Central Pattern Generators Augmented with Virtual Model Control for Quadruped Rough Terrain Locomotion

Mostafa Ajallooeian, Soha Pouya, Alexander Sproewitz and Auke J. Ijspeert
Biorobotics Laboratory, École Polytechnique fédérale de Lausanne (EPFL), CH-1015 Lausanne, Switzerland
Email: mostafa.ajallooeian@epfl.ch

***Abstract*—We present a modular controller for quadruped locomotion over unperceived rough terrain. Our approach is based on a computational Central Pattern Generator (CPG) model implemented as coupled nonlinear oscillators. Stumbling correction reflex is implemented as a sensory feedback mechanism affecting the CPG. We augment the outputs of the CPG with virtual model control torques responsible for posture control. The control strategy is validated on a 3D forward dynamics simulated quadruped robot platform of about the size and weight of a cat. To demonstrate the capabilities of the proposed approach, we perform locomotion over unperceived uneven terrain and slopes, as well as situations facing external pushes.**

I. Introduction

The problem of rough terrain locomotion has always been of great interest to roboticists. It is intriguing to understand the underlying mechanism of rough terrain locomotion and to be able to build robots which traverse over rough terrain. To do so, one needs to know how a legged robot can be controlled in irregular environments. In this paper we explore quadruped locomotion control over moderately difficult *unperceived* rough terrain with a dynamically simulated quadruped robot.

Quadruped locomotion research was pushed forward by the seminal work of Mark Raibert [1]. From that point up to now different quadruped robots has been built and tested. This includes Tekken [2], Puppy [3], Kolt [4], Cheetah [5], Rush [6], StarlETH [7], and HyQ [8]. However the majority of these robots where only tested on flat terrain, and only Tekken, StarlETH and HyQ has been tested in outside-the-lab environments[1]. In the same context, one should not forget to mention BigDog [9] as a robot capable of real world rough terrain locomotion, but the details are not publicly accessible.

Quadruped rough terrain locomotion has been recently studied by several groups in the context of the DARPA's learning locomotion project with the LittleDog robot [10], [11]. This project aimed at quadruped rough terrain locomotion facing *perceived* environments with provided detailed information of terrain, and several papers demonstrated appealing results [12]–[15]. As for the *unperceived* rough terrain locomotion, only Buchli et al. [16] explored it with the LittleDog robot. Their approach is composed of a low-gain PD-controller along with force and inverse dynamics control, and they demonstrated rough terrain locomotion over uneven terrain of more than 30% of leg length (ll) variations. They need the dynamics information of the robot such as inertia tensors as well as 3D force sensing for control. The demonstrated results in [16] are for low speed static walking gaits (of about[2] $0.3[\frac{BL}{s}]$).

Quadruped locomotion over *unperceived* rough terrain has also been studied in 3D forward dynamics simulation, but not extensively. One example is the work of Maufroy et al. [17]. They introduced a CPG control with phase modulations based on legs loading/unloading and tested their control approach one slopes and steps. Another example is the work of Coros et al. [18]. Their control approach consists of several modules for foot placement, joint space control, and task space virtual model control. They demonstrate successful quadruped locomotion with different speeds and gaits on flat terrain and they also experiment with stairs of known heights. Their approach is more suited for a computer graphics application as their simulated quadruped consists of more than 50 controlled degrees of freedom.

Here in this paper we present a bio-inspired approach to rough terrain locomotion based on Central Pattern Generators (CPG) [19] and Virtual Model Control (VMC) [20]. CPG models have proved to be useful [2], [21]–[24] for locomotion as they generate smooth rhythmic pattern which are stable against state perturbations. These properties allow for proper integration of sensory feedback signals and smooth modulation of the output signal. We implement stumbling corrective reflex as a sensory feedback mechanism affecting the CPG control. We propose to augment CPG control with VMC, as VMC is a tangible way to do feedback control in locomotion [20]. We use VMC for posture control when performing rough terrain locomotion and/or when facing external perturbations.

The problem of *unperceived* rough terrain locomotion control with dynamic gaits has not been extensively explored. The main contribution of this paper is to take steps towards creating simple to implement locomotion controllers for unperceived rough terrain locomotion. Moreover, we propose a way to correct the torques generated by a CPG model utilizing the VMC concept.

The rest of this document is organized as follows: section II details the proposed approach. We explain the simulated quadruped, the experimental setup and the obtained results in section III. We finally discuss the pros and cons of our approach in section IV. All the equations in this paper follow

[1]We are aware that StarlETH and HyQ are undergoing rough terrain locomotion experiments now.

[2]Estimated from the multimedia attachment of [16]. BL refers to body length (hip to hip distance).

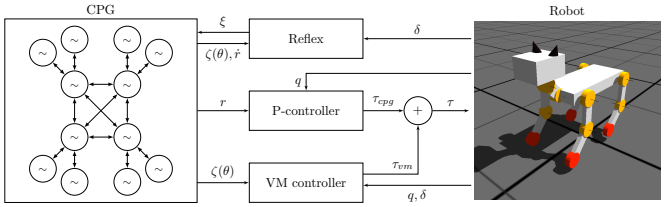

Fig. 1. The proposed architecture for locomotion control. CPG, implemented as coupled oscillators, generates the rhythmic joint angle patterns needed for an open-loop locomotion. The output of the CPG, r, is converted into motor torques, τ_{cpg}, through a P-controller. The reflex mechanism receives the contact sensing information, δ, from the robot and phase information from the CPG and generates reflex feedbacks, ξ, if needed. The CPG torques are augmented with virtual model control torques, τ_{vm}, which are generated based on contact and proprioceptive information received from the robot. The activity of the VM controller can be inhibited by the CPG by $\zeta(\theta)$, a phase-dependent term.

vector calculus, which makes them easier for a vector based implementation.

