Delft University of Technology

Delft Center for Systems and Control

Technical report 14-001

Modeling and control of legged locomotion via switching max-plus models*

G.A.D. Lopes, B. Kersbergen, T.J.J. van den Boom, B. De Schutter, and R. Babuška

If you want to cite this report, please use the following reference instead:
G.A.D. Lopes, B. Kersbergen, T.J.J. van den Boom, B. De Schutter, and R. Babuška, "Modeling and control of legged locomotion via switching max-plus models," *IEEE Transactions on Robotics*, vol. 30, no. 3, pp. 652–665, June 2014. doi:10.1109/TRO.2013.2296105

Delft Center for Systems and Control
Delft University of Technology
Mekelweg 2, 2628 CD Delft
The Netherlands
phone: +31-15-278.24.73 (secretary)
URL: https://www.dcsc.tudelft.nl

Modeling and Control of Legged Locomotion via Switching Max-Plus Models

G.A.D. Lopes, B. Kersbergen, T.J.J. van den Boom, B. De Schutter, and R. Babuška

Abstract—We present a gait generation framework for multi-legged robots based on max-plus algebra that is endowed with intrinsically safe gait transitions. The time schedule of each foot lift-off and touchdown is modeled by sets of max-plus linear equations. The resulting discrete-event system is translated to continuous time via piecewise constant leg phase velocities, thus, it is compatible with traditional central pattern generator approaches. Different gaits and gait parameters are interleaved by utilizing different max-plus system matrices. We present various gait transition schemes, and show that optimal transitions, in the sense of minimizing the stance time variation, allow for constant acceleration and deceleration on legged platforms. The framework presented in this paper relies on a compact representation of the gait space, provides guarantees regarding the transient and steady-state behavior, and results in simple implementations on legged robotic platforms.

Index Terms—Max-plus algebra, legged locomotion, robotics, gait generation, gait transition.

I. Introduction

Legged robots are becoming increasingly prominent in the robotics field. Their advantages on unstructured terrain combined with the challenges in mechatronics and control have fueled a community of academics and industry alike that aims to build truly autonomous legged robots with agility akin to animals. The recent successes by Boston Dynamics on quadrupeds, and the effort of the Japanese community on developing home assistance anthropomorphic robots contributes to this growing interest in legged robots.

A fundamental element in the control of a legged robot is the synchronization of its legs. For bipedal robots synchronization is usually addressed implicitly, since balancing is the biggest challenge. For robots with more than two legs, many different synchronizations can be chosen, resulting in the number of distinct gaits increasing with the number of legs (see Holmes et al. [1] for an extensive review on the elements of dynamic legged locomotion). This paper focuses on the systematic design of gait controllers for robots with many legs where the number of available gaits is high. From a control design point of view, legged locomotion can be implemented via a *gait reference generator* module and a *dynamic tracking controller* module, as illustrated in Figure 1. The first is a component that generates cyclic reference signals in a synchronized way, and the second translates the typically low-dimensional reference signals into the high-dimensional motion of the robot's limbs and implements other desirable dynamical properties such as balancing, see e.g., Vukobratovic and Borovac [2]. The advantage of this partition is that the gait reference generator can be designed without explicit knowledge of the mechanics of the robot (other than the number of legs) while the latter is designed specifically for each robot model. This paper focuses on the first subsystem: we introduce a novel type of gait reference generator.

All authors are with the Delft Center for Systems and Control, Delft University of Technology, The Netherlands, e-mail: {g.a.delgadolopes, b.kersbergen, a.j.j.vandenboom, b.deschutter, r.babuska}@tudelft.nl

Manuscript received xxxxxx, 20xx; revised xxxxx, 20xx.

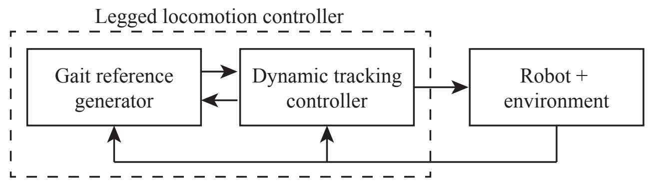

Fig. 1. The standard partitioning of a legged locomotion controller. The gait reference generator subsystem provides reference signals to the tracking controller. Feedback can exist from both the robot and the tracking controller to influence the reference signals.

Central pattern generators (CPGs) are currently the standard tool for designing gait reference generators. CPGs offer a natural bio-inspired control framework that address synchronization (see Ijspeert [3] for a survey on CPGs). Although used widespread, CPGs offer their own set of challenges due to the nature of their foundation as sets of coupled differential equations. As in normal systems modeled by differential equations, the transient behavior is typically less understood than the steady-state behavior. Transient behaviors exist during gait transitions, a very natural occurrence in nature. Animals change gait to accommodate for different types of terrain or locomoting speeds. Gait transition in the CPG framework has been addressed by Nagashino [4], Inagaki [5], [6], Zhang [7], Li [8], Aoi [9], Daun-Gruhn [10], Santos [11], and the references within [3]. Other work on gait transition without using CPGs in the continuous-time domain has been done by Haynes et al. [12], [13]. The traditional approach for gait transition in the CPG framework exploits the bifurcations that occur when changing parameters in the set of coupled differential equations. This can lead to intricate analysis of the global behavior due to the continuous-time models used.

In this paper we present an alternative to the continuous-time approach of CPGs by considering instead discrete-event models. Starting with circuits of timed event graphs (a subclass of Petri nets), each abstractly representing the phase of a leg, we write the evolution equations that describe the time instants of each feet touchdown and lift off, to find a compact linear representation in the max-plus algebra [14], [15], [16] which features maximization and addition as its basic operations. Max-plus linear discrete-event systems (MPL-DES) are a subclass of timed discrete-event systems (DES), classes of DES endowed with a time structure, that can be framed in systems of linear equations in the max-plus algebra. DES that enforce synchronization can be modeled in this framework. MPL systems inherit a large set of analysis and control synthesis tools thanks to many parallels between the max-plus-linear systems theory and the traditional linear systems theory. At the time of writing, the theory of max-plus algebra has been successfully applied to railroads [17], [18], queuing systems [19], resource allocation [20], and recently image processing [21]. In this paper we show that by modeling legged locomotion in the max-plus algebra one can take advantage of its well known properties to obtain guarantees such as *kinematic stance stability*[1] during gait transition, or the ability to compute gait transitions that minimize stance time variation. Finally, we show that our framework is compatible with current continuous-phase systems via a conversion map, avoiding the need to compute differential equations in real-time, as in the CPG case.

This paper presents a discrete-event based control synthesis tool for generating gait reference signals. We compare our method to CPGs and analyze specific properties that arise naturally from the discrete-event modeling approach. This paper addresses abstracted time models, while the kinematics and dynamics of the robot are not considered. As such, our notion of kinematic stance stability is not defined in terms of balancing dynamics, but in terms of allowed sets of states in a discrete-event system. Additionally, the tools presented here are more valuable for robots with many legs (more than 2) where many different gaits (leg synchronizations) are possible.

We start by revisiting the notion of CPGs for robotics and introduce our novel gait modeling approach based on discrete-event systems in Section II. In Section III we partially review the theory of max-plus algebra and in Section IV we demonstrate how to systematically model legged locomotion using max-plus linear systems. In Section Section V the control structure that implements the max-plus-linear system is presented. In Section VI we demonstrate how to compute transient gait parameters to obtain optimal gait transitions in terms of minimizing the leg time stance variation. Furthermore, we introduce transition gaits that enforce a constant leg stance time, and we present a constant acceleration/deceleration gait generator. In Section VII we show experimental and simulation results of the introduced legged locomotion controller and Section Section VIII concludes the paper.

II. Modeling legged locomotion

A large body of work has been dedicated to the modeling of legged locomotion in neuroscience (see references within [3], [1], [22], [23]). There, the focus is put on understanding how animals control their limbs rhythmically by analyzing the interaction between populations of neurons and, in some instances, the effect of sensory feedback. The neural networks that generate limb coordination patterns (called CPGs), have inspired the robotics community to design classes of gait generators or controllers that bear the same name. In this section we revisit the traditional coupled-differential equation approach to CPGs, revisit piecewise constant velocity phase models, and introduce an abstraction of the continuous-phase space into a class of Petri net circuits.

