Advanced Robotics

Rollover Prevention of Mobile Manipulators using Invariance Control and Recursive analytic Zmp Gradients

Sohee Lee a, Marion Leibold c, Martin Buss c & Frank Chongwoo Park b

a Intelligent Autonomous Systems Group, Technische Universität München, Munich, 80333, Germany
b Robotics Laboratory, Seoul National University, Seoul, 151-742, South Korea
c Institute of Automatic Control Engineering, Technische Universität München, Munich, 80333, Germany

Version of record first published: 10 Jul 2012

To cite this article: Sohee Lee, Marion Leibold, Martin Buss & Frank Chongwoo Park (2012): Rollover Prevention of Mobile Manipulators using Invariance Control and Recursive analytic Zmp Gradients, Advanced Robotics, DOI:10.1080/01691864.2012.690611

To link to this article: http://dx.doi.org/10.1080/01691864.2012.690611

PLEASE SCROLL DOWN FOR ARTICLE

Full terms and conditions of use: http://www.tandfonline.com/page/terms-and-conditions

This article may be used for research, teaching, and private study purposes. Any substantial or systematic reproduction, redistribution, reselling, loan, sub-licensing, systematic supply, or distribution in any form to anyone is expressly forbidden.

The publisher does not give any warranty express or implied or make any representation that the contents will be complete or accurate or up to date. The
accuracy of any instructions, formulae, and drug doses should be independently verified with primary sources. The publisher shall not be liable for any loss, actions, claims, proceedings, demand, or costs or damages whatsoever or howsoever caused arising directly or indirectly in connection with or arising out of the use of this material.
Rollover Prevention of Mobile Manipulators using Invariance Control and Recursive Analytic ZMP Gradients

Sohee Lee*, Marion Leiboldc, Martin Bussc and Frank Chongwoo Parkb

aIntelligent Autonomous Systems Group, Technische Universität München, Munich, 80333, Germany; bRobotics Laboratory, Seoul National University, Seoul, 151-742, South Korea; cInstitute of Automatic Control Engineering, Technische Universität München, Munich, 80333, Germany

Received 17 June 2011; accepted 30 August 2011

Abstract
We present a rollover prevention control law for wheeled mobile manipulators based on the invariance control framework, and that makes use of recursively calculated analytic gradients of the mobile manipulator’s zero moment point. Our controller relaxes many of the assumptions made in existing approaches, and enhances robustness through the use of exact gradient information. Numerical experiments demonstrate the improved performance of our controller vis-à-vis existing rollover prevention schemes.

© 2012 Taylor & Francis and The Robotics Society of Japan

Keywords
rollover prevention, zero moment point, mobile manipulation, balancing control

1. Introduction

As wheeled mobile manipulators become more compact in size, carry larger payloads, and move at higher speeds over more diverse terrain, the possibility of rollover (i.e. the mobile manipulator loses balance and tips over) increases considerably. Vehicle rollover is also a prevalent concern in cars and trucks, and usually the only available means to avoid rollover is through a combination of braking (or sometimes accelerating) and nimble steering. In contrast, wheeled mobile manipulators do not have fixed mass distributions like cars and trucks; they have the advantage of being able to move their arms (and thereby reshaping their overall inertia tensor) so as to prevent rollover. The broad aim of this paper is to develop such online rollover prevention control strategies for wheeled mobile manipulators.

* To whom correspondence should be addressed. E-mail: ssoheelee@gmail.com

© 2012 Taylor & Francis and The Robotics Society of Japan
DOI:10.1080/01691864.2012.690611
More generally, the problem of maintaining dynamic balance during motion is a recurring one in robotics—from humanoids to more diverse legged structures, as well as wheeled mobile robots—and it comes as no surprise that the tools used to characterize balance for legged robots, e.g. the zero moment point (ZMP) and associated support polygon [1], are just as relevant to wheeled mobile manipulators [2]. Existing rollover prevention control laws share the common feature of attempting to keep the zero moment point well inside the support polygon. There is, however, wide variation in the precise formulation of these control laws, and also in the underlying physical assumptions.

Huang et al. [3] are the first to develop a comprehensive mobile manipulator motion planning algorithm that takes into account nonholonomicity of the mobile base, dynamic balance, and end-effector manipulability. They assume a desired end-effector trajectory is given, determine the wheeled mobile base trajectory from velocity and acceleration considerations as well as workspace reachability requirements, and then determine arm motions consistent with the end-effector and base trajectories such that the ZMP stability criterion is satisfied; this latter procedure in general involves the numerical solution of a set of nonlinear differential equations, which are typically not amenable for implementation as a real-time control law.

Among those control laws designed to maintain ZMP stability in real-time, most are based on constructing potential functions that drive any null motions of the manipulator to keep the ZMP within the interior of the support polygon [4–6]. Before describing these methods in detail, it is instructive to first review how potential functions are used in a motion control setting. For this purpose, let \( q \in \mathbb{R}^n \) denote the state vector for the configuration space. Now, given a continuously differentiable potential function \( V : \mathbb{R}^n \to \mathbb{R} \), the system

\[
\dot{q} = -\frac{\partial V}{\partial q}(q),
\]

then follows solution trajectories that are everywhere directed along the negative gradient of \( V \). The equilibria are the critical points of \( V \). In typical motion control applications the goal state is defined to be the unique equilibrium of \( V \), with \( V \) increasing as \( q \) approaches obstacles. Systems of the form (1) are called gradient systems, and there is both a well-developed theory and abundant results on the qualitative behavior of such systems. The key point to keep in mind from the above discussion is that the potential function is necessarily a function of the state vector \( q \).

Returning now to our robot balancing problem, recall that balancing is inherently a task that involves the dynamics of the robot. For a typical unconstrained robot the dynamic equations assume the form

\[
M(q)\ddot{q} + b(q, \dot{q}) = \tau,
\]
where the vector of input joint torques $\tau \in \mathbb{R}^n$ is regarded as the control. Because the zero moment point depends on the dynamics, as such it contains terms explicitly dependent on $(q, \dot{q}, \ddot{q})$. Any potential function involving the ZMP will therefore depend explicitly on $\ddot{q}$ (as well as $q$ and $\dot{q}$). It is not difficult to see that the gradients of the ZMP potential function will likely turn out to be very complicated. More fundamentally, however, one can ask whether it is even meaningful to formulate a potential function dependent on $\ddot{q}$, given that the underlying system is only of second order (i.e. a differential equation in $\dot{q}$). Including a potential function gradient term in the control $\tau$ makes sense only when the underlying equations involve at least the third derivative of $q$.