II. Methodology

We propose a bio-inspired locomotion controller that consists of three modules: 1) CPG generating rhythms; 2) sensory feedback mechanisms generating reflexes; and 3) posture control mechanisms. Similar functional modules have been identified in the organization of locomotion control in vertebrates [25]. There are circuits located in the spinal cord generating synchronized rhythmic patterns (CPG), sensory feedback signals which affect the CPG through proprioceptive and cutaneous afferents, and a connection from the vestibular system to the brain stem and then to the motoneurons affecting motor actions (cf. fig. 1 in [25]). We implement a very simplified functional model of these modules as depicted in Figure 1. The CPG model is implemented as coupled nonlinear oscillators. The outputs of the CPG are converted to motor torques through a P-controller. Reflex mechanism receives cutaneous (touch sensing) signals and generates a stumbling correction reflex affecting the CPG. The posture controller is implemented as a virtual model controller receiving cutaneous and proprioceptive signals and directly generating motor torques needed for posture adjustments. The activity of the virtual model controller can be inhibited by the CPG module if there is a stumbling.

The following subsections will explain the functionality of introduced modules[3]. In the rest of this document the outputs of all modules are $N \times 1$ vectors ($N = 12$). For all vectors, other than the virtual forces in task space, the first 4 elements correspond to the hip adduction/abduction (A/A) joints, the next 4 elements to protraction/retraction (P/R) joints, and the last 4 to knee flexion/extension (F/E) joints. For virtual force vectors in the task space, each consecutive 3 elements relate to x, y and z coordinates respectively, and a {left fore, right fore, left hind, right hind} order is assumed.

[3]Of course all the modules implemented are very simplified and there are many feedback pathways and functionalities which are not included in our control strategy.

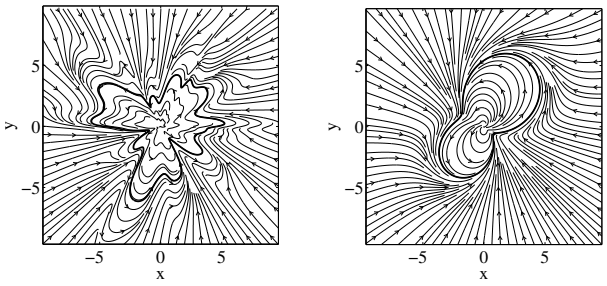

Fig. 2. Example phase portraits generated by phased-based scaled Hopf oscillators. The shape of the limit cycle can be any arbitrary function of phase. Left) $f(\theta) = \cos(4\theta + 0.4545\pi) + \tanh(\cos(10\theta)) + \tanh(10\sin(3\theta)) + 3.7$. Right) $f(\theta) = \sin(\theta) + 2\cos(2\theta - 2) + 3.6$. Note that the oscillator states are θ and r, and in this Cartesian illustration $x = rcos(\theta)$ and $y = r\sin(\theta)$.

A. Central Pattern Generator

We implement our CPG model as coupled nonlinear oscillators. Each oscillator controls the joint angle position of one degree of freedom (DoF) of the simulated robot. Since the desired joint angle profile for each joint might be different and the profile shape is not known in advance, we need nonlinear oscillators which can have arbitrary limit cycle shapes. Thus we introduce phase-based scaled Hopf oscillators:

$$\dot{\theta} = 2\pi\nu + (W \circ \sin(\Delta\Theta - \Phi))\,[1]_{[N\times 1]} \quad (1)$$

$$\Delta\Theta = [\theta]_{[1\times N]} - [\theta^T]_{[N\times 1]} = \begin{bmatrix} 0 & \theta_2-\theta_1 & \dots & \theta_N-\theta_1 \\ \theta_1-\theta_2 & 0 & \dots & \theta_N-\theta_2 \\ \vdots & & \ddots & \\ \theta_1-\theta_N & \theta_2-\theta_N & \dots & 0 \end{bmatrix} \quad (2)$$

$$\dot{r} = 2\pi\nu r \circ \frac{f'(\theta)}{f(\theta)} + \gamma\left([\mu]_{N\times 1} - \frac{r \circ r}{f(\theta) \circ f(\theta)}\right) \circ r + \xi \quad (3)$$

where ν is the frequency of locomotion, θ is the vector of the phases of the oscillators, Φ is the matrix of the desired phase differences, W is the matrix of the couping weights, r is the output vector, $f(\theta)$ is a vectorized phase-based function defining the shape of limit cycle of each DoF, γ is the convergence factor, μ is the radius of the Hopf oscillator's limit cycle, and ξ is the vector of external feedback signals (see section II-B). The $\circ$ operator is the Hadamard (entry-wise) product [26] and the divisions are likewise. $[x]_{[m,n]}$ concatenates m rows and n columns of copies of x in a matrix.

Phase-based scaled Hopf oscillators are very flexible in terms of the limit cycle shape. As long as W is consistent (sum of phase differences in each coupling loop equals $2k\pi, k \in \mathbb{Z}$) and all f are periodic positive C^1 differentiable functions then the oscillators asymptotically converge to the desired limit cycles and the basin of attraction is $\theta \in \mathbb{R}^N, r \in (\mathbb{R}^+)^N$. Example phase portraits generated by phase-based scaled Hopf oscillators are illustrated in Figure 2.

B. Stumbling Correction Reflex

With the right set of parameters, the open-loop CPG generates gaits suitable for flat environments, but correction mechanisms are needed as soon as there are external perturbations.

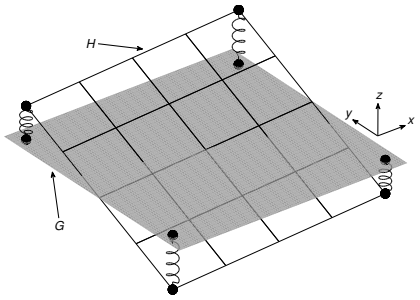

Fig. 3. A schematic figure depicting the virtual springs for attitude control. A virtual plane, H, is attached to the trunk of the robot. Virtual springs are connected between H, and another virtual plane G lying parallel to the ground and passing through robot's trunk. There virtual springs generate virtual forces which will correct the robot's attitude.

