Robotics: Science and Systems 2015
Rome, Italy, July 13-17, 2015

Quantitative Measures of a Robot's Ability to Balance

Roy Featherstone
Dept. Advanced Robotics, Istituto Italiano di Tecnologia
via Morego 30, 16163 Genova, Italy. email: roy.featherstone at ieee.org

***Abstract*—This paper shows how to define quantitative measures of a robot's ability to balance itself actively on a single point of support. These measures are expressed as ratios of velocities, and are called velocity gains. This paper builds on earlier work in this area by showing how these gains can be defined and calculated for the case of a general planar robot balancing on a general rolling-contact point in the plane, and the case of a general spatial robot balancing on a general rolling-contact point in 3D space. The case of balancing on a contact area with compliance is also considered. The paper concludes with two examples showing how to use velocity gains in the design of a triple pendulum and the analysis of a hydraulic quadruped.**

I. Introduction

The ability to balance actively on a single point or line of support, or on a small area of support, is something that we take for granted in the natural world [5, 12]. Robots used to lack this skill, but this is no longer the case. In the commercial realm, there is the Segway Personal Transporter [9] and an increasing number of low-cost telepresence robots such as the Double [1]. In the research laboratories there is the Ballbot [7], the AcroBOX [8], the Cubli [4], the spatial double inverted pendulum [12], the Acrobot [6], and many more.

Robotic balancing presents us with an interesting control problem, but this has led to an imbalance in the way the topic has been treated: too much attention on the control system and not enough on the plant. For example, how good are the above robot mechanisms at balancing? Could they be better? This is a separate issue from the performance of the control system, and has received relatively little attention.

In an attempt to redress this imbalance, Featherstone [3] studied the physical ability to balance of a planar double pendulum, and devised a quantitative measure of this particular robot's ability to balance, which he called the *velocity gain*. It was defined as the ratio of a step change in the velocity of the actuated joint to a step change in the velocity of the centre of mass relative to the support point, both steps being caused by an impulse at the actuated joint. In effect, the velocity gain measures the amount of active movement that is physically necessary in order to correct a balance error of a given magnitude: the higher the gain, the less movement is required.

This paper builds on the work of Featherstone by extending the definition of velocity gain, first to a general planar mechanism and then to a general spatial mechanism, so that a velocity gain can be defined for almost any system that balances on a single point of support, which can be a sharp point or a general rolling contact. Instructions for calculating these gains are presented, and the case of balancing on an area contact is also considered. The paper concludes with two examples, which illustrate the use of velocity gains in the design of a planar triple pendulum and the analysis of a hydraulic quadruped.

II. Velocity Gain

Figure 1 shows a simple planar self-balancing robot consisting of a lower link (link 1), which makes contact with a supporting surface (the ground) at a single point, and is connected to an upper link (link 2) via an actuated revolute joint (joint 2, with joint variable q_2). For the purpose of calculating the velocity gain, it is assumed that the ground is flat and horizontal, and that the lower link rolls without slipping or losing contact with the ground.

Although the lower link has been drawn as a leg with a foot, it could just as easily be a wheel. Any shape is acceptable, including sharp points, so long as the portion of the link that makes contact with the ground is strictly convex, so that it always makes contact with the ground at a single point.

This rolling contact between the lower link and the ground is effectively a one-degree-of-freedom (DoF) joint, which can be characterized by a single joint variable denoting an angle (joint 1, with joint variable q_1 in the diagram). If the robot is balancing on a sharp point then the rolling contact simplifies to a revolute joint at the contact point.

The vector $\boldsymbol{c}$, having components c_x and c_y, gives the position of the robot's centre of mass (CoM) relative to the contact point; and the angle ϕ gives the direction of $\boldsymbol{c}$ relative to the x axis. The vector $\boldsymbol{b}$ is described later.

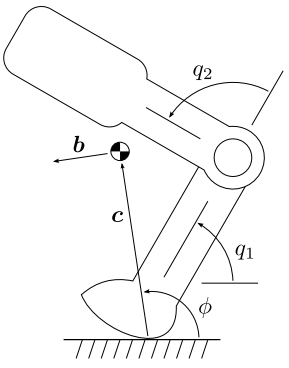

Fig. 1. Definition of velocity gain (based on Fig. 1 in Featherstone [3])

The objective of a balance controller is either to drive c_x to zero, or to drive ϕ to 90°, by motions of the actuated joint. Therefore, from the point of view of the balance controller, the robot can be regarded as a plant in which the output is c_x or ϕ and the input is q_2. So one could define the gain as

$$G_{\mathrm{v}} = \frac{\partial c_x}{\partial q_2} \qquad \text{or} \qquad G_\omega = \frac{\partial \phi}{\partial q_2}\,. \tag{1}$$

The larger the magnitudes of these gains, the greater the influence of the actuated joint over the motion of the CoM, and therefore the better the robot is at balancing.

However, it is not immediately obvious how to evaluate the partial derivatives in Eq. 1, so a better way to express the gains is

$$G_{\mathrm{v}} = \frac{\Delta \dot{c}_x}{\Delta \dot{q}_2} \qquad \text{and} \qquad G_\omega = \frac{\Delta \dot{\phi}}{\Delta \dot{q}_2}\,, \tag{2}$$

in which the quantities $\Delta\dot{c}_x$, $\Delta\dot{\phi}$ and $\Delta\dot{q}_2$ all denote step changes in the velocities $\dot{c}_x$, $\dot{\phi}$ and $\dot{q}_2$ caused by an impulse applied at the actuated joint. As the gains are now expressed as ratios of velocities, they are called *velocity gains*.

The subscripts v and ω indicate linear and angular velocity gains, respectively. (Featherstone [3] defines only the angular velocity gain, but calls it G_{v}.) The choice of which gain to use depends on circumstances. For example, G_ω is not defined when $|\boldsymbol{c}| = 0$, so it is not appropriate in situations where the CoM could coincide with the contact point. Likewise, G_{v} does not distinguish between $c_y > 0$ and $c_y < 0$, and can run into difficulty when $c_y = 0$, so it might not be appropriate in applications like swing-up control, where the CoM starts below the support point but must finish above it.

