A Reactive Controller Framework for Quadrupedal Locomotion on Challenging Terrain

Victor Barasuol*, Jonas Buchli†‡, Claudio Semini‡, Marco Frigerio‡, Edson R. De Pieri*, Darwin G. Caldwell‡

*PPGEAS - Dept. of Automation and Systems, Federal University of Santa Catarina (UFSC), Florianpolis, SC, Brazil 88040-970
victor@das.ufsc.br

†Agile & Dexterous Robotics Lab, ETH Zurich, Tannenstr. 3, 8092 Zürich
buchlij@ethz.ch

‡ Dept. of Advanced Robotics, Istituto Italiano di Tecnologia (IIT), via Morego, 30, 16163 Genova
<first name>.<last name>@iit.it

***Abstract*—We propose a reactive controller framework for robust quadrupedal locomotion, designed to cope with terrain irregularities, trajectory tracking errors and poor state estimation. The framework comprises two main modules: One related to the generation of elliptic trajectories for the feet and the other for control of the stability of the whole robot. We propose a task space CPG–based trajectory generation that can be modulated according to terrain irregularities and the posture of the robot trunk. To improve the robot's stability, we implemented a null space based attitude control for the trunk and a push recovery algorithm based on the concept of capture points. Simulations and experimental results on the hydraulically actuated quadruped robot HyQ will be presented to demonstrate the effectiveness of our framework.**

I. Introduction

Agile locomotion of legged robots over rough terrain requires all elements - from trajectory planning to control - to work in a well orchestrated manner. The different elements, e.g. planning and control should not interfere with each other. For example, a kinematic plan for the legs should never make the legs try to penetrate the ground. If ground contact occurs unexpectedly the kinematic plans have to be adjusted immediately. At the same time, the kinematic locomotion pattern should, as much as possible, not be disturbed by the controller responsible to keep the robot's trunk upright.

While there is a lot of work addressing single aspects of the overall locomotion gait planning and control problem, solutions that take all these elements together in a systematic and coherent fashion are rare.

In this contribution we present a reactive gait generation and control framework for a quadruped robot that has the following main goals:

- Creation of stable omni-directional periodic gait
- Robustness against disturbances from uneven ground and external forces on the trunk
- Foot slip avoidance
- Reduction of impact forces at the feet

To achieve these goals we make extensive use of kinematic and dynamic models of our hydraulic quadruped robot HyQ [1] and exploit the high performance torque-control available at all the joints [2] [3]. The resulting control framework exhibits the following features:

1 Avoid trajectories that would penetrate the ground (avoid high ground reaction forces – GRF);
2 Avoid weak contact or loss of contact (avoid slippage);
3 Avoid undesired leg internal forces (avoid slippage and waste of energy);
4 Reduce disturbances between joint position and trunk attitude controllers;
5 Reduce disturbances at the trunk due to poor state estimation (avoid excessive GRF);
6 Increase the locomotion robustness with respect to unexpected terrain irregularities (avoid excessive GRF);

Our main contributions include: a simple, yet reactive gait pattern generation; the introduction of the so-called *horizontal frame* to derive the equations of the control blocks, which allows to effectively decouple the foot trajectory planning from the trunk attitude control; the extension of the capture point approach on a quadruped robot addressing not only linear but also rotational disturbances (about the yaw axis).

We present simulations and experimental results demonstrating the performance of our controller framework on the HyQ robot, Fig. 1. A detailed description of the robot hardware can be found in [1]. All the experiments are executed on the robot performing a dynamic gait on rough terrain, namely a walking trot, and undergoing severe external perturbations (e.g. strong pushes on the trunk).

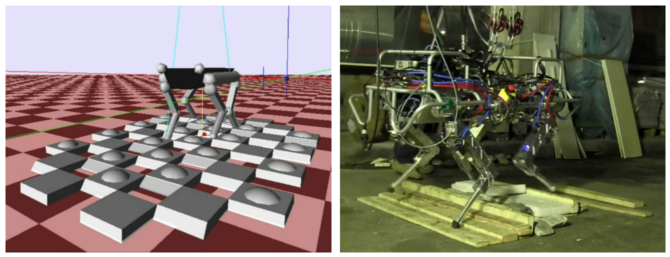


Fig. 1. IIT's hydraulic quadruped robot HyQ on rough terrain. *Left:* Robot model in simulation environment (SL [4]); *Right:* Picture of robot during a test run.

Related work

Extensive research has been conducted in the field of floating base kinematics [5] and dynamics [6], [7], [8] and our approach partially builds on this work. However, in our work, we specifically address the reactive generation of the locomotion pattern rather than the underlying whole body and floating base control problems.

Boston Dynamics' quadruped robots *BigDog* [9] and *LS3* have shown impressive locomotion performance in several online videos in the past years. However, no details on the hardware design and control methods have been published to date. The performances of the underlying control algorithms are therefore hard to verify and compare to other approaches.

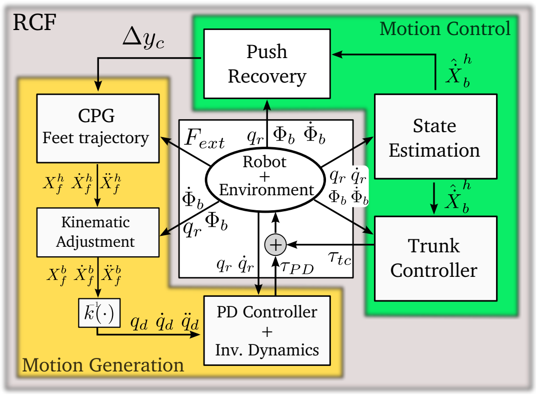
Autonomous Locomotion through rough terrain has recently been the focus of the DARPA Learning Locomotion Challenge (cf. IJRR special issue [10]). In [11] a rigid body model based controller has been shown to allow to lower the error feedback controller gains improving the robustness for walking over rough terrain. However, the therein presented approaches make use of a high precision terrain map, extensive foothold search and kinematic motion planning and mostly focus on statically stable locomotion, while we focus on reactive footstep planning in absence of a terrain map.