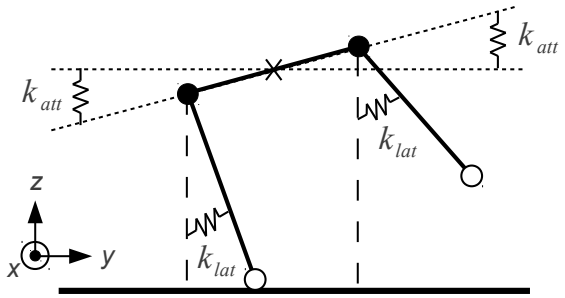

Fig. 4. Virtual components for attitude and lateral angle of attack control from a frontal view. The spring with k_{att} stiffness are responsible for attitude control (frontal view of Figure 3). The springs with k_{lat} stiffness try to keep the lateral angle of attack vertical w.r.t. world coordinates.

One of the important cases that should be handled by a feedback signal is when a swing leg hits an obstacle. If the obstacle is not perceived and a reflex is not initiated, then that swing leg will push into the obstacle and will make the robot prone to fall. There are several studies which indicate that there is a stumbling corrective reflex in animals when a swing leg hits an obstacle. [27] explains that a swing hit evokes an extra flexion which helps passing over the obstacle. We implement the stumbling correction reflex as an impulse feedback to the CPG to flex the knee joint. Additionally, since the CPG is already protracting the leg, we also send a retracting impulse to the P/R joint of the same leg to avoid pushing into the obstacle. This can be formulated as:

$$\xi = k_r \begin{bmatrix} [0]_{[1\times 4]} & [1]_{[1\times 4]} & -[1]_{[1\times 4]} \end{bmatrix}^T \circ \zeta(\theta) \circ \delta \circ (\dot{r} > 0) \quad (4)$$

where k_r is the reflex gain, and $\zeta(\theta)$ tells which joints belong to swing legs depending on the phase values. The activation domain of the $\zeta : \theta \longmapsto \{0,1\}^N$ function is estimated from the phase-contact history when both CPG and VMC are active. δ is a binary valued vector, and $\delta_i = 1$ if the ith joint belongs to a stance (contact) leg. Contact sensing is done by simple on/off contact sensors (bumpers) around the feet. The term $\dot{r} > 0$ is added to activate this reflex only when the leg is protracting. So the reflex is not activated if there is a swing hit just before the start of the stance phase when the leg is retracting. An example of this reflex is provided in `video1`.

C. Virtual Model Control

The two modules discussed make the robot locomote, and prevent the robot from stumbling. However locomotion over rough terrain always includes unwanted body rotations which can lead to a fall if not accounted for. We utilize virtual model control to adjust for these rotations. The main idea of virtual model control is to attach virtual components to a robot, as if they had existed, and generate joint torques which simulate them [20]. In our case, we want to attach springs to the robot to correct for body attitude, lateral angle of attack and direction during locomotion. The output of our virtual model controller is the total of the torques generated by these three components ($\tau_{vm} = \tau_{att} + \tau_{lat} + \tau_{trn}$). [4]

1) Attitude control: We assume a hypothetical plane connected to the center of trunk of the robot (H), and another plane passing through the center of the trunk lying horizontal w.r.t. world coordinates (G). As depicted in Figure 3, one can attach virtual springs between the corners of H and their vertical projections (w.r.t. world frame) on G. These virtual springs naturally generate forces which adjust the attitude of the body to be parallel to the ground:

$$P = R \begin{bmatrix} 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (5)$$

$$F_{att} = \begin{bmatrix} [0]_{[8\times 1]} \\ k_{att} \left(\begin{bmatrix} 0 & 0 & 1 \end{bmatrix} P \right)^T \end{bmatrix} \quad (6)$$

where R is the robot's rotation matrix w.r.t. world coordinates (sensed by an absolute rotation sensor), P are the relative coordinates of the corners of H, k_{att} is the gain and F_{att} is the vector of virtual forces for attitude adjustment. We use the stance legs to generate these forces. Each stance leg is used to generate forces generated by its corresponding virtual spring separately. If for example only two legs are on the ground, then they try to adjust the relative height of their hips, and the other two virtual forces are ignored. To convert the virtual forces to joint torques we use the Jacobian transpose [28]:

$$\tau_{att} = -(J^T F_{att}) \circ \delta \circ \tilde{\zeta}(\theta) \quad (7)$$

where J is the Jacobian of the forward kinematics of each foot's position in its hip frame aligned parallel to the world coordinates[5]. J does not include the partial derivatives of one foot position in one other hip frame, so J is a sparse 12×12 matrix. The activation of the attitude controller can be inhibited by the activation of the stumbling correction reflex, and this is done by the $\tilde{\zeta}(\theta)$ term in equation 7 ($\sim$ is negation). We switch the attitude controller off when there is only one stance leg.

[4]To implement the virtual model controller, we need three kinds of sensory information: current joint angles, on/off contact information, and body rotation sensing. This can be done by on-board sensors (i.e. without external sensing) with e.g. encoders, bumpers, and absolute rotation sensors.

[5]So if a virtual force F is supposed to be generated at hip, we instead generate $-F$ at the foot.

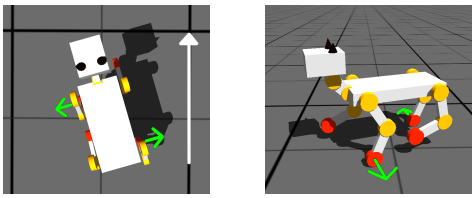

Fig. 5. Virtual forces generated to correct the locomotion direction. The white arrow shows the desired direction. The green arrows on left fore and right hind feet are the virtual forces of about $6[N]$ which, through the Jacobian transpose method, generate joint torques needed for turning.

2) Lateral angle of attack control: The attitude control component is responsible for correcting body attitude if there are unwanted pitch and roll rotations. But the attitude control component is active only when the robot is in contact with the ground. However when traversing over rough terrain at moderately high speeds, there are always cases when there are no ground contacts (stance legs). Swing legs can be used to adjust the angle of attack in situations like the one mentioned. We continuously adjust the lateral angle of attack by introducing virtual torsion springs at the hip abduction/adduction joints. The rest position of each virtual torsion spring is such that the corresponding leg is vertical if that spring is at rest. So:

$$\tau_{lat} = -\begin{bmatrix}[k_{lat}]_{[1\times 4]} & [0]_{[1\times 8]}\end{bmatrix}^T \circ ([\varphi_x]_{[12\times 1]} - q) \qquad (8)$$

where φ_x is the trunk roll angle, and q is the vector of actual joint angles. We keep this component also active for the stance legs, so this component is active all the time. An illustration of this component is given in Figure 4.

