2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)
November 3 - 5, 2015, Seoul, Korea

Biologically Inspired Deadbeat control for running on 3D stepping stones

Johannes Englsberger, Paweł Kozłowski, Christian Ott

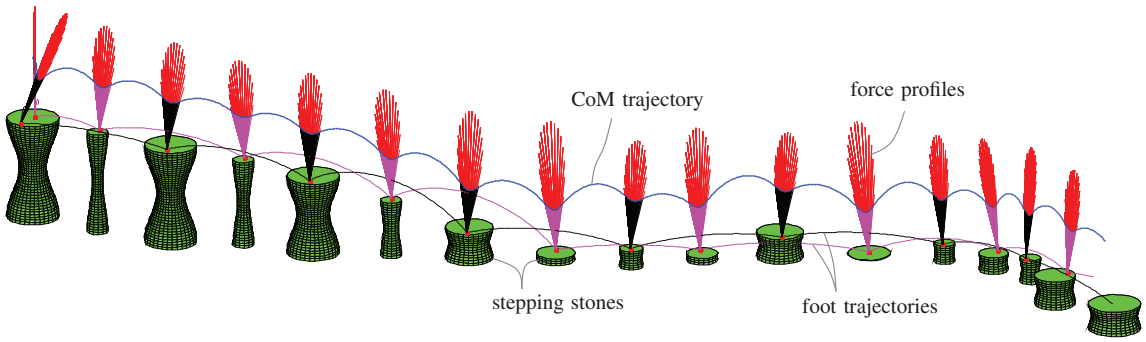

Fig. 1: Bipedal point-mass model running on 3D stepping stones based on Biologically Inspired Dead-beat (BID) control.

***Abstract*— This paper enhances the Biologically Inspired Dead-beat (BID) controller from [1], to not only enable three-dimensional bipedal running on a horizontal plane but also on 3D stepping-stones. Further contributions of the paper are explicit foot step targeting during running, leg cross-over avoidance and the embedding of the BID controller into a QP-based whole-body controller. The BID controller is based on the encoding of leg forces and CoM trajectories during stance as polynomial splines, allowing for intuitive and primarily analytical controller design. It allows a real-time implementation, is highly robust against perturbations and enables versatile running patterns. The performance of the control framework is tested in various simulations for a bipedal point-mass model and an articulated multi-body model of the humanoid robot Toro.**

I. Introduction

Robotic mobility has recently attracted more and more attention. From an engineering point of view, gaited forms of locomotion - once fully understood - promise highly increased mobility of machines as compared to wheel-based locomotion. Overcoming a set of stepping stones, as shown in Fig. 1, is one possible example where a legged robot may have advantages over machines of similar size and weight.

The first efforts in robotic bipedal locomotion have been put in the subdomain of bipedal walking. Over the decades, the field of bipedal walking control has made major progress. Alongside successes in passive dynamic walking [2], one of the major breakthroughs has been the introduction of zero moment point control [3], [4] for bipedal walking. More recently, several successful walking control algorithms have been presented, e.g. [5]–[13], to name but a few. Recently, bipedal walking algorithms have reached a level that is close to actual application in real-world scenarios. Most walking algorithms attempt to keep the robot in a fully controllable state, which facilitates the use of standard control methods.

In contrast, during flight some of the robot's states are unavoidably underactuated, which makes running and hopping challenging tasks. Running provides a number of assets such as high achievable speed and efficiency [14]. Aside from few exceptions such as [15]–[17], most running algorithms are based on the spring-loaded inverted pendulum (SLIP) [14]. Dadashzadeh et al. [18] present a SLIP-based two-level controller for running simulations of the ATRIAS robot. Carver et al. [19] show that the number of required recovery steps depends on the goals of the control mechanism and present a SLIP-based controller for two-step recovery using synergies. Vejdani et al. [20] introduce bio-inspired swing leg control for running on ground with unexpected height disturbances. Wu et al. [21] present a deadbeat controller for the 3D SLIP model that can cope with unknown ground height variations of up to 30% of the leg length. It is based on multi-dimensional look-up tables and achieves deadbeat control of apex height and heading direction. Yet, dissipative losses are not considered in their method. Koepl and Hurst [22] control the stance phase impulse of a planar SLIP model and achieve robust running. Recently, Wensing and Orin [23] presented an algorithm that computes periodic trajectories of the 3D-SLIP offline and applies a linearized control law to stabilize the virtual SLIP model around the periodic solutions. The desired leg forces are passed to a whole-body controller and bipedal running of a simulated humanoid robot is achieved. Yet, the method in [23] comes with two major drawbacks: Firstly, it requires offline computation of periodic SLIP gaits. Second, the motions are designed to be periodic, which makes the design of non-periodic running motions such as accelerating and turning a challenge. Also recently, Park et al. [24] presented quadrupedal galloping with the MIT Cheetah 2 based on impulse control. They used 3-rd order

The authors are with Institute of Robotics and Mechatronics, German Aerospace Center (DLR), 82234 Wessling, Germany. E-mail: johannes.englsberger@dlr.de

978-1-4799-6885-5/15/$31.00 ©2015 IEEE

Bezier polynomials to encode the leg force profiles. Yet, their method is nominally unstable and designed for constant speeds, such that heuristic PD control laws have to be applied to achieve stability and speed control.

Several drawbacks of the previously mentioned works were eliminated in [1]. Here, we proposed the so called Biologically Inspired Deadbeat[1] (BID) controller that is real-time capable, enables versatile running motions and is very robust against external perturbations. It has been inspired by observations from human running experiments (see Fig. 2) and uses polynomial splines to encode the robot's CoM motion and leg forces during stance. The control design process is very intuitive and comprehensible. Different running speeds and transitions between them are handled in a clean way. One major advantage of the algorithm is that the next two upcoming foot aim points on the ground (i.e. the left and the right one) are predicted at all times, which facilitates the design of appropriate foot trajectories.

One disadvantage of the method as presented in [1] was that the foot positions could not be controlled directly, which caused the danger of leg cross-over (especially when running in sharp turns). In this paper, we extend the original method to achieve precise foot placement and running on 3D stepping stones (see Fig. 1). The precise foot placement now enables explicit leg cross-over avoidance. Another contribution of this paper is the embedding of our running controller into a QP-based whole-body control framework.

The paper is organized as follows: Section II motivates the use of polynomial splines via observations from human running experiments. Sections III and IV give a short outline of our planning and control framework and recapitulate the flight dynamics. In Section V, the vertical and horizontal boundary conditions are solved, which facilitates the design of our feedback controller presented in section VI. Sections VII and VIII outline the embedding of the BID controller into a whole-body controller and present various simulations for point-mass and whole-body running. Sections IX and X discuss the proposed control framework's assets and limitations and conclude the paper.

II. Human running experiments as motivation

The main idea in this paper is to *design desired CoM trajectories* that produce *approximately natural ground reaction force (GRF) profiles* while fulfilling several *boundary conditions*. It is well known that some physical template models, such as the SLIP [14], generate GRF similar to the ones observed in human running. The lack of closed form solutions for the SLIP motivates us to find an alternative way of encoding the leg force ($\boldsymbol{F}_{leg}$, equivals GRF). Figure 2 shows a typical GRF profile that was recorded during a human running experiment via force plate. Except for the impact phenomenon at the beginning of stance, the human GRF profiles can be approximated quite well by polynomials of order 2 in the vertical direction and of order 3 in the

[1]In this work, we use the term "deadbeat" for a controller that brings the system back to the desired state after just one stance phase.