Existing online mobile manipulator dynamic balancing control strategies for the most part include ZMP potential functions (or more precisely, their gradients, and in some cases even their Hessians) directly into the input joint torque $\tau$. Because of the difficulty in calculating gradients, these approaches are forced to settle for approximations of the gradient, e.g. by ignoring the derivative terms with respect to $\ddot{q}$, and using finite difference approximations to numerically calculate the gradients. Such approximations amount to ignoring the effects of the acceleration terms, and can have dire consequences on ZMP stability.

Many of the same issues discussed above arise in humanoid balancing. In [7], they are resolved in a physically and mathematically consistent way by invoking the invariance control framework [8]. Specifically, by further considering the motor dynamics and performing feedback linearization of the equations of motion, the underlying system is now recast into a third-order system (i.e. involving third-order derivatives of $q$); in this way, it now makes sense to formulate ZMP potential functions dependent on $\ddot{q}$. The main ideas are illustrated in [7] for a simple planar humanoid model maintaining dynamic balance.

With respect to rollover prevention, the specific contributions of this work are twofold. First, we extend the invariance control framework of [8] to the wheeled mobile manipulator dynamic balancing problem, possibly with a nonholonomic wheeled base. Our formulation makes the definition of a ZMP potential function mathematically consistent and physically plausible, and correctly accounts for changes in acceleration when determining the balance compensation torques.

As our second contribution, we derive efficient recursive algorithms for evaluating exact analytic gradients (with respect to $q$, $\dot{q}$, and $\ddot{q}$) of the ZMP potential function. Analytic gradients are critical to robust, real-time performance: previous studies [9] have shown that using finite difference approximations of these gradients, for example, not only considerably increases the computational burden, but the resulting cumulative numerical precision errors will more often than not lead to numerical instability and poor convergence. Our recursive algorithms can also be used in various optimization-based ZMP compensation algorithms of the type discussed in [10].

The specific balancing problem we address in this work assumes a single manipulator mounted on a wheeled (holonomic or nonholonomic) mobile base
maneuvering on a horizontal planar surface. Although we assume the mobile base’s motion is given at first, it is entirely straightforward to apply our approach to more general scenarios where the base motion is not specified, and in which further operational tasks may be given as in [3,5,6]. We describe the formulations for each of these two cases in Section 2.3.2. Our rollover prevention control law is applied to diverse case studies involving wheeled mobile manipulators, and for each case study we show that considerable improvements in performance can be obtained vis-à-vis existing approaches.

The paper is organized as follows. Section 2 derives the equations of motion for wheeled mobile manipulators, reviews the ZMP dynamic stability criterion, and formulates dynamic balancing within an invariance control framework. Section 3 presents the recursive algorithm for evaluating the gradients of the ZMP potential function, as part of the recursive dynamics computation. Section 4 examines several detailed rollover prevention case studies using our approach, and shows that our algorithm leads to marked improvements in convergence and computational performance over existing approaches, even preserving ZMP stability in scenarios where other approaches fail. The results reported in this paper are an extension and evolution of the preliminary work reported in [11,12].

2. Rollover prevention control law

2.1. Wheeled mobile manipulator model

Mobile manipulators can vary considerably depending on the drive mechanism of the mobile base as well as the kinematic structure of the manipulator arm. For the purposes of this paper, we consider two representative mobile platform designs, a (holonomic) omnidirectional platform, and a (nonholonomic) differential drive platform (see Fig. 1). For both platforms we assume that only a single open chain manipulator is mounted.

2.1.1. Kinematics

Assuming the mobile platform moves only in the horizontal $x$-$y$ plane, denote the kinematic configuration of the mobile base by

$$q_b = \begin{pmatrix} x_b \\ y_b \\ \theta \end{pmatrix}^T,$$

where $(x_b, y_b)$ denotes the Cartesian position of a reference frame $G_{xyz}$ attached to the mobile base, and $\theta$ denotes its orientation relative to the fixed frame $O_{xyz}$. Let $q_a \in \mathbb{R}^n$ denote the vector of joint positions for the manipulator arm, and define...
The velocity kinematics of the mobile manipulator can be expressed in the form

\[ \dot{q} = S(q) \eta \]

\[ \eta = \begin{pmatrix} \eta_b \\ \eta_a \end{pmatrix} \]

where \( \eta_b \) is defined as follows:
- If the mobile platform is holonomic, \( S(q) \) is the \((n+3) \times (n+3)\) identity matrix, and

\[ \eta_b = \begin{pmatrix} v_x \\ v_y \\ w_b \end{pmatrix}^T \]

where \( v_x = \dot{x}_b \), \( v_y = \dot{y}_b \), and \( w_b = \dot{\theta} \).
- If the mobile base is nonholonomic, the constraint, \( \dot{x}_b \sin \theta - \dot{y}_b \cos \theta = 0 \), is also in effect. In this case

\[ S(q) = \begin{pmatrix} \cos \theta & 0 & 0 \\ \sin \theta & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \in \mathbb{R}^{(n+3)\times(n+2)} \]

and

\[ \eta_b = \begin{pmatrix} v_b \\ w_b \end{pmatrix} \]

where \( v_b \) is the velocity in the forward direction and \( w_b \) is the angular velocity with respect to the z-axis.
The precise form of this velocity kinematics equation depends on the type of the mobile platform [13]. For example, if the mobile base of our model has two fixed wheels which the wheel axes are collinear and the reference frame, \( G_{xyz} \), is attached to the center of mass, assuming that no slip occurs in the forward direction,

\[
S(q) = \begin{pmatrix}
\overline{S}(q) & 0 \\
\mathbf{0} & I
\end{pmatrix},
\]

\[
\overline{S}(q) = \frac{r}{2b} \times \begin{pmatrix}
b \cos \theta - d \sin \theta & b \cos \theta + d \sin \theta \\
b \sin \theta + d \cos \theta & b \sin \theta - d \cos \theta \\
I & -I
\end{pmatrix}
\]

and

\[
\eta_s = \begin{pmatrix}
\omega_r \\
\omega_l
\end{pmatrix}^T,
\]

where \( r \) is the radius of the wheel, \( b \) is half the distance between the wheels, \( d \) is the distance from the center of mass to the wheel axis, and \( \omega_r \) and \( \omega_l \) are the angular velocities of the right and left wheels, respectively.

Here, we adopt the general form of (7) as the kinematics model of the nonholonomic mobile manipulator.

In this paper, we shall assume, as done elsewhere, that the mobile manipulator’s task is specified in terms of task coordinates \( \xi \in \mathbb{R}^m \) (typically \( \xi \) is taken to be the end-effector frame coordinates). The forward kinematics can then be expressed as \( \xi = f(q) \), and its differential version as

\[
\dot{\xi} = J(q)\eta,
\]

where \( J(q) = J(q)S(q) \) and \( J(q) = \frac{\partial f}{\partial q}(q) \in \mathbb{R}^{m \times (n+3)} \).