3) Direction control: We implement a locomotion direction controller as virtual forces compensating for wrong heading direction. Based on the deviation of the robot from the desired direction, we generate sideways virtual forces with opposing signs in front and back of the robot:

$$F_{trn} = k_{trn}\begin{bmatrix}\begin{pmatrix}[1 & 1 & -1 & -1]_{[1\times 2]}\\ [0]_{[4\times 1]}\end{pmatrix}^T\end{bmatrix} \circ \begin{bmatrix}[\sin\Delta\varphi_z]_{[4\times 1]}\\ [\cos\Delta\varphi_z]_{[4\times 1]}\\ [0]_{[4\times 1]}\end{bmatrix} \qquad (9)$$

with $\Delta\varphi_z$ being the difference between the desired heading angle and the current one. Finally equation 7 is used to convert F_{trn} to τ_{trn}. Figure 5 depicts an example situation and the generated virtual forces.

III. Experiments

This section describes the simulated quadruped robot, control setup, experiment scenarios, and the results obtained. All the body properties and control parameters are given in Table I. The forward dynamics physics simulation is done at $1[kHz]$, and the control loop is working at $250[Hz]$.

A. The Simulated Quadruped

We do all the experiments with a forward dynamics simulated quadruped robot modeled in Webots$^{\text{TM}}$robot simulation software [29]. The quadruped is about the weight and size of a cat (Table I). All the body parts have uniform density distribution. There are 3 active DoF per leg, first hip abduction/adduction (lateral hip joint), then leg protraction/retraction (sagittal hip joint), and finally knee flexion/extension (sagittal knee joint). The zero joint angles are equal to being fully abducted and retracted ($\frac{pi}{6}[rad]$ and $\frac{pi}{3}[rad]$ w.r.t. vertical standing posture), and completely extended. All the joints are passively damped to increase the numerical stability of the simulation (with a damping factor of $b = 1$), and this also is why a P-only controller is used instead of a PD-controller. There is no displacement between lateral and sagittal hip joints. All limb segments have equal lengths. The robot is equipped with four on/off contact sensors (bumpers) around the feet, encoders for joint angle sensing, and an absolute rotation sensor placed in the trunk.

TABLE I
Body and control parameters of the simulated quadruped

Property	Value
Total mass	$5.75[kg]$
Head to total mass percentage	7%
Limb to total mass percentage	7%
Headless length	$0.4[m]$
Sagittal hip to hip distance	$0.3[m]$
Lateral hip to hip distance	$0.24[m]$
Standing leg length	$0.28[m]$
Limb segment length	$0.14[m]$
Foot radius	$0.035[m]$
Foot width	$0.025[m]$

Control parameter	Value
k_p	50
k_{att}	250
k_{lat}, k_{trn}	25
k_r	150
knots of $f_{fore-P/R}$	$\frac{\pi}{3} + [-0.3, 0, -0.3, -0.6]$
knots of $f_{hind-P/R}$	$\frac{\pi}{3} + [-0.4, -0.1, -0.4, -0.7]$
knots of $f_{fore-F/E}$	$[0.6, 1.2, 0.6, 0.9]$
knots of $f_{hind-F/E}$	$[0.7, 1.3, 0.7, 0.85]$
knots of $f_{A/A}$	$[\frac{\pi}{6}, \frac{\pi}{6}, \frac{\pi}{6}, \frac{\pi}{6}]$
ν	$2[Hz]$
γ	10
μ	1

B. Control Parameters

There are two sets of control variables which should be set: CPG parameters and control gains. There are 16 parameters that can be set to tune the open-loop gait generated by CPG, and 4 gain values to define the strength of different modules. One can use optimization techniques to find a proper CPG gait and also the gains, however, we did not find that necessary. It took about 25 manual trials to find an acceptable gait and proper control gains:

1) CPG parameters: We implement and experiment with trot gaits, so the phase difference for ipsilateral and contralateral pairs are similarly $\pi[rad]$, and $0[rad]$ for the diagonal pairs. A phase difference of $\frac{\pi}{4}[rad]$ is introduced between hips and knees. These values are always fixed and we do not use them to tune the controller. We implement limit cycle shaping functions f as piecewise cubic Hermite polynomials [30] with 4 equally spaced knots in $\theta \in [0, 2\pi)$, for each dimension. Joints of contralateral legs use identical f because of symmetry. The desired f for all A/A joints are set to $\frac{\pi}{6}[rad]$ (vertical standing posture), and the other 16 knots for fore and

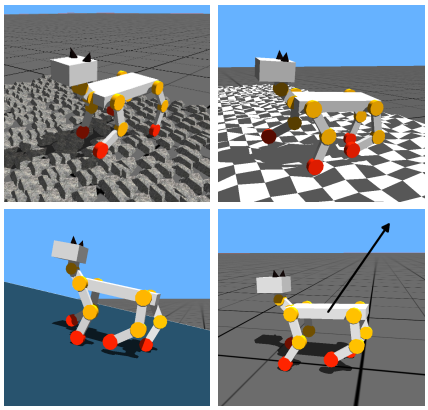

Fig. 6. Experiment scenarios. Top-left) passing through a rocky setup. Top-right) traversing over environments with uneven terrain. Bottom-left) climbing over slopes. Bottom-right) Handling external perturbations.

hind P/R and F/E joints are used to tune the open loop gait. We normally start by front-hind symmetry and sinusoidal joint angle profiles and then slightly alter them. We also benefit from the ideas given in [31] to use single-peak P/R and double-peak F/E profiles.

