot{\theta} + \dot{J}\dot{q} = 0 \tag{18}$$

If we express $\omega, \upsilon$ and Eqs. (16)–(18) with respect to the spin frame, the angular and linear accelerations become decoupled, i.e., $K_v = L_w = 0$. Also, $J_v$ and $L_v$ become identity matrices, and $J_w = -[GP]$.[3] This gives us

$$K_w\dot{\omega} + K_\theta\ddot{\theta} + \dot{K}\dot{q} = GP \times f \tag{19}$$

$$m\dot{\upsilon} + L_\theta\ddot{\theta} + \dot{L}\dot{q} = f + mg \tag{20}$$

$$-GP \times \dot{\omega} + \dot{\upsilon} + J_\theta\ddot{\theta} + \dot{J}\dot{q} = 0 \tag{21}$$

Since we are not interested in the linear motion of the trunk, we will eliminate $\dot{\upsilon}$ by substituting Eq. (21) into Eq. (20) to get

$$GP \times \dot{\omega} + \left(\frac{1}{m}L_\theta - J_\theta\right)\ddot{\theta} + \frac{1}{m}\dot{L}\dot{q} - \dot{J}\dot{q} = \frac{1}{m}f + g \tag{22}$$

Equations (19) and (22) show the dynamic coupling among the angular acceleration of the trunk, joint accelerations, and GRF under kinematic constraints. Obviously, the joint accelerations $\ddot{\theta}$ completely determine both $\dot{\omega}$ and $f$ and it is possible to eliminate $f$ to express $\dot{\omega}$ in terms of only $\ddot{\theta}$. However, we will include $f$ in the governing equations because $f$ conveniently describes the constraints (contact force should be inside friction cone) as well as the global linear motion of the robot, i.e., $\ddot{r}_G = f/m + g$ where $r_G$ is the CoM.

**4.2 Fall Control Command.** Our goal is to determine $\ddot{\theta}$ and $f$ such that the desired orientation of the trunk is achieved at the time of ground impact. Let us assume that we have the desired value of the rotational acceleration of the trunk $\dot{\omega}_d$ as well as the desired joint accelerations $\ddot{\theta}_d$ and GRF $f_d$. Given the desired values, determining $\ddot{q}$ and $f$ from Eqs. (18) and (20) can be described as a constrained least-squares problem of the form

$$\begin{aligned} &\min (1-w)\|Ax-b\|^2 + w\|x - x_d\|^2 \\ &\text{s.t. joint limit constraints} \end{aligned} \tag{23}$$

where

$$x^T = (\dot{\omega}^T, \ddot{\theta}^T, f^T) \tag{24}$$

$$A = \begin{bmatrix} K_w & K_\theta & -[GP] \\ [GP] & \left(\frac{L_\theta}{m} - J_\theta\right) & -\frac{1}{m}I \end{bmatrix} \tag{25}$$

[3] $[\cdot]$ denotes a $3 \times 3$ skew symmetric matrix representing a vector cross product as in $[a]b = a \times b$.

$$b = \begin{bmatrix} -\dot{K}\dot{q} \\ -\frac{1}{m}\dot{L}\dot{q} + \dot{J}\dot{q} + g \end{bmatrix} \tag{26}$$

$x_d^T = (\dot{\omega}_d^T, \ddot{\theta}_d^T, f_d^T)$ is the desired value for $x$. Since $Ax = b$ is an under-determined problem, we include an additional cost function $||x - x_d||^2$ to find the solution close to the desired values $x_d$, with a small weighting factor $w$ (=0.01 in our experiment). We enforce inequality constraints on $\ddot{\theta}$ to satisfy joint limits. It is done rather heuristically, e.g., by giving smaller upper bound on a joint acceleration if the joint gets close to the upper limit.

The details of setting the components of $x_d$ are as follows. At each control time step, the desired angular acceleration of the trunk $\dot{\omega}_d$ is set as

$$\dot{\omega}_d = \gamma(\omega_d - \omega) \tag{27}$$

where $\gamma$ is a feedback gain parameter. We set $\gamma = 20$ in our experiment through trial and error.