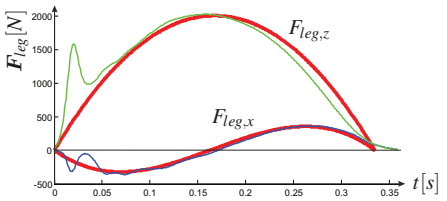

Fig. 2: Comparison of experimentally measured human leg forces (blue/green) and polynomial approximations (red).

$x-$direction. Therefore, our original idea was to approximate the leg force profile during stance via polynomials [1]. The total force $\boldsymbol{F}_{CoM}$ acting on the CoM can be computed from the leg force $\boldsymbol{F}_{leg}$ and the gravitational force $\boldsymbol{F}_g$ via

$$\boldsymbol{F}_{CoM} = \boldsymbol{F}_{leg} + \boldsymbol{F}_g = \boldsymbol{F}_{leg} + m\,\boldsymbol{g} \quad . \tag{1}$$

Here, m is the robot's total mass and $\boldsymbol{g} = [0\,0\,-g]^T$ denotes the gravitational acceleration vector. The constant offset between $\boldsymbol{F}_{CoM}$ and $\boldsymbol{F}_{leg}$ in (1) and Newton's 2nd law[2] (CoM acceleration $\ddot{\boldsymbol{x}} = \frac{\boldsymbol{F}_{CoM}}{m}$) motivate us to use - during stance - a 4th order polynomial to encode the vertical CoM position z and 5th order polynomials to encode the horizontal CoM positions x and y, as this correlates to 2nd and 3rd order polynomials for the CoM accelerations $\ddot{x}$, $\ddot{y}$, $\ddot{z}$ and thus leg forces. This polynomial encoding can be written as:

$$\begin{bmatrix}\sigma(t)\\ \dot{\sigma}(t)\\ \ddot{\sigma}(t)\end{bmatrix} = \underbrace{\begin{bmatrix}1 & t & t^2 & t^3 & t^4 & t^5\\ 0 & 1 & 2t & 3t^2 & 4t^3 & 5t^4\\ 0 & 0 & 2 & 6t & 12t^2 & 20t^3\end{bmatrix}}_{\begin{bmatrix}\boldsymbol{t}_\sigma^T(t)\\ \boldsymbol{t}_{\dot{\sigma}}^T(t)\\ \boldsymbol{t}_{\ddot{\sigma}}^T(t)\end{bmatrix}} \boldsymbol{p}_\sigma, \quad \sigma \in \{x,y,z\} \tag{2}$$

Here, $\boldsymbol{t}_\sigma^T(t)$, $\boldsymbol{t}_{\dot{\sigma}}^T(t)$ and $\boldsymbol{t}_{\ddot{\sigma}}^T(t)$ denote the time-mapping row vectors that - for a given time t - map the polynomial parameter vectors $\boldsymbol{p}_\sigma$ to CoM positions $\sigma(t)$, velocities $\dot{\sigma}(t)$ and accelerations $\ddot{\sigma}(t)$. The last elements of the vectors are greyed out to indicate that they are only used for the horizontal directions, but not for the vertical one.

III. Outline of BID control method

In this work, we use a preview of typically four upcoming stance and flight phases, as drafted in Fig. 3. The desired relative apex and touch-down heights $\Delta z_{apex,des}$ and $\Delta z_{TD,des}$ are used as *design parameters*. They indicate how high over the floor (at $z_{floor,i}$) the apex of each flight curve (i.e. $\dot{z} = 0$) should be and at what CoM height the touch-down (TD) is supposed to happen. $z_{floor,i}$ denotes the height level of the i-th step. Another design parameter, used in this work, is the total stance time T_s, whereas the total flight time T_f results from the boundary conditions chosen in section V-A. To keep track of the current running state, we use a state machine. It switches from flight to stance, if the CoM is below $z_{TD} = z_{floor,i} + \Delta z_{TD}$ and the vertical velocity is

[2]Newton's 2nd law - and thus the controller derived in this work - is valid for any real system, not only for simplified models.

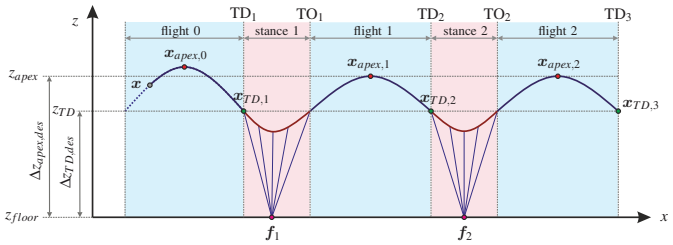

Fig. 3: Preview of upcoming flight and stance phases (planar sketch) - used for design of boundary conditions. For readability, a constant floor height z_{floor} is shown here.

negative, and from stance to flight when the total stance time is over. A timer provides the time in stance $t_s \in [0, T_s]$ and the time in flight $t_f \in [0, T_f]$. They are reset at state transitions.

IV. CoM dynamics during flight

Running is a locomotion pattern, which employs alternate flight and (single leg supporting) stance phases. During flight, the CoM cannot be controlled, i.e. it follows its natural dynamics (parabolic path through space). For a given time t, the CoM position $\boldsymbol{x}(t) = [x(t), y(t), z(t)]^T$ and velocity $\dot{\boldsymbol{x}}(t) = [\dot{x}(t), \dot{y}(t), \dot{z}(t)]^T$ can be computed as

$$\boldsymbol{x}(t) = \boldsymbol{x}_0 + \dot{\boldsymbol{x}}_0\, t + \boldsymbol{g}\, \frac{t^2}{2} \quad , \tag{3}$$

$$\dot{\boldsymbol{x}}(t) = \dot{\boldsymbol{x}}_0 + \boldsymbol{g}\, t \, , \tag{4}$$

where $\boldsymbol{x}_0$ and $\dot{\boldsymbol{x}}_0$ are the initial ($t = 0$) CoM position and velocity. One typical task in running control is to achieve a certain apex height. The apex is the highest point in the ballistic flight curve, i.e. the vertical CoM velocity is zero ($\dot{z}_{apex} = 0$). Using this condition and the current vertical CoM velocity $\dot{z}$ instead $\dot{z}_0$ in the third row of (4), we find the current time to apex Δt_{apex} as

$$\Delta t_{apex} = \frac{\dot{z}}{g} \quad . \tag{5}$$

If Δt_{apex} is negative (true for $\dot{z} < 0$), then the CoM is already on the descending path of the ballistic flight curve and thus the time of apex is in the past. In the same way, we find the remaining time until the i-th touch-down (TD) as

$$\Delta t_{TD} = \Delta t_{apex} + \sqrt{\Delta t_{apex}^2 + \frac{2}{g}\,(z - z_{TD,i})} \quad . \tag{6}$$

Here, $z_{TD,i} = z_{floor,i} + \Delta z_{TD}$ is the CoM height at which the i-th touch-down (flight to stance transition) is previewed to happen. The relative touch-down height is computed as

$$\Delta z_{TD} = min(\Delta z_{TD,des},\ z - z_{floor,i} + \frac{\dot{z}^2}{2g} - \Delta_{apex,TD,min}) \quad . \tag{7}$$

That way, nominally the desired relative touch-down height $\Delta z_{TD,des}$ (one of our *design variables*) is achieved, while for challenging initial conditions or perturbations a minimum height difference between apex and touch-down $\Delta_{apex,TD,min}$ is guaranteed and the solution of (6) is assured to be real.