2.1.2. Rigid body dynamics

We now write down the dynamic equations for the mobile manipulator. The detailed derivation of the dynamic equations for general mobile manipulators is given in, e.g. [14].

For the nonholonomic case, the equations are of the form

\[
M(q)\ddot{q} + b(q, \dot{q}) = E(q)\tau + A(q)^T\lambda,
\]

where \( M(q) \in \mathbb{R}^{(n+3) \times (n+3)} \) is the mass matrix, \( b(q, \dot{q}) \in \mathbb{R}^{(n+3)} \) represents the Coriolis, centrifugal, and gravity terms, \( \tau \in \mathbb{R}^{(n+2)} \) is the vector of actuated
2.1.3. Motor dynamics

In order to be able to use the invariance control framework for dynamic balancing, the dynamics of the actuators also need to be considered. For the purposes of this paper, we shall consider, without loss of generality, a simplified first-order motor dynamics model of the form

$$\dot{\tau} + K\tau = u,$$

where $u$ represents the input currents, and the constant diagonal matrix $K$ corresponds to the time constants of the motor dynamics. Taking into consideration the previously derived rigid body dynamics model (12), the overall system dynamics can now be written in the form

$$\dot{x} = \begin{pmatrix} 0 & S(q) & 0 \\ 0 & 0 & I \\ 0 & 0 & 0 \end{pmatrix} x + \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} v, \quad x = \begin{pmatrix} q \\ \eta \end{pmatrix},$$

where

$$u(v) = \dot{M}(q)v + \ddot{M}(q)M^{-1}(\tau - \ddot{b}(q, \eta)) + K\tau + \ddot{\tilde{b}}(q, \eta).$$

If the mobile base is holonomic, then $S(q)$ is the identity matrix, and the system equations assume the same form as in [7]. Note that alternative models for the motor dynamics can also be included.

2.2. Balance condition

2.2.1. Zero moment point

Intuitively, the ZMP is the point on the ground about which the sum of all the moments of active forces (e.g. gravity, internal forces of the manipulator, exter-
nal forces from the environment) becomes zero [1]. The ZMP, denoted $P_{zmp}$, can be represented by:

$$P_{zmp} = \begin{pmatrix} \frac{M_y}{f_z} \\ -\frac{M_x}{f_z} \\ 0 \end{pmatrix},$$

where $(M_x, M_y)$ denotes the moments about the $x$ and $y$ axes generated by the resultant interaction force, and $f_z$ is the $z$-component of the resultant interaction force. Later in Section 3 we derive a recursive algorithm for computing the ZMP and its gradient.

2.2.2. Support polygon and balance condition

The support polygon associated with a mobile manipulator is the convex hull of all the points of the mobile manipulator (typically the points of contact between the wheels or casters and ground). If the ZMP lies within the support polygon, the mobile manipulator will not tip over. If $r_p$ denotes the radius of the largest circle inscribed inside the support polygon, then the ZMP balance requirement can be expressed conservatively as the following inequality constraint:

$$h(x) = (P_{zmp,1}(x))^2 + (P_{zmp,2}(x))^2 - r_p^2 \leq 0.$$  \hspace{1cm} (17)

Tighter constraints can of course be formulated, although the form of the inequality becomes more complex.

2.3. Balance control algorithm

This section describes our online control algorithm for dynamic balancing. To simplify the exposition we shall assume that the motion of the mobile base is given and the motion is tractable. To compare with other approaches, the additional scenario to execute the prescribed operational task is considered. In that case, the manipulator arm is kinematically redundant (i.e. possesses seven or more degrees of freedom), so that the manipulator must now use its available null motions to prevent rollover. Our approach can be extended in a straightforward manner even when these assumptions are relaxed.

2.3.1. Invariance control framework

As reported in [8], invariance control is a means of controlling systems subject to state inequality constraints. The main idea behind invariance control is to ren-
der a subset of the state space positively invariant by means of a switching controller. Under normal operating conditions, the system is controlled by a nominal controller that achieves the control objectives. When the system trajectory approaches the boundary of the admissible region of the state space, the control is switched to a corrective controller, which drives the system back into the interior of the admissible set; in this mode the system moves along the boundary of the admissible set. Control is switched back to the nominal controller as soon as the nominal control no longer results in any violation of state space constraints. We refer the reader to [8] for a more detailed analysis of the invariance control framework. Here, instead, we review the application to balance control of a planar biped robot standing on one leg [7].

The invariance control algorithm considers nonlinear control affine systems

\[ \dot{x} = f(x) + g(x)v, \quad x \in \mathbb{R}^n, \; v \in \mathbb{R}^m \]  

and the state constraints are defined by the zero sublevel set of a possible output

\[ y = h(x) \leq 0, \quad y \in \mathbb{R}. \]  

The functions \( f, g, \) and \( h \) are assumed to be sufficiently smooth to allow for the calculation of time derivatives of the output \( y \) up to the relative degree. For the special case of constraints with relative degree one, the maximal constraint admissible set is defined as

\[ G = \{ x \mid h(x) \leq 0 \} \]  

with a boundary \( \partial G = \{ x \mid h(x) = 0 \} \). The set \( G \) is invariant if and only if the control \( u \) satisfies

\[ \frac{\partial h}{\partial x} [f(x) + g(x)v] \leq 0 \]  

for all \( x \) on the boundary \( \partial G \). To keep the constraint set invariant, control is switched between nominal control and corrective control.

\[ v = \begin{cases} 
    v_{\text{nom}}, & y(x) < 0 \\
    v_{\text{nom}}, & y(x) \geq 0, \; \dot{y}(x, v_{\text{nom}}) < 0 \\
    v_{\text{corr}}, & y(x) \geq 0, \; \dot{y}(x, v_{\text{nom}}) \geq 0 
\end{cases} \]  

The corrective control input \( v_{\text{corr}} \) is derived from the equation

\[ \dot{y}(x, v) = 0 \iff \frac{\partial h}{\partial x} g(x)v = -\frac{\partial h}{\partial x} f(x). \]
Set by $\frac{\partial}{\partial x} g(x) = a^T$ and $\frac{\partial}{\partial x} f(x) = b$, the solution of the underdetermined linear equation $a^T v = b$ with minimal norm $\| v - v_{\text{nom}} \|$ can be obtained as follows:

$$v = (a^T)^\dagger b + N(a^T)v_{\text{nom}}, \quad (24)$$

where $(a^T)^\dagger$ is the Moore–Penrose pseudoinverse of $a^T$ and $N(a) = I - (a^T)^\dagger a^T$ is the null space of $a^T$. Other solutions are possible as referred in [7]. Based on the general formulation of the invariance control scheme, the balance control for the planar biped robot was introduced. The dynamic equations of motion can be transformed as defining $x = (q, \dot{q}, \ddot{q})$:

$$\dot{x} = \begin{pmatrix} \dot{q} \\ \dot{\dot{q}} \\ 0 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & I \end{pmatrix} v = f(x) + g(x)v. \quad (25)$$