A. Central pattern generators

In robotics, CPGs are usually implemented by solving sets of coupled differential equations online. An abstract phase $\theta_i \in \mathbb{S}^1$ is associated to each leg i representing its periodic motion, with $\mathbb{S}^1$ representing the circle. The dynamical equations for the full phase state $\theta = [\theta_1 \ \cdots \ \theta_n]^T \in \mathbb{T}^n$ can be written as:

$$\dot{\theta}(\tau) = V + h(\theta(\tau)), \tag{1}$$

where $\mathbb{T}^n$ is the n-torus (the Cartesian product of n circles), $V \in \mathbb{R}^n$ represents the desired phase velocity vector, τ represents time, and the function h includes the desired coupling between each phase.A common realization of (1) is presented below:

$$\dot{\theta}_i(\tau) = v + \sum_j w_{ij} \sin(\theta_j(\tau) - \theta_i(\tau) - \phi_{ij}) \tag{2}$$

where $v \in \mathbb{R}$ is a common phase velocity, the weights w_{ij} represent the coupling strength between phases $\theta_i(\tau)$ and $\theta_j(\tau)$, and ϕ_{ij} is their phase difference (typically $\phi_{ij} = -\phi_{ji}$). In traditional robotic applications that use CPGs, the phase θ is utilized to generate reference trajectories for the "limbs" of the robot via a parameterized map g:

$$q_{\mathrm{ref}}(\tau) = g(p, \theta(\tau)), \tag{3}$$

where $q_{\mathrm{ref}}(\tau)$ represents the reference trajectories of each actuator at time τ, and p is a set of parameters that modulate the shape of the resulting phase curves into a physical motion in space. The desired reference trajectory q_{ref} is then fed into a tracking controller, or a reference vector field (as a function of $\theta(\tau)$) that can be pushed back through g (if g is differentiable [24]). Equation (1) corresponds to the subsystem 1 in Figure 1, while equation (3) corresponds to subsystem 2.

Designing gaits in the CPG framework is accomplished by choosing the parameters w_{ij}, ϕ_{ij}, and p. Despite the widespread use of CPGs and their straightforward implementation, there are some disadvantages to this approach that should be considered. First, it is necessary to continuously solve the differential equation (1) in real-time. Many approaches have been taken, including dedicated analog CPG implementations (see references within [3]). Second, the transient behavior of (1) may be difficult to describe. This is more so when the parameters of (1) are a function of time (i.e., $w_{ij}(\tau)$ and $\phi_{ij}(\tau)$), as in the case of gait transitions or variable velocity, since changing parameters in dynamical systems typically results in bifurcations. Such behavior can be difficult to analyze.

[1] Since our framework is developed on an abstracted supervisory control layer, we define "kinematic stance stability" in terms of requirements of leg stance, i.e., certain combinations of multiple legs simultaneously swinging are not allowed.

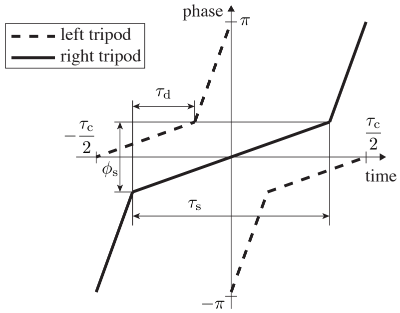

Fig. 2. The “Buehler clock” model for a hexapod robot: piecewise constant phase velocity (Figure reproduced from [25]). Each trajectory corresponds to the reference phase of a group of legs in time.

B. Buehler clock

An alternative approach to CPGs for the synchronization of cyclic systems is called the “Buehler clock” [25], illustrated in Figure 2 for a hexapod robot. In this framework, piecewise constant velocity references are generated based on the set of these parameters:

τ_{c} is the cycle time
τ_{s} is the stance time
ϕ_{s} is the “stance phase”
τ_{d} is the double stance time, with $\tau_{\mathrm{d}} = \tau_{\mathrm{s}} - \tau_{\mathrm{c}}/2$

The stance phase ϕ_{s} represents the section of the abstracted phase when the legs are assumed to be in stance. For a gait where the legs are divided into two groups the mathematical model can be written as:

$$\theta_1(\tau) = \begin{cases} \dfrac{\phi_{\mathrm{s}}}{\tau_{\mathrm{s}}}\,\bar{\tau} & \text{if} \quad -\dfrac{\tau_{\mathrm{s}}}{2} < \bar{\tau} < \dfrac{\tau_{\mathrm{s}}}{2} \\ \dfrac{\pi - \phi_{\mathrm{s}}}{\tau_{\mathrm{c}} - \tau_{\mathrm{s}}}\left(\bar{\tau} - \dfrac{\tau_{\mathrm{s}}}{2}\right) + \dfrac{\phi_{\mathrm{s}}}{2} & \text{if} \quad \bar{\tau} \geqslant \dfrac{\tau_{\mathrm{s}}}{2} \\ \dfrac{\pi - \phi_{\mathrm{s}}}{\tau_{\mathrm{c}} - \tau_{\mathrm{s}}}\left(\bar{\tau} + \dfrac{\tau_{\mathrm{s}}}{2}\right) - \dfrac{\phi_{\mathrm{s}}}{2} & \text{if} \quad \bar{\tau} \leqslant -\dfrac{\tau_{\mathrm{s}}}{2} \end{cases} \tag{4}$$

$$\theta_2(\tau) = \theta_1\left(\tau + \frac{\tau_{\mathrm{c}}}{2}\right) \tag{5}$$

with $\bar{\tau} = ((\tau + \tau_{\mathrm{c}}/2) \text{ modulo } \tau_{\mathrm{c}}) - \tau_{\mathrm{c}}/2$. The reference phases $\theta_1(\tau)$ and $\theta_2(\tau)$ in (4) and (5), represent the right and left tripod of a hexapod robot respectively, as in Figure 2, and τ represents the current time instant. In [25], $\theta_1(\tau)$ is used as a phase reference for legs 1, 4, and 5; and $\theta_2(\tau)$ is used for legs 2, 3, and 6, following the notation of the left-most image in Figure 3. The advantage of the Buehler clock is that, since it is constructed as a piecewise function, its computation is very simple, as opposed to solving differential equations in the case of CPGs. The methodology we propose next generalizes the Buehler clock.

C. Timed event graphs

We propose a different approach to model legged locomotion by considering only two physical states of a leg: stance and swing, and also the time of their respective transition events: the moment the foot touches down and lifts off. Petri nets [27] naturally capture these concepts by assigning swing and stance to places and feet touchdown and lift off to transitions. When additionally considering that there exists a time structure associated with the Petri net, e.g., leg swing and stance take a finite time to execute, then it is convenient to utilize the notion of *timed event graphs*.

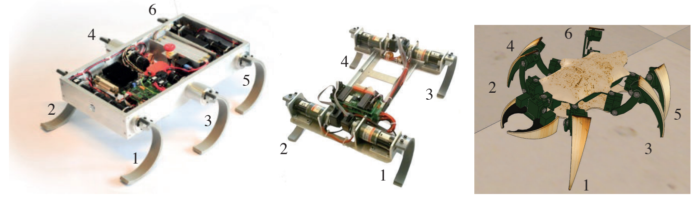

Fig. 3. Left: Zebro and Rquad robots developed at the Delft Center for Systems and Control, morphologically identical to RHex [25]. Right: ant-inspired 23-dof hexapod robot in the V-Rep simulation environment [26]. The numbers indicate the leg labeling for each robot.

Definition 1. *[16] A timed Petri net $\mathcal{G}$ is characterized by a set of places $\mathcal{P}$, a set of transitions $\mathcal{Q}$, a set of arcs $\mathcal{D}$ from transitions to places and vice versa, an initial marking $\mathcal{M}_0$, and a holding time vector $\mathcal{T}$. If each place has exactly one upstream and one downstream transition, then the timed Petri net is called a* timed event graph*.*

For simplicity, consider a robot with 2 legs. For each leg one assigns a circuit composed of 2 transitions (t_i for touchdown and l_i for lift off) and 2 places (f_i for leg swing, or foot in flight; and g_i for leg stance, or foot on the ground), as illustrated in Figure 4-a1. Each circuit is initialized with a token in the stance places, representing that the robot starts standing on two legs. A token in a place can be seen as the fulfilment of the condition of the place, e.g., the leg is in stance, or swing. A minimum time (holding time, see [15] Definition 2.43) is added to each place such that each leg must stay at least τ_{g} time units in stance and τ_{f} time units in swing. When a transition fires the event associated to the transition takes place and one token of each of the upstream places of the transition are removed and tokens are added to the downstream places of the transition. A transition can fire if all of its upstream places have tokens and have held them for the required holding time.