Calculation Method

The equations below show how to calculate G_{v} and G_ω for the special case of a sharp contact point, in which the robot simplifies to a planar 2R mechanism. The case of general rolling contact is covered in Section V. To calculate the velocity gains, we assume that the following items are available:

1) a data structure rob containing a dynamic model of the robot (kinematic and inertia parameters, connectivity and joint type data);
2) a function $\boldsymbol{H} = \mathrm{jsim}(rob, \boldsymbol{q})$ to calculate the joint-space inertia matrix $\boldsymbol{H}$ of the mechanism described by rob in the configuration described by the vector of joint position variables $\boldsymbol{q}$; and
3) a function $[\boldsymbol{c}_o, \dot{\boldsymbol{c}}_o] = \mathrm{cmpv}(rob, \boldsymbol{q}, \dot{\boldsymbol{q}})$ to calculate the position and velocity, $\boldsymbol{c}_o$ and $\dot{\boldsymbol{c}}_o$, of the CoM expressed in base coordinates, given rob, $\boldsymbol{q}$ and a vector of joint velocity variables $\dot{\boldsymbol{q}}$.

Note that $\boldsymbol{c}_o$ and $\dot{\boldsymbol{c}}_o$ give the position and velocity of the CoM relative to the origin of base coordinates, whereas $\boldsymbol{c}$ and $\dot{\boldsymbol{c}}$ are the position and velocity relative to the support point. In Section V these quantities will be different. However, for the special case of balancing on a sharp point we can make the origin of base coordinates coincide with the contact point so that $\boldsymbol{c}_o = \boldsymbol{c}$ and $\dot{\boldsymbol{c}}_o = \dot{\boldsymbol{c}}$.

Given these functions, the gains can be calculated in two steps. In the first step, we use the impulsive equation of motion to calculate the velocity step changes caused by an impulse ι (iota) at the actuated joint. The equation is

$$\begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix} \begin{bmatrix} \Delta\dot{q}_1 \\ \Delta\dot{q}_2 \end{bmatrix} = \begin{bmatrix} 0 \\ \iota \end{bmatrix}. \tag{3}$$

Setting $\Delta\dot{q}_2 = 1$ gives

$$\Delta\dot{q}_1 = \frac{-H_{12}}{H_{11}}\,. \tag{4}$$

In the second step, we calculate $\boldsymbol{c}$ and $\Delta\dot{\boldsymbol{c}}$ using

$$[\boldsymbol{c},\, \Delta\dot{\boldsymbol{c}}] = \mathrm{cmpv}(rob, \boldsymbol{q}, \Delta\dot{\boldsymbol{q}}) \quad \text{with} \quad \Delta\dot{\boldsymbol{q}} = \begin{bmatrix} \Delta\dot{q}_1 \\ 1 \end{bmatrix}. \tag{5}$$

The gains are then calculated as follows:

$$G_{\mathrm{v}} = \Delta\dot{c}_x \qquad \text{and} \qquad G_\omega = \Delta\dot{\phi} = \frac{\boldsymbol{b} \cdot \Delta\dot{\boldsymbol{c}}}{|\boldsymbol{c}|} \tag{6}$$

where $\boldsymbol{b}$ is a unit vector at right-angles to $\boldsymbol{c}$ in the direction of increasing ϕ, and is given by $\boldsymbol{b} = [-c_y\ c_x]^{\mathrm{T}}/|\boldsymbol{c}|$.

Some Properties

Several properties of G_ω are listed in Featherstone [3], including the fact that it is a ratio of two angular velocities, and therefore a dimensionless quantity. This means that G_ω is independent of both the overall mass and the overall size of the robot. G_{v} is also independent of overall mass, but not overall size. The mass-independence property carries over into 3D, and also to the case of robots with more than one actuator. The scale-invariance property of G_ω also carries over into 3D, but it carries over into the multiple-actuator case only if all of the actuators used for balancing have joint variables that are angles.

As mentioned earlier, G_ω is not defined when $\boldsymbol{c} = \boldsymbol{0}$. It can also be shown that neither G_ω nor G_{v} is defined where there is a step change in the local curvature at the contact point, since a step change in curvature causes a step change in the values of the velocity gains. If the gains have opposite signs on each side of the step, then it is impossible to balance the mechanism at that configuration because the controller has two ways to correct a balance error in one direction and no way to correct an error in the other direction. Balancing is also a practical impossibility within a neighbourhood of a continuous zero crossing.

Featherstone [3] also points out that mass can be redistributed within a mechanism in certain ways without altering its equation of motion, as explained in Featherstone [2, §9.7]. This fact, along with the invariances mentioned above, means that if a designer can find a single mechanism with a desirable balancing property, then a multi-parameter family of other mechanisms with the same property can immediately be generated simply by varying the parameters and combinations of parameters that have no effect on the velocity gains.

Velocity gain resembles the CoM Jacobian defined in Sugihara et al. [11]. The difference is that the CoM Jacobian is a mapping from a complete joint velocity vector to the motion

of the CoM, whereas velocity gain is a mapping from active joint velocity to the motion of the CoM, in which the passive joint motion is deduced from the active joint motion.

Area Contact

Although velocity gain has been defined for a robot making a single point contact with the ground, it can easily be extended to robots that make an area contact if any of the following are true: (1) the ground is elastic, (2) the foot or leg is elastic, or (3) there is an elastic element between the actuator and the joint. If any of these are true then we can modify the definition of velocity gain to use the centre of pressure in place of the contact point. If item 3 is the only one that is true then it is necessary to use the velocity of the actuator in place of $\dot{q}_2$.

III. Extension to Multiple Actuated Joints

We now consider a general planar mechanism having n DoF, which includes the passive rolling contact with the ground. The vector of position variables is now $\boldsymbol{q} = [q_1 \ \boldsymbol{q}_\mathrm{a}^\mathrm{T}]^\mathrm{T}$, where q_1 is the angle of the rolling contact and $\boldsymbol{q}_\mathrm{a}$ is the vector of position variables for the internal motion freedoms of the mechanism, which are assumed to be fully actuated. If the mechanism is a kinematic tree then $\boldsymbol{q}_\mathrm{a}$ is the vector of joint position variables; otherwise it is a vector of generalized coordinates from which the joint variables can be calculated, in which case $\boldsymbol{H}$ is a generalized inertia matrix. Instructions for calculating such matrices can be found in several textbooks. We shall assume that a function $\mathrm{gim}(rob, \boldsymbol{q})$, analogous to jsim, is available to calculate this matrix.

From a balancing point of view, the fundamental difference between a single actuated freedom and multiple actuated freedoms is that in the latter case the control system has the freedom to choose what movements to make in order to achieve or maintain balance. This choice affects the velocity gain, since some movements have more influence on the CoM than others. Indeed, it will usually be possible to find $n-2$ directions of motion in which the velocity gain is zero. To cater for this new possibility, we modify Eq. 6 as follows:

$$G_\mathrm{v}(\Delta\dot{\boldsymbol{q}}_\mathrm{a}) = \Delta\dot{c}_x\,, \qquad G_\omega(\Delta\dot{\boldsymbol{q}}_\mathrm{a}) = \Delta\dot{\phi} = \frac{\boldsymbol{b}\cdot\Delta\dot{\boldsymbol{c}}}{|\boldsymbol{c}|}\,, \tag{7}$$

where $\Delta\dot{\boldsymbol{q}}_\mathrm{a}$ is a velocity step chosen by the user, and $\Delta\dot{\boldsymbol{c}}$ and $\Delta\dot{\phi}$ are the steps in $\dot{\boldsymbol{c}}$ and $\dot{\phi}$ resulting from the actuation impulse $\boldsymbol{\iota}$ that causes $\Delta\dot{\boldsymbol{q}}_\mathrm{a}$.