V. Method for boundary condition satisfaction as basic module for deadbeat controller

A. Vertical planning and boundary conditions

As mentioned above, the vertical CoM trajectory during stance is encoded via a 4th order polynomial, i.e. it has 5 polynomial parameters. These can be derived using 5 boundary conditions. Fig. 3 graphically displays the used preview of upcoming flight and stance sequences and the corresponding boundary conditions. In this work, - for each previewed contact phase - we make use of four linear vertical boundary conditions that can be combined as

$$\underbrace{\begin{bmatrix} z_{TD,i} \\ \dot{z}_{TD,i} \\ -g \\ -g \end{bmatrix}}_{\boldsymbol{b}_{z,i}} = \underbrace{\begin{bmatrix} \boldsymbol{t}_z^T(0) \\ \boldsymbol{t}_{\dot{z}}^T(0) \\ \boldsymbol{t}_{\ddot{z}}^T(0) \\ \boldsymbol{t}_{\ddot{z}}^T(T_s) \end{bmatrix}}_{\boldsymbol{B}_{z,i}} \boldsymbol{p}_{z,i} \quad . \tag{8}$$

Here, i denotes the index of the considered step and $\boldsymbol{b}_{z,i}$, $\boldsymbol{B}_{z,i}$ and $\boldsymbol{p}_{z,i}$ denote the corresponding boundary condition vector, boundary condition mapping matrix and vertical polynomial parameter vector, respectively. The first two elements in $\boldsymbol{b}_{z,i}$ imply that CoM position and velocity at the beginning of stance equal the CoM state at touch-down. The other two elements say that the acceleration of the CoM at the beginning and the end of the stance phase equals minus gravity, meaning that the vertical leg force is zero. The general solution of the linear system $\boldsymbol{B}_{z,i}\, \boldsymbol{p}_{z,i} = \boldsymbol{b}_{z,i}$ is

$$\boldsymbol{p}_{z,i} = \underbrace{\boldsymbol{B}_{z,i}^T (\boldsymbol{B}_{z,i}\, \boldsymbol{B}_{z,i}^T)^{-1}\, \boldsymbol{b}_{z,i}}_{\boldsymbol{p}_{z,i,0}} + \boldsymbol{r}_{z,i}\, \tilde{p}_{z,i} \, . \tag{9}$$

The nullspace base vector $\boldsymbol{r}_{z,i}$ ensures that $\boldsymbol{B}_{z,i}\, \boldsymbol{r}_{z,i} = 0$. The whole (one-dimensional) nullspace of $\boldsymbol{B}_{z,i}$ is represented by the scalar variable $\tilde{p}_{z,i}$. The vector $\boldsymbol{r}_{z,i}$ is computed as

$$\boldsymbol{r}_{z,i} = \begin{bmatrix} -\boldsymbol{B}_{z,i,square}^{-1}\, \boldsymbol{b}_{z,i,final} \\ 1 \end{bmatrix} \quad , \tag{10}$$

where $\boldsymbol{b}_{z,i,final}$ is the last column in $\boldsymbol{B}_{z,i}$, while $\boldsymbol{B}_{z,i,square}$ consists of all other columns. With $\boldsymbol{p}_{z,i}$ and (2), the vertical CoM state of the i-th take-off (at end of stance) is

$$z_{TO,i} = \boldsymbol{t}_z^T(T_s)\, \boldsymbol{p}_{z,i} \tag{11}$$

$$\dot{z}_{TO,i} = \boldsymbol{t}_{\dot{z}}^T(T_s)\, \boldsymbol{p}_{z,i} \quad . \tag{12}$$

Equation (8) encodes the four *linear* previously described vertical boundary conditions. The fifth boundary condition that we aim to fulfill is the apex height $z_{apex,i}$ of the CoM during the upcoming flight phase (see Fig. 3). With (3) and (5), the i-th apex height can be computed as

$$z_{apex,i} = z_{TO,i} + \frac{\dot{z}_{TO,i}^2}{2g} \, . \tag{13}$$

We are looking for a parameter vector $\boldsymbol{p}_{z,i}$ that will result in the desired apex height $z_{apex,i,des}$, which can be computed as

$$z_{apex,i,des} = z_{floor,i+1} + \Delta z_{apex,des} \, . \tag{14}$$

Note that here we use the height $z_{floor,i+1}$ of the upcoming step. Inserting (11) and (12) into (13) leads to a quadratic equation in the unknown scalar variable $\tilde{p}_z$

$$\begin{aligned} 0 &= \frac{\boldsymbol{t}_z^T \boldsymbol{r}_{z,i}}{2g}\,\tilde{p}_{z,i}^2 + (\boldsymbol{t}_z^T \boldsymbol{r}_{z,i} + \frac{\boldsymbol{t}_z^T \boldsymbol{p}_{z,i,0}\;\boldsymbol{t}_z^T \boldsymbol{r}_{z,i}}{g})\,\tilde{p}_{z,i} + \\ &+ \frac{(\boldsymbol{t}_z^T \boldsymbol{p}_{z,i,0})^2}{2g} - z_{apex,i,des} \end{aligned} \tag{15}$$

It can be shown that the only valid solution (yielding positive vertical take-off velocities) to (15) is

$$\begin{aligned} \tilde{p}_{z,i} &= \frac{2\,\dot{z}_{TD,i} - gT_s - \Gamma}{4\,T_s^3}\,, \\ \Gamma &= \sqrt{g(gT_s^2 - 4\,\dot{z}_{TD,i}\,T_s + 8\,(z_{apex,i,des} - z_{TD,i}))} \end{aligned} \tag{16}$$

B. Horizontal planning and boundary conditions

In Newton's 2nd law $[\ddot{x},\ddot{y},\ddot{z}]^T = \frac{1}{m}\,[f_{com,x}, f_{com,y}, f_{com,z}]^T$, each spatial direction is independent. In our framework, the derivation for the x- and y-component is equivalent. We use the letter χ to indicate horizontal quantities, i.e. $\chi \in \{x,y\}$. We choose - for each previewed contact phase - the following five linear horizontal boundary conditions:

$$\underbrace{\begin{bmatrix} \chi_{TD,i} \\ \dot{\chi}_{TD,i} \\ 0 \\ 0 \\ \chi_{TD,i+1,des} \end{bmatrix}}_{\boldsymbol{b}_{\chi,i}} = \underbrace{\begin{bmatrix} \boldsymbol{t}_\chi^T(0) \\ \boldsymbol{t}_{\dot{\chi}}^T(0) \\ \boldsymbol{t}_{\ddot{\chi}}^T(0) \\ \boldsymbol{t}_{\ddot{\chi}}^T(T_s) \\ \boldsymbol{t}_\chi^T(T_s) + T_{f,i}\,\boldsymbol{t}_{\dot{\chi}}^T(T_s) \end{bmatrix}}_{\boldsymbol{B}_{\chi,i}} \boldsymbol{p}_{\chi,i} \quad . \tag{17}$$