2) Control gains: We first set the P-controller gain k_p while the other gains are set to zero. We do not choose high values for k_p as it makes the robot very stiff, and k_p is set high enough to generate acceptable joint amplitudes. Consequently the joint angle plan generated by CPG is not perfectly tracked, however, this does not pose a problem since the CPG pattern itself is under tuning. After that, gains for the virtual model controller are set. We choose k_{trn} to be equal to k_{lat}. The value of k_{att} indicates the importance of the attitude control compared to following the CPG pattern. We set k_{att} to be about 5 times bigger than k_p, and this means that the momentary attitude control is more important than momentary forward progression, if they contradict. Finally we extract the swing onsets to define $\zeta(.)$ functions, and set the stumbling corrective reflex gain k_r, which is set such that the robot could move over a an obstacle of about 20% of the leg length.

C. Experimental Setup

We test our simulated robot in the following scenarios:

- passing through a rocky setup;
- traversing over randomized uneven terrain of $7-11\%$ of leg length variation in $0.1[m]$ spaced vertices;
- going over $14-21\%$ slopes;
- handling external pushes of $\{15[N], 0.5[s]\}$ magnitude while locomotion over flat terrain.

The first 3 types of aforementioned scenarios are repeated from 25 different initial conditions. The last scenario is repeated 27 times where an external force of $15[N]$ pushes the robot for a duration of $0.5[s]$ with a random timing and variable direction. All the experiments are executed in a time window of $20[s]$ out of which the first $5-7[s]$ are used for initiation on a flat terrain and unperturbed. Figure 6 illustrates these scenarios. In all of these scenarios the robot does not have *any* prior knowledge about the environment, even the subtle information of "what kind of environment am I facing?". So there is no possibility to anticipate. All the experiments are done with a trot gait at $2[Hz]$ which gives an average speed of more than $0.6[\frac{m}{s}]$ equal to $2[\frac{BL}{s}]$.

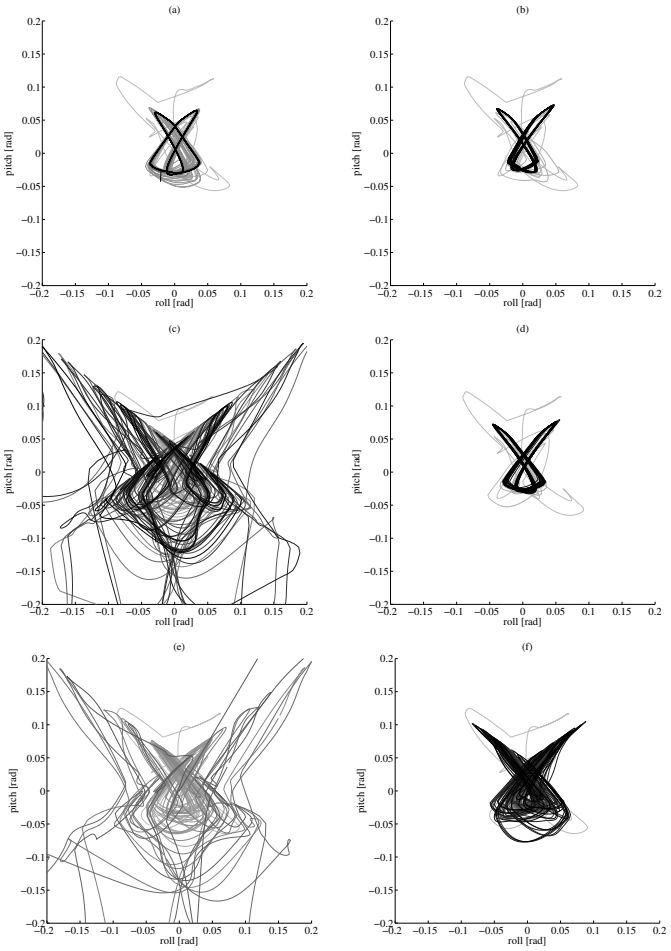

Fig. 7. Comparison between the roll-pitch variations (RPV) of open-loop gaits (left) and their closed-loop counterparts (right). All the plots are for $40[s]$ runs equal to 80 gait cycles. The line color starts from gray in the beginning of the recording and becomes black in the end of recording at $t = 40[s]$. (a) A not-well-tuned open-loop gait. (b) Same gait as 'a', but reflex and VM controller modules are active. (c) A bad open-loop gait. The gait behavior is not regular. (d) Same gait as 'c', but the loop is closed. The RPV is much more periodic. (e) A not open-loop stable gait. The robot falls after about $20[s]$. (f) The not open-loop stable gait can be turned into a sufficiently stable gait by closing the loop. The RPV are bounded, but of course not as regular as 'b' and 'd' because of the open-loop gait used.

D. Results

Before going into the results obtained for the experimental scenarios, we would like to show the effect of the virtual model controller on badly designed open-loop gaits, and on increasing the periodicity of the gait. Figure 7-(a) shows the roll-pitch variations (RPV) of a typical not well-tuned open-loop gait on flat terrain, where the settling time is long (about $12[s]$). Figure 7-(b) depicts the RPV of the mentioned gait, but now with the VM controller active, which shows a better periodicity and faster settling (about $4.5[s]$). The

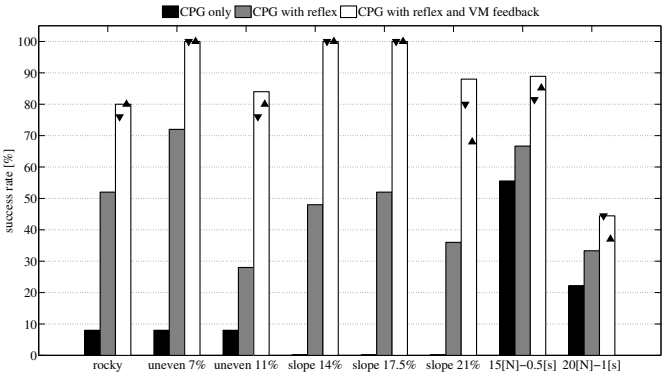

Fig. 8. Results for the experiment scenarios. The open-loop gait is mostly unable to successfully pass the scenarios, other than $15[N]$ pushes where a 56% success rate is obtained. The results for activating the reflex mechanism are consistently better than the open-loop control, and similarly, the results obtained by additionally activating the VM controller is consistently better than the reflex-only case. The ▲ and ▼ markers on the white bars are for controllers with $\pm 20\%$ VM controller gains, as a measure of the sensitivity of the controller to the choice of its parameters. We additionally present results for a $20[N] - 1[s]$ pushes scenario (column 8) where the pushes are just too big to be handled. Nevertheless, the reflex and VM controller modules improve the result even for this case.