Next, we need to determine $\ddot{\theta}_d$ and $f_d$. We set $\ddot{\theta}_d$ for the arm joints such that the arms follow the prescribed motion, in our case maintaining the default arm pose, and set $\ddot{\theta}_d = 0$ for the leg joints to give preference to minimal leg movement.

Setting $f_d$ is more involved and has significant effect on the behavior of the fall controller. Setting $f_d$ too high will cause the robot to jump into the air, whereas setting it too low will make it collapse before achieving our control goal. Our approach is to set $f_d$ such that the CoM of the robot moves as if the robot were a falling 3D point-mass pendulum. We set the position of the point-mass $r$ to the CoM of the robot and set the angular velocity $\dot{\phi}$ of the pendulum such that the velocity of the point-mass (i.e., $\dot{\phi} \times r$) is equal to the normal velocity of the CoM of the robot to the lean line. Then, the desired GRF is set to the GRF of the 3D point-mass pendulum as follows:

$$f_d = -mr\left(\frac{r^T g}{r^T r} + \dot{\phi}^T \dot{\phi}\right) \tag{28}$$

where $r = PG$. Equation (28) can be derived by the relations of a 3D point-mass pendulum:

$$m\ddot{r} = f + mg \tag{29}$$

$$\dot{r} = \dot{\phi} \times r \tag{30}$$

$$f \times r = 0 \tag{31}$$

where $r = \overrightarrow{PG}$, $f$ is the GRF, and $\dot{\phi}$ is the angular velocity. Equation (31) ensures that the GRF is collinear with $\overrightarrow{PG}$.

Given $\dot{\omega}_d$, $\ddot{\theta}_d$, and $f_d$, we solve Eq. (23) to compute $\ddot{\theta}$ and $f$, which are then used as inputs for the inverse dynamics to

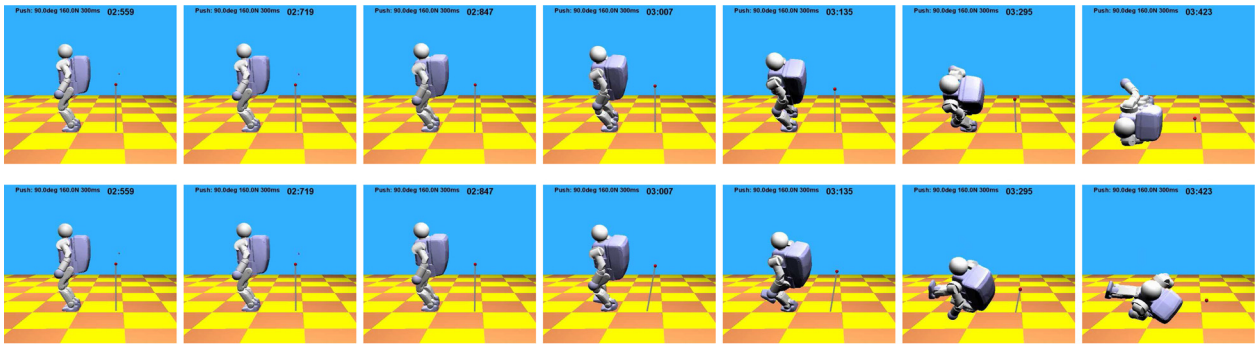



**Fig. 7 An external force of 160 N applied at the CoM of the robot to its left for 300 ms makes the robot fall. Top row: The robot locks all the joints without triggering the fall controller. It falls sideways. Bottom row: The robot engages only the spinning strategy. It lifts and rotate its right leg to control the spin angle of the trunk. The trunk bending strategy is not used. The trunk spins noticeably, but not enough to fall on the backpack. As a result, the robot's hand hits the ground first.**