The ZMP balance condition $y_1 \leq r_{\text{ZMP}}(x) \leq y_u$ can be restated by two constrained output functions:

$$y_1 = r_{\text{ZMP}}(x) - y_u \leq 0, \quad y_2 = -r_{\text{ZMP}}(x) + y_1 \leq 0. \quad (26)$$

Trajectory tracking of desired trajectories is realized as nominal behavior.

$$v_{\text{nom}} = \dot{q}_d + K_A \dot{e} + K_D e + K_P \ddot{e}, \quad (27)$$

where $q_d$ is the desired reference trajectory, $e = q_d - q$ represents tracking error, and $K_A, K_D$ and $K_P$ are control parameters which are selected $v_{\text{nom}}$ so as to result in asymptotic tracking of the reference trajectory. From (24), the corrective control input $v_{\text{corr}}$ is obtained with

$$a^T = \frac{\partial r_{\text{ZMP}}}{\partial \dot{q}} \quad \text{and} \quad b = -\frac{\partial r_{\text{ZMP}}}{\partial q} \ddot{q} - \frac{\partial r_{\text{ZMP}}}{\partial \dot{q}} \dot{q}. \quad (28)$$

We now show how to extend the invariance control framework to our mobile manipulator balancing problem. In contrast to the planar biped robot analyzed in [7], in which the planted foot is stationary, the mobile base moves with a prescribed motion trajectory. Moreover, the zero moment point now becomes two-dimensional rather than a scalar, leading to much more involved expressions for the ZMP and its gradient. We address the question of stability of the multiple control input system with the proposed switching controller. Our overall control framework is described in detail in Section 2.3.2.
2.3.2. Invariance controller design for wheeled mobile manipulator

The invariance balance controller for wheeled mobile manipulator is formulated for two types of dynamic balancing problems. In the first case, the motion of the mobile base is given in such a way that loss of dynamic stability becomes possible; the objective is to generate appropriate arm motions so as to maintain stability. In the second case, an operational task—in the form of a prespecified end-effector trajectory—is also assumed given, and any null motions of the arm are exploited to maintain dynamic stability. Although we assume the mobile base is given at first, it is straightforward to extend the formulation to more general scenarios where the base motion is not specified. Among the general scenarios, the second case is selected to compare our algorithm with the previous methods, since most balancing algorithms for mobile manipulators assume the operational task is given.

With the system definition (14) and the state constraint function (17), the switching controller can be designed with the same form as (22).

\[
v = \begin{cases} 
  v_{\text{nom}} & h(x) < 0 \\
  v_{\text{nom}} & h(x) \geq 0, \ h(x, v_{\text{nom}}) < 0 \\
  v_{\text{corr}} & h(x) \geq 0, \ h(x, v_{\text{nom}}) \geq 0 
\end{cases}
\] (29)

Here, \( \dot{h}(x, v) = \frac{\partial h}{\partial x} \dot{x} \) can be written as

\[
\dot{h}(x, v) = \frac{\partial h}{\partial q_a} \dot{q}_a + \frac{\partial h}{\partial \dot{q}_a} \ddot{q}_a + \frac{\partial h}{\partial q_b} \dot{q}_b + \frac{\partial h}{\partial \dot{q}_b} \ddot{q}_b + \frac{\partial h}{\partial \eta_b} \dot{\eta}_b + \frac{\partial h}{\partial \dot{\eta}_b} \ddot{\eta}_b,
\] (30)

with \( \eta = \begin{pmatrix} \eta_b^T & \dot{q}_a^T \end{pmatrix}^T \).

**Case 1. The motion of the mobile base is given**

In this case, \( \eta_b, \dot{\eta}_b, \ddot{\eta}_b \), are given and the goal is to evaluate an appropriate stabilizing joint trajectory by calculating \( q_a (= v_a) \). The motion of the mobile manipulator is then obtained from \( v = \begin{pmatrix} \eta_b^T & v_a^T \end{pmatrix}^T \).

- Nominal controller: The nominal controller is a general trajectory tracking controller as (27). If a reference joint angle trajectory for the manipulator is not given, the control that maintains the most stable posture (for example, if the arm is attached to the center of the mobile base, the zero-position can then be regarded as the most stable pose) can be specified to be the nominal trajectory.

\[
v_{a, \text{nom}} = \ddot{q}_a + K_A \ddot{\epsilon} + K_B \dot{\epsilon} + K_P \epsilon,
\] (31)
where \( \mathbf{q}_d(t) \in \mathbb{C}^3(0, T] \) is the desired reference trajectory for the arm (\( T \) is total operational time) and \( \mathbf{e}(t) = \mathbf{q}_d(t) - \mathbf{q}_a(t) \).

- **Corrective controller:** From (29) and (30), the \( \mathbf{v}_{a,\text{corr}} \) is selected such that

\[
\mathbf{a} \mathbf{v}_{a,\text{corr}} = \mathbf{b},
\]

where \( \mathbf{a} = \frac{\partial h}{\partial \mathbf{q}_a} \) and \( \mathbf{b} = -\left( \frac{\partial h}{\partial \mathbf{q}_a} \mathbf{q}_a + \frac{\partial h}{\partial \mathbf{q}_b} \mathbf{q}_b + \frac{\partial h}{\partial \mathbf{\eta}_b} \mathbf{\eta}_b \right) \). The objective of this case is to seek the control input to track the reference trajectory for the arm as well as to satisfy the stability condition. The appropriate solution is selected as:

\[
\mathbf{v}_{a,\text{corr}} = \mathbf{a}^\dagger \mathbf{b} + \mathbf{N}(\mathbf{a})\mathbf{v}_{a,\text{nom}},
\]

where the null space of \( \mathbf{a} \), \( \mathbf{N}(\mathbf{a}) = I - \mathbf{a}^\dagger \mathbf{a} \).

**Case 2. The operational task is specified**

Suppose a task \( \mathbf{\xi}_d(t) \in \mathbb{C}^3(0, T] \), where \( T \) is the total operational time, is given. For example, \( \mathbf{\xi}_d(t) \) can be regarded as the trajectory of the position of tip of the manipulator. In this case, we seek a control input \( \mathbf{v} \) that simultaneously satisfies the stability constraint and task requirement; the task requirement is reflected to the nominal controller.