In this paper, we propose a push recovery algorithm based on the concept of *N-step capturability*, described in [12]. This concept has been used by some authors for push recovery and generation of trajectories in bipeds, by modeling the robots with simple linear models. For example, the 3D Linear Inverted Pendulum [13], the Linear Inverted Pendulum plus Flywheel [14], the Linear Inverted Pendulum with finite-size foot and reactive mass [15]. However, all these models do not consider the effect of rotational motion, which is relevant for the long trunk of a quadruped. Moreover, an analysis for balance recovery in quadrupeds based on N-step capturability is still missing in the literature.

Central pattern generators observed in animals have been a major source of inspiration for trajectory generation in legged robots [16], [17]. In robotics, the majority of CPG-inspired methods for trajectory generation is applied in joint space [18]. However, feet trajectories mapped into joint space are complex signals that cannot be modeled well by few harmonics. In addition, the relationship between the parameters of the generator in joint space and the gait features (e.g. step height and length) are very non-intuitive. To overcome such drawbacks of CPG-inspired methods in joint space, some authors proposed to use a Cartesian space CPG [19]. In that work the authors used a neural network model in which the parameters are still non-intuitive, have no independent effect on the feet trajectory and require a mapping analysis to be tuned.
Our contribution is a CPG-inspired foot task space trajectory generator with a very simple structure where all the parameters have intuitive meaning and can be adjusted with independent effect on the feet trajectories.

II. Reactive Controller Framework

The Reactive Controller Framework (RCF) presented in this work consists of two main parts: the *Motion Generation* and the *Motion Control*. Both of them comprise three functional blocks which will be detailed in Section III and IV, respectively. Fig. 2 illustrates the layout of the various control blocks along with the main information flows between such blocks and the robot/environment.



RCF
Δy_c
Push Recovery
Motion Control
$\hat{\dot{X}}_b^h$
CPG Feet trajectory
State Estimation
F_{ext} q_r Φ_b $\dot{\Phi}_b$
Robot + Environment
X_f^h $\dot{X}_f^h$ $\ddot{X}_f^h$
q_r $\dot{q}_r$ Φ_b $\dot{\Phi}_b$
$\dot{\Phi}_b$ q_r Φ_b
$\hat{\dot{X}}_b^h$
Kinematic Adjustment
Trunk Controller
X_f^b $\dot{X}_f^b$ $\ddot{X}_f^b$
q_r $\dot{q}_r$ τ_{PD} τ_{tc}
$k^{-1}(\cdot)$
q_d $\dot{q}_d$ $\ddot{q}_d$
PD Controller + Inv. Dynamics
Motion Generation

Fig. 2. Overview diagram of the Reactive Controller Framework (RCF), highlighting the main functional blocks and the information flows. The small block with k^{-1} represents the inverse kinematics routine. All the other variables and the blocks are explained in Section III and IV.

An important element of our framework is the *horizontal frame*, which we will use throughout the whole paper. A horizontal frame is a reference frame whose xy plane is always horizontal (i.e. orthogonal to the gravity vector $\vec{g}$), such that the projection of its x axis on the horizontal plane is parallel to the same projection of the x axis of the robot (that is, the horizontal frame has the same yaw angle as the robot, with respect to the world frame). A horizontal frame can be attached to the robot (it is then said to be floating), or fixed somewhere in the environment, as illustrated in Fig. 3.

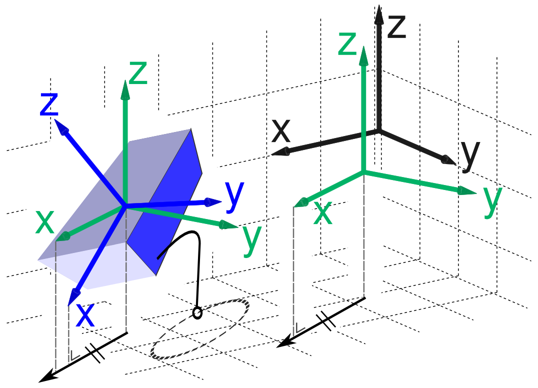


Fig. 3. Horizontal reference frames (in green) and the robot frame (in blue – the parallelepiped represents the robot trunk; see also Fig. 5); horizontal frames share the same yaw angle with respect to the world reference frame (in black).

Choosing such a horizontal frame as the coordinate frame for motion generation and control provides several advantages. In general, it makes the trajectory generation of the CPG block independent from the trunk attitude, therefore the influence of the trunk attitude controller on the feet trajectories is minimized. This feature is very important for improved locomotion stability and for push recovery, as we will show in Section V.

III. Motion Generation

The purpose of this part of the framework is to generate stereotypic and reactive motions for the feet. The most important sub-modules are: a CPG-inspired trajectory generator that provides elliptical trajectories for the feet (the *primitives*), which dynamically adapt to the terrain profile; a kinematic adjustment function that corrects the feet trajectory according to the actual trunk attitude; a joint space controller which tracks the desired trajectories. These modules are detailed in the following subsections.

A. CPG-inspired trajectory generation

Our approach for the generation of the reference trajectories for the feet is loosely inspired by the CPGs of animals. Ellipse-shaped trajectories are generated by a network of four non-linear oscillators, whose state represents the Cartesian coordinates of each foot [20].

We extend our previous work by adding non-linear filters coupled to the output of the network of oscillators. During each leg's swing phase the filter output tracks the output of its corresponding oscillator. The filters also receive information about the foot contact on the ground, allowing them to adapt the trajectories according to the actual terrain profile. This is achieved by *cutting* the ellipses, as explained later in this section. The shape of the adapted trajectories are illustrated in Fig. 4 on the right.

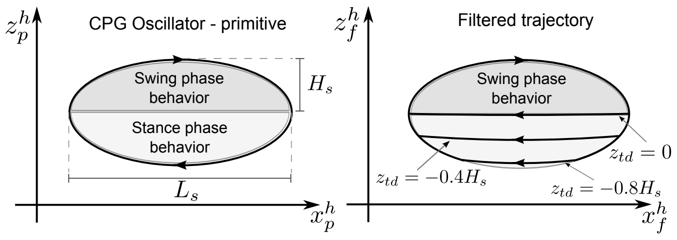


Fig. 4. The foot trajectory generated by the CPG oscillator (on the left) and the trajectory modulated by the non linear filter (on the right). L_s and H_s are respectively the length and the height of a single step; z_{td} is the filter parameter which determines where the original elliptic trajectory has to be interrupted (to start the stance phase). The h on the axes labels stands for horizontal frame.