These gains are calculated in almost the same way as before. First we solve the impulsive equation of motion for $\Delta\dot{q}_1$:

$$\begin{bmatrix} H_{11} & \boldsymbol{H}_{1\mathrm{a}} \\ \boldsymbol{H}_{\mathrm{a}1} & \boldsymbol{H}_{\mathrm{aa}} \end{bmatrix} \begin{bmatrix} \Delta\dot{q}_1 \\ \Delta\dot{\boldsymbol{q}}_\mathrm{a} \end{bmatrix} = \begin{bmatrix} 0 \\ \boldsymbol{\iota} \end{bmatrix} \tag{8}$$

giving

$$\Delta\dot{q}_1 = \frac{-1}{H_{11}}\,\boldsymbol{H}_{1\mathrm{a}}\Delta\dot{\boldsymbol{q}}_\mathrm{a}\,. \tag{9}$$

($\boldsymbol{H}_{1\mathrm{a}}$ is a $1\times(n-1)$ matrix.) Then we calculate $\boldsymbol{c}$ and $\Delta\dot{\boldsymbol{c}}$ from

$$[\boldsymbol{c},\, \Delta\dot{\boldsymbol{c}}] = \mathrm{cmpv}(rob, \boldsymbol{q}, \begin{bmatrix} \Delta\dot{q}_1 \\ \Delta\dot{\boldsymbol{q}}_\mathrm{a} \end{bmatrix})\,. \tag{10}$$

To understand the definitions in Eq. 7, suppose that the balance controller is using a *virtual joint*, with joint variable q_v, to perform its balancing task. This joint is mapped to the actuated motion freedoms according to $q_i = f_i(q_\mathrm{v})$, $i = 2 \ldots n$, where the functions f_i are chosen by the user. The virtual velocity variable therefore maps to the actuated velocity variables according to $\dot{q}_i = (\partial f_i/\partial q_\mathrm{v})\dot{q}_\mathrm{v}$. A unit-magnitude velocity step of the virtual joint in the positive direction is obtained by setting $\dot{q}_\mathrm{v} = 1$, which gives

$$\Delta\dot{\boldsymbol{q}}_\mathrm{a} = \begin{bmatrix} \frac{\partial f_2}{\partial q_\mathrm{v}} & \frac{\partial f_3}{\partial q_\mathrm{v}} & \cdots & \frac{\partial f_n}{\partial q_\mathrm{v}} \end{bmatrix}^\mathrm{T}. \tag{11}$$

So $\Delta\dot{\boldsymbol{q}}_\mathrm{a}$ is actually a kind of Jacobian: mapping velocities, and therefore also velocity steps, from the virtual joint to the physical ones.

One new property of the velocity gains in Eq. 7 is that they are linear in their arguments. Thus, for any two scalars α_1 and α_2 and any two vectors $\Delta\dot{\boldsymbol{q}}_{\mathrm{a}1}$ and $\Delta\dot{\boldsymbol{q}}_{\mathrm{a}2}$ we have

$$G_\mathrm{v}(\alpha_1\Delta\dot{\boldsymbol{q}}_{\mathrm{a}1}+\alpha_2\Delta\dot{\boldsymbol{q}}_{\mathrm{a}2}) = \alpha_1 G_\mathrm{v}(\Delta\dot{\boldsymbol{q}}_{\mathrm{a}1})+\alpha_2 G_\mathrm{v}(\Delta\dot{\boldsymbol{q}}_{\mathrm{a}2}) \tag{12}$$

and

$$G_\omega(\alpha_1\Delta\dot{\boldsymbol{q}}_{\mathrm{a}1} + \alpha_2\Delta\dot{\boldsymbol{q}}_{\mathrm{a}2}) = \alpha_1 G_\omega(\Delta\dot{\boldsymbol{q}}_{\mathrm{a}1}) + \alpha_2 G_\omega(\Delta\dot{\boldsymbol{q}}_{\mathrm{a}2})\,. \tag{13}$$

An immediate consequence is that we only need to know the gains for the individual variables in $\dot{\boldsymbol{q}}_\mathrm{a}$, since all other gains are just linear combinations of these ones. If we define $G_{\mathrm{v}i}$ and $G_{\omega i}$ to be the gains associated with variable i then we can define matrices

$$\boldsymbol{G}_\mathrm{v} = \begin{bmatrix} G_{\mathrm{v}2} & G_{\mathrm{v}3} & \cdots & G_{\mathrm{v}n} \end{bmatrix} \tag{14}$$

and

$$\boldsymbol{G}_\omega = \begin{bmatrix} G_{\omega 2} & G_{\omega 3} & \cdots & G_{\omega n} \end{bmatrix} \tag{15}$$

such that

$$G_\mathrm{v}(\Delta\dot{\boldsymbol{q}}_\mathrm{a}) = \boldsymbol{G}_\mathrm{v}\,\Delta\dot{\boldsymbol{q}}_\mathrm{a} \tag{16}$$

and

$$G_\omega(\Delta\dot{\boldsymbol{q}}_\mathrm{a}) = \boldsymbol{G}_\omega\,\Delta\dot{\boldsymbol{q}}_\mathrm{a}\,. \tag{17}$$

A Simple Example

Consider a planar triple pendulum containing two revolute joints and balancing on a sharp point. This robot is effectively a planar 3R mechanism in which joint 1 is passive and joints 2 and 3 are active. The vector of joint variables is $\boldsymbol{q} = [q_1 \ q_2 \ q_3]^\mathrm{T}$ and the vector of actuated joint variables is $\boldsymbol{q}_\mathrm{a} = [q_2 \ q_3]^\mathrm{T}$. If the user is interested in using joint 2 for balancing then the relevant velocity gains are $G_\mathrm{v}(\Delta\dot{\boldsymbol{q}}_\mathrm{a})$ and $G_\omega(\Delta\dot{\boldsymbol{q}}_\mathrm{a})$ with $\Delta\dot{\boldsymbol{q}}_\mathrm{a} = [1 \ 0]^\mathrm{T}$, and the two gains are equal to $G_{\mathrm{v}2}$ and $G_{\omega 2}$. Likewise, if the user is interested in using joint 3 for balancing then $\Delta\dot{\boldsymbol{q}}_\mathrm{a} = [0 \ 1]^\mathrm{T}$, and the two gains are equal to $G_{\mathrm{v}3}$ and $G_{\omega 3}$. Alternatively, if the user (or the balance controller) wishes to distribute the balancing motion equally between the two joints then the relevant gains are $G_\mathrm{v}(\Delta\dot{\boldsymbol{q}}_\mathrm{a})$ and $G_\omega(\Delta\dot{\boldsymbol{q}}_\mathrm{a})$ with $\Delta\dot{\boldsymbol{q}}_\mathrm{a} = [0.5 \ 0.5]^\mathrm{T}$.