- **Nominal controller:** The nominal controller that achieves the given task is of the form

\[
\mathbf{v}_{\text{task}} = \mathbf{J}^\dagger (\mathbf{\xi} - \mathbf{J} \dot{\mathbf{q}} - 2\mathbf{J} \ddot{\mathbf{q}}),
\]

where \( \mathbf{J} \) is the Jacobian matrix as defined in (9), \( \mathbf{J}^\dagger \) is the Moore–Penrose pseudoinverse, and \( \mathbf{\xi} = \mathbf{\xi}_d + \mathbf{K}_A (\mathbf{\xi}_d - \mathbf{\xi}) + \mathbf{K}_D (\mathbf{\xi}_d - \mathbf{\xi}) + \mathbf{K}_P (\mathbf{\xi}_d - \mathbf{\xi}) \). (34) is derived from twice differentiating the kinematics equation (9).

- **Corrective controller:** Similar to the first case, the control that corrects the motion to satisfy the stability constraint is calculated as

\[
\mathbf{v}_{\text{corr,stab}} = \mathbf{a}^\dagger \mathbf{b}, \quad \text{where} \quad \mathbf{a} = \frac{\partial h}{\partial \mathbf{\eta}} \quad \text{and} \quad \mathbf{b} = -\left( \frac{\partial h}{\partial \mathbf{q}_a} + \frac{\partial h}{\partial \mathbf{\eta}_a} \mathbf{\eta}_a \right),
\]

We now derive the corrective controller that takes into consideration the given task using the null space projection, or a weighted sum of the corrective control for stability compensation and the nominal control. The former corrective control is then derived as

\[
\mathbf{v}_{\text{corr}} = \mathbf{v}_{\text{corr,stab}} + \mathbf{N}(\mathbf{a})\mathbf{v}_{\text{task}}.
\]
Stability

The stability condition for the invariance control system with a single control input is suggested and proved in [8]. With the same perspective of that proof, a sufficient condition for existence of a Lyapunov function for our system with the switching control is also derived. It is assumed that the nominal controller stabilizes the system with respect to a desired trajectory and that a suitable Lyapunov function \( V(t,x) \) is available. For example, with the trajectory tracking control as the nominal control (31), the error dynamics can be written as a first-order linear system and the suitable Lyapunov function can be easily obtained with the proper control parameters.

If we can choose the corrective control input \( v_{corr} \) to satisfy

\[
\mathcal{L}_g V(t,x)v_{corr} \leq \mathcal{L}_g V(t,x)v_{nom},
\]

(37)

where \( \mathcal{L}_g V = \frac{\partial V(x)}{\partial x} g(x) \), the Lyapunov function \( V(t,x) \) is also valid for the switching control system. It is easily derived from the Lyapunov stability condition that \( V<0 \). Therefore, if the corrective control violates the Lyapunov stability condition, the corrective control is corrected again to be

\[
v_{corr} = c^T d + N(c)v_{nom},
\]

(38)

where \( c = \left( \frac{\partial V}{\partial \eta} \right) \) and \( d = \left( -\frac{\partial V}{\partial \eta} \eta + \frac{\partial V}{\partial \dot{\eta}} \dot{\eta} \right) \) are augmented matrices for the balance and stability conditions. Even though the stability constraint is added, the redundancy of the motion is still remained, so that the solution satisfying both constraints exists.

3. A recursive algorithm for analytic gradients of the ZMP

In this section, we present a recursive algorithm for evaluating analytic gradients of the ZMP potential function. The algorithm has its origins in the recursive algorithm for multibody dynamics presented in [15], as well as the recursive algorithm for differentiating the dynamics as presented in [9]. The gradient of the ZMP potential function requires differentiation of the mobile manipulator’s dynamics not only with respect to all the joint angles and velocities, but also joint accelerations. The recursive algorithm can be derived through a straightforward but involved calculation; due to space limitations we omit the details of the derivation, but provide the complete computational procedure.

Assuming the manipulator arm to be an \( n \)-link serial chain, each link frame \( k \) is assumed attached to the link center of mass. With respect to this choice of link frame, the \( 6 \times 6 \) inertial \( J_k \) of link \( k \) is defined from the rotational inertia matrix and the mass of the link. Let \( f_{k-1,k} \) be the displacement between link frame \( k - 1 \) and \( k \), where \( M_{k-1,k} \in SE(3) \) and \( S_k \in se(3) \). Define \( V_0 \in se(3) \) to be the
velocity of the base frame, expressed in base frame coordinates; in most typical circumstances it will be set to zero. Let \( g \in \mathbb{R}^3 \) represent the gravity vector expressed in terms of the base frame, and \( \dot{V}_0 = (0, g) \in se(3) \) be the acceleration assuming the base’s linear and angular accelerations are zero. \( F_{n+1} \) is the external moment and force applied to the tip, expressed in tip frame coordinates.

Given an element \( G \in SE(3) \), we also define the adjoint mapping \( Ad_G : se(3) \to se(3) \), as \( Ad_G(g) = GgG^{-1} \).

Given an element \( g_1 \in se(3) \), the Lie bracket \( ad_{g_1} : se(3) \to se(3) \) is given by \( ad_{g_1}(g_2) = g_1g_2 - g_2g_1 \). The two adjoint mappings are by definition linear, and can also be represented in matrix form. The corresponding dual adjoint mappings \( Ad^*_G : se^*(3) \to se^*(3) \) and \( ad^*_g : se^*(3) \to se^*(3) \) can also be represented in matrix form as described in [15]. With these definitions, the inverse dynamics of a serial chain can be computed recursively as follows:

- **Initialization**
  \[
  V_0, \dot{V}_0, F_{n+1}.
  \]

- **Forward recursion:** for \( k = 1 \) to \( n \)
  \[
  f_{k-1,k} = M_k e^{S_k q_k},
  \]
  \[
  V_k = Ad_{f_{k-1,k}} V_{k-1} + S_k \dot{q}_k,
  \]
  \[
  \dot{V}_k = Ad_{f_{k-1,k}} \dot{V}_{k-1} + S_k \ddot{q}_k + ad_{\dot{V}_k} S_k \dot{q}_k.
  \]

- **Backward recursion:** for \( k = n \) to \( 0 \)
  \[
  F_k = Ad_{f_{k,k+1}}^{-1} F_{k+1} + J_k \dot{V}_k - ad_{\dot{V}_k} J_k V_k,
  \]
  \[
  \tau_k = S_k^T F_k.
  \]