Here, $\boldsymbol{b}_{\chi,i}$, $\boldsymbol{B}_{\chi,i}$ and $\boldsymbol{p}_{\chi,i}$ denote the horizontal boundary condition vector, boundary condition mapping matrix and polynomial parameter vector, respectively. As in Sec. V-A, the first two elements of $\boldsymbol{b}_{\chi,i}$ imply that the initial CoM state is equal to the CoM touch-down state. The next two elements assure that initial and final CoM acceleration are zero, i.e. horizontal leg forces are zero. The fifth element specifies the horizontal CoM touch-down position $\chi_{TD,i+1,des}$ of the upcoming step. Since - in case of no perturbations - the horizontal velocity during flight is constant, we can propagate the take-off state to each upcoming touch-down position via

$$\chi_{TD,i+1,des} = \chi_{TO,i} + T_{f,i}\,\dot{\chi}_{TO,i} = (\boldsymbol{t}_{\chi,i}^T(T_s) + T_{f,i}\,\boldsymbol{t}_{\dot{\chi}}^T(T_s))\,\boldsymbol{p}_{\chi,i} \quad , \tag{18}$$

Here, the i-th time of flight $T_{f,i}$ is computed via (6). Note: $z_{TO,i}$ and $\dot{z}_{TO,i}$ are computed from the vertical polynomial parameter vector $\boldsymbol{p}_{z,i}$. Thus, the vertical boundary conditions are solved before the horizontal ones. A more simple option as compared to CoM touch-down position would be to specify the horizontal take-off velocity, which might be used by a purely velocity-based controller.

The general solution of (17) is

$$\boldsymbol{p}_{\chi,i} = \underbrace{\boldsymbol{B}_{\chi,i}^T(\boldsymbol{B}_{\chi,i}\,\boldsymbol{B}_{\chi,i}^T)^{-1}\,\boldsymbol{b}_{\chi,i}}_{\boldsymbol{p}_{\chi,i,0}} + \boldsymbol{r}_{\chi,i}\,\tilde{p}_{\chi,i} \quad . \tag{19}$$

The nullspace base vector $\boldsymbol{r}_{\chi,i}$ is computed via the equivalent of (10). The horizontal directions have one more polynomial parameter than the vertical one - and thus one more degree of freedom (DOF). This DOF, represented by the scalar $\tilde{p}_{\chi,i}$ in (19), has an effect on the geometry of the leg force rays in space (see Fig. 4). Our goal is to find the value for $\tilde{p}_{\chi,i}$, which produces the best possible focusing of leg forces, such that these are best feasible for finite-sized (or even point-) feet. To this end, - for each previewed step - we compute the time-dependent intersection point $\boldsymbol{x}_{int,i} = [x_{int,i}, y_{int,i}, z_{floor,i}]$ of the leg force $\boldsymbol{F}_{leg}$ with the floor and minimize the integral of the mean square of the deviation from its mean value $\overline{x}_{int,i}$. For a given time in stance t_s, it's horizontal components are

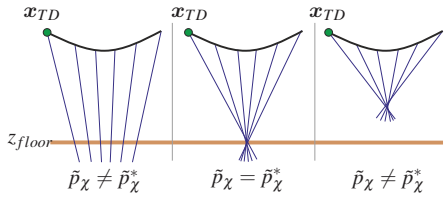

Fig. 4: Effect of $\tilde{p}_\chi$ on force ray focusing (lines of action).

$$\begin{aligned} \chi_{int,i}(t_s) &= \chi(t_s) - \frac{f_{leg,\chi,i}(t_s)}{f_{leg,z,i}(t_s)}\,(z(t_s) - z_{floor,i}) \\ &= \underbrace{(\boldsymbol{t}_\chi^T(t_s) - \frac{(\boldsymbol{t}_z^T(t_s)\boldsymbol{p}_{z,i} - z_{floor,i})\,\boldsymbol{t}_{\ddot{\chi}}^T(t_s)}{\boldsymbol{t}_{\ddot{z}}^T(t_s)\boldsymbol{p}_{z,i} + g})}_{\boldsymbol{d}_i^T(t_s)}\,\boldsymbol{p}_{\chi,i} \quad . \end{aligned} \tag{20}$$

Here, $f_{leg,\chi,i}(t_s)$ and $f_{leg,z,i}(t_s)$ are the horizontal and vertical components of the leg force $\boldsymbol{F}_{leg}$ and $z(t_s)$ is the height of the CoM. The horizontal components of the mean intersection point $\overline{\boldsymbol{x}}_{int,i} = [\overline{x}_{int,i}, \overline{y}_{int,i}, z_{floor,i}]$ can be computed via

$$\overline{\chi}_{int,i} = \frac{1}{T_s}\int_{t_s=0}^{T_s} \chi_{int,i}(t_s)\,dt_s = \underbrace{\frac{1}{T_s}\int_{t_s=0}^{T_s} \boldsymbol{d}_i^T(t_s)\,dt_s}_{\boldsymbol{e}_i^T}\,\boldsymbol{p}_{\chi,i} \quad . \tag{21}$$

Here, $\boldsymbol{e}_i^T$ is a constant row vector. The deviation of the i-th time-varying intersection point from its mean value is

$$\Delta\chi_{int,i}(t_s) = \chi_{int,i}(t_s) - \overline{\chi}_{int,i} = \underbrace{(\boldsymbol{d}_i^T(t_s) - \boldsymbol{e}_i^T)}_{\boldsymbol{k}_i^T(t_s)}\,\boldsymbol{p}_{\chi,i} \quad . \tag{22}$$

The square of the deviation at a given time t_s is

$$\Delta\chi_{int,i}^2(t_s) = \boldsymbol{p}_{\chi,i}^T\,\underbrace{\boldsymbol{k}_i(t_s)\,\boldsymbol{k}_i^T(t_s)}_{\boldsymbol{L}_i(t_s)}\,\boldsymbol{p}_{\chi,i} \quad . \tag{23}$$

In order to obtain the mean square of the deviation $\chi_{int,i,ms}$ we once again integrate and insert (19) to achieve

$$\begin{aligned} \chi_{int,i,ms} &= \boldsymbol{p}_{\chi,i}^T\,\underbrace{\frac{1}{T_s}\int_{t_s=0}^{T_s} \boldsymbol{L}_i(t_s)\,dt_s}_{\boldsymbol{M}_i}\,\boldsymbol{p}_{\chi,i} = \\ &= \boldsymbol{r}_{\chi,i}^T\boldsymbol{M}_i\,\boldsymbol{r}_{\chi,i}\,\tilde{p}_{\chi,i}^2 + 2\,\boldsymbol{r}_{\chi,i}^T\boldsymbol{M}_i\,\boldsymbol{p}_{\chi,i,0}\,\tilde{p}_{\chi,i} + \boldsymbol{p}_{\chi,i,0}^T\,\boldsymbol{M}_i\,\boldsymbol{p}_{\chi,i,0} \quad . \end{aligned} \tag{24}$$