Figure 4-a2 illustrates a sample simulation where the events of the timed event graph do not fire immediately, but randomly with a bounded uniform distribution for illustration purposes. In this simulation, plotted in time, the gray/blue rectangles represent leg stance and white space represent leg swing. One can observe that since the timed event graph in Figure 4-a1 is composed of two concurrent circuits, no synchronization takes place, resulting in the lift-off and touchdown events for each leg to evolve independently. We can now define a notion of synchronization:

Definition 2. *We say that the legs of a robot are* synchronized *if each leg's lift-off event is a function of the touchdown events of other legs.*

Figure 4-b1 illustrates a synchronized timed event graph

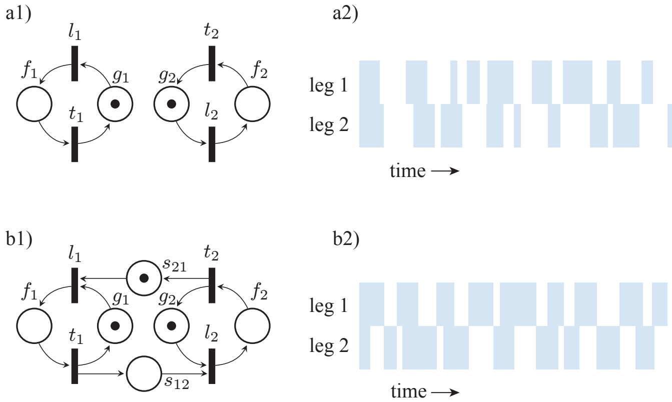

Fig. 4. Two examples of timed event graphs. In Subfigures a1 and b1 the transitions are represented by the bars, the places are represented by the circles, and the tokens by dots in the places. Subfigures a2 and b2 illustrate sample time evolutions of the associated timed event graphs, where transitions are allowed to fire according to a bounded uniform distribution after a minimum holding time has expired. Gray/Blue rectangles represent the stance places g_i and white represents the swing places f_i. The top figure represents the unsynchronized behavior, in the sense that for each circuit of the Petri net the firing is independent, resulting in moments where both legs are in swing; the bottom figure represents the synchronized behavior, according to Definition 2, i.e., a swing can only occur when the other leg is in stance.

where each lift-off event has an incoming arc from the opposing circuit. Here, the initial marking contains tokens on the stance places with one additional token in the synchronization place s_1 such that the net is alive. Figure 4-b2 illustrates a simulation where synchronization is present.

A general procedure to design a timed event graph that captures the synchronization of the legs can be summarized as follows:

1) For each leg i define a circuit with two events: leg touchdown t_i and lift off l_i; and two places: stance g_i and swing f_i
2) To synchronize event l_i with event t_j add a new place s_{ji} connecting t_j to l_i.
3) Initialize the marking such that all stance places have a token, i.e., the robot starts with all legs on the ground. Add one token to the minimum number of synchronization places s_{ji} such that all the upstream places of the transitions associated to the lift-off events of the legs that lift off first have one token.

By following the previous procedure it is clear that all places have a single incoming arc and a single outgoing arc. Using timed event graphs allows for most periodic gaits with intermediate ground contact to be modeled. By carefully adding synchronization places and choosing appropriate holding times, traditional gaits, such as trotting, pacing, etc. can be generated, i.e., a time schedule for the foot touchdown and lift off can be created. Time schedules for gaits with aerial phases can also be modeled in this fashion, but in this situation "negative"[2] holding times must be used in the synchronization places since the some legs lift off before the other legs have touched down. If Definition 2 is relaxed, to allow any types of synchronization other than the lift-off time of legs being a function of the touchdown of other legs, then other types of gaits can be modeled. Once an *event schedule* S (consisting of a matrix of real values that encode the desired time at which the feet should touchdown and lift off) is computed for a specific gait, it can be used to generate continuous-time reference phase trajectories via some periodic function f in time, resulting in the set of equations

$$\begin{aligned} \theta(\tau) &= f(\tau, S) \\ q_{\mathrm{ref}}(\tau) &= g(p, \theta(\tau)). \end{aligned}$$

In Section V we show how f can be constructed as a map, thus not requiring to solve a differential equation as in (1).

In this paper we focus on a class of gaits following Definition 2. As such, we write the equations that describe the behavior of the timed event graphs as sets of nonlinear equations. Given a timed event graph the process to obtain the associated evolution equations is:

1) For each event Ψ_i of the timed event graph assign the state variable $\psi_i(k) \in \mathbb{R}$ that represents the time at which the event Ψ_i fires for the k-th turn, with $k \in \mathbb{N}$.
2) Let $\mathcal{S}(\Psi_i)$ be the set of the indices of all events that have outgoing arcs a_{ji} to places that connect to Ψ_i. Let ν_j be the holding time of the origin place j of the arc a_{ji}, and let κ_j be the number of tokens in that place. Then write the equations:

$$\psi_i(k) = \max_{j \in \mathcal{S}(\Psi_i)} \left(\psi_i(k-\kappa_j) + \nu_j\right) \tag{6}$$

Equation (6) models timed event graphs where the events fire as soon as they are enabled, hence the use of the operator max. Consider the timed event graph example in Figure 4-b1, now with its events also firing as soon as they are enabled. Associate the holding time τ_{g} to the stance places g_i, the holding time τ_{f} to the swing places f_i, and the double-stance time τ_Δ to the synchronization places s_i. We now define the state variables as:

- $t_i(k)$ is the time instant the foot of leg i touches down for the kth cycle
- $l_i(k)$ is the time instant the foot of leg i lifts off the ground for the kth cycle

Following the previously described process we obtain the time evolution equations:

$$t_1(k) = l_1(k) + \tau_{\mathrm{f}} \tag{7}$$
$$t_2(k) = l_2(k) + \tau_{\mathrm{f}} \tag{8}$$
$$l_1(k) = \max\left(t_1(k-1) + \tau_{\mathrm{g}}, t_2(k-1) + \tau_\Delta\right) \tag{9}$$
$$l_2(k) = \max\left(t_2(k-1) + \tau_{\mathrm{g}}, t_1(k) + \tau_\Delta\right) \tag{10}$$

Equations (7)–(10) capture the synchronization requirements of the legs for a traditional biped walk. Equation (7) states that foot 1 touches down τ_{f} time units after it has lifted off the ground. Equation (9) states that foot 1 will lift off the ground after both feet have spent a total of τ_{g} time units in stance and τ_Δ time units after foot 2 has touched down. Equations (8) and (10) have an analogous interpretation. Note that the time parameters τ_{f}, τ_{g}, and τ_Δ represent the *minimal* swing, stance, and double-stance times, respectively, as opposed to

[2]Although negative holding times are not defined in the timed event graph framework, they can be safely used in the max-plus algebra framework that we adopt in Section III.

their exact times. Equation (6) (and (7)–(10)) contain only the $\max$ and $+$ operations, motivating the use of the max-plus algebra: first, (6) is nonlinear in the traditional algebra, but it is linear in the max-plus algebra; second, the theory of the max-plus algebra is well developed, and as such many properties can be inferred from the system matrices of the max-plus linear system. In the next section we explore these properties.