RPV is illustrated for a badly designed open-loop gait in Figure 7-(c). Flat terrain locomotion with this gait did not lead to a fall, but the robot was irregularly disturbed, and as it is obvious, the RPV is not periodic. We get Figure 7-(d) by activating the VM controller for this gait. Again, the RPV is greatly reduced, and the outcome looks more periodic and symmetrical. A similar experiment is presented in 7-(e-f), however this time the open-loop gait is not even open-loop stable and the robot falls after about 20 seconds. After activating the VM controller the robot could locomote for more than $100[s]$ where we stopped the recording. For all these experiments the same parameters given in Table I are used (other than CPG shaping functions f which is used to generate different open-loop gaits). `video1` compares the bad open-loop gait and its closed-loop counterpart.

Figure 8 presents the results obtained for the experiment scenarios. Though the open-loop gait (with parameters given in Table I) is stable on the flat terrain, it is mostly unsuccessful in the scenarios. In case of rocky and uneven environments, there are two typical fall cases: 1) stumbling, and 2) bad foothold leading to unwanted body roll and pitch which are not corrected (since there is no feedback). In case of the slopes environments, the robot stumbles in transition to the slope, or the trunk's pitch angle gradually increases and consequently leads to a fall. When facing external pushes, pitch and roll angles are not continuously corrected and robot falls for about 44% of the times.

In contrast, the closed-loop control is successful in most of the scenarios. In case of the rocky environment, the robot is successful for 20 out of 25 experiments. Normally the robot can go over the rocky terrain with minor difficulties. But since there is no anticipation, there are cases where the robot is greatly disturbed and needs to reactively correct the attitude. Uneven terrain is passed with a good success rate. For uneven terrain of about 7% of leg length variation, all the experiments where successful. The performance starts to degrade at higher terrain variations, as the momentary corrections are not sufficient to correct the attitude of the robot.

Closed-loop controller handles slope environments with a 100% success rate for $14 - 17.5\%$ slopes. Fall cases start to happen after 21% slopes. The attitude controller forces the robot to have a horizontal attitude w.r.t. world coordinates, and this has positive and negative effects when facing slopes. The positive is that the pitch angle is contained so it is less probable to fall backwards. The negative is that when the slopes become bigger, the generated torques for the attitude control also consistently become bigger and start to alter the CPG plan needed for forward progression. One would use the robot orientation along with the posture of the contact legs to estimate the slope [20], and use this to regulate the attitude controller. We are planning to extend and experiment our control with this element in the future.

There were 3 fall cases out of 27 runs in $15[N] - 0.5[s]$ pushes scenario. In two of the fall cases the external push was backwards and sideways, and only sideways in the third fall. We additionally tested our controller with external pushes of $20[N] - 1[s]$ which are very big for the robot to be handled. Nevertheless, the performance of the closed-loop controller is 2 times better than the open-loop controller facing these external pushes (please see the last column in Figure 8).

We also tested the closed-loop controller with $\pm 20\%$ VM gains as a partial measure of robustness to parameter changes. The results for these tests are presented with ▲ and ▼ markers in Figure 8. The results have not changed for less difficult setups like $uneven$ 7% or $slopes$ $14 - 17.5\%$. Performance starts to be sensitive to the choice of parameters at rougher terrains, since it makes sense to assume that the basin of attraction of the whole system (forward dynamics plus the closed-loop control) is naturally smaller in a more irregular environment.

A collection of rough terrain locomotion examples can be found in `video1`. One should note that there is a limit to the presented control strategy as there is no prior knowledge about the environment, and many situations are very difficult to handle if there is no anticipation.

E. Additional Test: A Faster Gait

To be able to show the generality of the introduced control methodology, we ran the experiment scenarios with another trot gait working at $2.5[Hz]$ and a speed of about $0.9[\frac{m}{s}]$ which is $3[\frac{BL}{s}]$ (about 50% faster). We reused the gait parameters given in Table I, changed the frequency ν to $2.5[Hz]$, and also added a virtual force of $5[N]$ pulling the robot in forward direction to obtain a faster locomotion speed. The control gains are kept the same as the ones in Table I. The results for the scenario experiments for this gait are illustrated in Figure 9. As expected, the obtained success rates are still good, and comparable to the ones obtained in section III-D. There is a

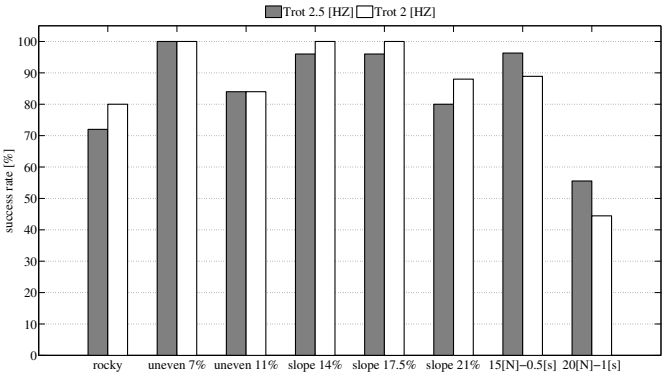

Fig. 9. Results of the experiment scenarios for a faster trot gait at $2.5[Hz]$ compared the ones obtained for the gait used in section III-D. There is not much variation in the results even though different gaits are used. The control gains are not tuned for the faster gait, and are kept as they were for the $2[Hz]$ gait. The faster gait is more robust against external pushes.

minor occasional drop in performance since the control gains are not tuned for this new faster gait.