Due to the nonlinearity of (20), solving for $\boldsymbol{M}_i$ analytically is computationally expensive. Instead, $\boldsymbol{d}_i^T$, $\boldsymbol{e}_i^T$, $\boldsymbol{k}_i^T$, $\boldsymbol{L}_i$ and $\boldsymbol{M}_i$ are approximated numerically by evaluating the above

equations for n_{approx} time samples equally spread along the stance period. That way, the integrals turn into sums, which highly facilitates computation. We found that $n_{approx} = 10$ yields sufficient accuracy. Now, with the approximate matrix $\boldsymbol{M}_{i,approx}$ and differentiating (24) with respect to $\tilde{p}_{\chi,i}$, we find the optimal parameter

$$\tilde{p}^*_{\chi,i} = -\frac{\boldsymbol{r}^T_{\chi,i}\ \boldsymbol{M}_{i,approx}\ \boldsymbol{p}_{\chi,i,0}}{\boldsymbol{r}^T_{\chi,i}\ \boldsymbol{M}_{i,approx}\ \boldsymbol{r}_{\chi,i}}\ , \tag{25}$$

which minimizes the mean square deviation as defined above. With (25), (19) turns into

$$\boldsymbol{p}_{\chi,i} = \underbrace{(\boldsymbol{I} - \frac{\boldsymbol{r}_{\chi,i}\ \boldsymbol{r}^T_{\chi,i}\ \boldsymbol{M}_{i,approx}}{\boldsymbol{r}^T_{\chi,i}\ \boldsymbol{M}_{i,approx}\ \boldsymbol{r}_{\chi,i}})}_{\boldsymbol{\Omega}_i}\ \underbrace{\boldsymbol{B}^T_{\chi,i}(\boldsymbol{B}_{\chi,i}\,\boldsymbol{B}^T_{\chi,i})^{-1}}_{\boldsymbol{B}^+_{\chi,i}}\ \boldsymbol{b}_{\chi,i}\quad . \tag{26}$$

This is the direct mapping from the horizontal boundary conditions/tasks $\boldsymbol{b}_{\chi,i}$ to appropriate polynomial parameter vectors $\boldsymbol{p}_{\chi,i}$ (including best force focus). If - as in [1] - horizontal CoM touch-down target positions (or similarly: take-off velocities) are used as boundary conditions, (26) provides the solution to the problem.

C. Foot step targeting and leg cross-over avoidance

In this paper, we aim at an explicit solution for foot-step targeting. Setting $\overline{\chi}_{int,i} = \chi_{foot,i}$ and $\boldsymbol{e}^T_i = \boldsymbol{e}^T_{i,approx}$ in (21), and with (26), we find the following linear relation

$$\boldsymbol{p}_{\chi,i} = \underbrace{[(\boldsymbol{I} - \boldsymbol{e}^{\oplus}_i\ \boldsymbol{e}^T_{i,approx})\ \boldsymbol{\Omega}_i\ \boldsymbol{\Pi}_i}_{\boldsymbol{A}_{TD,i}},\quad \boldsymbol{e}^{\oplus}_i] \begin{bmatrix} \chi_{TD,i} \\ \dot{\chi}_{TD,i} \\ \chi_{foot,i} \end{bmatrix}\quad . \tag{27}$$

Here, $\boldsymbol{A}_{TD,i}$ maps the i-th touch-down state to $\boldsymbol{p}_{\chi,i}$ and the specific pseudo-inverse $\boldsymbol{e}^{\oplus}_i = \frac{\boldsymbol{\Omega}_i\,\boldsymbol{\pi}_i}{\boldsymbol{e}^T_{i,approx}\,\boldsymbol{\Omega}_i\,\boldsymbol{\pi}_i}$ of $\boldsymbol{e}_{i,approx}$ maps the i-th foot position. The matrix $\boldsymbol{\Pi}_i$ combines the first two column vectors of $\boldsymbol{B}^+_{\chi,i}$, while $\boldsymbol{\pi}_i$ is its final column vector. Due to the linearity of (27), using all previewed desired footholds $\chi_{foot,i}$ (except for the first one) and a terminal condition (e.g. final take-off velocity $\dot{\chi}_{TO,final} = \boldsymbol{0}$) as constraints and propagating each $\boldsymbol{p}_{\chi,i}$ to the upcoming touch-down state $[\chi_{TD,i+1}, \dot{\chi}_{TD,i+1}]^T$ as in (18), we solve for the first foothold $\chi_{foot,1}$ (*control variable*) and all future horizontal parameter vectors $\boldsymbol{p}_{\chi,i}$, which yield perfect tracking of the future desired footholds. For lack of space, the detailed derivation is omitted here. The first and second footholds $\chi_{foot,1}$ and $\chi_{foot,2}$ are known at all times, which facilitates foot trajectory generation. In this work, we implemented the foot trajectories as polynomial splines. The achieved precise foot targeting is particularly interesting for running over 3D stepping stones or other restricted ground surfaces.

An additional feature of precise foothold targeting is that leg cross-over can be explicitly avoided. This feature is especially helpful for running in sharp turns. Therefore, the originally preplanned footholds can be adjusted such that the left foot always passes by the right foot on the left, and vice versa. At the same time, the Euclidean distance of the adjusted footholds from the originally planned ones

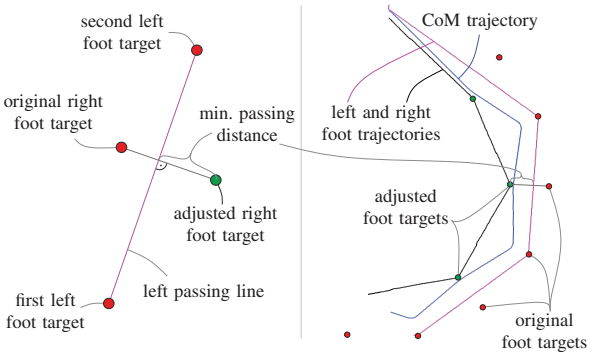

Fig. 5: Leg cross-over avoidance,
left: scheme (depicted for left pass), right: simulation output

should be minimal. This way, the legs can be prevented from twisting around each other. In this paper, to achieve this goal, we use an adjustment heuristic as shown in Fig. 5. In total we preview four foot positions, i.e. two for each foot. The method adjusts the second/third previewed footstep (i.e. the projection method shown in Fig. 5, left, is applied twice), such that the swing feet can safely swing from the first/second foothold to the third/fourth one. The fourth foothold remains unchanged to achieve good long term tracking of the original desired foot locations.