Based on the above, we now derive a corresponding recursive algorithm for calculating the gradients of the ZMP. Recall that the ZMP is defined to be a function of the inverse dynamics, with joint angles, joint velocities, and joint accelerations as the explicit variables. The ZMP also varies with the velocity and acceleration of the mobile base. We shall therefore require the partial derivatives about the base’s velocity and acceleration, as well as with respect to the joint angles, joint velocities, and joint accelerations. The joint variables are written as \( q, \dot{q}, \ddot{q} \), while the base’s kinematics are represented in terms of the velocity \( V_0 \) and acceleration \( \dot{V}_0 \). The formulation is represented in Table 1.
Table 1.
Recursive algorithm for calculating gradients of inverse dynamics for WMM

<table>
<thead>
<tr>
<th>Initialization</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\frac{\partial F_0}{\partial q_0} , \frac{\partial F_0}{\partial \dot{q}_0}, \frac{\partial F_0}{\partial \ddot{q}<em>0} = 0</em>{6\times 0}$</td>
</tr>
<tr>
<td>$\frac{\partial F_0}{\partial q_0} , \frac{\partial F_0}{\partial \dot{q}_0}, \frac{\partial F_0}{\partial \ddot{q}<em>0} = 0</em>{6\times 0}$</td>
</tr>
<tr>
<td>$\frac{\partial F_0}{\partial q_0} , \frac{\partial F_0}{\partial \dot{q}_0}, \frac{\partial F_0}{\partial \ddot{q}<em>0} = 0</em>{6\times 0}$</td>
</tr>
<tr>
<td>$\frac{\partial F_0}{\partial q_0} , \frac{\partial F_0}{\partial \dot{q}_0}, \frac{\partial F_0}{\partial \ddot{q}<em>0} = 0</em>{6\times 0}$</td>
</tr>
<tr>
<td>$\frac{\partial F_0}{\partial q_0} = I_{6\times 6}$</td>
</tr>
<tr>
<td>$\frac{\partial F_0}{\partial q_0} , \frac{\partial F_0}{\partial \dot{q}_0}, \frac{\partial F_0}{\partial \ddot{q}<em>0} = 0</em>{6\times 0}$</td>
</tr>
</tbody>
</table>

Forward recursion: for $k = 1$ to $n$

$\frac{\partial F_k}{\partial q_0} = Ad_{f_{k-1}} \frac{\partial F_{k-1}}{\partial q_0} - \delta_{jk} \partial_{f_{k-1}} V_k$

$\frac{\partial F_k}{\partial \dot{q}_0} = Ad_{f_{k-1}} \frac{\partial F_{k-1}}{\partial \dot{q}_0} - \Delta_{jk} \partial_{f_{k-1}} \dot{V}_k$

$\frac{\partial F_k}{\partial \ddot{q}_0} = Ad_{f_{k-1}} \frac{\partial F_{k-1}}{\partial \ddot{q}_0} - \Delta_{jk} \partial_{f_{k-1}} \ddot{V}_k$

$\frac{\partial F_k}{\partial \dot{q}_0} = Ad_{f_{k-1}} \frac{\partial F_{k-1}}{\partial \dot{q}_0} + \Delta_{jk} \partial_{f_{k-1}} \dot{V}_k - \partial_{f_{k-1}} S_k$

Backward recursion: for $k = n$ to $0$

$\frac{\partial F_k}{\partial q_0} = Ad_{f_{k+1}} \left( \delta_{j,k+1} \partial_{f_{k+1}} F_{k+1} + \partial_{f_{k+1}} \dot{S}_{k+1} \right) + J_k \frac{\partial F_k}{\partial q_0} - \partial_{f_{k-1}} J_k \dot{V}_k - \partial_{f_{k-1}} J_k \ddot{V}_k$

$\frac{\partial F_k}{\partial \dot{q}_0} = Ad_{f_{k+1}} \left( \delta_{j,k+1} \partial_{f_{k+1}} \dot{F}_{k+1} + \partial_{f_{k+1}} \dot{\dot{S}}_{k+1} \right) + J_k \frac{\partial F_k}{\partial \dot{q}_0} - \partial_{f_{k-1}} J_k \dot{V}_k - \partial_{f_{k-1}} J_k \ddot{V}_k$

$\frac{\partial F_k}{\partial \ddot{q}_0} = Ad_{f_{k+1}} \left( \delta_{j,k+1} \partial_{f_{k+1}} \ddot{F}_{k+1} + \partial_{f_{k+1}} \ddot{S}_{k+1} \right) + J_k \frac{\partial F_k}{\partial \ddot{q}_0} - \partial_{f_{k-1}} J_k \dot{V}_k - \partial_{f_{k-1}} J_k \ddot{V}_k$
Choosing the z-axis of the body-fixed vehicle coordinate system to be perpendicular to ground, the ZMP $P_{zmp}$ can then be obtained as follows:

$$P_{zmp} = \left( \begin{array}{c} -F_{0.2}/F_{0.6} \\ F_{0.1}/F_{0.6} \\ 0 \end{array} \right),$$

(45)

where $(F_{0.1}, F_{0.2})$ is the $(x,y)$ component of the moment, and $F_{0.6}$ is the z-component of the ground interaction force. Given these components, the ZMP follows directly from the inverse dynamics, and the gradient of the ZMP can be evaluated as follows:

$$\frac{\partial P_{zmp,1}}{\partial p_i} = -\frac{\partial F_{0.2}}{\partial p_i} F_{0.6} + F_{0.2} \frac{\partial F_{0.6}}{\partial p_i},$$

(46)

$$\frac{\partial P_{zmp,2}}{\partial p_i} = \frac{\partial F_{0.1}}{\partial p_i} F_{0.6} - F_{0.1} \frac{\partial F_{0.6}}{\partial p_i},$$

(47)

where $p = [q^T \dot{q}^T \ddot{q}^T V_0^T V_\theta^T]$. If the mobile platform is holonomic, $V_\theta = [0 0 w_b v_x v_y 0]^T$, and if the platform is nonholonomic, then $V_\theta = [0 0 w_b v_b 0 0]^T$.

4. Case studies

The performance of our proposed rollover prevention algorithm is now evaluated through several case studies with two robot models.