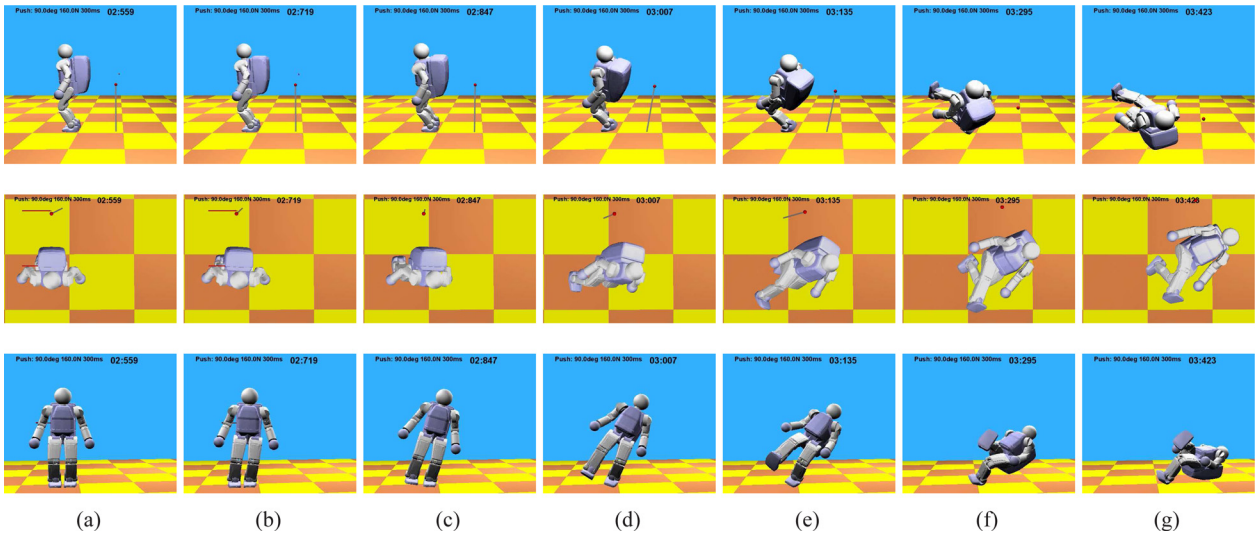



**Fig. 8 Under the same external force as in Fig. 7, the robot uses both the spinning and bending strategies. It successfully touches the ground with the backpack. The top, middle, and bottom rows show the side, top, and front views, respectively, of the simulation.**

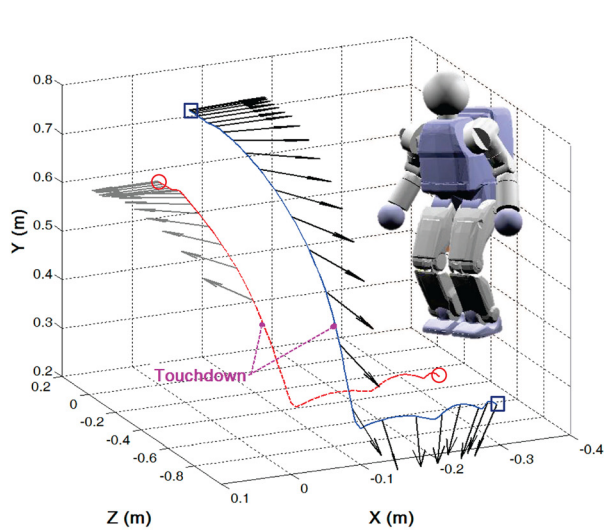
determine joint torques. We use the hybrid system dynamics algorithm [19] for this purpose.

**4.3 Maintaining a Point Contact.** While the robot is falling, we control the support foot so that it maintains a point contact with the ground because edge or planar contact would incur a large friction that will hinder the robot from spinning. To this end, the robot either flexes or extends the support foot if the CoP is not at one of the corners of the foot. The foot is flexed if the CoP is closer to the hill so that the CoP is nudged to the hill, and it is extended if the CoP is closer to the toe so as to push the CoP towards the toe.