F. Additional Skill: Turning

Voluntary turning can be acquired by giving constant sideways virtual forces like the ones used for direction control in II-C. We could obtain a maximum of $90[\frac{deg}{s}]$ turning rate without greatly disturbing the robot. For this maximum turning rate we have $k_{trn} = 50$. This maximum turning is acquired at a locomotion speed of more than $2[\frac{BL}{s}]$ and a minimum turning radius of about $0.4[m]$. Figure 10 depicts snapshots of a fast turning, and a demonstration of this skill is provided in `video1`.

IV. Discussion

A. Summary

We presented a bio-inspired locomotion controller based on Central Pattern Generators (CPG) and Virtual Model Control (VMC). The CPG is implemented as coupled phase-based scaled Hopf oscillators which are flexible tools to design nonlinear oscillators with arbitrary limit cycle shape and large basins of attraction. We added the stumbling correction reflex to the CPG as an additive feedback signal. Finally we enriched the control by adding virtual model control. The VM controller consists of three components, namely attitude, lateral swing angle of attack, and direction controllers. The direction controller can also be used for voluntary turning.

We demonstrated results of unperceived rough terrain locomotion with 80%+ success rates in environments that cannot be traversed with a CPG-only controller. Results are obtained with a trot gait of more than $2[\frac{BL}{s}]$ forward velocity. This is a demonstration of dynamic locomotion over moderately difficult and unperceived rough terrain.

B. Comparison

Our work presents similarities with the CPG-based approach of Fukuoka et al. [2] on the Tekken robot. Compared to their work, we offer simplicity. The control strategy presented in

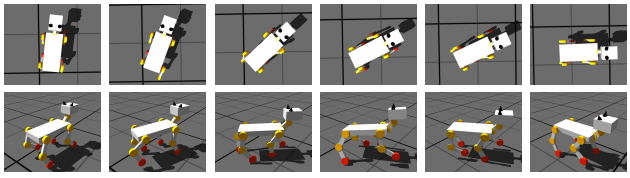

Fig. 10. Snapshots of the turning skill. The robot turns with $k_{trn} = 50$ for $1[s]$. A turning rate of about $90[\frac{deg}{s}]$ is obtained. This fast turning does not disturb locomotion, and the robot continues to normally locomote afterwards.

[2] demonstrates impressive results, however their approach is quite complex and consists of different parametric interconnected mechanisms. What we introduce here is rather simple and modular, and a step-by-step procedure can be employed to tune each module with consistent results (refer to Figure 8 showing consistent improvement of results by adding reflex and VM controller modules). We also did systematic tests in different experiment scenarios to validate what we have presented. We believe that the approach introduced here can be easily implemented and tuned for another robot, and we are implementing our approach on a hardware robot.

Buchli et al. [16] presented unperceived rough terrain locomotion with LittleDog robot. Compared to what we presented in this paper, they can go over much rougher terrains (more than 30% of leg length variations), but at also slower speed (about $0.3[\frac{BL}{s}]$). Assuming a support polygon stability definition [32], rough terrain locomotion with a static walk is done in [16], and we demonstrate dynamic locomotion over rough terrain. Moreover, our kind of control does not need any information about the dynamic properties of the robot (like inertia tensors), nor 3D force sensing. Our approach needs the rotation information of the robot's trunk, sensing of the joint angle values, and binary-valued contact sensors.

At any rate, comparing the rough terrain capabilities of different control methods is not easy if a same shared platform is not used. An ankle joint, the contact material, compliance, etc are just some factors that can affect the comparison of different platforms. We are well aware that our experiments are on a simulated robot, and not a hardware one. Our purpose here is to show a control methodology for quadruped rough terrain locomotion.

C. Advantages, Limitations and Prospect

We have proposed a simple way to design a locomotion controller for unperceived quadruped rough terrain locomotion. A controller which is quite robust even without anticipation can be a good basis for rich locomotion with additional sensing. Moreover, what we present here is modular, and modules can be designed and tuned on top of each other. This is a very useful feature which makes the design process easier.

One limitation of the presented approach is the disability to anticipate obstacles and terrain variations. Of course this is the nature of the problem that we are trying to solve, but in the future, we need to enrich our robot with additional sensing needed for anticipation, like stereo vision. The other limitation

is that the robot is mechanically stiff, so the maximum roughness that the robot can traverse is limited. We believe adding passive compliance to the mechanics will damp impacts and help the robot overcoming more difficult situations.

One main point that we have not explored in this paper is how to affect the phase states of CPG based on the situation. [33] and [34] suggest to add a phase resetting behavior in case of an early stance onset, or leg extension in case of a late stance onset, and we are testing such feedback mechanisms. We are currently working on implementing the proposed controller on our newly designed quadruped Oncilla, which benefits from compliant pantograph legs [31].

ACKNOWLEDGMENT

This work was supported by the European Communitys Seventh Framework Programme FP7/2007-2013 under grant agreements 1) No 248311 - AMARSi, and 2) No 231688 - Locomorph.

APPENDIX I: LIST OF THE ACCOMPANIED MATERIAL

material	description
`video1`	Video demonstrating open-loop and closed-loop rough terrain locomotion, stumbling correction reflex, RPV correction and fast turning

REFERENCES

[1] M. H. Raibert, "Trotting, pacing and bounding by a quadruped robot," *Journal of Biomechanics*, vol. 23, no. Supplement 1, pp. 79–81, 1990.

[2] Y. Fukuoka, H. Kimura, and A. H. Cohen, "Adaptive dynamic walking of a quadruped robot on irregular terrain based on biological concepts," *The International Journal of Robotics Research*, vol. 22, no. 3-4, pp. 187–202, 2003.

[3] F. Iida and R. Pfeifer, "Cheap rapid locomotion of a quadruped robot: Self-stabilization of bounding gait," *Intelligent Autonomous Systems*, 2004.

[4] J. Estremera and K. J. Waldron, "Thrust control, stabilization and energetics of a quadruped running robot," *The International Journal of Robotics Research*, vol. 27, no. 10, pp. 1135–1151, 2008.

[5] S. Rutishauser, A. Spröwitz, L. Righetti, and A. J. Ijspeert, "Passive compliant quadruped robot using central pattern generators for locomotion control," in *2008 IEEE International Conference on Biomedical Robotics and Biomechatronics*, 2008.