The equations governing the network of oscillator inspired by the CPG are the following:

$$\dot{x}_{p_i}^h = \alpha \left(1 - \frac{4\bar{x}_i^2}{{L_s}^2} - \frac{\bar{z}_i^2}{H_{s_i}^2}\right)\bar{x}_i + \frac{w_{si}L_s}{2H_{s_i}}\bar{z}_i \tag{1}$$

$$\dot{y}_{p_i}^h = \beta\,(\bar{y}_i + \Delta y_{ci}) \tag{2}$$

$$\dot{z}_{p_i}^h = \gamma \left(1 - \frac{4\bar{x}_i^2}{{L_s}^2} - \frac{\bar{z}_i^2}{H_{s_i}^2}\right)\bar{z}_i - \frac{w_{si}2H_{s_i}}{L_s}\bar{x}_i + \sum \mathbb{C}_{ij}\frac{H_{s_i}}{H_{s_j}}\bar{z}_j \tag{3}$$

$$w_{si} = \pi\frac{V_f}{L_s}\left(\frac{D_f}{1-D_f}\sigma_{p_{1i}}(\bar{z}_i) + \sigma_{p_{2i}}(\bar{z}_i)\right) \tag{4}$$

$$\sigma_{p_{1i}}(\bar{z}_{p_i}) = (e^{-b_p\bar{z}_{p_i}} + 1)^{-1} \tag{5}$$

$$\sigma_{p_{2i}}(\bar{z}_{p_i}) = (e^{b_p\bar{z}_{p_i}} + 1)^{-1} \tag{6}$$

where $x_{p_i}^h$, $y_{p_i}^h$ and $z_{p_i}^h$ are the i-th oscillator outputs that compose the position reference vector $X_{p_i}^h = [x_{p_i}^h \; y_{p_i}^h \; z_{p_i}^h]^T$ of the i-th leg expressed in the horizontal frame. Each ellipse is positioned in the horizontal frame by using the variables $\bar{x}_i = x_{p_i}^h - x_{p0_i}^h$, $\bar{y}_i = y_{p_i}^h - y_{p0_i}^h$ and $\bar{z}_i = z_{p_i}^h - z_{p0_i}^h$, where $(x_{p0_i}^h, y_{p0_i}^h, z_{p0_i}^h)$ are the coordinates of the i-th ellipse origin. The four main parameters that can be set by the user (or a higher level controller) are the following: the step length L_s, the step height H_s, the step duty factor D_f and forward velocity V_f. The angular frequency w_s is calculated as $w_s = \pi V_f/L_s$. The angular velocity of the limit cycle, during stance and swing phases, is changed according to the functions $\sigma_{p_{1i}}(\bar{z}_{p_i})$, $\sigma_{p_{2i}}(\bar{z}_{p_i})$ and the duty factor. The constant b_p changes the transition rate of $\sigma_{p_{1i}}(\bar{z}_{p_i})$ and $\sigma_{p_{2i}}(\bar{z}_{p_i})$. α, β and γ affect the convergence rate to the limit cycle.

The last term in (3) is the coupling term that allows independent modulation of each step height without disturbing the synchronization. The gait pattern is selected according to the coupling matrix $\mathbb{C}$ (see [21] for trot, walk, bound and pace gaits). All simulations and experiments presented in this paper are based on a trot.

The output filter is written as

$$\dot{X}_{f_i}^h = (\dot{X}_{p_i}^h + K_c(X_{p_i}^h - X_{f_i}^h))\sigma_{f_{1i}}(\bar{z}_{pi}) - V_i\sigma_{f_{2i}}(\bar{z}_{pi}) \tag{7}$$

$$\sigma_{f_{1i}}(\bar{z}_{p_i}) = (e^{-b_f(\bar{z}_{p_i} - z_{td_i})} + 1)^{-1} \tag{8}$$

$$\sigma_{f_{2i}}(\bar{z}_{p_i}) = (e^{b_f(\bar{z}_{p_i} - z_{td_i})} + 1)^{-1} \tag{9}$$

The functions $\sigma_{f_{1i}}(\bar{z}_i)$ and $\sigma_{f_{2i}}(\bar{z}_i)$ are responsible for switching the behaviour of the limit cycle between swing and stance phases according to the touchdown position. During stance phase, the filter output becomes $\dot{X}_{f_i}^h = -V$, where V is computed to provide omni-directional motion. The higher the value of the constants b_p and b_f, the faster the transitions between swing and stance phase behaviours. The idea of using exponential functions to achieve smooth transitions was first introduced by [21], [22].

The *step depth* parameter z_{td} affects the reshaping of the trajectory by determining at which height the ellipse has to be interrupted, as depicted in Fig. 4 on the right.
If a terrain map is available the swing to stance transition can be planned in advance, reducing the impact forces. In absence of a map (i.e. the robot is walking *blindly*), the feet trajectories can be dynamically adjusted as soon as touchdown is detected; this feature makes the locomotion more robust also with respect to poor state estimation.
During the execution of the trajectory, the foot touchdown event is recognized (e.g. by force sensors, simple binary switch sensors, or more complex estimators fusing different data). z_{td} is consequently adjusted to match the actual step height/depth (for bumps or holes, respectively) and the filter changes the shape of the trajectory.

B. Kinematic adjustment

The kinematic adjustment has fundamental importance within the whole RCF. This algorithm transforms the generated desired foot trajectory from the horizontal frame (X_f^h,$\dot{X}_f^h$,$\ddot{X}_f^h$)

to the robot base frame (X_f^b,$\dot{X}_f^b$,$\ddot{X}_f^b$), using the information about the actual attitude of the trunk (i.e. the robot base), Φ and $\dot{\Phi}$.
Having a dedicated, specific module for this purpose allows the CPG module to be independent of Φ and $\dot{\Phi}$, effectively decoupling the corresponding controllers that therefore do not conflict or *fight* each other. Such a separation reflects the different nature of the two problems (generation of periodic trajectories and taking care of the attitude) and is also effective from the software implementation point of view.