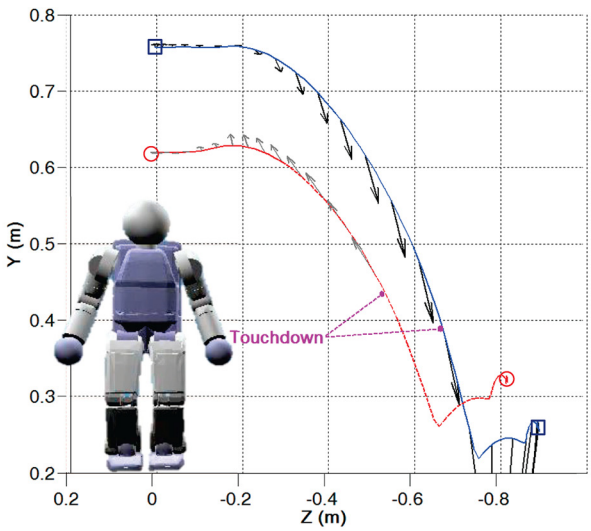
## 5 Simulation Results

We tested the proposed controller with a full-sized humanoid robot model of which the height and weight are about 120 cm and 50 kg. In Fig. 7, the robot is given an external push of 160 N applied to the left for 300 ms. The push makes the robot fall sideways (Fig. 7, top) when no fall control is engaged. The bottom row of Fig. 7 shows the result when the fall controller uses only the spinning strategy, without activating the bending strategy. The robot lifts and rotates the right leg, which creates a noticeable spin of the trunk as can be seen compared with the no-control case. However, the robot fails to touchdown with its backpack and instead its hand first collides with the ground as shown in Fig. 7(*g*).

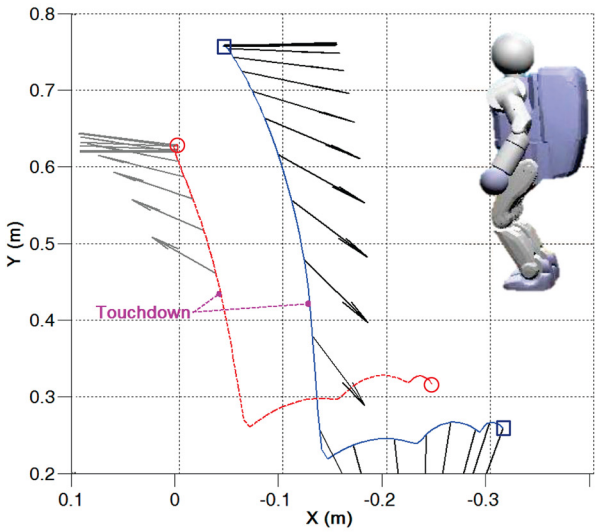
Figure 8 shows the result of using both the spinning and bending strategies under the same push as in Fig. 7. By lifting its swing leg, the robot tries to rotate the trunk counterclockwise about the spin axis and increases the possibility of touchdown with the backpack by rotating the trunk backward. As a result, it successfully touches down with the backpack (Fig. 8(*g*)). To visualize the temporal changes of the configurations of the backpack during the fall control, Fig. 9 shows different views of the trajectories of the