[6] Z. G. Zhang and H. Kimura, "Rush: a simple and autonomous quadruped running robot," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 223, no. 3, pp. 323–336, 2009.

[7] 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.

[8] 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," *IMechE, Part I: Journal of Systems and Control Engineering*, vol. 225, no. 6, pp. 831 –849, 2011.

[9] M. Raibert, K. Blankespoor, G. Nelson, and R. Playter, "BigDog, the Rough–Terrain quadruped robot," in *Proceedings of the 17th IFAC World Congress*, (COEX, South Korea), pp. 10823—10825, 2008.

[10] J. Pippine, D. Hackett, and A. Watson, "An overview of the defense advanced research projects agencys learning locomotion program," *The International Journal of Robotics Research*, vol. 30, no. 2, pp. 141–144, 2011.

[11] M. P. Murphy, A. Saunders, C. Moreira, A. A. Rizzi, and M. Raibert, "The littledog robot," *The International Journal of Robotics Research*, vol. 30, no. 2, pp. 145–149, 2011.

[12] J. Zico Kolter and A. Y. Ng, "The stanford littledog: A learning and rapid replanning approach to quadruped locomotion," *The International Journal of Robotics Research*, vol. 30, no. 2, pp. 150–174, 2011.

[13] M. Zucker, N. Ratliff, M. Stolle, J. Chestnutt, J. A. Bagnell, C. G. Atkeson, and J. Kuffner, "Optimization and learning for rough terrain legged locomotion," *The International Journal of Robotics Research*, vol. 30, no. 2, pp. 175–191, 2011.

[14] A. Shkolnik, M. Levashov, I. R. Manchester, and R. Tedrake, "Bounding on rough terrain with the littledog robot," *The International Journal of Robotics Research*, vol. 30, no. 2, pp. 192–215, 2011.

[15] M. Kalakrishnan, J. Buchli, P. Pastor, M. Mistry, and S. Schaal, "Learning, planning, and control for quadruped locomotion over challenging terrain," *The International Journal of Robotics Research*, vol. 30, no. 2, pp. 236–258, 2011.

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

[17] C. Maufroy, H. Kimura, and K. Takase, "Integration of posture and rhythmic motion controls in quadrupedal dynamic walking using phase modulations based on leg loading/unloading," *Autonomous Robots*, vol. 28, no. 3, pp. 331–353, 2010.

[18] S. Coros, A. Karpathy, B. Jones, L. Reveret, and M. van de Panne, "Locomotion skills for simulated quadrupeds," in *ACM SIGGRAPH 2011 papers*, SIGGRAPH '11, (New York, NY, USA), pp. 59:1–59:12, ACM, 2011.

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

[20] J. Pratt, C.-M. Chew, A. Torres, P. Dilworth, and G. Pratt, "Virtual model control: An intuitive approach for bipedal locomotion," *The International Journal of Robotics Research*, vol. 20, no. 2, pp. 129–143, 2001.

[21] K. Tsujita, H. Toui, and K. Tsuchiya, "Dynamic turning control of a quadruped robot using nonlinear oscillators," in *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, vol. 1, pp. 969 – 974 vol.1, sept.-2 oct. 2004.

[22] S. Dégallier Rochat, L. Righetti, S. Gay, and A. Ijspeert, "Towards simple control for complex, autonomous robotic applications: Combining discrete and rhythmic motor primitives," *Autonomous Robots*, vol. 31, no. 2, pp. 155–181, 2011.

[23] J. Buchli and A. J. Ijspeert, "Self-organized adaptive legged locomotion in a compliant quadruped robot," *Autonomous Robots*, vol. 25, no. 4, pp. 331–347, 2008.

[24] L. Righetti and A. J. Ijspeert, "Pattern generators with sensory feedback for the control of quadruped locomotion," in *Proceedings of the 2008 IEEE International Conference on Robotics and Automation (ICRA 2008)*, pp. 819–824, 2008.

[25] S. Rossignol, R. Dubuc, and J.-P. Gossard, "Dynamic sensorimotor interactions in locomotion," *Physiol. Rev.*, vol. 86, no. 1, pp. 89–154, 2006.

[26] R. A. Horn and C. R. Johnson, *Matrix analysis*. New York, NY, USA: Cambridge University Press, 1986.

[27] H. Forssberg, "Stumbling corrective reaction: a phase-dependent compensatory reaction during locomotion," *Journal of Neurophysiology*, vol. 42, no. 4, pp. 936–953, 1979.

[28] R. P. Paul, *Robot Manipulators: Mathematics, Programming, and Control*. Cambridge, MA, USA: MIT Press, 1st ed., 1982.

[29] O. Michel, "Webots: Professional mobile robot simulation," *Journal of Advanced Robotics Systems*, vol. 1, no. 1, pp. 39–42, 2004.

[30] F. N. Fritsch and R. E. Carlson, "Monotone piecewise cubic interpolation," *SIAM Journal on Numerical Analysis*, vol. 17, no. 2, pp. pp. 238–246, 1980.

[31] A. Sproewitz, L. Kuechler, A. Tuleu, M. Ajallooeian, M. DHaene, R. Moeckel, and A. Ijspeert, "Oncilla robot, a light-weight bio-inspired quadruped robot for fast locomotion in rough terrain," in *Symposium on Adaptive Motion of Animals and Machines (AMAM2011)*, pp. 63–64, 2011.

[32] R. McGhee and A. Frank, "On the stability properties of quadruped creeping gaits," *Mathematical Biosciences*, vol. 3, no. 0, pp. 331 – 351, 1968.

[33] L. Righetti and A. Ijspeert, "Pattern generators with sensory feedback for the control of quadruped locomotion," in *Proceedings of the 2008 IEEE International Conference on Robotics and Automation (ICRA 2008)*, pp. 819–824, 2008.

[34] M. A. Daley and A. A. Biewener, "Running over rough terrain reveals limb control for intrinsic stability," *Proceedings of the National Academy of Sciences*, vol. 103, no. 42, pp. 15681–15686, 2006.