This last choice illustrates why it is important to let the user supply $\Delta\dot{\boldsymbol{q}}_\mathrm{a}$. By specifying $[0.5 \ 0.5]^\mathrm{T}$ as the definition of a

unit velocity step, the user is tacitly adopting a 1-norm. But why not a 2-norm, or an ∞-norm, or some kind of weighted norm? The only reasonable answer is that this has to be left to the discretion of the user, since one cannot anticipate exactly what problem the user is trying to solve.

IV. Extension to 3D

We now consider the case of a general spatial mechanism making a single point contact with a horizontal supporting surface located in the x–y plane of a Cartesian coordinate system with the z axis pointing up (world coordinates). The link that makes this contact (the leg) is able to roll without slipping in both the x and y directions, and it is able to spin about the contact normal. It therefore has three degrees of instantaneous motion freedom relative to the support surface. However, rolling in 3D is a well-known example of a non-holonomic constraint, and this particular constraint needs five position variables and three velocity variables.

We therefore characterize the motion of the leg relative to the support with position variables q_x, q_y, q_1, q_2 and q_3, and velocity variables $\dot{q}_1$, $\dot{q}_2$ and $\dot{q}_3$. The variables q_1, q_2 and q_3 are a set of Euler angles describing the orientation of the leg, and $\dot{q}_1$, $\dot{q}_2$ and $\dot{q}_3$ are their derivatives. The variables q_x and q_y give the x and y coordinates of the contact point, and cannot be computed directly from the other variables. Instead, they are calculated by integrating the velocity of the contact point, which can be computed directly from the other variables.

For the special case of balancing on a sharp contact point, the rolling contact simplifies to a spherical joint, and the position variables q_x and q_y become constants. In this case we choose the origin of the world coordinate system to be at the contact point, so that $q_x = q_y = 0$.

Overall, the robot has n degrees of velocity freedom, which can be partitioned into 3 passive DoF at the contact and $n-3$ actuated DoF. We therefore define $\dot{\boldsymbol{q}}_\mathrm{p} = [\dot{q}_1\ \dot{q}_2\ \dot{q}_3]^\mathrm{T}$, $\dot{\boldsymbol{q}}_\mathrm{a} = [\dot{q}_4 \cdots \dot{q}_n]^\mathrm{T}$ and $\dot{\boldsymbol{q}} = [\dot{\boldsymbol{q}}_\mathrm{p}^\mathrm{T}\ \dot{\boldsymbol{q}}_\mathrm{a}^\mathrm{T}]^\mathrm{T}$. We continue to assume the existence of data structure rob, describing both the robot and its contact with the ground, and the three functions jsim, gim and cmpv, which must now perform their calculations in 3D. The vectors $\boldsymbol{c}$ and $\dot{\boldsymbol{c}}$, describing the position and velocity of the CoM relative to the contact point, are now 3D vectors.

The job of a balance controller in 3D is to bring the CoM directly above the support point. This means either driving both c_x and c_y to zero, or making $\boldsymbol{c}$ point upwards. We therefore define the 3D versions of the velocity gains as follows:

$$G_\mathrm{v}(\Delta\dot{\boldsymbol{q}}_\mathrm{a}) = \begin{bmatrix}\Delta\dot{c}_x \\ \Delta\dot{c}_y\end{bmatrix} \quad \text{and} \quad G_\omega(\Delta\dot{\boldsymbol{q}}_\mathrm{a}) = \frac{\boldsymbol{c}\times\Delta\dot{\boldsymbol{c}}}{|\boldsymbol{c}|^2}\,. \tag{18}$$

G_ω is now the angular velocity vector that is perpendicular to both $\boldsymbol{c}$ and $\dot{\boldsymbol{c}}$, and that describes the rate of change of the direction of $\boldsymbol{c}$. These definitions are compatible with their planar counterparts, and simplify to the planar versions if the robot happens to be planar and moving in a vertical plane.

As the balance controller now needs to control two degrees of rolling freedom, it requires two virtual joints with linearly independent velocity gains. Motions about these two virtual joints will cause rolling of the leg in two different directions, and the set of all linear combinations of motions about these joints spans the set of all rolling directions. The robot can be regarded as being good at balancing if it has a high velocity gain in every direction. If the robot's ability to balance must be described by a single scalar, then choose the smallest gain in this set, because this is the direction in which the robot is least able to balance, and therefore most likely to fall over.

Another difference between the 2D and 3D cases is that in the 2D case there is a single passive DoF and the controller is required to control it, whereas in the 3D case there are 3 passive DoF but the controller is only required to control two of them. The third passive DoF is the freedom of the leg to spin about the contact normal. This motion must be taken into account, because spinning motions do occur during balancing, but there is no need to control these motions for the purpose of achieving and maintaining balance. Indeed, the spin freedom is sometimes described as intrinsically uncontrollable, although this is only correct if spinning is frictionless. The subject of spin control is separate from balance control, and is outside the scope of this paper.

The spatial velocity gains are calculated in a similar manner to their planar counterparts. Assuming a sharp contact point, the first step is to calculate $\Delta\dot{\boldsymbol{q}}_\mathrm{p}$ from the given value of $\Delta\dot{\boldsymbol{q}}_\mathrm{a}$ using the impulsive equation of motion:

$$\begin{bmatrix}\boldsymbol{H}_\mathrm{pp} & \boldsymbol{H}_\mathrm{pa} \\ \boldsymbol{H}_\mathrm{ap} & \boldsymbol{H}_\mathrm{aa}\end{bmatrix}\begin{bmatrix}\Delta\dot{\boldsymbol{q}}_\mathrm{p} \\ \Delta\dot{\boldsymbol{q}}_\mathrm{a}\end{bmatrix} = \begin{bmatrix}\boldsymbol{0} \\ \boldsymbol{\iota}\end{bmatrix} \tag{19}$$

giving

$$\Delta\dot{\boldsymbol{q}}_\mathrm{p} = -\boldsymbol{H}_\mathrm{pp}^{-1}\,\boldsymbol{H}_\mathrm{pa}\,\Delta\dot{\boldsymbol{q}}_\mathrm{a}\,. \tag{20}$$

We then calculate $\boldsymbol{c}$ and $\Delta\dot{\boldsymbol{c}}$ using

$$[\boldsymbol{c},\,\Delta\dot{\boldsymbol{c}}] = \mathrm{cmpv}(rob, \boldsymbol{q}, \begin{bmatrix}\Delta\dot{\boldsymbol{q}}_\mathrm{p} \\ \Delta\dot{\boldsymbol{q}}_\mathrm{a}\end{bmatrix})\,, \tag{21}$$

and then calculate the gains using Eq. 18. The calculation method for a general rolling contact is the subject of the next section.