This block changes the trajectories according to the body inclination so that each origin of the CPG ellipse *lies close* to the ground. This reduces the risk of a weak or missed contact.

C. PD controller and inverse dynamics

The actual joint space controllers used on the robot to track the desired trajectories are regular PD position and torque controllers plus a floating–base inverse dynamics routine providing feed–forward commands [6]. The inverse dynamics block allows to lower the gains of the PD joint position controller resulting in a low virtual joint stiffness (beneficial to reduce disturbances between trunk and feet) without compromising the tracking performance. Both components are implemented in our simulation and real–time control software, SL [4]. A detailed description of this block would go beyond the focus of this paper.

IV. Motion control

The purpose of the motion control block is to induce actions that allow better control of the trunk motion and reject external disturbances created by unexpected terrain irregularities and external forces applied to the trunk.

This section presents the three algorithms of the motion control block: the *trunk controller* that affects the trunk attitude and motion during the stance phase; the *push recovery* that estimates footholds for the swing legs which will lead to a corrective reaction during stance; the *state/velocity estimation* that computes the translational velocities of the trunk, which are used by the other two algorithms.

A. Trunk controller

The purpose of this control block is to provide joint commands that result in the application of a certain force to the trunk of the robot, for example to correct its attitude. The foundation of the algorithm lies in the computation of the Jacobian matrix that gives the velocities of the feet according to the velocities of the joints *and* of the floating base [23]. This Jacobian is obtained by the derivation of the forward kinematics expressed in a fixed horizontal frame (see Section II).
This formulation uses the following definitions:

- $q_{r_i} \in \mathbb{R}^{n\times 1}$: vector of joint positions for leg i (n is the number of joints per leg);
- $\Phi_b \in \mathbb{R}^{3\times 1}$: vector of orientation angles of the base (roll, pitch and yaw), with respect to the horizontal frame (yaw is always 0);
- $R_b^h(\Phi_b)$: rotation matrix from the base frame to horizontal frame;
- $X_{f_i}^b \in \mathbb{R}^{3\times 1}$: vector of foot coordinates in the base frame (which we can write as $k(q_{r_i})$, where k is the forward kinematics function);
- $X_{f_i}^h \in \mathbb{R}^{3\times 1}$: foot coordinates in the fixed horizontal frame;
- $X_b^h \in \mathbb{R}^{3\times 1}$: base coordinates in the fixed horizontal frame.
- $J_b(q_{r_i}) \in \mathbb{R}^{3\times n}$: Jacobian that relates the foot velocity to the joint velocities of leg i in the base frame. For simplicity, $J_b(q_{r_i})$ will be written as J_{b_i};
- $M(\Phi_b, q_{r_i}) \in \mathbb{R}^{3\times 3}$: matrix that relates the foot velocity of leg i to the body angular velocities in the horizontal frame.

For each foot i of the robot we can write:

$$X_{f_i}^h = X_b^h + R_b^h X_{f_i}^b \tag{10}$$

Differentiating (10) with respect to time, yields:

$$\dot{X}_{f_i}^h = \dot{X}_b^h + \dot{R}_b^h X_{f_i}^b + R_b^h \dot{X}_{f_i}^b \tag{11}$$

$$\dot{X}_{f_i}^h = \dot{X}_b^h + \frac{\partial R_b^h}{\partial \Phi_b}\dot{\Phi}_b k(q_{r_i}) + R_b^h J_{b_i}\dot{q}_{r_i} \tag{12}$$

$$\dot{X}_{f_i}^h = [R_b^h J_{b_i} \quad I \quad M(\Phi_b, q_{r_i})][\dot{q}_{r_i}^T \quad (\dot{X}_b^h)^T \quad \dot{\Phi}_b^T]^T \tag{13}$$

$$\dot{X}_{f_i}^h = J_{h_i}(\Phi_b, q_{r_i}) \; [\dot{q}_{r_i}^T \quad {}^h\dot{X}_b^T \quad \dot{\Phi}_b^T]^T \tag{14}$$

The actual Jacobian used for trunk control ($J_H(\Phi_b, q_r)$, or simply J_H) is built by stacking only the J_{h_i} associated with the stance legs, therefore the number of rows changes. For instance, for a quadruped robot with all the legs in stance phase we have $J_H \in \mathbb{R}^{[12\times 4n+6]}$. The structure of J_H is such that we can write:

$$\underbrace{[\dot{X}_{f_1}^T \dots \dot{X}_{f_j}^T]^T}_{\dot{X}_f^h} = J_H(\Phi_b, q_r) \; \underbrace{[\dot{q}_{r_1}^T \dots \dot{q}_{r_j}^T \quad {}^h\dot{X}_b^T \quad \dot{\Phi}_b^T]^T}_{\dot{q}_h} \tag{15}$$

where j is the number of stance legs.

The user or a high level locomotion controller chooses a vector of desired forces $\Upsilon_{h_{des}} \in \mathbb{R}^{(jn+6)\times 1}$; a typical example are the forces to be applied to the trunk to compensate for a tilted attitude.
The idea is to then map this generalized forces into torque commands $\tau_{tc} \in \mathbb{R}^{n\times 1}$ for the joints, trying not to move the feet positions in the horizontal frame. Therefore such torques are extracted from the projection of the desired force vector $\Upsilon_{h_{des}}$ into the null space of J_H^T [7]:

$$\tau_{tc} = S(I - J_H^T J_H^{+T})\Upsilon_{h_{des}} \tag{16}$$

where $S \in \mathbb{R}^{(nj)\times(nj+6)}$ is the selection matrix that preserves only the torques associated to the actuated joints.

The matrix J_H^+ in (16) is the right generalized inverse of J_H:

$$J_H^+ = W^{-1} J_H^T (J_H W^{-1} J_H^T)^{-1} \tag{17}$$

In our approach we select a weighting matrix W ($\in \mathbb{R}^{(jn+6)\times(jn+6)}$) which results in the minimization of the floating-base forces and therefore in a more effective use of the actuated joints.