VI. State Feedback Control

In the nominal case (no perturbations), the force profiles and foot aim points as derived in the previous sections assure that - for any initial conditions - after the first stance phase all desired boundary conditions from sections V-A and V-B are perfectly fulfilled (deadbeat control). Therefore, planning once per step or even pre-planning a whole sequence of upcoming steps would be sufficient. Yet, to cope with perturbations, we propose a state feed-back control method, which is based on the continuous re-planning of the desired contact forces (and corresponding polynomial parameters) throughout both flight and stance phases. To this end, during flight - based on the current CoM state ($\boldsymbol{x}$, $\dot{\boldsymbol{x}}$) - the previewed CoM touch-down state is updated by inserting (6) in (3) and (4). In contrast to [1] (no feedback during stance), during stance, the first take-off state is predicted via

$$\begin{bmatrix} \sigma_{TO,1} \\ \dot{\sigma}_{TO,1} \end{bmatrix} = \underbrace{\begin{bmatrix} \sigma \\ \dot{\sigma} \end{bmatrix}}_{feedback} + \underbrace{\begin{bmatrix} \boldsymbol{t}^T_{\sigma}(T_s) - \boldsymbol{t}^T_{\sigma}(t_s) \\ \dot{\boldsymbol{t}}^T_{\sigma}(T_s) - \dot{\boldsymbol{t}}^T_{\sigma}(t_s) \end{bmatrix} \boldsymbol{p}_{\sigma,1}}_{preview}\ ,\quad \sigma \in \{x,y,z\} \tag{28}$$

Here, $\boldsymbol{t}^T_{\sigma}(t)$ and $\dot{\boldsymbol{t}}^T_{\sigma}(t)$ are the time-mapping row vectors from (2). They are evaluated for the total stance time T_s and the current time in stance t_s to predict how much of an offset is expected if for the remaining time in step the current force profile (encoded by $\boldsymbol{p}_{\sigma,1}$) is applied. This offset is added to the current measured state to predict the take-off state, which in turn is used to compute the upcoming CoM touch-down state. This continuous re-planning is facilitated by the low computational demand of the proposed algorithm. Note: after touch-down, the force profile of the current stance

phase is frozen and commanded to the robot as feed-forward. The main advantage of our state feedback during stance is that the foot aim points are continuously updated, such that discontinuities in the foot reference trajectories are avoided.

Note: during flight, the first upcoming foot position is one of the main *control variables*. This means that, whilst all other future footsteps rapidly converge to the desired foot target locations, the nominal position of the first foot is an output of the controller. Thus, - depending on the limitations at hand (e.g. limited allowable supporting area) - this nominal foot aim point may have to be projected to a feasible one, resulting in deviations from the nominal deadbeat behavior.

VII. Guaranteeing Feasibility

The desired three-dimensional force acting on the CoM can be computed for a given time in stance t_s as

$$\boldsymbol{F}_{CoM,des}(t_s) = m \begin{bmatrix} \boldsymbol{t}_{\ddot{x}}^T(t_s)\, \boldsymbol{p}_{x,1} \\ \boldsymbol{t}_{\ddot{y}}^T(t_s)\, \boldsymbol{p}_{y,1} \\ \boldsymbol{t}_{\ddot{z}}^T(t_s)\, \boldsymbol{p}_{z,1} \end{bmatrix} , \tag{29}$$

i.e. the polynomial of the first force profile is evaluated. The corresponding desired leg force $\boldsymbol{F}_{leg,des}$ is found by reordering (1). The polynomial parameters were chosen to result in the best achievable focus of the leg forces with the ground. Yet, for physical robots feasibility is not guaranteed.

1) Point-mass point-feet model: One obvious example is when the robot is modeled as point-mass with point feet. In that case, the leg force is constraint to point along the unit vector $\boldsymbol{u}_{x,f}$ pointing from CoM to point foot. As the other two spatial directions are unactuated, the desired leg force $\boldsymbol{F}_{leg,des}$ has to be projected to the feasible direction[3]:

$$\boldsymbol{F}_{leg,f} = \boldsymbol{u}_{x,f}\boldsymbol{u}_{x,f}^T\boldsymbol{F}_{leg,des} \quad . \tag{30}$$

$\boldsymbol{F}_{leg,f}$ can be commanded to the point-mass point-foot model.

2) Articulated multi-body model: As in our previous work on walking [12], the main idea of our BID control concept is to first focus on the robot's CoM dynamics and the problem of foot placement, which from our point of view are the key challenges of locomotion. The use of a point-mass model can be sufficient to address these issues. Once CoM dynamics and foot placement are solved, they need to be embedded into a more general control framework in order to make them available for articulated multi-body models such as simulated or real humanoid robots. To this end, we use an inverse dynamics based whole-body control framework similar to [23], [25]. It solves a single quadratic program (QP) that tries to satisfy the specified tasks as best as possible while guaranteeing feasibility. The tasks include foot trajectory tracking, upper-body posture control, overall joint posture control and a centroidal momentum task [26], which can be subdivided into a linear and an angular momentum task. Most of the tasks include a task space PD control component. The desired linear force on the CoM from the BID controller (29) directly corresponds to the change in linear momentum and can thus be handled by our linear momentum task formulation. The angular momentum task aims to regulate the robot's overall angular momentum to zero. The foot trajectories from the BID controller form the direct input to the whole-body foot task.

[3]Note: for more complex robots this projection may not be necessary.

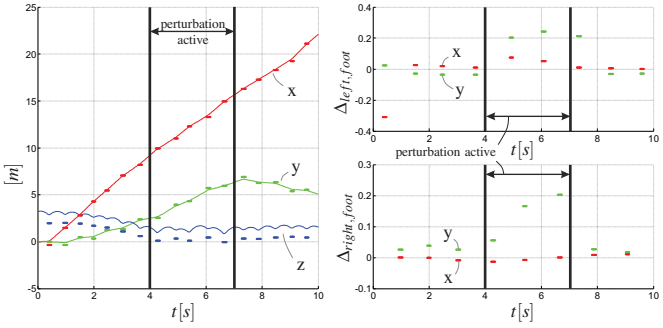

Fig. 6: CoM trajectories and foot target tracking performance in nominal and perturbed case (point mass simulation over stepping stones). Perturbing force: $30N$ in y-direction.

VIII. Simulations

To test the performance and robustness of the proposed control framework, we performed numerous simulations. For the first set of simulations, we considered a point-mass (50kg) with two massless point-feet. Figure 6 shows the result of a simulation in which the point-mass robot was running over three-dimensional stepping stones (see Fig. 1). The left subplot shows the robot's foot positions (bars, only active during stance) and CoM positions (continuous curves). The right subplots show the difference between the desired and achieved foot positions. Nominally, the foot target positions are tracked well, whereas in case of perturbations they deviate. This is necessary to stabilize the CoM motion against the unknown perturbation. After the perturbation is removed, good tracking is regained after a single step.

In another simulation, the point-mass robot was perturbed by an unknown lateral force of 10.000 N, i.e. 200 times the robot's weight. The controller handled even this perturbation successfully. However, the required leg length was up to 78 meters. This is unrealistic for a real robot, but it shows the very high robustness of the controller.

The controller is most sensitive against strong unknown perturbations that point towards the ground. Here, the maximum permanent force the controller could withstand in simulation (for $\Delta z_{apex,des} = 1.1m$, $\Delta z_{TD,des} = 1m$ and $T_s = 0.15s$) was $-1000N$, i.e. double the robot's weight. For higher forces, the robot's CoM would hit the ground.