V. General Rolling Contact

In the preceding sections, definitions were given for a general rolling contact, but calculation methods were restricted to the case of balancing on a sharp point. This section presents the calculation methods for a general rolling contact. The general method is to augment the robot model with extra degrees of freedom, and then impose the kinematic constraints of the rolling contact. This approach allows us to reuse the functions jsim, gim and cmpv defined earlier, and requires only one new function, roll, which is explained below. However, the 2D and 3D cases differ in their details because the former is holonomic and the latter nonholonomic.

Rolling in 2D

Figure 2 shows the kinematics of a planar rolling contact. A coordinate frame is fixed in the leg, and the three variables q_x, q_y and q_1 give the position and orientation of this frame

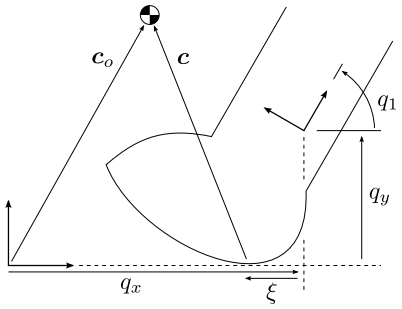

Fig. 2. Kinematics of rolling in 2D

relative to the base coordinate frame, which is positioned so that its x axis lies on the support surface. A fourth variable, ξ, serves to locate the contact point. The vectors $\boldsymbol{c}$ and $\boldsymbol{c}_o$ are as defined earlier. However, whereas we previously assumed that the contact point was located at the origin of base coordinates, so that $\boldsymbol{c} = \boldsymbol{c}_o$, we now have a new relationship between them:

$$\boldsymbol{c} = \boldsymbol{c}_o - \begin{bmatrix} q_x + \xi \\ 0 \end{bmatrix}. \tag{22}$$

(ξ is negative in the diagram.)

In the data structure rob, the leg is connected to the ground via a planar joint (i.e., two prismatic joints and a revolute joint), so that the leg has a full 3 DoF relative to the ground. Thus, the robot has been augmented with an extra 2 DoF (it now has $n+2$ DoF), and the variables q_x and q_y are being treated as independent. It therefore follows that the functions jsim, gim and cmpv must be given an augmented position vector, $\boldsymbol{q}'$, as argument, and also an augmented velocity vector, $\dot{\boldsymbol{q}}'$, in the case of cmpv. These are defined as follows:

$$\boldsymbol{q}' = \begin{bmatrix} q_x \\ q_y \\ q_1 \\ \boldsymbol{q}_\mathrm{a} \end{bmatrix} \qquad \text{and} \qquad \dot{\boldsymbol{q}}' = \begin{bmatrix} \dot{q}_x \\ \dot{q}_y \\ \dot{q}_1 \\ \dot{\boldsymbol{q}}_\mathrm{a} \end{bmatrix}. \tag{23}$$

We now introduce a new function, $\mathrm{roll}(rob, q_1)$, to calculate the kinematics of the rolling contact. Specifically, the following quantities are calculated: q_x, q_y, ξ, X ($=\mathrm{d}q_x/\mathrm{d}q_1$), Y ($=\mathrm{d}q_y/\mathrm{d}q_1$) and Ξ ($=\mathrm{d}\xi/\mathrm{d}q_1$). Observe that q_1 is the independent variable, and that everything in this list is computed as a function of q_1. rob appears in the argument list because roll needs to know the shape of the foot, and it is assumed that this data is stored in rob.

It was mentioned earlier that the velocity gain is not defined if there is a step change in foot curvature at the contact point. This effect enters into the calculation via the derivatives X, Y and Ξ, which depend on the curvature at the contact point, and which undergo a step change in value if there is a step change in curvature.

The functions jsim and gim calculate an augmented inertia matrix, $\boldsymbol{H}'$, which is a $(n+2)\times(n+2)$ matrix. This is converted to the $n\times n$ generalized inertia matrix $\boldsymbol{H}$, which includes the kinematics of the rolling contact, as follows:

$$\boldsymbol{H} = \boldsymbol{G}^\mathrm{T}\, \boldsymbol{H}'\, \boldsymbol{G} \tag{24}$$

where

$$\boldsymbol{G} = \begin{bmatrix} X & \boldsymbol{0} \\ Y & \boldsymbol{0} \\ 1 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{1}_{(n-1)\times(n-1)} \end{bmatrix}. \tag{25}$$

Given $\boldsymbol{H}$, the next step is to calculate $\Delta\dot{\boldsymbol{q}}_1$ from Eqs. 8 and 9. One can then calculate $\boldsymbol{c}_o$ and $\Delta\dot{\boldsymbol{c}}_o$ from

$$[\boldsymbol{c}_o, \Delta\dot{\boldsymbol{c}}_o] = \mathrm{cmpv}(rob, \boldsymbol{q}', \Delta\dot{\boldsymbol{q}}') \tag{26}$$

where

$$\Delta\dot{\boldsymbol{q}}' = \begin{bmatrix} X\Delta\dot{q}_1 \\ Y\Delta\dot{q}_1 \\ \Delta\dot{q}_1 \\ \Delta\dot{\boldsymbol{q}}_\mathrm{a} \end{bmatrix}. \tag{27}$$

Finally, $\boldsymbol{c}$ is calculated from Eq. 22, and $\Delta\dot{\boldsymbol{c}}$ is calculated from

$$\Delta\dot{\boldsymbol{c}} = \Delta\dot{\boldsymbol{c}}_o - \begin{bmatrix} (X+\Xi)\Delta\dot{q}_1 \\ 0 \end{bmatrix}. \tag{28}$$

To summarize, the calculation procedure for obtaining the velocity gains in the case of a general planar rolling contact is as follows:

1) call roll to obtain q_x, q_y, etc.;
2) form $\boldsymbol{q}'$ as per Eq. 23, and call either jsim or gim to obtain $\boldsymbol{H}'$;
3) calculate $\boldsymbol{H}$ using Eqs. 24 and 25;
4) calculate $\Delta\dot{q}_1$ using Eq. 9;
5) form $\Delta\dot{\boldsymbol{q}}'$ as per Eq. 27 and calculate $\boldsymbol{c}_o$ and $\Delta\dot{\boldsymbol{c}}_o$ using Eq. 26;
6) calculate $\boldsymbol{c}$ and $\Delta\dot{\boldsymbol{c}}$ using Eqs. 22 and 28; and
7) calculate the velocity gains using Eq. 7.