B. Push recovery based on capture points

The purpose of the push recovery algorithm is to dampen out disturbances that cause undesirable lateral and rotational motion of the trunk. The idea is to find proper footholds which naturally counteract the disturbances and make the robot stop. We based our analysis on the concept of *N-step capturability*, described in [12]; it considers the states and actions that allow a legged system to eventually come to a stop and it provides a metric about the probability of the robot to fall. In our case we are interested in the *instantaneous capture points*, which allow the robot to stop with a single step.

Our contribution consists of a linear model that considers two stance legs and allows to calculate instantaneous capture points according to the yaw motion and the lateral velocity. The proposed model, illustrated in Fig. 5, is a quadruped model described in the horizontal frame and simplified by considering massless legs and no roll and pitch motion.

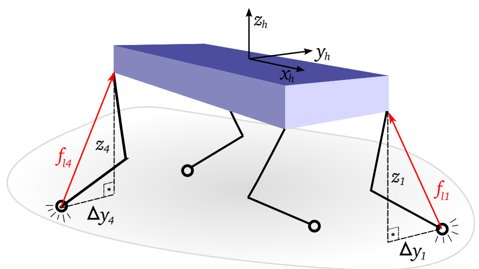


Fig. 5. Simplified quadruped model with massless legs, showing the robot frame attached to the geometric center of the trunk. The red arrows represent the linear forces exerted on the trunk by the stance legs.

The resulting equations of motion for this simplified model are given by:

$$m_b \ddot{y}_b = -f_{l1}\frac{\Delta y_1}{l_1} - f_{l4}\frac{\Delta y_4}{l_4} \quad (18)$$

$$m_b \ddot{z}_b = -f_{l1}\frac{z_1}{l_1} - f_{l4}\frac{z_4}{l_4} - m_b g \quad (19)$$

$$I_{bz}\ddot{\psi}_b = -r f_{l1}\frac{\Delta y_1}{l_1} + r f_{l4}\frac{\Delta y_4}{l_4} \quad (20)$$

$$I_{by}\ddot{\theta}_b = -r f_{l1}\frac{z_1}{l_1} + r f_{l4}\frac{z_4}{l_4} \quad (21)$$

where m_b is the robot mass and I_{by} and I_{bz} are the rotational inertias around y_h and z_h axes, respectively. The variable z_j is the foot position along the z_h axis, Δy_j is a relative position along the y_h axis between hip j and foot j. The distance between each hip j and the relative contact point is defined as l_j, g is the gravitational acceleration constant and f_{lj} is the linear actuation force on leg j.

Assuming that each hip height is kept constant during the stance phase, which implies $\dot{\theta}_b = \dot{z}_b = 0$, we get a linear system of equations:

$$\ddot{y}_b = \frac{g}{2}\left(\frac{\Delta y_1}{z_{01}} + \frac{\Delta y_4}{z_{04}}\right) \quad (22)$$

$$\frac{I_{bz}}{m_b r}\ddot{\psi}_b = \frac{g}{2}\left(\frac{\Delta y_1}{z_{01}} - \frac{\Delta y_4}{z_{04}}\right) \quad (23)$$

Adding and subtracting equation (22) from equation (23) yields:

$$\ddot{y}_b + \frac{I_{bz}}{m_b r}\ddot{\psi}_b = g\frac{\Delta y_{c1}}{z_{01}} \quad (24)$$

$$\ddot{y}_b - \frac{I_{bz}}{m_b r}\ddot{\psi}_b = g\frac{\Delta y_{c4}}{z_{04}} \quad (25)$$

From equations (24) and (25) we can then independently derive a conserved quantity, called Orbital Energy [24], relative to the motion of each hip associated to the leg j:

$$E_{hip_j} = \frac{1}{2}(\dot{y}_b + \frac{I_{bz}}{m_b r}\dot{\psi}_b)^2 + \frac{1}{2}\frac{g\Delta y_{cj}^2}{z_{0j}} \quad for \quad j = 1, 2 \quad (26)$$

$$E_{hip_j} = \frac{1}{2}(\dot{y}_b - \frac{I_{bz}}{m_b r}\dot{\psi}_b)^2 + \frac{1}{2}\frac{g\Delta y_{cj}^2}{z_{0j}} \quad for \quad j = 3, 4 \quad (27)$$

If $E_{hip_j} = 0$, then the hip j will come to rest over the foot j. Solving the equations (26) and (27) for zero orbital energy results in the Δy_c quantities that match the instantaneous capture points for each foot j:

$$\Delta y_{cj} = \sqrt{\frac{-z_{0j}}{g}}(\dot{y}_b + \frac{I_{bz}}{m_b r}\dot{\psi}_b) \quad for \quad j = 1, 2 \quad (28)$$

$$\Delta y_{cj} = \sqrt{\frac{-z_{0j}}{g}}(\dot{y}_b - \frac{I_{bz}}{m_b r}\dot{\psi}_b) \quad for \quad j = 3, 4 \quad (29)$$

C. Trunk state estimation for translational velocities

The performance of the motion control techniques described above heavily depends on the quality of the estimation of the base state. Quantities like the lateral velocities, required by the push recovery module, are the most critical ones since we do not have a direct measurement of them (as opposed to angular velocities that are directly measured by the gyro-sensor of the Inertia Measurement Unit (IMU)).
Since both the push recovery and the trunk control work with coordinates in the horizontal frame, it is sensible to use the same frame for the state estimation. Our IMU provides estimates of the angular velocities and the orientation of the robot's trunk already in the horizontal frame (e.g. roll and pitch angles express how tilted a body is with respect to a horizontal reference).

To estimate the trunk lateral velocities we use the general expression (12) assuming that the feet are naturally constrained by the friction forces during the stance phase, i.e. $\dot{X}^h_{f_i} = 0$. This procedure is analogous to what was adopted in [25] and provides the estimation of the trunk translational velocities in the horizontal frame:

$$\hat{\dot{X}}^h_b = J_{Hest}(\Phi_b, q_{r_i})[\dot{q}^T_{r_1} \dots \, \dot{q}^T_{r_j} \;\; \dot{\Phi}^T_b]^T \quad (30)$$

where $J_{Hest} \in \mathbb{R}^{jn \times jn+3}$ is the Jacobian for state estimation.

V. Simulation and Experimental Results

In this section we present a series of results to show the robot balance improvement achieved by the proposed RCF. More specifically, these results show the balance and locomotion performance by fusing a push recovery algorithm, an adaptive trajectory generation and a horizontal frame kinematic adjustment.