III. Max-plus algebra

A. Background

The max-plus algebra was introduced in the sixties by Giffler [28] and Cuninghame-Green [29]. In the late seventies the second author wrote the first book on the topic [14], and in the eighties Cohen et al. [30] presented a system-theoretic view. A few additional books have been published on the topic including [15], [16]. For a historical overview see [31]. The structure of the max-plus algebra is as follows: let $\varepsilon := -\infty$, $e := 0$, and $\mathbb{R}_{\max} = \mathbb{R} \cup \{\varepsilon\}$. Define the operations $\oplus, \otimes : \mathbb{R}_{\max} \times \mathbb{R}_{\max} \to \mathbb{R}_{\max}$ by:

$$\begin{aligned} x \oplus y &:= \max(x, y) \\ x \otimes y &:= x + y \end{aligned}$$

Definition 3. *The set $\mathbb{R}_{\max}$ with the operations $\oplus$ and $\otimes$ is called the max-plus algebra, denoted by $\mathcal{R}_{\max} = (\mathbb{R}_{\max}, \oplus, \otimes, \varepsilon, e)$.*

Theorem 4. [15] *The max-plus algebra $\mathcal{R}_{\max}$ has the algebraic structure of a commutative idempotent semiring.*

The max-plus algebra can be interpreted as the traditional linear algebra with the operations '$+$' and '$\times$' replaced by the operators '$\max$' and '$+$', respectively, with the supplemental difference that the additive inverse does not exist, thus resulting in a semiring. Matrices can be defined by taking Cartesian products of $\mathbb{R}_{\max}$. Define the matrix sum $\oplus$, matrix product $\otimes$, and matrix power operations by:

$$[A \oplus B]_{ij} = a_{ij} \oplus b_{ij} := \max(a_{ij}, b_{ij}) \tag{11}$$

$$[A \otimes C]_{ij} = \bigoplus_{k=1}^{m} a_{ik} \otimes c_{kj} := \max_{k=1,\dots,m} (a_{ik} + c_{kj}) \tag{12}$$

$$D^{\otimes k} := \underbrace{D \otimes D \otimes \ldots \otimes D}_{k \text{ times}}, \; k \in \mathbb{N} \backslash \{0\} \tag{13}$$

where $A, B \in \mathbb{R}_{\max}^{n \times m}$, $C \in \mathbb{R}_{\max}^{m \times p}$, $D \in \mathbb{R}_{\max}^{n \times n}$, and the i, j element of A is denoted by $a_{ij} = [A]_{ij}$. In this context, the max-plus zero $\mathcal{E}$, and (square) identity E matrices are defined by:

$$[\mathcal{E}]_{ij} = \varepsilon; \qquad [E]_{ij} = \begin{cases} e & \text{if } i = j \\ \varepsilon & \text{otherwise.} \end{cases}$$

Throughout this paper the dimensions of the matrices $\mathcal{E}$ and E are omitted since they can be unambiguously derived from the context. Finally, we define $D^{\otimes 0} := E$ and $x^{\otimes 0} := e$.

Theorem 5 (see [15], Th 3.17)**.** *Consider the following system of linear equations in the max-plus algebra:*

$$x = A \otimes x \oplus b \tag{14}$$

with $A \in \mathbb{R}_{\max}^{n \times n}$ and $b, x \in \mathbb{R}_{\max}^{n \times 1}$. Now let

$$A^* := \bigoplus_{p=0}^{\infty} A^{\otimes p} \; .$$

If A^* exists in $\mathbb{R}_{\max}^{n \times n}$ then

$$x = A^* \otimes b \tag{15}$$

solves the system of max-plus linear equations (14)*.*

Definition 6. *The matrix $A \in \mathbb{R}_{\max}^{n \times n}$ is called nilpotent if there exists a finite positive integer p_0 such that for all integers $p \geqslant p_0$ we have $A^{\otimes p} = \mathcal{E}$.*

Max-plus eigenvectors λ and eigenvalues v are defined in the same way as in the traditional algebra, where $v \neq \mathcal{E}$:

$$A \otimes v = \lambda \otimes v$$

For max-plus linear systems the max-plus eigenvalue of the system matrix represents the total cycle time, whereas the max-plus eigenvector represents the steady-state behavior. As an example, consider the following max-plus linear system where the initial condition is an eigenvector of A:

$$x(k) = A \otimes x(k-1); \qquad x(0) = v$$

The solution of the previous system can then be written as a function of the initial condition:

$$x(k) = \underbrace{A \otimes \cdots \otimes A}_{k \text{ times}} \otimes x(0) = \lambda^{\otimes k} \otimes v$$

So then the behavior of the state vector $x(k)$, i.e., the time instances at which each event fire, is a max-plus scaled version of the eigenvector v. Written in the traditional algebra we have that $x(k) = k\lambda\mathbb{1} + v$, where $\mathbb{1}$ is a column vector of 1's. So if the initial state is an eigenvalue, then at each cycle all events fire exactly λ time units after the last time they have fired. We now present conditions for the steady-state (eigenvector) to be reached given an arbitrary initial condition.

Definition 7. *A* permutation *matrix in max-plus algebra is a square matrix with a single 0 in every row and column and ε everywhere else.*

Definition 8. *The (square) matrix A is called* irreducible *if no permutation matrix B exists such that the matrix $\bar{A}$, defined by $\bar{A} = B^T \otimes A \otimes B$, has an upper triangular block structure (an alternative definition states that a matrix A is irreducible if its communication graph is strongly connected [16]).*

Theorem 9. [15] *If A is irreducible, there exists one and only one eigenvalue (but possibly several eigenvectors).*

Theorem 10. [30], [15] *Let A be an irreducible matrix. Then there exists $c \in \mathbb{N}$ (the cyclicity of A), $\lambda \in \mathbb{R}$ (the unique max-plus eigenvalue of A), and $k_0 \in \mathbb{N}$ (the coupling time of A) such that*

$$\forall p \geqslant k_0 \; : \; A^{\otimes p+c} = \lambda^{\otimes c} \otimes A^{\otimes p}$$

For a matrix with cyclicity one the coupling time states that given any initial condition $x(0)$ the system $x(k) = A \otimes x(k-1)$ reaches steady-state in at most k_0 steps, i.e.

$$A^{\otimes k_0} \otimes x(0) = \alpha \otimes v$$

where v is the eigenvector of A and $\alpha > 0$ is a scalar.

In this section we have presented two important elements of the theory of max-plus linear systems. First, Theorem 5 states that under proper assumptions implicit max-plus linear equations can be made explicit. The models we present next are first written in an implicit form (such as in (7)–(10)) and then translated to an explicit set of equations that are simple to solve. Second, Theorem 10 introduces the notion of the coupling time k_0. If k_0 can be computed, then in practice this means that a robot modeled as a max-plus-linear system can reach a steady-state walking pattern in at most k_0 steps given any initial state of the legs and any finite disturbance[3]. E.g., gait transitions stabilize in at most k_0 steps, all disturbances are rejected in k_0 steps, etc. This is similar to having stable limit cycles for CPGs.

IV. Gait scheduler

At the end of Section II-C we have introduced a set of nonlinear equations that model the time at which events occur during legged locomotion of a simple biped robot. Here, we take advantage of the max-plus algebra theory to systematically construct gait generators for robots with an arbitrary (larger than 1) number of legs.

A. Model

We start by rewriting (7)–(10) into a max-plus linear state-space representation:

$$\begin{bmatrix} t_1(k) \\ t_2(k) \\ l_1(k) \\ l_2(k) \end{bmatrix} = \left[\begin{array}{cc|cc} \varepsilon & \varepsilon & \tau_{\mathrm{f}} & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \tau_{\mathrm{f}} \\ \hline \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \tau_\Delta & \varepsilon & \varepsilon & \varepsilon \end{array}\right] \otimes \begin{bmatrix} t_1(k) \\ t_2(k) \\ l_1(k) \\ l_2(k) \end{bmatrix} \oplus \left[\begin{array}{cc|cc} \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \hline \tau_{\mathrm{g}} & \tau_\Delta & \varepsilon & \varepsilon \\ \varepsilon & \tau_{\mathrm{g}} & \varepsilon & \varepsilon \end{array}\right] \otimes \begin{bmatrix} t_1(k-1) \\ t_2(k-1) \\ l_1(k-1) \\ l_2(k-1) \end{bmatrix} \tag{16}$$

System (16) exhibits a clear structure that we now explore. By grouping together all the touchdown or lift-off events the system matrices in (16) are max-plus zero in the block diagonals, with all the parameters lying in the off-diagonal blocks. System (16) can be generalized in the following way: consider an n-legged robot, with the full discrete-event state vector defined by:

$$x(k) = [\underbrace{t_1(k) \ \cdots \ t_n(k)}_{t(k)} \ \underbrace{l_1(k) \ \cdots \ l_n(k)}_{l(k)}]^T.$$