Although the kinematics of a general rolling contact can be very complicated, it is worth mentioning that if the leg happens to be a circular wheel, and the leg coordinate frame is at the centre of the wheel, then $\xi = \Xi = Y = 0$ and $y = X = r$ (the radius of the wheel).

Rolling in 3D

The calculation procedure for a general rolling contact in 3D follows the same pattern as for a general rolling contact in 2D, but with some differences in the details, as follows. First, the leg is now connected to the world coordinate frame by a 6-DoF joint, consisting of three translations and three rotations. Thus, the robot model in rob has been augmented with three translational DoF, with corresponding variables q_x, q_y and q_z, and now has $n+3$ DoF in total. It therefore follows that the augmented vectors $\boldsymbol{q}'$ and $\Delta\dot{\boldsymbol{q}}'$ and the matrix $\boldsymbol{G}$ all have an extra row corresponding to the new z coordinate, and that $\boldsymbol{H}'$ is now a $(n+3)\times(n+3)$ matrix.

Next, $\boldsymbol{q}_\mathrm{p}$ and $\Delta\dot{\boldsymbol{q}}_\mathrm{p}$ replace q_1 and $\Delta\dot{q}_1$, and this implies some changes in the dimensions of other quantities. In particular, $\boldsymbol{X}$ and $\boldsymbol{Y}$ are now 1×3 matrices, and $\boldsymbol{\Xi}$ is a 2×3 matrix because $\boldsymbol{\xi}$ is now a 2D vector with x and y components. The relationship between $\boldsymbol{c}$ and $\boldsymbol{c}_o$ is now

$$\boldsymbol{c} = \boldsymbol{c}_o - \begin{bmatrix} q_x - \xi_x \\ q_y - \xi_y \\ 0 \end{bmatrix}, \tag{29}$$

and the relationship between $\Delta\dot{\boldsymbol{c}}$ and $\Delta\dot{\boldsymbol{c}}_o$ is

$$\Delta\dot{\boldsymbol{c}} = \Delta\dot{\boldsymbol{c}}_o - \begin{bmatrix} \left(\begin{bmatrix} \boldsymbol{X} \\ \boldsymbol{Y} \end{bmatrix} + \boldsymbol{\Xi} \right) \Delta\dot{\boldsymbol{q}}_{\mathrm{p}} \\ 0 \end{bmatrix}. \tag{30}$$

Finally, $\mathrm{roll}(rob, \boldsymbol{q}_{\mathrm{p}})$ must now calculate the quantities q_z and $\boldsymbol{Z}$ ($=\mathrm{d}q_z/\mathrm{d}\boldsymbol{q}_{\mathrm{p}}$, a 1×3 matrix), but it can no longer calculate q_x and q_y because the rolling contact is nonholonomic. These variables must instead be calculated by integrating $\dot{q}_x$ and $\dot{q}_y$, which can be calculated from $\boldsymbol{X}$, $\boldsymbol{Y}$ and $\dot{\boldsymbol{q}}_{\mathrm{p}}$. However, it turns out that this calculation is not necessary because the velocity gains do not depend on the values of these two variables, and they can be set to any arbitrary value, such as zero. (The same is also true of q_x in the planar rolling case.) Note, however, that the velocity gains do depend on the velocity variables $\Delta\dot{q}_x$ and $\Delta\dot{q}_y$, which therefore must be calculated as described above.

If the leg happens to be a sphere, and the leg coordinate frame is located at the centre of the sphere, then $\boldsymbol{\xi}$, $\boldsymbol{\Xi}$ and $\boldsymbol{Z}$ are all zero, and q_z equals the radius of the sphere.

VI. A Design Example

This section illustrates the use of velocity gain to design a planar triple pendulum that is good at balancing. Joints 2 and 3 in this mechanism are actuated, and joint 1 is passive.

Figure 3 shows contour plots of the angular velocity gain for joints 2 and 3 of an initial design and an improved design of the triple pendulum, plotted against q_2 from 0 to π and q_3 from $-\pi$ to π. The plot for negative values of q_2 is identical to the one shown, rotated by $180°$ about $(0, 0)$. The gain is independent of q_1, and the robot is straight when $q_2 = q_3 = 0$. The colours red, dark orange, light orange and yellow onwards can be regarded as bad, poor, OK and good. Brown corresponds to positive values, and the boundary between red and brown is the set of configurations where balancing is physically impossible using the chosen joint.

Graphs like these can be used as maps for a high-level controller, telling it which configurations are good for balancing and which are bad, and telling it also which joints, or combinations of joints, are most effective for maintaining balance in each region of configuration space.

The initial design consists of three identical links of length 0.3m and mass 0.5kg, the mass being concentrated at a point at the far end of each link. The improved design has link lengths of 0.2m, 0.25m and 0.35m, and masses of 0.7kg, 0.5kg and 0.3kg. As G_ω is both mass- and scale-invariant, neither the absolute masses not the absolute link lengths matter: only the ratios are important.

The improved design was obtained by manually exploring the effects that parameter changes had on the velocity gains until a parameter set was found with significantly better gains. This is feasible when the mechanism is as simple as a triple pendulum. However, in general it is preferable to use an automated optimization process in which the designer supplies weights indicating the relative importance of each region of configuration space, plus information indicating which joints and joint combinations are to be used for balancing.

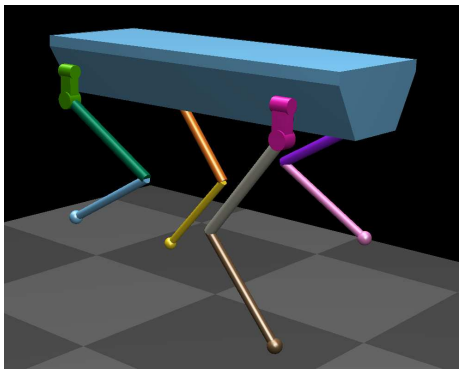

Fig. 4. HyQ balancing on diametrically opposite legs (torso height 0.6m, leg configuration C in Table I)

The graphs show that the improved design has better velocity gains in most of its configuration space, although there is a problem area in the vicinity of $(\pi, 0)$ where the gain changes sign. This may or may not be an issue. For example, if q_2 cannot get close to π because of motion limits then this region of configuration space is unreachable and the velocity gains in this region are irrelevant.