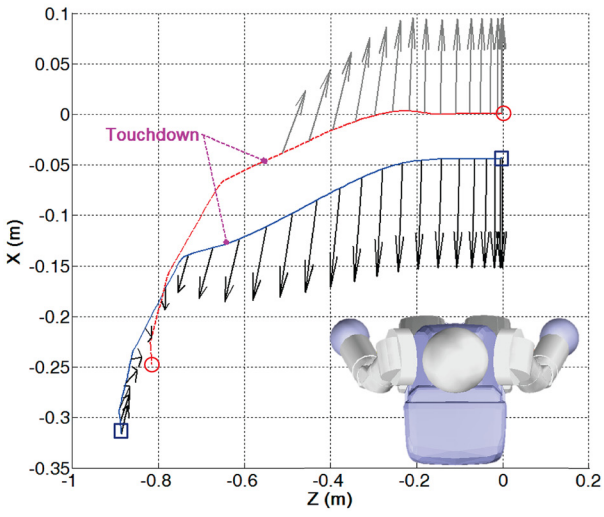

(a) Perspective view




(b) Frontal view

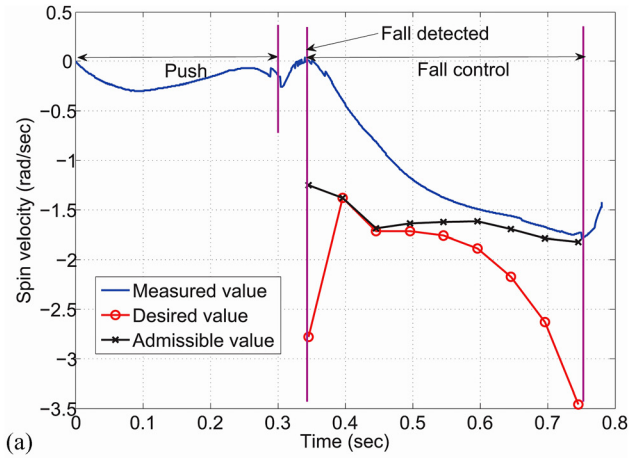



(c) Side view




(d) Top view

**Fig. 9 Plots of CoM (dotted line marked with circles), frontal axis *t* of the spin frame (gray arrows), the position of the trunk frame (solid line marked with squares), and the normal vector *n* to the backpack (black arrows, see Fig. 5 for *n*) for the simulation of Fig. 8. Frontal axes and the normal vectors are drawn at every 50 ms. The inset robot figure in each graph indicates the viewpoint in which the graph is drawn. The vector *n* is not orienting completely downward at touchdown time, but the robot still manages to touch the ground with the backpack.**



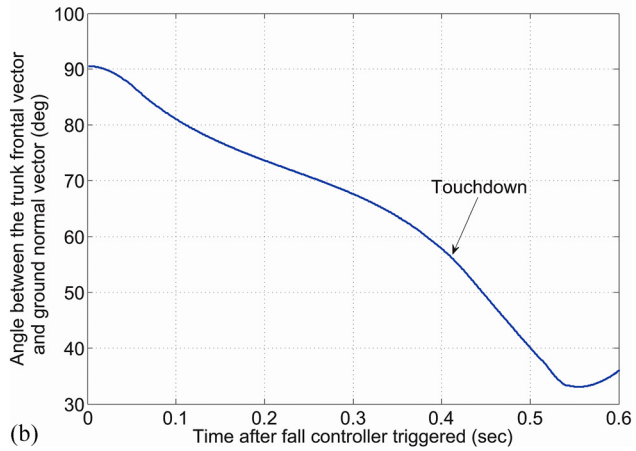
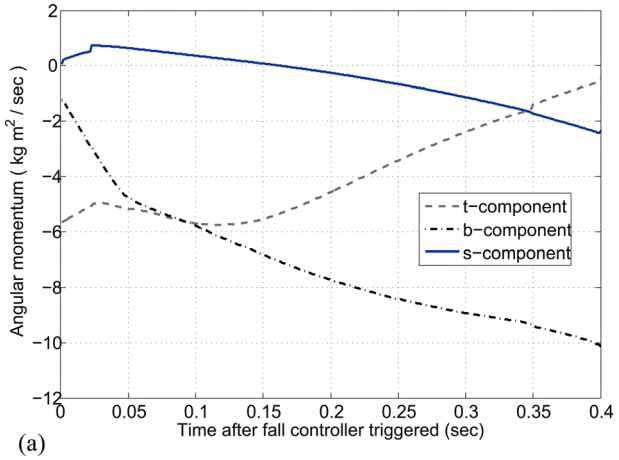




**Fig. 10 Spin velocity (top) and the angle between the normal direction of the backpack and the downward normal direction of the ground (bottom) of the fall control experiment in Fig. 8**