To proof the applicability of the biologically inspired deadbeat (BID) control framework, we embedded it into a QP-based whole-body controller (as described in Sec. VII-.2) and performed full-body simulations of the humanoid robot Toro [27] in OpenHRP [28]. Fig. 7 shows Toro ($76.4kg$) running at $1.8m/s$. The stance time was set to $0.1s$, the desired touch-down and apex height were $0.87m$ and $0.95m$, respectively. The target velocity ramped up from $0m/s$ to $1.8m/s$ until second 3 and then stayed constant. The chosen

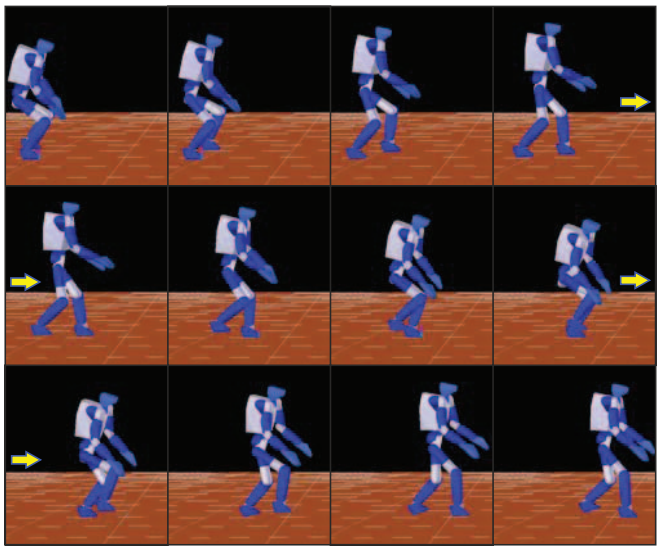

Fig. 7: Toro [27] running in OpenHRP [28] at 1.8 m/s.

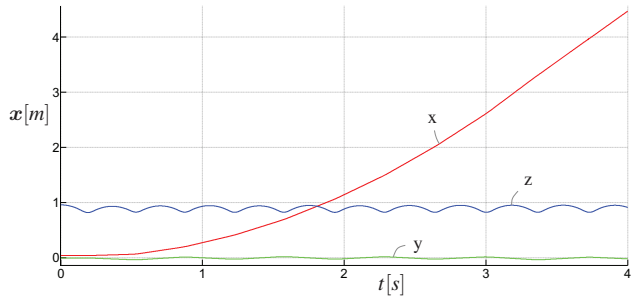

Fig. 8: Toro's CoM position during running (simulation)

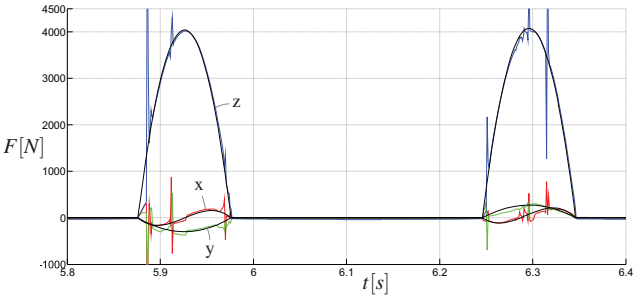

Fig. 9: Comparison of desired (black) and achieved (colored) GRF for Toro running in OpenHRP.

parameters resulted in a steady-state flight duration of about $0.258s$ and a step length of $0.643m$ (stride length $1.285m$). Fig. 8 shows Toro's CoM components over time. The CoM motion is very steady and follows the desired velocity reference nicely (the actual reference were corresponding foot targets, as described in Sec. V-C). Fig. 9 shows two subsequent ground reaction force profiles. Aside from some jerks that are related to the rather stiff contact model, the desired force profiles from the BID controller are tracked quite well. This simulation demonstrates that the embedding of reduced model controllers into whole-body controllers can be sufficient to get running to work with multi-body dynamics. It has to be noted though that Toro's actual hardware limits (such as joint speed and torque) were not considered in this simulation. Our BID control framework assumes impact-free state transitions (compare Fig. 2). The OpenHRP simulations show the controller's robustness against impact losses.

IX. Discussion and Outlook

For our simulations, we used a standard PC (3.3 GHz, quad-core, Win7 64bit). Using Matlab/Simulink, we were able to compute 1 simulated second per 0.78 real second (excluding whole-body QP) for a sampling time of 1ms, i.e. real-time computation of the BID controller is achieved. On a real-time system, even faster computation is expected.

Although the method proposed in [12] (based on Divergent Component of Motion (DCM), a.k.a. Capture Point) handles a different form of locomotion, namely walking, on closer inspection its overall control framework shows similarities with the BID controller proposed in this paper. The first analogy is the preview of several (typically three to four) future footsteps and the derivation of feasible force profiles that nominally track them. The second analogy is related to the modulation and potential projection of the desired forces, such that they comply with the contact constraints. In case of DCM control this modulation/projection consists of leg force modulation and projection of a desired center of pressure (CoP) to the feasible foot supporting area, respectively. The proposed BID controller, in comparison, modulates the first upcoming stance foot position and all previewed leg force profiles, while projecting the foot position to a feasible one in case of limited allowable contact area (see Fig. 10).

Note: The BID algorithm may also be applied to problems such as hopping and jumping. We also expect quadrupedal gaits such as galloping and trotting could be achieved.

As future work, we want to explicitly limit the required leg length, extend the algorithm to jumping over obstacles and show push-recovery (similar to [23]) with Toro running in OpenHRP. Also, we plan to implement the BID controller on our recently developed planar compliant leg robot.

X. Conclusion

In this paper, we extended the Biologically Inspired Deadbeat (BID) controller from [1] to achieve bipedal running on three-dimensional stepping stones. To this end, the original method was adjusted to achieve explicit foot targeting. Additionally, an explicit method for leg cross-over avoidance was introduced. Just like in the original version [1], the proposed controller has deadbeat properties, i.e. in the nominal case it reaches the desired boundary conditions after just one stance phase. The controller facilitates agile, precise and versatile running motions and is very robust against external perturbations. Additionally, we embedded the BID controller into a QP-based whole-body controller (similar to [23]) to achieve running with the humanoid Toro [27] in simulation.

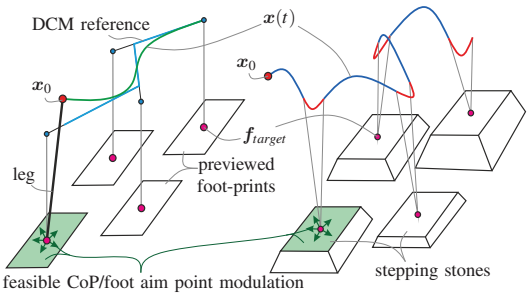

Fig. 10: Analogy of DCM (→ walking) and BID (→ running); stepping stones can be compared with finite-sized feet.

Acknowledgements

The authors want to thank the staff from Autonomous Systems Lab (ETH Zürich) for organizing the summer school "Dynamic Walking and Running with Robots" in 2011. This summer school gave the inspiration for this work.