Both simulation and experimental results were performed on our torque-controlled quadruped robot platform (named HyQ [1]). It stands 1 m tall, weighs around 70 Kg and is currently tethered to an external power supply. Each of the 12 actuated revolute joints feature a range of motion of 120^o. While the hip abduction/adduction joints are actuated by DC brushless electric motors, the hip and knee flexion/extension joints are driven by fast and strong hydraulic cylinders. Information about angular position and velocity of the trunk are provided by an embedded IMU (MicroStrain 3DM-GX3-25). Currently our robot detects the foot touchdown by monitoring the foot forces estimated by the foot Jacobian and the joint torques.

A. Simulation results

The balance improvement due to the push recovery algorithm is evaluated by subjecting the robot's trunk to different rotational and lateral disturbances. Rotational (yaw) disturbances were created by applying different constant torque values to the robot trunk creating positive moments around the z_h axes. Lateral disturbances are applied perpendicularly to the center of the trunk's side. Each push is applied as a step input of one second duration.

The robot receives the disturbances during a trot gait and always at the beginning of the swing phase of the same pair of diagonal legs. The gait features a step frequency of 1.65 Hz, a duty factor of 0.55 and a step height of 8 cm. The desired torques $\Upsilon_{h_{des}}$ used in the trunk controller (see Section IV-A) are computed according to a PD action on the robot roll and pitch angular errors. The desired roll and pitch angles are zero. The response for rotational and lateral disturbances are shown separately.

First, to analyse the yaw disturbance rejection three different constant torques were applied to the robot's trunk. The total robot yaw displacements due to each applied torque, with and without push recovery, are shown in Table I.

TABLE I
Push recovery results showing yaw displacement angles after rotational disturbance

Mode\Torque	100 Nm	200 Nm	300 Nm
Push R. ON	23.5^o	47.0^o	60.7^o
Push R. OFF	31.5^o	60.2^o	85.4^o

The results in Table I show that the push recovery algorithm is able to reduce the yaw displacement up to 29%. These results indicate that the push recovery cannot stop all the yaw motion, which was expected since the push recovery has no effect during the swing phase (and thus at the beginning of the applied disturbance of one second).

Next, to analyze the lateral push recovery performance we show the robot's ability to keep balance by turning on/off blocks in the RCF. We change the simulation sets by: turning on/off the push recovery algorithm (PR) to show its contribution for lateral balancing; turning on/off the kinematic adjustment (KA) to show its benefits to the push recovery; and turning on/off both algorithms. Many disturbance trials were simulated by pushing the robot laterally with constant force during one second. Three pushing force values were applied (400 N, 500 N and 600 N) for each one of the above cases, as shown in the three plots of Fig. 6.

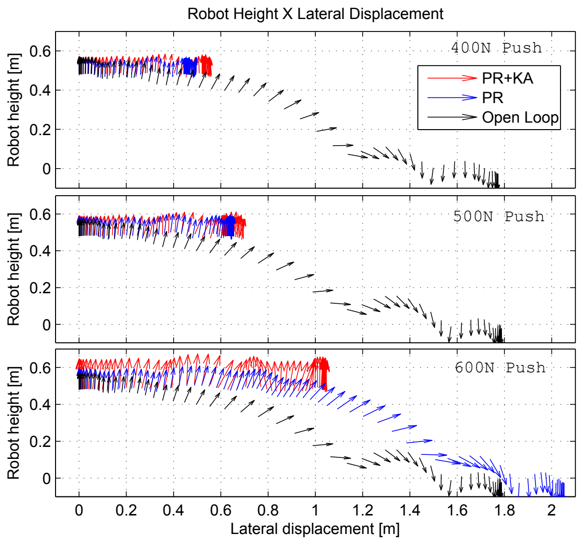


Fig. 6. Simulation results illustrating the push recovery response for three different trunk push forces. The arrows represent the z axis of the robot base frame illustrating the sequence of the trunk's motion (lateral displacement, robot height and roll angle) after the lateral force disturbance. The three simulated cases are: Push recovery + Kinematic adjustment (red), Push recovery only (blue) and neither (black). Arrow sequences that reach zero robot height show a falling robot and therefore lost balance.

This figure also shows that the robot falls down for all three pushing forces when the PR is off. Actually, a robot in a narrow stance (left and right legs almost parallel) is very sensitive to lateral disturbances. In simulation, the robot in narrow stance can only hold lateral pushes up to 120 N if the PR is disabled. With the PR turned on, all disturbances can successfully be coped with, showing a substantial increase in motion robustness. In theory, with push recovery based on capture points, a robot starts falling only when the instantaneous capture points are out of the robot's workspace or if the joint position controllers are not able to track the desired trajectory.

Up to a 500 N push, the robot with PR or PR+KA presents similar stabilization. However, around 600 N, the PR is not sufficient to keep the balance and at that point the benefits of the KA (and thus the horizontal frame) become noticeable. A hard lateral push excites the roll motion due to the azimuthal lever arm between the foot position and the point where the force was applied. If the trajectories were generated in the robot base frame, instead of using the horizontal frame approach, the roll motion would drive the generated trajectories to penetrate the ground. In this case, the foot reaches the ground in a foothold that is earlier than the predicted instantaneous capture point. As a consequence, the robot becomes incapable of cancelling the lateral motion and the roll motion continues, leading to a fall.

B. Experimental results

In the experiments we assessed the improvements in robot balance during push recovery tests and trotting on irregular terrain.

For the push recovery test we use a force sensor ATI Mini45 to measure the level of disturbance forces applied to the robot. The sensor is mounted on a bar laterally fixed to the robot's trunk. With the robot IMU we acquire the roll angular position as the output signal of our experiment. The results are presented in Fig. 7.