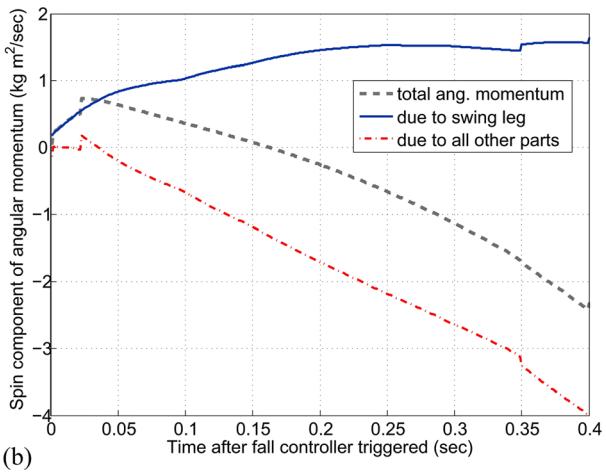




**Fig. 11 Angular momentum of the robot in the experiment of Fig. 8 Top: angular momentum with respect to the spin frame. Bottom: Decomposition of the spin component of the angular momentum (dotted line, middle) into two sources, one due to swing leg (solid line, top) and the other due to all other body parts (dotted line, bottom).**

CoM as well as the frontal axes $t$ of the spin frame and the normal vectors $n$ to the backpack snapshot at every 50 ms during the simulation of Fig. 8.

Figure 10 shows the trajectories of spin velocity and the angle between the normal vector of the trunk ($n$ in Fig. 5) and the downward normal direction of the ground for the experiment shown in Fig. 8. In our experiment, the fall controller is triggered when the angle between the lean line and the ground normal direction exceeds 15 deg, which is detected about 30 ms after the push. The duration for which the fall controller is active is about 400 ms, and the planning of the fall is executed at every 50 ms. One can see that the actual spin velocity is progressively approaching the desired spin velocity, but it cannot achieve the desired value because it is not feasible, as can be seen by the discrepancy between the admissible and the desired spin velocities. As a result, the trunk normal vector forms about 57 deg with the downward normal direction of the ground at the touchdown (Fig. 10, bottom). Still with this angle, the robot successfully touches down with the backpack because of the 3D profile of the backpack, which is protruded backward moderately.

Figure 11 shows the trajectory of the three components of angular momentum of the robot. The spin component should be constant in theory, but it gradually changes, although less than the other two components, as shown in Fig. 11($a$). Possible reasons for this error may be due to our simplifying assumptions on the dynamic model such as that the foot and the ground create a point contact whose location is static during the fall. In fact, the contact point will travel along the surface of the foot as the orientation of the foot changes during the fall. Figure 11($b$) shows the decomposition of this spin angular momentum into two groups of body parts: swing leg and the rest of the body. As expected, the angular momentum of the swing leg is in the opposite direction to that of the rest of the body, which verifies that the angular momentum of the body (including trunk) is controlled by counter-rotating the swing leg.

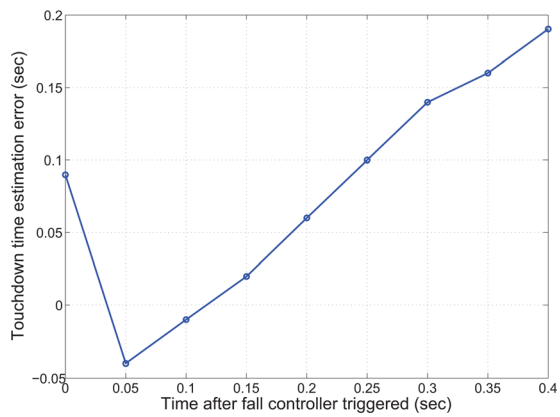
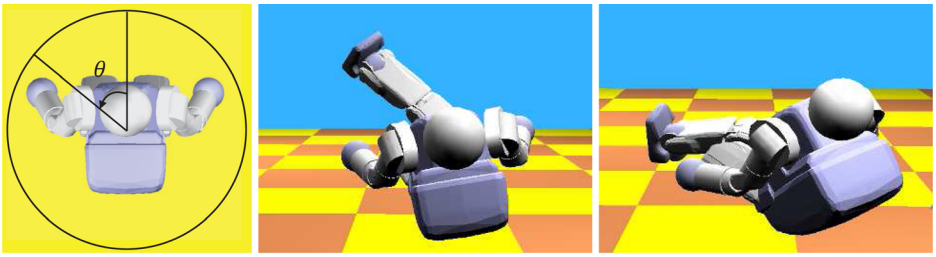