The graphs also show that joint 2 has a better velocity gain than joint 3 almost everywhere, and that the velocity gain of joint 2 improves with increasing angle. These results are no surprise: joint 2 moves more mass than joint 3, so it is hardly surprising that it has more effect on the CoM; and increasing q_2 folds the robot so as to bring the CoM closer to the support point, so that $|\boldsymbol{c}|$ in the denominator of Eq. 7 becomes smaller. The blue region in the graphs for joint 3 occur at configurations where the mechanism is curled up, and the CoM is relatively close to the support point.

An investigation of this kind can be used to analyze existing robots as well as design new ones. With an existing robot, one cannot alter the joint velocity gains, unless one is prepared to add masses here and there. However, one can compose maps like those in Figure 3 to show which configurations are good for balancing and which are not, and to show which joints, or combinations of joints, are the best to use. Adding mass is not necessarily a bad idea: consider, for example, the balancing pole of a tightrope walker—a relatively small additional mass that improves substantially the artist's ability to balance. A velocity-gain analysis can show how much mass is needed, where best to put it, and how big an effect it will have.

VII. HyQ Balancing

This section presents an analysis of the balancing ability of the hydraulic quadruped HyQ [10]. It is proposed to make this robot balance on two diametrically opposite legs in postures like the one shown in Figure 4. Specifically, the two raised legs will be held in a fixed configuration, so that they behave like an extension of the torso, and the robot will balance by tipping the torso about a line joining the hip centres (the centres of

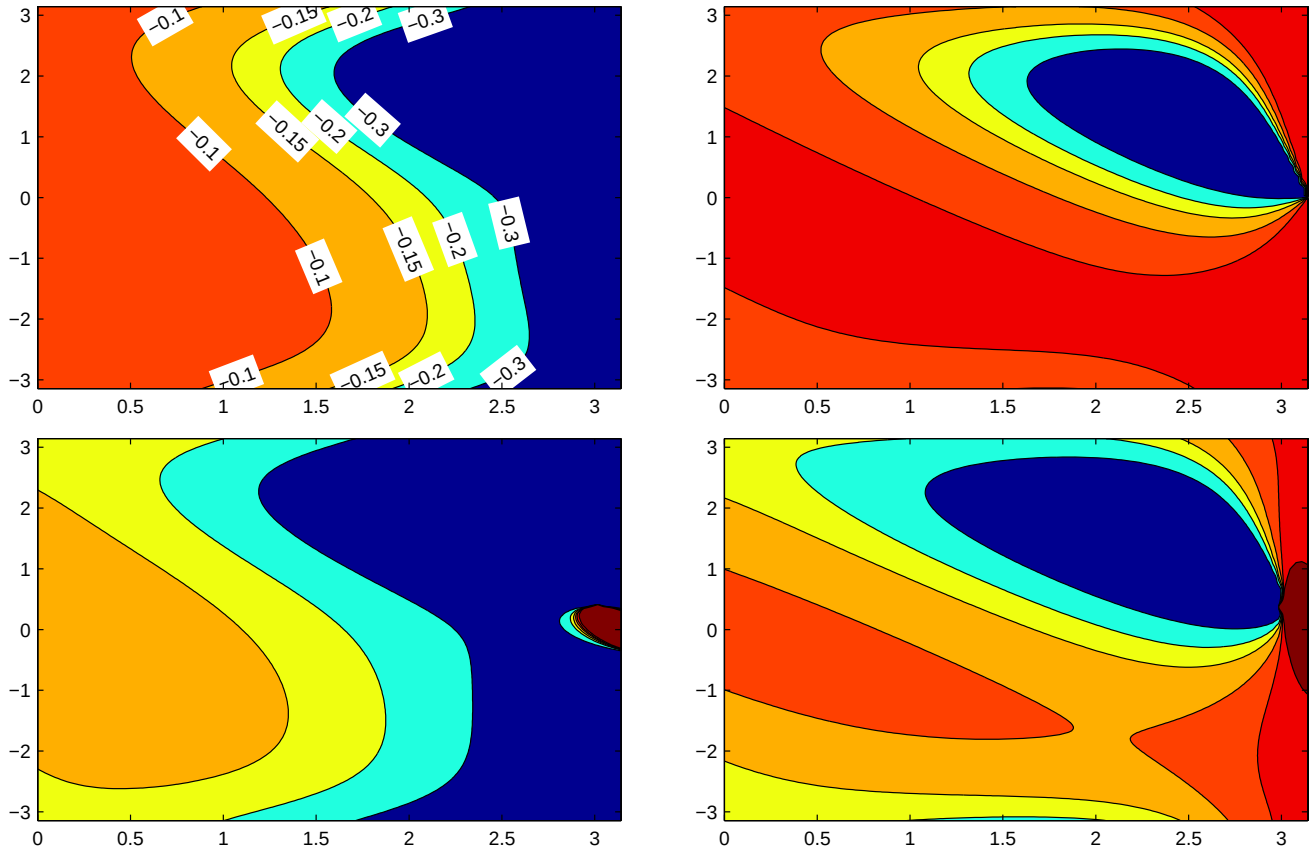

Fig. 3. Contour plots of angular velocity gain (G_ω) for joint 2 (left column) and joint 3 (right column) of an initial design (top row) and improved design (bottom row) of planar triple pendulum. Horizontal axis: q_2; vertical axis: q_3; contours: brown >0, red 0 to -0.06, others as shown in the top left plot.

the top cylinders in the figure) of the two supporting legs. The supporting feet are positioned directly below the hip centres. The available parameters are: the angles defining the posture of the raised legs, and the height of the torso above the ground. The latter is measured from the origin of the torso coordinate frame, which lies on the line of tipping.

To calculate the velocity gains of this robot, for the purpose of balancing in this particular manner, the first step is to define a kinematic mapping from the operational space of balancing to the joint space of the robot. The latter consists of the 12 revolute joints of the robot mechanism plus the torso's 6 DoF relative to the ground. The former consists of two virtual revolute joints: a rotation about the line joining the two supporting feet, with joint variable θ_1, which represents the robot's passive freedom to fall over, and a rotation about the line joining the two supporting hip centres, with joint variable θ_2, which represents the movement that the robot uses to balance. In effect, the kinematics function maps an inverted double pendulum onto the robot. The calculation of velocity gain then proceeds as explained in Section II, but with the robot's dynamics expressed in terms of the virtual joint variables θ_1 and θ_2 instead of q_1 and q_2.