4.1. Simulation models

The robot models we use for simulations consist of two types of manipulators. The first is a manipulator with three degrees of freedom, in which all three rota-

<table>
<thead>
<tr>
<th>Link</th>
<th>Length (height(m), radius(m))</th>
<th>Mass (kg)</th>
<th>Rotational axis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base</td>
<td>(0.4, 0.2)</td>
<td>5.0</td>
<td>None</td>
</tr>
<tr>
<td>Link1</td>
<td>(0.3, 0.03)</td>
<td>2.0</td>
<td>(0, -1, 0)</td>
</tr>
<tr>
<td>Link2</td>
<td>(0.3, 0.03)</td>
<td>2.0</td>
<td>(0, -1, 0)</td>
</tr>
<tr>
<td>Link3</td>
<td>(0.3, 0.03)</td>
<td>2.0</td>
<td>(0, -1, 0)</td>
</tr>
<tr>
<td>Payload</td>
<td>(0.08)</td>
<td>4.0</td>
<td>None</td>
</tr>
</tbody>
</table>

Fig. 2. The first model.
tional joint axes are parallel. The second model consists of a manipulator with five degrees of freedom whose kinematic structure is similar to that of the human arm. The kinematic and inertial parameters of our models are presented in Tables 2 and 3. The topologies of the robots are displayed in Figs. 2 and 3, respectively. All simulations are performed in Matlab v. 7.10 running on 2.67 GHz Core2 Quad desktop computer. For both models the stability region is set to be a maximum circle inscribed inside the triangle formed by the rear two wheels and the front caster. The radius of the circle is measured as 0.19 m, with the center of the circle at the origin of the mobile base, i.e. $r_p = 0.19$ in (17).

### 4.2. Experiments

#### 4.2.1. Stability compensation for linearly accelerated moving base

Here, the linear acceleration is given in terms of a sine function profile. The resulting velocity of the mobile platform is also given as a sine function profile. The maximum velocity is set to 8 m/s. The other required term is derived by differentiation of the given acceleration profile. The angular velocity is set to zero. The objective of the nominal controller is to keep the arm in an upward straight pose (its zero-position), which happens to be the most stable pose when the mobile platform is not moving.

Fig. 4 depicts the ZMP trajectories for the nominal and compensated motions. The two parallel dashed lines on the graph of the ZMP trajectories denote the boundary of the support polygon. In the original motion, the ZMP goes out of the stable region periodically because of the sine profile of the acceleration of the mobile base; clearly these situations are not what we desire. In the compensated motion, the ZMP trajectory is concentrated within the stable region simply by manipulating the arm. The compensated angle trajectories of the three joints in the figure show that the compensated motion is characterized by a primary motion of the first joint. This is because the first joint is the most effective way to compensate for ZMP stability. In a similar fashion to the way a human leans

back at the waist when a bus suddenly accelerates, it is observed that the robot also leans in the direction opposite to the direction of acceleration of the vehicle.

4.2.2. Stability compensation when holding a payload

Initially we assume that the vehicle is static, and the nominal controller seeks to move the arm from an arbitrary initial pose to a pose in which the arm points vertically up. In the resulting compensated motion, the manipulator slowly lifts the load so as to maintain balance. Had the robot lifted faster, ZMP stability may potentially be violated and the robot subject to tipping over. Of course, if time is limited, the robot may not be able to raise the load within the given time. If the weight of the load is increased to 6 kg, the invariance balance controller fails to correct the ZMP outside the support polygon. However, the algorithm can easily be modified for the case where the mobile base is free, and from Equation (35), the controller can now calculate the required velocity and acceleration of the mobile base in order to compensate for stability during lifting of the heavy load. By moving the base, the robot can now safely lift even heavy loads within the prescribed time. Fig. 5 shows the compensated ZMP trajectories and the resultant base’s movement for the case where the mobile base is free.

For this example, we compare our method with the gradient descent method-based task consistent control framework described in [6]. In that method, the gradient of the ZMP potential function is calculated only with respect to joint angle (i.e. using $\dot{q} = -\frac{\partial V(q, \dot{q})}{\partial q}$). Though the paper purports to use the Hessian of the
ZMP function, it is in fact only an approximation. Moreover, derivative terms with respect to the joint acceleration are ignored.

We now show explicitly the drawbacks of the proposed control scheme and the approximated gradient used. Because the given task is formulated on the configuration space so that the task does not have a null space, we use the weighted sum of control inputs to formulate the task consistent control framework. Even though we adjust the parameters (e.g. control gains and weights) or change relative priorities priority between the balance constraint and task requirement, we fail to satisfy simultaneously the stability condition and the requirement that the given task is executed on time as shown in Fig. 6. The thick straight line in Fig. 6(d) also denotes the boundary of the stable region. Although the first joint moves backward to assume a more stable pose, the resultant control input causes the ZMP constraint function \( h(P_{zmp}) \) to increase after some time has elapsed. The motion moves toward a stable posture because the term \( \frac{\partial h(q, \dot{q}, \ddot{q})}{\partial q} \) is considered, but there still remains the possibility of falling down while moving. Also, we observe that with the algorithm, the robot cannot sufficiently utilize the mobile base to balance as does our method.

4.2.3. Stability compensation for following a crooked road

When the vehicle maneuvers on a highly curved path, it is easy to rollover even when it moves with a linearly constant velocity. The mobile base’s movement is
given in such a way that the linear velocity is constant and the angular velocity is varied as the robot moves along a curve as shown in Fig. 7(e). Fig. 7(a) and (b) show the original ZMP trajectory and the compensated ZMP trajectory. Fig. 7(c) shows that the compensated ZMP trajectory remains within the support polygon. If there are insufficient available joints to correct for the ZMP trajectory in the direction of acceleration, the robot then manipulates the z-axis rotational joints. Similar to the result of the first case study, the compensated motion is characterized by a primary motion of the third joint that is the most effective way to compensate for ZMP stability (Fig. 7(d)). The other methods proposed in the literature cannot be applied to this situation.

4.2.4. Stability compensation for a given operational task

Using the switching controller in the second case, we can solve task consistent stability prevention problems as in [3–6]. The task is specified in the form of a trajectory for the tip of a manipulator, \( \xi_d(t) \), with the trajectory is prescribed to follow a straight line. Here, we compare our method with the control scheme-based methods using the approximated gradient of the ZMP potential function [5,6]. Both these methods ignore the derivative terms with respect to the joint acceleration.

We calculate the gradient (desired null velocity) and the Hessian (desired null acceleration) of the ZMP function as suggested in these two papers and using [16], obtain the motion to minimize the tracking error for the desired null values while satisfying the first task requirement. The result is shown in Fig. 8. Using our method, the manipulator follows the given trajectory within an acceptable error, and the ZMP always remains inside the stable region. However, with the same task, the previous two methods violate the stability condition in the early
phase, although it gradually improves until finally assuming a more stable motion. If the stability condition has a higher order of priority than the given task, the robot then fails to follow the given trajectory.

In the second simulation involving following the bell-shaped trajectory, we investigate the enhanced method [6] of two approximated methods, and note that the difference in performance with our method is even more pronounced. Even by adjusting the gains, changing the task priorities, or formulating a switching control law, the stability condition is not satisfied for some time as evident from Fig. 9.