Also, we cordially thank Roy Müller and Christian Rode from Department of Motion Science (Jena, Germany) for providing the human experimental data.

This research is partly supported by the Initiative and Networking Fund of Helmholtz Association through a Helmholtz Young Investigators Group (Grant no. VH-NG-808).

References

[1] J. Englsberger, P. Kozlowski, and C. Ott, "Biologically inspired deadbeat controller for bipedal running in 3d," in *IEEE/RSJ Int. Conf. on Intell. Robots and Systems (accepted)*, 2015.

[2] S. Collins, A. Ruina, R. Tedrake, and M. Wisse, "Efficient bipedal robots based on passive dynamic walkers," *Science*, vol. 307, no. 5712, pp. 1082–1085, 2005.

[3] M. Vukobratovic and Y. Stepanenko, "On the stability of anthropomorphic systems," *Mathematical Biosciences*, vol. 15, pp. 1–37, 1972.

[4] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Biped walking pattern generation by using preview control of zero-moment point," in *IEEE Int. Conf. on Robotics and Automation*, 2003, pp. 1620–1626.

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

[6] K. Nishiwaki and S. Kagami, "Sensor feedback modification methods that are suitable for the short cycle pattern generation of humanoid walking," in *Int. Conf. on Intell. Robots and Systems*, 2007, pp. 4214–4220.

[7] S. Kajita, M. Morisawa, K. Miura, S. Nakaoka, K. Harada, K. Kaneko, F. Kanehiro, and K. Yokoi, "Biped walking stabilization based on linear inverted pendulum tracking," in *Int. Conf. on Intell. Robots and Systems*, 2010, pp. 4489–4496.

[8] T. Takenaka, T. Matsumoto, and T. Yoshiike, "Real time motion generation and control for biped robot, 1st report: Walking gait pattern generation," in *Int. Conf. on Intell. Robots and Systems*, 2009.

[9] T. Koolen, T. D. Boer, J. Rebula, A. Goswami, and J. E. Pratt, "Capturability-based analysis and control of legged locomotion. part 1: Theory and application to three simple gait models," *Int. J. of Robotics Research*, vol. 31, no. 9, pp. 1094–1113, 2012.

[10] J. Urata, K. Nshiwaki, Y. Nakanishi, K. Okada, S. Kagami, and M. Inaba, "Online walking pattern generation for push recovery and minimum delay to commanded change of direction and speed," in *Int. Conf. on Intell. Robots and Systems*, 2012.

[11] Y. Zhao and L. Sentis, "A three dimensional foot placement planner for locomotion in very rough terrains," in *IEEE-RAS Int. Conf. on Humanoid Robots*, 2012, pp. 726–733.

[12] J. Englsberger, C. Ott, and A. Albu-Schäffer, "Three-dimensional bipedal walking control based on divergent component of motion," *Robotics, IEEE Transactions on*, vol. 31, no. 2, pp. 355–368, 2015.

[13] R. Tedrake, S. Kuindersma, R. Deits, and K. Miura, "A closed-form solution for real-time zmp gait generation and feedback stabilization," under review. [Online]. Available: http://groups.csail.mit.edu/robotics-center/public_papers/Tedrake15.pdf

[14] H. Geyer, A. Seyfarth, and R. Blickhan, "Compliant leg behaviour explains basic dynamics of walking and running," *Proceedings of the Royal Society B: Biological Sciences*, pp. 2861–2867, 2006.

[15] T. Takenaka, T. Matsumoto, T. Yoshiike, T. Hasegawa, S. Shirokura, H. Kaneko, and A. Orita, "Real time motion generation and control for biped robot -4th report: Integrated balance control-," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, 2009.

[16] S. Cotton, I. M. C. Olaru, M. Bellman, T. van der Ven, J. Godowski, and J. Pratt, "Fastrunner: A fast, efficient and robust bipedal robot. concept and planar simulation," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 2358–2364.

[17] D. Lakatos, C. Rode, A. Seyfarth, and A. Albu-Schäffer, "Design and control of compliantly actuated bipedal running robots: Concepts to exploit natural system dynamics," in *IEEE-RAS Int. Conf. on Humanoid Robots*, 2014.

[18] B. Dadashzadeh, H. Vejdani, and J. Hurst, "From template to anchor: A novel control strategy for spring-mass running of bipedal robots," in *Int. Conf. on Intell. Robots and Systems*, 2014.

[19] S. G. Carver, N. J. Cowan, and J. M. Guckenheimer, "Lateral stability of the spring-mass hopper suggests a two-step control strategy for running," *Chaos*, vol. 19, no. 2, 2009.

[20] H. R. Vejdani, Y. Blum, M. A. Daley, and J. W. Hurst, "Bio-inspired swing leg control for spring-mass robots running on ground with unexpected height disturbance," *Bioinspiration and Biomimetics*, vol. 8, no. 4, p. 046006, 2013.

[21] A. Wu and H. Geyer, "The 3-d spring-mass model reveals a time-based deadbeat control for highly robust running and steering in uncertain environments," *Robotics, IEEE Transactions on*, vol. 29, no. 5, pp. 1114–1124, Oct 2013.

[22] D. Koepl and J. Hurst, "Impulse control for planar spring-mass running," *Journal of Intelligent & Robotic Systems*, vol. 74, no. 3-4, pp. 589–603, 2014.

[23] P. M. Wensing and D. E. Orin, "High-speed humanoid running through control with a 3d-slip model," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, 2013, pp. 5134–5140.

[24] H.-W. Park, S. Park, and S. Kim, "Variable-speed quadrupedal bounding using impulse planning: Untethered high-speed 3d running of mit cheetah 2," in *IEEE Int. Conf. on Robotics and Automation*, 2015, pp. 5163–5170.

[25] T. Koolen, J. Smith, G. Thomas, S. Bertrand, J. Carff, N. Mertins, D. Stephen, P. Abeles, J. Englsberger, S. McCrory, J. van Egmond, M. Griffioen, M. Floyd, S. Kobus, N. Manor, S. Alsheikh, D. Duran, L. Bunch, E. Morphis, L. Colasanto, K.-L. H. Hoang, B. Layton, P. Neuhaus, M. Johnson, and J. Pratt, "Summary of team ihmc's virtual robotics challenge entry," in *Int. Conf. on Humanoid Robots*, 2013.

[26] D. E. Orin, A. Goswami, and S.-H. Lee, "Centroidal dynamics of a humanoid robot," *Auton. Robots*, vol. 35, no. 2-3, pp. 161–176, 2013.

[27] J. Englsberger, A. Werner, C. Ott, B. Henze, M. A. Roa, G. Garofalo, R. Burger, A. Beyer, O. Eiberger, K. Schmid, and A. Albu-Schäffer, "Overview of the torque-controlled humanoid robot toro," in *IEEE-RAS Int. Conf. on Humanoid Robots*, 2014.

[28] F. Kanehiro, K. Fujiwara, S. Kajita, K. Yokoi, K. Kaneko, H. Hirukawa, Y. Nakamura, and K. Yamane, "Open architecture humanoid robotics platform," in *IEEE Int. Conf. on Robotics and Automation*, 2002.