The $2n$-dimensional system equations exemplified by (16) can be written in the general form:

$$\begin{bmatrix} t(k) \\ l(k) \end{bmatrix} = \left[\begin{array}{c|c} \mathcal{E} & \tau_{\mathrm{f}} \otimes E \\ \hline P & \mathcal{E} \end{array}\right] \otimes \begin{bmatrix} t(k) \\ l(k) \end{bmatrix} \oplus \left[\begin{array}{c|c} \mathcal{E} & \mathcal{E} \\ \hline \tau_{\mathrm{g}} \otimes E \oplus Q & \mathcal{E} \end{array}\right] \otimes \begin{bmatrix} t(k-1) \\ l(k-1) \end{bmatrix} \tag{17}$$

When the system in (17) reaches steady state all legs follow the same rhythm, i.e., all legs cycle with the same period of at least $\tau_{\mathrm{f}} + \tau_{\mathrm{g}}$ seconds (this is due to the terms $\tau_{\mathrm{f}} \otimes E$ and $\tau_{\mathrm{g}} \otimes E$ in the off-diagonal blocks). Following Definition 2, it is assumed that all leg synchronizations are achieved by enforcing a relation between the next lift-off time of a leg with the touchdown time of other legs. This assumption is expressed by the additional matrices P and Q, that we address later in this section. If one introduces identity matrices in system (17) that implement the extra trivial constraints $t(k+1) \geqslant t(k)$ and $l(k+1) \geqslant l(k)$, then the resulting system matrix is irreducible [32], facilitating the analysis of the system properties. We obtain the resulting model:

$$\begin{bmatrix} t(k) \\ l(k) \end{bmatrix} = \left[\begin{array}{c|c} \mathcal{E} & \tau_{\mathrm{f}} \otimes E \\ \hline P & \mathcal{E} \end{array}\right] \otimes \begin{bmatrix} t(k) \\ l(k) \end{bmatrix} \oplus \left[\begin{array}{c|c} E & \mathcal{E} \\ \hline \tau_{\mathrm{g}} \otimes E \oplus Q & E \end{array}\right] \otimes \begin{bmatrix} t(k-1) \\ l(k-1) \end{bmatrix} \tag{18}$$

By defining the matrices

$$A_0 = \left[\begin{array}{c|c} \mathcal{E} & \tau_{\mathrm{f}} \otimes E \\ \hline P & \mathcal{E} \end{array}\right] \quad \text{and} \quad A_1 = \left[\begin{array}{c|c} E & \mathcal{E} \\ \hline \tau_{\mathrm{g}} \otimes E \oplus Q & E \end{array}\right]$$

system (18) can be written in simplified notation:

$$x(k) = A_0 \otimes x(k) \oplus A_1 \otimes x(k-1). \tag{19}$$

System (18), written in an implicit form, models a class of n two-state circuits as illustrated in Figure 4, where the term $\tau_{\mathrm{g}} \otimes E$ represents the g_i places; $\tau_{\mathrm{f}} \otimes E$ represents the f_i places; and the matrices P and Q encode the s_{ji} places.

B. Gait parameterization

Towards constructing the matrices P and Q systematically, we consider the following notation: for a robot with n legs let $\ell_1, \ldots, \ell_m$ be sets of integers such that

$$\begin{aligned} &\bigcup_{p=1}^{m} \ell_p = \{1, \ldots, n\}, \\ &\forall i \neq j, \ell_i \cap \ell_j = \emptyset, \ \text{and} \\ &\forall p, \quad \ell_p \neq \emptyset \end{aligned}$$

i.e., each set ℓ_p is not empty, takes elements of $\{1, \ldots, n\}$ with no overlap between sets, and the union of all sets equals $\{1, \ldots, n\}$. We consider that each ℓ_p contains the indices of a set of legs that swing simultaneously. As such, m represents the number of groups of legs that are synchronized in phase. A gait G is defined as an ordering relation of ordered sets of leg indexes[4]:

$$\mathrm{G} = \ell_1 \prec \ell_2 \prec \cdots \prec \ell_m \tag{20}$$

This ordering relation is interpreted in the following manner: each leg in the set ℓ_{i+1} swings τ_Δ time units after all the legs in the set ℓ_i have reached stance. For example, a trotting gait on a quadruped robot where the legs are sorted as in Figure 3, is represented by:

$$\mathrm{G}_{\mathrm{trot}} = \{1, 4\} \prec \{2, 3\}$$

[3] Any event that causes a delay in the touch down or lift off of a leg is considered a disturbance

[4] This definition does not capture gaits where there are a multiplicity of cycles between legs, e.g., one leg cycles twice in the time another leg cycles once.

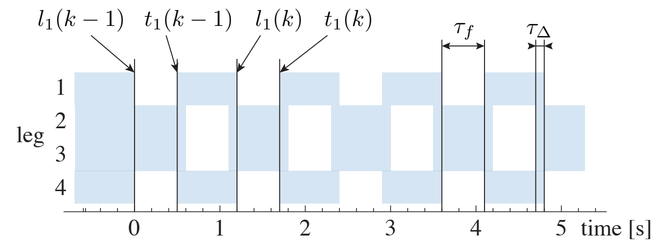

Fig. 5. A trot gait on a quadruped robot, using Hildebrand's diagram notation [34]. Gray/Blue rectangles represent stance and white swing.

Describing a swing leg ordering using the notation of $\mathrm{G}_{\mathrm{trot}}$, as an ordered set of sets, is equivalent to specifying legs phase offsets in the CPG method via the parameters ϕ_{ij} in (2). Given this notation, the matrices P and Q in (18) can be generated by:

$$[P]_{pq} = \begin{cases} \tau_\Delta & \forall j \in \{1,\ldots,m-1\}; \forall p \in \ell_{j+1}; \forall q \in \ell_j \\ \varepsilon & \text{otherwise} \end{cases} \tag{21}$$

$$[Q]_{pq} = \begin{cases} \tau_\Delta & \forall p \in \ell_1; \forall q \in \ell_m \\ \varepsilon & \text{otherwise} \end{cases} \tag{22}$$

For the trotting gait $\mathrm{G}_{\mathrm{trot}}$ we obtain:

$$P = \begin{bmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \tau_\Delta & \varepsilon & \varepsilon & \tau_\Delta \\ \tau_\Delta & \varepsilon & \varepsilon & \tau_\Delta \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \end{bmatrix} \text{ and } Q = \begin{bmatrix} \varepsilon & \tau_\Delta & \tau_\Delta & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ \varepsilon & \tau_\Delta & \tau_\Delta & \varepsilon \end{bmatrix} \tag{23}$$

In [32] we have shown that P is always max-plus nilpotent for gaits generated by expressions (21) and (22).

Lemma 11. [33] *A sufficient condition for A_0^* to exist in $\mathbb{R}_{\max}^{n\times n}$ is that the matrix P is nilpotent in the max-plus sense.*

We now have all the ingredients in place to write system (19) explicitly. Using Theorem 5 we define the *system matrix* A to be:

$$A := A_0^* \otimes A_1 \tag{24}$$

Equation (19) can be rewritten as:

$$\begin{aligned} x(k) &= A_0 \otimes x(k) \oplus A_1 \otimes x(k-1) \\ &= A_0^* \otimes A_1 \otimes x(k-1) \\ &= A \otimes x(k-1). \end{aligned} \tag{25}$$

If the gait parameters are chosen[5] to be $\tau_{\mathrm{f}} = 0.5$s, $\tau_{\mathrm{g}} = 0.7$s, and $\tau_\Delta = 0.1$s, equation (25) generates the following event schedule (that we previously denoted by S) for the first 4 steps:

k	$t_1(k)$	$t_2(k)$	$t_3(k)$	$t_4(k)$	$l_1(k)$	$l_2(k)$	$l_3(k)$	$l_4(k)$
1	0.5	1.1	1.1	0.5	0.0	0.6	0.6	0.0
2	1.7	2.3	2.3	1.7	1.2	1.8	1.8	1.2
3	2.9	3.5	3.5	2.9	2.4	3.0	3.0	2.4
4	4.1	4.7	4.7	4.1	3.6	4.2	4.2	3.6

This table is interpreted as follows: legs 1 and 4 lift off the ground when time equals zero and touch down after 0.5 seconds. When the time counter equals 0.6 seconds, legs 2 and 3 lift off the ground, and so forth, as illustrated in Figure 5. The previous table stores at what time instants certain events should occur.