4.3. Discussion

The above simulations demonstrate the utility of our balance control algorithm. With a slight change of formulation, the controller does not break out of the sta-

![Fig. 7. Stability compensation for driving on a crooked road: (a)-(b) The original ZMP trajectory and the compensated ZMP trajectory; (c) The support polygon and ZMP trajectory; (d) The nominal motion and the compensated motion; (e) The final simulation result.](image)

![Fig. 8. Comparison with invariance control and approximated gradient method for tracking a straight line: (a) The ZMP trajectory; (b) Distance from the boundary of the support polygon; (c) Task errors.](image)
bility region during execution of the task. Also, the recursive calculation of the gradient of the ZMP and the extended invariance control framework improves the computational performance and the accuracy of the balance control. Our numerical experiments support our claims that the accuracy of the analytic calculation of the gradient leads to more robust control. The computational performance for the proposed scenarios was enhanced from about 22 frames per second (fps) on average when using the forward finite difference method (which is the simplest numerical method) to about 52 fps on average when we use the recursive calculation. As the robot model and the scenario become more complex, the difference in the computational performance increases as shown in Table 4. However, the balance controller fails when the values of $\frac{\partial h}{\partial q}$ and (35) are sufficiently small so as to be unable to find a feasible control input to satisfy the invariance condition as pointed out in [7]. For example, if the mobile base is constantly accelerated in the same direction, at first the arm bends back to balance, but in the long run, the robot ultimately will tip over. The same is also true for human balancing; typically the human will grab onto something to maintain balance. If the human were aware of this situation in advance, the human could assume a stable pose in preparation for this situation, e.g. when riding a bike around a corner, the rider may lean into the turn to maintain balance.

Table 4.
Comparison of computation speed with finite difference method and recursive calculation

<table>
<thead>
<tr>
<th>Case studies</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computation speed (fps) (Matlab)</td>
<td>Forward finite difference</td>
<td>36.31</td>
<td>34.78</td>
<td>16.02</td>
<td>14.85</td>
</tr>
<tr>
<td></td>
<td>Recursive calculation</td>
<td>84.57</td>
<td>77.53</td>
<td>37.55</td>
<td>35.24</td>
</tr>
</tbody>
</table>

Fig. 9. Comparison with invariance control and approximated gradient method for tracking a curved line: (a) The ZMP trajectory; (b) Distance from the boundary of the support polygon; (c) Task errors.
In our balance controller, the term $\frac{\partial h}{\partial q}$ can be used to find a stable pose: we can make the controller track a stable feed-forward motion which is computed offline as the nominal control input, and compensate the motion in real-time using our feedback controller.

5. Conclusions

This paper has presented a rollover control law for wheeled mobile manipulators based on the invariance control framework, and that makes use of recursively calculated analytic gradients of the mobile manipulator’s ZMP. Because the ZMP necessarily involves the dynamics, and explicitly depends on the joint accelerations, to be consistent any ZMP potential function should be formulated for a third-order system (that is, the underlying state equations involve third-order derivatives of the state). In our approach, we extend the invariance control framework to the mobile manipulator balancing problem, and take the motor dynamics into consideration when formulating the governing equations of motion.

Our controller also relaxes many of the assumptions made in existing approaches. Robustness is enhanced through the use of exact gradient information, and our controller also correctly accounts for derivatives of the ZMP function with respect to accelerations. Several numerical experiments have been undertaken to demonstrate the improved robustness of our proposed controller over existing approaches. We expect that the general methods presented here can also be extended to other mobile robot systems, e.g. humanoids and other legged robots.

Acknowledgment

This research was supported in part by the KIST Center for Intelligent Robotics, the Advanced Intelligent Manipulation Center, ROSAEC ERC, KSPO Sports Technology Initiative, SNU-IAMD, and the SNU Engineering Research Foundation.

References


About the Authors

Sohee Lee received the BS degree in Earth and Environmental Sciences in 2003 and the PhD degree in Mechanical and Aerospace Engineering in 2011, both from Seoul National University. Her research interests are in mobile manipulation, dynamics-based motion planning and control, and humanoid movement generation. In 2011, she was a postdoctoral researcher at Humanoid Robot Research Center, Korean Advanced Institute of Science and Technology, and currently is working at Intelligent Autonomous Systems group in Technische Universität München.
Marion Leibold received her diploma degree in Applied Mathematics from Technische Universität München in 2002. She received the PhD degree from the Institute of Automatic Control Engineering, Faculty of Electrical Engineering, Technische Universität München in 2007. Currently she is a postdoctoral researcher at the Institute of Automatic Control Engineering of TU München. Her research interests include hybrid dynamical systems, optimal control, and legged robots.

Martin Buss received the diploma engineer degree in Electrical Engineering in 1990 from the Technical University Darmstadt, Germany and the doctor of Engineering degree in Electrical Engineering from the University of Tokyo, Japan, in 1994. In 2000 he finished his habilitation in the Department of Electrical Engineering and Information Technology, Technische Universität München, Munich, Germany. In 1988, he was a research student at the Science University of Tokyo, Japan, for one year. As a postdoctoral researcher he stayed with the Department of Systems Engineering, Australian National University, Canberra, Australia, in 1994/5. From 1995 to 2000, he has been a senior research assistant and lecturer at the Institute of Automatic Control Engineering, Department of Electrical Engineering and Information Technology, Technical University Munich, Germany. He has been appointed full professor, head of the control systems group, and deputy director of the Institute of Energy and Automation Technology, Faculty IV–Electrical Engineering and Computer Science, Technical University Berlin, Germany, from 2000 to 2003. Since 2003, he is a full professor (chair) at the Institute of Automatic Control Engineering, Technische Universität München, Germany. Since 2006, he is the coordinator of the DFG Excellence Research Cluster “Cognition for Technical Systems (CoTeSys).” His research interests include automatic control, mechatronics, multimodal human-system interfaces, optimization, nonlinear, and hybrid discrete-continuous systems.

Frank Chongwoo Park received the BS degree in electrical engineering and computer science from MIT in 1985 and the PhD degree in applied mathematics from Harvard University in 1991. From 1991 to 1994, he was an assistant professor of mechanical engineering at the University of California, Irvine. Since 1995, he has been with the School of Mechanical and Aerospace Engineering at Seoul National University, Seoul, Korea, where he is currently a professor and department chair of mechanical engineering. He is also an adjunct professor of interactive computing at the College of Computing, Georgia Institute of Technology. His research interests are in robotics, mathematical systems theory, and related areas of applied mathematics. He is a 2007–2008 IEEE Robotics and Automation Society Distinguished lecturer, and served as a senior editor of the IEEE Transactions on Robotics, Springer’s Handbook of Robotics and Tracts in Advanced Robotics, and as a secretary of the Robotics and Automation Society from 2008 to 2009.