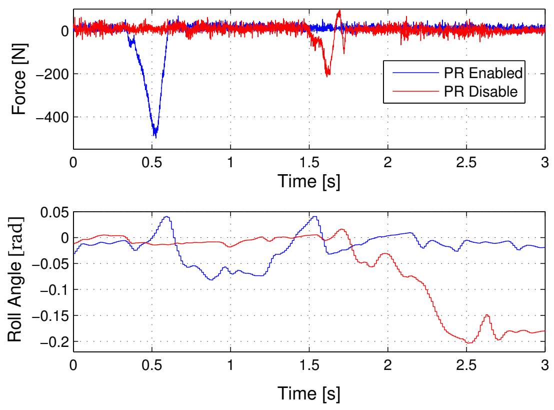


Fig. 7. Experimental results of the lateral push tests: Lateral external disturbance forces (top) and roll angle of the trunk (bottom) shown for push recovery (PR) enabled (blue) and disabled (red). Much larger absolute disturbance forces (at around $t = 0.5s$) can be tolerated when the PR controller is on, compared to a smaller force (at around $t = 1.6s$) that leads to a growing absolute roll angle (the robot starts to fall) if the PR is off. Note that from around $t = 2.5s$ the safety harness pulls the robot to prevent it from falling.

As predicted in the simulated results, the robot without the push recovery skills is very sensitive to lateral pushes and is not able to keep the balance even for low level of push forces.

To evaluate the effectiveness of the adaptive trajectories, the robot's navigation skills are tested on a challenging terrain made with battens, pieces of foam and stones. The obstacles are up to $10cm$ high, which represents around 25% of the maximum leg extension range. The robot trot features step height $H_s = 12cm$, step length $L_s = 12cm$, duty factor $D_f = 0.55$ and desired forward velocity $V_f = 0.35m/s$ (resulting in a step frequency of $1.65Hz$). It is important to mention that there is no yaw heading control on the robot.

We qualitatively evaluated the trotting robustness with and without the step depth variables (z_{td_i}) that adapts the feet trajectories. The non-adaptive trotting experiments are performed by fixing the step depth value equal to zero ($z_{td_i} = 0$), corresponding to the assumption of a flat terrain. The evaluation assesses the trajectory of the robot while crossing the terrain. To have a measurement of the robot trajectory we integrate the estimated translational velocities of the robot (see Section IV-C). The results are presented in Fig. 8.

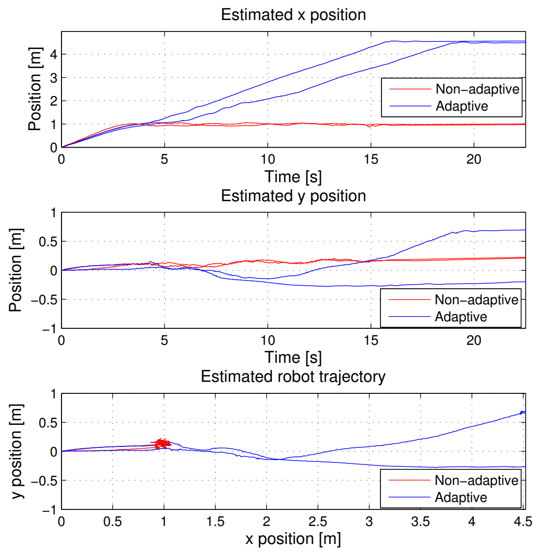


Fig. 8. Experimental results of the rough terrain tests: The three plots show trials with the foot trajectory adaptation enabled (blue, 2 trials) and disabled (red). The top and center plot show the estimated world frame x and y position versus time. The bottom plot shows a bird's-eye view of the trials, illustrating that during the non-adaptive trial the robot was not able to advance beyond $x = 1m$.

The red line in Fig. 8 shows that the robot is not able to cross the terrain without adaptive trajectory generation. When the feet trajectories are not adapted, the reaction forces appear proportionally to the height of the obstacles. For a trot without trajectory adaptation, the ground reaction forces opposing the direction of forward velocity are so strong that the robot cannot move forward.

On the other hand, the blue line in Fig. 8 illustrates that the robot is able to cross the terrain with the adaptive generation turned on. By enabling this adaptation, each z_{td_i} is not fixed as $z_{td_i} = 0$ anymore, but is set at each foot touchdown moment instead (as described in Section III-A). The adaptive action tries to adjust the trajectories during stance phase according to the terrain surface. This action leads to a substantial reduction in the generated ground reaction forces that point in the opposite direction of the robot's desired motion (consequently reducing the disturbances transmitted to the trunk). Therefore, the generation of adaptive trajectories allows to substantially improve the robot's capability to cross irregular terrains.

VI. Discussion and Conclusion

In this paper we proposed a Reactive Controller Framework for quadrupedal locomotion, comprising algorithms for both the generation of periodic yet reactive feet trajectories and for the stabilization of the whole robot.
We introduced the horizontal reference frame for deriving the algebra of our controllers, since it allows to effectively

decouple the generation of the feet trajectories and the control of the trunk motion (with focus on the attitude).

The contributions of this work include: a CPG-inspired trajectory generator for the feet. This CPG works in the task space of the feet and its parameters reflect intuitively some of the main gait parameters (such as the step height and length). The CPG is capable of smoothly adapting to unexpected terrains. We applied the concept of *N-point capturability* to a real quadruped robot, extending it to cope also with rotational disturbances to the trunk, about the yaw axis. Furthermore, we implemented a trunk motion control based on the null space of the Jacobian that relates the feet velocities and the robot velocities at the joints and at the trunk.

A wide set of experimental results, both in simulation and with a real hydraulic quadruped robot, have demonstrated the capabilities of our control framework in terms of the robustness for locomotion. This includes the capability of traversing challenging terrain even without any terrain maps.

Future works include further developments in the *capturability* analysis, to be able to include also disturbances for the roll and the pitch angles.
A challenging and interesting development will be to integrate our approach with other frameworks addressing different issues, like higher level planning and vision-based state estimation, into a sound and coherent architecture.

Appendix – Video Contents

The video shows experimental (EXP) and a few simulation (SIM) results: introduction scenes (EXP), the horizontal frame (EXP), rough terrain (EXP+SIM) and push recovery trials (EXP+SIM).

Acknowledgment

This research has been funded by the Fondazione Istituto Italiano di Tecnologia. J.B. is supported by a Swiss National Science Foundation professorship.