Figure 5 plots angular velocity gain against torso height for the raised-leg configurations listed in Table I. These configurations are: (A) legs fully extended, pointing sideways; (B) legs fully retracted, pointing sideways; and (C) legs partly retracted, pointing down (as shown in Fig. 4).

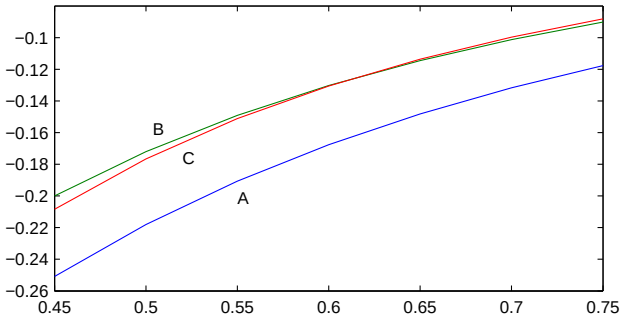

Fig. 5. Angular velocity gain versus torso height (m) for the raised-leg configurations listed in Table I

	left front leg			right hind leg		
A	$-90°$	$10°$	$-20°$	$-90°$	$-10°$	$20°$
B	$-90°$	$70°$	$-140°$	$-90°$	$-70°$	$140°$
C	$0°$	$60°$	$-120°$	$0°$	$-60°$	$120°$

TABLE I
RAISED-LEG CONFIGURATIONS

The first thing to notice about this graph is that there is a factor of 3 difference between the best and the worst

configuration; so the choice of configuration can be expected to have a substantial effect on the quality of the robot's balancing, and on the risk of falling over. This alone is enough to justify a velocity-gain analysis of this robot.

The next thing to notice is that the main determining factor is torso height, which accounts for more than a factor of two variation in velocity gain over the range considered, whereas extending the raised legs sideways produces only a 20–30% improvement. When the raised legs are not extended, they have little effect. One can deduce form this data that the robot's safest strategy is to crouch down as much as possible, unload the two non-supporting legs, and then move them directly out to configuration A.

There are two sources of disturbance acting on this robot: IMU sensor noise and unknown forces exerted by its umbilical. The IMU data sheet states a noise magnitude of $0.5°$, which will make the robot wobble. At the worst-case velocity gain of 0.08 (ignoring the sign), a $0.5°$ error would require a $6°$ correction if the robot could do it instantaneously. However, as the robot cannot move infinitely quickly, the actual correction magnitude will be significantly in excess of $6°$ because gravity will be acting to exacerbate the balance error during the course of the motion. Furthermore, the control system is aiming not merely to arrest the robot's fall, but to bring the robot back to its commanded position. This requires tipping the robot in the opposite direction, which means that the control system must necessarily overshoot.

The exact magnitude of the resulting wobbles will depend on many details, but simulation studies on other robots suggest wobble magnitudes of 2 to 3 times the theoretical minimum predicted by the velocity gain, so roughly 12–$18°$ for the HyQ. Add to this a disturbance from the umbilical, and one can see that there is a significant possibility that the robot will reach its $30°$ torso rotation limit and lose its balance. In the best-case configuration, the wobbles will be smaller by a factor of 3, and the risk of losing balance will be greatly reduced.

VIII. Conclusion

This paper has further developed the idea of velocity gain, which was proposed originally in Featherstone [3] as a means of measuring the ability of a planar double pendulum to balance itself actively on a single point of support. This paper extends the idea to the case of a robot with more than one actuated degree of freedom, and to the case of a robot balancing in 3D. A distinction is drawn between linear and angular velocity gains, which have slightly different properties and ranges of applicability; and detailed instructions have been given for calculating these quantities, including the case of balancing on a general rolling contact. The applicability of velocity gain to robots that balance on an area of support rather than a single point is discussed briefly.

The paper concludes with two examples of use: designing a triple pendulum to be good at balancing, and analyzing the balancing ability of an existing quadruped in order to choose a good configuration for a balancing experiment. The latter requires a kinematics function that maps the robot's balancing motion onto the physical joints.

References

[1] Double Robotics. 'Double' telepresence robot. `www.doublerobotics.com`, accessed Jan. 2015.

[2] R. Featherstone. *Rigid Body Dynamics Algorithms*. Springer, New York, 2008.

[3] R. Featherstone. Analysis and Design of Planar Self-Balancing Double-Pendulum Robots. *in* Padois, Bidaud & Khatib (eds.), *RoManSy 19 — Robot Design, Dynamics and Control*, pp. 259–266, Springer, Vienna, 2013.

[4] M. Gajamohan, M. Merz, I. Thommen and R. D'Andrea. The Cubli: A Cube that can Jump up and Balance. *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, Vilamoura, Portugal, Oct. 7–12, pp. 3722–3727, 2012. DOI

[5] J. W. Grizzle, C. H. Moog and C. Chevallereau. Nonlinear Control of Mechanical Systems With an Unactuated Cyclic Variable. *IEEE Trans. Automatic Control*, 50(5):559–576, 2005. DOI

[6] J. Hauser and R. M. Murray. Nonlinear Controllers for Non-Integrable Systems: the Acrobot Example. *Proc. American Control Conf.*, Nov. 3–5, pp. 669–671, 1990.

[7] T. B. Lauwers, G. A. Kantor and R. L. Hollis. A Dynamically Stable Single-Wheeled Mobile Robot with Inverse Mouse-Ball Drive. *Proc. IEEE Int. Conf. Robotics and Automation*, Orlando, FL, May 15–19, pp. 2884–2889, 2006. DOI

[8] N. Miyashita, M. Kishikawa and M. Yamakita. 3D Motion Control of 2 Links (5 DOF) Underactuated Manipulator Named AcroBOX. *Proc. American Control Conf.*, Minneapolis, MN, June 14–16, pp. 5614–5619, 2006.

[9] Segway Inc. Personal Transporter. `www.segway.com`, accessed Jan. 2015.

[10] C. Semini. HyQ – Design and Development of a Hydraulically Actuated Quadruped Robot. *Ph.D. Thesis*, University of Genoa, Italy, 2010.

[11] T. Sugihara, Y. Nakamura and H. Inoue. Realtime Humanoid Motion Generation through ZPM Manipulation based on Inverted Pendulum Control. *Proc. IEEE Int. Conf. Robots and Automation*, Washington, DC, May 11–15, pp. 1404–1409, 2002. DOI

[12] Xinjilefu, V. Hayward and H. Michalska. Stabilization of the Spatial Double Inverted Pendulum Using Stochastic Programming Seen as a Model of Standing Posture Control. *Proc. 9th IEEE Int. Conf. Humanoid Robots*, Paris, France, Dec. 7–10, pp. 367–372, 2009. DOI