**Fig. 12 Estimation errors, calculated as the difference between the actual and the estimated touchdown times, in each of the planning stages of the experiment in Fig. 8**




(a) Push angle and direction (b) 70 degrees (c) 110 degrees

**Fig. 13 Experiments with different push directions**

Figure 12 shows the errors in estimating the touchdown time in each planning stage. The errors are relatively small in the early stages of the fall, but grow significantly as time proceeds. We conjecture that the error increases because the dynamics of the falling robot becomes increasingly vigorous and complex. Additionally the robot diverges significantly from that of a 3D point mass inverted pendulum on which the touchdown time estimation is based.

In our simulation experiments, the fall controller succeeded up to 70 deg of the push angle ($\theta$ in Fig. 13(*a*)). Figure 13 shows the snapshots of the robot when it touches down with the ground with different push angles. The swing leg is lifted more when the push angle is 70 deg (Fig. 13(*b*)) because it requires more spinning than when it is pushed 110 deg (Fig. 13(*c*)).

## 6 Discussions and Future Work

We showed that a humanoid robot can fall on a target body segment by utilizing the dynamic coupling of the body parts. The fall control technique proposed in this paper can make the robot fall on the backpack even if its default fall direction was significantly different. However, there is a limit in the range of fall directions for which our fall control can be reasonably effective. For instance, the robot cannot touchdown with the backpack if it falls forward.

The effective range of the fall control is determined by many factors such as the mass distribution and joint configuration of a robot. The range will greatly increase if a robot has a waist joint that allows for the vertical rotation of the upper body.

One possible remedy to deal with the limitation of the effective range is to attach shock-absorbing materials to multiple locations on the robot. For example, if the robot has cushioning pads on the knees, the robot will be able to reduce the damage by touching down with knees first when it is falling forward. In order to have multiple target body parts, one would need to develop a controller that first selects the optimal target body part and then tries to fall with that body part.

There is also a possibility that our fall controller is hindered from achieving the goal by a collision with a nearby obstacle. A control strategy avoiding the collision requires a more sophisticated planning step, which remains to be investigated in the future.

There also can be a different method of controlling the target body segment to collide with the ground than what is introduced here. For example, changing the foot-ground contact configuration can be effective for this purpose. If a robot can step while falling, the robot can change the falling direction, and it may also be possible to change the orientation of the trunk as reported in Refs. [2,3]. Changing the location of the contact point of the foot, for example, from the heel to the toe, may also be useful for controlling the trunk orientation.

In this paper, we made a simplifying assumption that the robot is not surrounded by objects that can possibly collide with the robot. In the real situation where a falling robot can collide with surrounding objects or humans, it is important to develop an integrated approach that considers reducing the damage to the environment as well as the damage to the robot itself.

## 7 Conclusion

In order for humanoid robots to coexist with humans for extended time in the real world, their accidental fall must be managed so as to minimize the damage they suffers. In this paper, we propose a novel method to control a falling robot to touchdown with a targeted body part, in this case the backpack. This is achieved principally by spinning the trunk using dynamic coupling through counter-rotating the swing leg about the spin axis. This effort is helped by bending the trunk backward to increase the possibility of fall on the backpack. The assumption is that the design of the backpack is able to survive an impact much more than other parts of the robot.

## Acknowledgment

This work was mainly done while SHL was with HRI. SHL was also supported in part by the Basic Science Program of NRF funded by MEST, Korea (2011-0027158).

## References

[1] Pratt, J., Carff, J., Drakunov, S., and Goswami, A., 2006, "Capture Point: A Step Toward Humanoid Push Recovery," Proceedings of the Humanoids 2006 Conference, Genoa, Italy, Dec. 2–4.

[2] Yun, S.-K., Goswami, A., and Sakagami, Y., 2009, "Safe Fall: Humanoid Robot Fall Direction Change Through Intelligent Stepping and Inertia Shaping," Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 781–787.