The authors would like to thank CAPES for the scholarship granted to V. Barasuol (Grant Procs. 6463-11-8). The authors would like to thank also the colleagues that collaborated for the success of this project: Hamza Khan, Jake Goldsmith, Thiago Boaventura, Michele Focchi, Ioannis Havoutis and our team of technicians.

References

[1] C. Semini, N. G. Tsagarakis, E. Guglielmino, M. Focchi, F. Cannella, and D. G. Caldwell, "Design of HyQ - a hydraulically and electrically actuated quadruped robot," *Journal of Systems and Control Engineering*, vol. 225, no. 6, pp. 831–849, 2011.

[2] T. Boaventura, C. Semini, J. Buchli, M. Frigerio, M. Focchi, and D. G. Caldwell, "Dynamic torque control of a hydraulic quadruped robot," in *IEEE International Conference in Robotics and Automation*, 2012.

[3] M. Focchi, T. Boaventura, C. Semini, M. Frigerio, J. Buchli, and D. G. Caldwell, "Torque-control based compliant actuation of a quadruped robot," in *12th IEEE International Workshop on Advanced Motion Control (AMC)*, 2012.

[4] S. Schaal, "The SL simulation and real-time control software package," CLMC lab, University of Southern California, Tech. Rep., 2009.

[5] M. Mistry, J. Nakanishi, G. Cheng, and S. Schaal, "Inverse kinematics with floating base and constraints for full body humanoid robot control," in *8th IEEE-RAS International Conference on Humanoid Robots*, December 2008, pp. 22–27.

[6] M. Mistry, J. Buchli, and S. Schaal, "Inverse dynamics control of floating base systems using orthogonal decomposition." in *ICRA*. IEEE, 2010, pp. 3406–3412.

[7] L. Sentis and O. Khatib, "Synthesis of whole-body behaviors through hierarchical control of behavioral primitives," *International Journal of Humanoid Robotics*, vol. 2, no. 4, pp. 505–518, 2005.

[8] M. Hutter, M. Hoepflinger, C. Gehring, M. Bloesch, C. D. Remy, and R. Siegwart, "Hybrid operational space control for compliant legged systems," in *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2012.

[9] M. Raibert, K. Blankespoor, G. Nelson, R. Playter, and the Big-Dog Team, "Bigdog, the rough-terrain quadruped robot," in *Proceedings of the 17th World Congress The International Federation of Automatic Control (IFAC)*, 2008.

[10] J. Buchli, J. Pratt, and N. Roy, "Editorial – special issue on legged locomotion," *Int. J. Robotics Research*, vol. 30, no. 2, 2011.

[11] J. Buchli, M. Kalakrishnan, M. Mistry, P. Pastor, and S. Schaal, "Compliant quadruped locomotion over rough terrain," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, october 2009, pp. 814–820.

[12] T. Koolen, T. de Boer, J. Rebula, A. Goswami, and J. Pratt, "Capturability-based analysis and control of legged locomotion, part 1: Theory and application to three simple gait models," *The International Journal of Robotics Research*, 2012.

[13] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa, "The 3d linear inverted pendulum mode: a simple modeling for a biped walking pattern generation," in *Intelligent Robots and Systems, 2001. Proceedings. 2001 IEEE/RSJ International Conference on*, vol. 1, 2001, pp. 239 –246.

[14] J. Pratt, J. Carff, S. Drakunov, and A. Goswami, "Capture point: A step toward humanoid push recovery," in *Humanoid Robots, 2006 6th IEEE-RAS International Conference on*, dec. 2006, pp. 200 –207.

[15] J. Pratt, T. Koolen, T. De Boer, J. Rebula, S. Cotton, J. Carff, M. Johnson, and P. Neuhaus, "Capturability-based analysis and control of legged locomotion, part 2: Application to m2v2, a lower-body humanoid," *Int. J. Rob. Res.*, vol. 31, no. 10, pp. 1117–1133, 2012.

[16] A. Ijspeert, "Central pattern generators for locomotion control in animals and robots: A review," *Neural Networks*, vol. 21, no. 4, pp. 642–653, May 2008.

[17] Q. Wu, C. Liu, J. Zhang, and Q. Chen, "Survey of locomotion control of legged robots inspired by biological concept," *Science in China Series F: Inf. Sciences*, vol. 52, 2009.

[18] J. Morimoto, G. Endo, J. Nakanishi, and G. Cheng, "A biologically inspired biped locomotion strategy for humanoid robots: Modulation of sinusoidal patterns by a coupled oscillator model," *Robotics, IEEE Transactions on*, vol. 24, no. 1, pp. 185 –191, feb. 2008.

[19] C. J. Liu, Q. J. Chen, and D. W. Wang, "CPG-inspired workspace trajectory generation and adaptive locomotion control for quadruped robots," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions*, 2011.

[20] V. Barasuol, V. J. D. Negri, and E. R. D. Pieri, "Wcpg: A central pattern generator for legged robots based on workspace intentions," *ASME Conference Proceedings*, vol. 2011, no. 54761, pp. 111–114, 2011.

[21] L. Righetti and A. Ijspeert, "Pattern generators with sensory feedback for the control of quadruped locomotion," in *Robotics and Automation, 2008. ICRA 2008.*, May 2008.

[22] S. Rutishauser, A. Sprowitz, L. Righetti, and A. Ijspeert, "Passive compliant quadruped robot using central pattern generators for locomotion control," in *Biomedical Robotics and Biomechatronics, 2008*, Oct. 2008.

[23] A. Shkolnik and R. Tedrake, "Inverse kinematics for a point-foot quadruped robot with dynamic redundancy resolution," in *Robotics and Automation, 2007 IEEE International Conference on*, april 2007, pp. 4331 –4336.

[24] S. Kajita, K. Tani, and A. Kobayashi, "Dynamic walk control of a biped robot along the potential energy conserving orbit," in *Intelligent Robots and Systems '90. 'Towards a New Frontier of Applications', Proceedings. IROS '90. IEEE International Workshop on*, jul 1990, pp. 789 –794 vol.2.

[25] M. Mistry, J. Nakanishi, G. Cheng, and S. Schaal, "Inverse kinematics with floating base and constraints for full body humanoid robot control," in *Humanoid Robots, 2008. Humanoids 2008. 8th IEEE-RAS International Conference on*, dec. 2008, pp. 22 –27.