C. Properties

The system matrix A, defined by (24), has a number of mathematical properties that shed light on the resulting gait behavior. The max-plus eigenvalue of A is the total cycle time, its max-plus eigenvector represents the steady-state behavior, and the coupling time of A describes the transient behavior [15].

In order for us to determine the eigenvalue and eigenvector some assumptions have to be made:

Assumption A1: $\tau_{\mathrm{f}} > 0$, and $\tau_{\mathrm{g}} > 0$

This assumption is always true in practice since the swing and stance times are always positive numbers.

Lemma 12. [35] *If assumption A1 is satisfied then*

$$\lambda := (\tau_{\mathrm{f}} \otimes \tau_\Delta)^{\otimes m} \oplus \tau_{\mathrm{f}} \otimes \tau_{\mathrm{g}} \tag{26}$$

is a max-plus eigenvalue of the system matrix A defined by equations (25), and $v \in \mathbb{R}_{\max}^{2n}$ defined by

$$\forall j \in \{1,\ldots,m\}, \forall q \in \ell_j : \; [v]_q := \tau_{\mathrm{f}} \otimes (\tau_{\mathrm{f}} \otimes \tau_\Delta)^{\otimes j-1} \tag{27}$$

$$[v]_{q+n} := (\tau_{\mathrm{f}} \otimes \tau_\Delta)^{\otimes j-1} \tag{28}$$

is a max-plus eigenvector of A.

The relevance of the previous lemma is that (27) and (28) provide a closed-form expression of a max-plus eigenvector of the system matrix A.

Assumption A2: $(\tau_{\mathrm{f}} \otimes \tau_\Delta)^{\otimes m} \geqslant \tau_{\mathrm{f}} \otimes \tau_{\mathrm{g}}$

This assumption can be interpreted as a restriction on the choice of the parameters τ_{f}, τ_{g}, and τ_Δ.

Lemma 13. [35] *Given assumptions A1, A2, the coupling time for the max-plus-linear system defined by (25) is $k_0 = 2$ with cyclicity $c = 1$.*

The significance of this lemma is in stating that the steady-state of system (25) is reached in at most 2 steps. This result is important in robotics since it shows that it is possible for a robot to transition between arbitrary gaits without stopping, and it will return to its steady-state behavior after a disturbance, in at most 2 steps. Note that by a single step we mean a single cycle of the discrete-event system, i.e., all the n legs of the robot go through a swing/stance cycle.

V. Control structure

In this section we present a modular control structure that implements the presented max-plus framework both in simulation and in reality on the legged robots illustrated in Figure 3. This structure, illustrated in Figure 6, consists of four control

[5]The parameters $\tau_{\mathrm{f}}, \tau_{\mathrm{g}}$, and τ_Δ are design parameters that can be freely chosen by the user, as long as the assumptions A1, A2 presented in Section V.C below are verified.

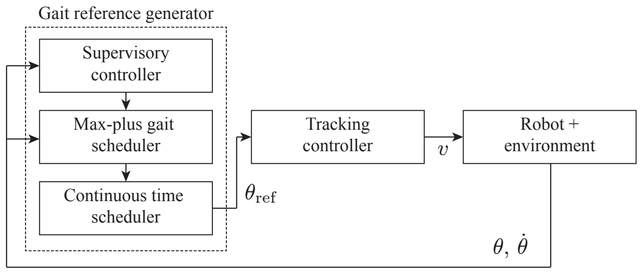

Fig. 6. Block diagram of the control structure using the max-plus algebra framework for legged locomotion presented in this paper.

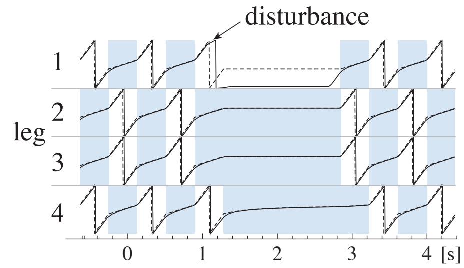

Fig. 7. Experimental run on a quadruped: the dashed lines represent the reference phase, ranging from $(-\pi, \pi]$ (vertical lines represent the phase wrapping around), the solid lines represent the actual phase of each leg. In this experiment, leg 1 is prevented from touching down, resulting in the phase of all other legs to be delayed.

blocks: the supervisory controller, tasked with choosing a gait; the max-plus gait generator, which generates an event schedule; the continuous time scheduler, which transforms the event schedule into a continuous time reference trajectory; and finally the tracking controller, which enforces the desired reference trajectory. Note that both the supervisory controller and the max-plus gait generator blocks use feedback on the phase state to update the internal scheduling.

The choice of the supervisory controller can be a function of the terrain, desired speed, or other considerations. Section VI is dedicated to gait transitions, further elaborating on the operation of the supervisory control block.

The event schedule $S \in \mathbb{R}_{\max}^{2n\times(2p+1)}$ for $p \geqslant 1$ (in the case of the robots utilized in this paper we use $p = 1$) is defined to be the matrix

$$S = \begin{bmatrix} x(k-p) & \cdots & x(k) & \cdots & x(k+p) \end{bmatrix}.$$

The parameters p and k are chosen such that at the time instant τ for each row of S we get that

$$\min([S]_{i,\cdot}) < \tau < \max([S]_{i,\cdot}),$$

i.e., for each leg S contains both events that have happened in the past and events that are scheduled to occur in the future (in a practical implementation the matrix S is updated at discrete time instants that are a function of the total cycle time, thus S is actually a function of time). If we consider that foot i always touches down when its phase is at a fixed value θ_{t} and always lifts off the ground at the fixed phase θ_{l} then it is possible to generate a reference phase via the function

$$\theta_{\text{ref}}(\tau, S(\tau)) : \mathbb{R} \times \mathbb{R}_{\max}^{2n\times(2p+1)} \to \mathbb{T}^n$$

that takes as inputs time $\tau \in \mathbb{R}$ and the event schedule and outputs a piecewise affine trajectory for each of the leg's phases:

$$[\theta_{\text{ref}}]_i := \begin{cases} \dfrac{\theta_{\text{l}}\left(t_i(k_{\text{t}i}) - \tau\right) + (\theta_{\text{t}} + 2\pi)\left(\tau - l_i(k_{\text{l}i})\right)}{t_i(k_{\text{t}i}) - l_i(k_{\text{l}i})} \\ \qquad\qquad \text{if } \tau \in [l_i(k_{\text{l}i}), t_i(k_{\text{t}i})) \\ \dfrac{\theta_{\text{t}}\left(l_i(k_{\text{l}i}+1) - \tau\right) + \theta_{\text{l}}\left(\tau - t_i(k_{\text{t}i})\right)}{l_i(k_{\text{l}i}+1) - t_i(k_{\text{t}i})} \\ \qquad\qquad \text{if } \tau \in [t_i(k_{\text{t}i}), l_i(k_{\text{l}i}+1)) \end{cases} \tag{29}$$

The indices in the event counter variables $k_{\text{t}i}$ and $k_{\text{l}i}$ are used here to distinguish that for each leg i a different event counter is in use for interpolation. The interpretation of expression (29) is as follows: the function θ_{ref} interpolates the phase parameters θ_{t} and θ_{l} linearly in time τ. For a specific leg i, if it is in stance, then the interval $[t_i(k_{\text{t}i}), l_i(k_{\text{l}i}+1))$ is used for interpolation of the phase, such that at time $\tau = t_i(k_{\text{t}i})$ the reference phase is θ_{t} and at time $\tau = l_i(k_{\text{l}i}+1)$ the reference phase for leg i is θ_{l}. If the leg is in swing, then the interpolation interval $[l_i(k_{\text{l}i}), t_i(k_{\text{t}i}))$ is used instead. Figure 7 illustrates a sample simulation where the reference phase is represented by the dashed lines. For each leg, each graph ranges from $-\pi$ to π and the phase "wraps around" when crossing π, as illustrated by the vertical lines.