[3] Nagarajan, U., and Goswami, A., 2010, "Generalized Direction Changing Fall Control of Humanoid Robots Among Multiple Objects," Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 3316–3322.

[4] Fujiwara, K., Kanehiro, F., Kajita, S., Kaneko, K., Yokoi, K., and Hirukawa, H., 2002, "UKEMI: Falling Motion Control to Minimize Damage to Biped Humanoid Robot," Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2521–2526.

[5] Fujiwara, K. F. K., Kajita, S., Yokoi, K., Saito, H., Harada, K., Kaneko, K., and Hirukawa, H., 2003, "The First Human-Size Humanoid That can Fall Over Safely and Stand-Up Again," Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, pp. 1920–1926.

[6] Fujiwara, K., Kanehiro, F., Kajita, S., and Hirukawa, H., 2004, "Safe Knee Landing of a Human-Size Humanoid Robot While Falling Forward," Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Sendai, Japan, pp. 503–508.

[7] Fujiwara, K. F. K., Saito, H., Kajita, S., Harada, K., and Hirukawa, H., 2004, "Falling Motion Control of a Humanoid Robot Trained by Virtual Supplementary Tests," Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), New Orleans, LA, pp. 1077–1082.

[8] Fujiwara, K., Kajita, S., Harada, K., Kaneko, K., Morisawa, M., Kanehiro, F., Nakaoka, S., Harada, S., and Hirukawa, H., 2006, "Towards an Optimal Falling Motion for a Humanoid Robot," Proceedings of the Humanoids 2006 Conference, Genoa, Italy, pp. 524–529.

[9] Fujiwara, K., Kajita, S., Harada, K., Kaneko, K., Morisawa, M., Kanehiro, F., Nakaoka, S., Harada, S., and Hirukawa, H., 2007, "An Optimal Planning of Falling Motions of a Humanoid Robot," Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 456–462.

[10] Ishida, T., Kuroki, Y., and Takahashi, T., 2004, "Analysis of Motions of a Small Biped Entertainment Robot," Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 142–147.
[11] Ogata, K., Terada, K., and Kuniyoshi, Y., 2007, "Falling Motion Control for Humanoid Robots While Walking," Proceedings of the Humanoids 2007 Conference, Pittsburgh, PA.
[12] Ogata, K., Terada, K., and Kuniyoshi, Y., 2008, "Real-Time Selection and Generation of Fall Damagae Reduction Actions for Humanoid Robots," Proceedings of the Humanoids 2008, Daejeon, Korea, pp. 233–238.
[13] del Solar, J. R., Palma-Amestoy, R., Marchant, R., Parra-Tsunekawa, I., and Zegers, P., 2009, "Learning to Fall: Designing Low Damage Fall Sequences for Humanoid Soccer Robots," Rob. Auton. Syst., **57**(8), pp. 796–807.
[14] Forner Cordero, A., 2003, "Human Gait, Stumble and . . . Fall?," Ph.D. thesis, University of Twente, Enschede, The Netherlands.
[15] Robinovitch, S. R., Hsiao, E. T., Sandler, R., Cortez, J., Liu, Q., and Paiment, G. D., 2000, "Prevention of Falls and Fall-Related Fractures Through Biomechanics," Exercise Sport Sci. Rev., **28**(2), pp. 74–79.
[16] Robinovitch, S. N., Brumer, R., and Maurer, J., 2004, "Effect of the 'Squat Protective Response' on Impact Velocity During Backward Falls," J. Biomech., **37**(9), pp. 1329–1337.
[17] Li, Y., Wang, W., Crompton, R., and Gunther, M., 2001, "Free Vertical Moments and Transverse Forces in Human Walking and Their Role in Relation to Arm-Swing," J. Exp. Biol., **204**, pp. 47–58.
[18] Orin, D., and Goswami, A., 2008, "Centroidal Momentum Matrix of a Humanoid Robot: Structure and Properties," Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Nice, France.
[19] Featherstone, R., 1987, *Robot Dynamics Algorithms*, Kluwer Academic Publishers, Berlin.