Note that function $\theta_{\text{ref}}(\tau, S(\tau))$ is general, and can be used in place of a CPG type generator, as in (1), resulting in a new discrete-event type of reference trajectory generator for the actuators of the robot:

$$q_{\text{ref}}(\tau) = g(p, \theta_{\text{ref}}(\tau, S(\tau))) \tag{30}$$

For a tripod gait $\{1, 4, 5\} \prec \{2, 3, 6\}$ with the parameters $\phi_{\text{s}} = \theta_{\text{t}} + \theta_{\text{l}}$, (29) results in the Buehler Clock equations (4)-(5). Thus, the switching max-plus method is a generalization of the Buehler Clock. We can now establish a comparison between the standard CPG versus the switching max-plus methodology, illustrated in Table I.

VI. Gait switching

We now address the problem of choosing gaits and their transition parameters when changing rhythms. In Section VI-A we discuss how to choose gaits to obtain the best possible transitions given the models presented earlier. In Section VI-B we introduce a new scheme that results in an equal stance time for all legs during transitions. This result is used in Section VI-C to enable constant acceleration/deceleration in legged robots while switching gaits.

A. Compatible gaits for switching

Let G_n, called the *gait space*, be the set of all gaits defined according to expression (20) for an n-legged robot. Also, let $\mathcal{A}_n$ be the set of all system matrices for gaits generated from (20) with equation (25):

$$\mathcal{A}_n = \{A(1), \ldots, A(n)\}$$

TABLE I
COMPARISON BETWEEN STANDARD CPG AND SWITCHING MAX-PLUS METHODS.

Property	CPG	Switching max-plus
Dynamics	continuous	discrete
System representation	differential equation (1)	max-plus linear system (18)
Control parameterization	set of phase offset parameters and gains: w_{ij}, ϕ_{ij}	ordered set of numbers $\ell_1 \prec \ell_2 \prec \cdots \prec \ell_m$ and time parameters $\tau_\mathrm{f}, \tau_\mathrm{g}, \tau_\Delta$
Steady state	limit cycle	max-plus eigenvector
Cycle time	depending on the gain	max-plus eigenvalue
Convergence	depending on the gain	maximum 2 cycles
Transitions with constraint guarantees	obstacles encoded in vector fields	switch state matrices
Implementation	numerical differential equation solver	additions, maximizations, linear interpolation
Output smoothness	C^∞	C^n with n finite

One can write the switching max-plus linear system

$$x(k+1) = A(\mu(k)) \otimes x(k)$$

where $\mu(k)$ is a "switching" integer function whose value is determined by the supervisory controller based on the desired gait. By construction, gait switching is kinematic stance stable, in the sense that for two different gaits that swing at most q_i and q_j legs simultaneously, we will have at most $\max(q_i, q_j)$ legs swinging during the transition between both. For example during the transition between a walk and a trot on a quadruped robot, no more than two legs can swing simultaneously (note that since we are not taking into consideration the dynamics of the robot this measure of "stability" applies only to the discrete-event supervisory controller). By looking at the definition of a gait in expression (20) it is clear that the size of the gait space G_n is combinatorial in n (in fact $\#\mathsf{G}_n = n! \times (2^{(n-1)} - 1)$, i.e., the number of permutations of n elements times the number of possible set partitions, excluding the partition consisting of a set with n elements). However, different representations for a gait as an ordered set of ordered sets can lead to the same exact robot physical motion behavior, as in the following example:

$$\begin{aligned} \mathrm{G}_1 &= \{1,2\} \prec \{3,4\} \prec \{5,6\} \\ \mathrm{G}_2 &= \{2,1\} \prec \{3,4\} \prec \{5,6\} \\ \mathrm{G}_3 &= \{5,6\} \prec \{1,2\} \prec \{3,4\} \\ \mathrm{G}_4 &= \{4,3\} \prec \{6,5\} \prec \{2,1\} \\ &\cdots \end{aligned}$$

The difference lies in the fact that the transition between the above defined gaits and a new different gait, say $\mathrm{G}_5 = \{3,4,6\} \prec \{1,2,5\}$, will result in a different transient behavior, as illustrated in the examples of Figures 10a) and 10b). This poses the question of how to optimally switch gaits, in the sense of minimizing the variation of the leg stance time during gait switching. For applications of climbing robots [36] it is fundamental that all legs exert the same force on the attaching wall at all times, thus motivating constant foot velocity (viewed from a frame attached to the robot). The same is valid for walking robots, as different leg velocities can result in turning moments that can make the legged platform unstable. For the n-legged robot with gaits represented by (20) suppose the gait switching mechanism consists in moving a single leg from one group of legs ℓ_i to a different group of legs ℓ_j with $0 < i, j \leqslant m$. By inspecting the max-plus eigenvector (thus assuming steady-state behavior), one can observe that the moment that a leg in the set ℓ_i lifts off the ground happens at the time instant

$$(\tau_\mathrm{f} \otimes \tau_\Delta)^{\otimes i},$$

assuming the cycle starts at zero time. Analogously, for a leg in the set ℓ_j we get the lift-off time to be:

$$(\tau_\mathrm{f} \otimes \tau_\Delta)^{\otimes j},$$

Moving a leg from the set ℓ_i to the set ℓ_j results in a change of lift-off time of

$$(\tau_\mathrm{f} \otimes \tau_\Delta)^{\otimes (j-i)}$$

If $j > i$, then the switching leg will stay in stance for an extra $(\tau_\mathrm{f} \otimes \tau_\Delta)^{\otimes (j-i)}$ time units during the transition to synchronize with the new leg group. This is always the case since the time of flight τ_f is fixed. If $j < i$ then all the legs in the original group of the switching leg will have their lift-off times postponed by $(\tau_\mathrm{f} \otimes \tau_\Delta)^{\otimes (i-j)}$ time units. Thus, the larger the magnitude of $j - i$ the larger the stance time variation during the transition will be. For instance, the gait transition of

$$\{1, \mathbf{2}\} \prec \{3,4\} \prec \{5\} \prec \{6\} \to \{1\} \prec \{\mathbf{2},3,4\} \prec \{5\} \prec \{6\}$$

has less stance time variation than the transition

$$\{1, \mathbf{2}\} \prec \{3,4\} \prec \{5\} \prec \{6\} \to \{1\} \prec \{3,4\} \prec \{\mathbf{2},5\} \prec \{6\}$$

The same is true when changing the number of leg groups, e.g., the gait transition of

$$\{1,2,3\} \prec \{4,5,6\} \to \{1,2\} \prec \{3,4\} \prec \{5,6\}$$

has less stance time variation then the transition

$$\{1,2,3\} \prec \{4,5,6\} \to \{5,6\} \prec \{1,2\} \prec \{3,4\}$$

This provides a simple mechanism for choosing gaits without requiring to search the gait space for all structurally equivalent gaits. Figure 10 illustrates the comparison of a non-optimal gait switch a) with an optimal one b). To quantify the quality of a gait transition, we introduce the following measure:

$$\bar{\sigma} = \frac{1}{\tau_\mathrm{g}} \sqrt{\frac{1}{n} \sum_{i=1}^{n} (\bar{\tau}_{\mathrm{g}i} - \bar{\bar{\tau}}_\mathrm{g})^2} \tag{31}$$

where $\bar{\tau}_{\mathrm{g}i}$ is the true stance time of leg i, and $\bar{\bar{\tau}}_\mathrm{g}$ is the average stance time for all legs, both during the transition. In formula (31) we divide the unbiased standard deviation of $\bar{\tau}_{\mathrm{g}i}$ by the desired stance time τ_g to obtain a non-dimensional measure. If $\bar{\sigma} = 0$ then the transition maintains a constant stance time for all legs. Note that minimizing $\bar{\sigma}$ results in minimizing the variation of the foot velocities during stance (assuming a constant foot velocity for the stance phase range), as exemplified in Figure 10 .

B. Variable swing time, constant stance model

As shown before, by selecting the leg indices in the proper way when switching a gait, one can achieve a better switching behavior. However, by construction, since the synchronization happens at the lift-off time, during gait transitions some legs will inevitably stay longer on the ground, which can cause instabilities to the robotic platform. We now show that by manipulating the flight time of each leg independently one can achieve a unique stance time for all legs under well defined assumptions. Consider the new model:

$$\begin{bmatrix} t(k) \\ l(k) \end{bmatrix} = \left[\begin{array}{c|c} \mathcal{E} & R \\ \hline P & \mathcal{E} \end{array}\right] \otimes \begin{bmatrix} t(k) \\ l(k) \end{bmatrix} \oplus \left[\begin{array}{c|c} E & \mathcal{E} \\ \hline \tau_{\mathrm{g}} \otimes E \oplus Q & E \end{array}\right] \otimes \begin{bmatrix} t(k-1) \\ l(k-1) \end{bmatrix} \quad (32)$$

where the diagonal matrix R represents different swing times:

$$R = \begin{bmatrix} \tau_{\mathrm{f}1} & \varepsilon & \cdots & \varepsilon \\ \varepsilon & \tau_{\mathrm{f}2} & & \\ \vdots & & \ddots & \\ \varepsilon & & & \tau_{\mathrm{f}n} \end{bmatrix}$$

Following the definition (24) let

$$\bar{A}(\mathrm{G}, R, \tau_{\mathrm{g}}, \tau_{\Delta}) := \left[\begin{array}{c|c} \mathcal{E} & R \\ \hline P_{\mathrm{G},\tau_\Delta} & \mathcal{E} \end{array}\right]^* \otimes \left[\begin{array}{c|c} E & \mathcal{E} \\ \hline \tau_{\mathrm{g}} \otimes E \oplus Q_{\mathrm{G},\tau_\Delta} & E \end{array}\right]$$

where the matrices $P_{\mathrm{G},\tau_\Delta}$ and $Q_{\mathrm{G},\tau_\Delta}$ are constructed according expressions (21) and (22), respectively, for a gait G. Then, the system matrix of (25) is parameterized as:

$$\bar{A}(\mathrm{G}, \tau_{\mathrm{f}} \otimes E, \tau_{\mathrm{g}}, \tau_{\Delta})$$

and the resulting system matrix of (32) is parameterized by:

$$\bar{A}(\mathrm{G}, R, \tau_{\mathrm{g}}, \tau_{\Delta})$$

Let $\max_v : \mathbb{R}^n \to \mathbb{R}$ and $\min_v : \mathbb{R}^n \to \mathbb{R}$ be operators on vectors that return the maximum or the minimum element of a vector, respectively. Now consider two different gaits G_1 and G_2 with respective eigenvectors $v_{\mathrm{G}_1} = [t_{\mathrm{G}_1}^T \; l_{\mathrm{G}_1}^T]^T$ and $v_{\mathrm{G}_2} = [t_{\mathrm{G}_2}^T \; l_{\mathrm{G}_2}^T]^T$. During a transition from gait G_1 to the gait G_2 the extra time each leg will stay in stance can be computed by:

$$\Gamma = (l_{\mathrm{G}_2} - t_{\mathrm{G}_1}) - \min_v(l_{\mathrm{G}_2} - t_{\mathrm{G}_1}) \quad (33)$$

A transition system matrix $\bar{A}(\mathrm{G}_1, R_1, \tau_{\mathrm{g}}, \tau_{\Delta})$ can be constructed such that for each leg an element of the "extra time" vector $\Gamma \in \mathbb{R}^n_{\max}$ is subtracted from the flight time τ_{f} so that in the next cycle, now using gait G_2, will make the real stance time $\bar{\tau_{\mathrm{g}}}$ the same for each leg. Note that this is only possible if

$$\tau_{\mathrm{fG}_1} \geqslant \max_v(\Gamma),$$

where τ_{fG_1} is the swing time parameter for gait G_1. If that is not the case, then an additional transition matrix, now using gait G_2, can be constructed as $\bar{A}(\mathrm{G}_2, R_2, \tau_{\mathrm{g}}, \tau_{\Delta})$ such that the time that cannot be subtracted from the transition matrix R_1 is subtracted from the matrix R_2. The resulting transition algorithm is summarized as follows:

1) Given two gaits G_1 and G_2 compute Γ via (33).
2) If $\tau_{\mathrm{fG}_1} \geqslant \max_v(\Gamma)$ then compute the vector:

$$\Gamma_{t1} = [(\tau_{\mathrm{fG}_1} - [\Gamma]_1) \; \cdots \; (\tau_{\mathrm{fG}_1} - [\Gamma]_n)]^T$$

and the system matrix

$$\bar{A}(\mathrm{G}_1, \mathrm{diag}(\Gamma_{t1}), \tau_{\mathrm{gG}_1}, \tau_{\Delta\mathrm{G}_1})$$

where $\mathrm{diag} : \mathbb{R}^n \to \mathbb{R}^{n\times n}$ returns a matrix with the elements of a vector on the leading diagonal. The transition sequence is obtained by the following sequence of system matrices:

$$\begin{aligned} A(\mu(k-p)) &= \bar{A}(\mathrm{G}_1, \tau_{\mathrm{fG}_1} \otimes E, \tau_{\mathrm{gG}_1}, \tau_{\Delta\mathrm{G}_1}) \\ &\vdots \\ A(\mu(k-1)) &= \bar{A}(\mathrm{G}_1, \tau_{\mathrm{fG}_1} \otimes E, \tau_{\mathrm{gG}_1}, \tau_{\Delta\mathrm{G}_1}) \\ A(\mu(k)) &= \bar{A}(\mathrm{G}_1, \mathrm{diag}(\Gamma_{t1}), \tau_{\mathrm{gG}_1}, \tau_{\Delta\mathrm{G}_1}) \\ A(\mu(k+1)) &= \bar{A}(\mathrm{G}_2, \tau_{\mathrm{fG}_2} \otimes E, \tau_{\mathrm{gG}_2}, \tau_{\Delta\mathrm{G}_2}) \\ &\vdots \\ A(\mu(k+p)) &= \bar{A}(\mathrm{G}_2, \tau_{\mathrm{fG}_2} \otimes E, \tau_{\mathrm{gG}_2}, \tau_{\Delta\mathrm{G}_2}) \end{aligned}$$

3) If $\tau_{\mathrm{fG}_1} < \max_v(\Gamma)$ then create two transition matrices

$$\bar{A}(\mathrm{G}_1, \mathrm{diag}(\Gamma_{t1}), \tau_{\mathrm{gG}_1}, \tau_{\Delta\mathrm{G}_1})$$

and

$$\bar{A}(\mathrm{G}_2, \mathrm{diag}(\Gamma_{t2}), \tau_{\mathrm{gG}_2}, \tau_{\Delta\mathrm{G}_2})$$

where

$$[\Gamma_{t1}]_i = \max(\min([\Gamma]_i, \tau_{\mathrm{fG}_1}), \tau_{f\min})$$

with $\tau_{f\min} > 0$ the minimum leg swing time, and

$$[\Gamma_{t2}]_i = \tau_{\mathrm{fG}_2} - ([\Gamma_{t1}]_i - [\Gamma]_i) - \min_v(\Gamma_{t1} - \Gamma)$$

The transition sequence is obtained by the following sequence of system matrices:

$$\begin{aligned} A(\mu(k-p)) &= \bar{A}(\mathrm{G}_1, \tau_{\mathrm{fG}_1} \otimes E, \tau_{\mathrm{gG}_1}, \tau_{\Delta\mathrm{G}_1}) \\ &\vdots \\ A(\mu(k-1)) &= \bar{A}(\mathrm{G}_1, \tau_{\mathrm{fG}_1} \otimes E, \tau_{\mathrm{gG}_1}, \tau_{\Delta\mathrm{G}_1}) \\ A(\mu(k)) &= \bar{A}(\mathrm{G}_1, \mathrm{diag}(\Gamma_{t1}), \tau_{\mathrm{gG}_1}, \tau_{\Delta\mathrm{G}_1}) \\ A(\mu(k+1)) &= \bar{A}(\mathrm{G}_2, \mathrm{diag}(\Gamma_{t2}), \tau_{\mathrm{gG}_2}, \tau_{\Delta\mathrm{G}_2}) \\ A(\mu(k+2)) &= \bar{A}(\mathrm{G}_2, \tau_{\mathrm{fG}_2} \otimes E, \tau_{\mathrm{gG}_2}, \tau_{\Delta\mathrm{G}_2}) \\ &\vdots \\ A(\mu(k+p)) &= \bar{A}(\mathrm{G}_2, \tau_{\mathrm{fG}_2} \otimes E, \tau_{\mathrm{gG}_2}, \tau_{\Delta\mathrm{G}_2}) \end{aligned}$$

Figure 8.c) illustrates an example transition with constant stance times τ_{g} and different τ_{f} for each leg during the transitions, highlighted by the green shades of color.

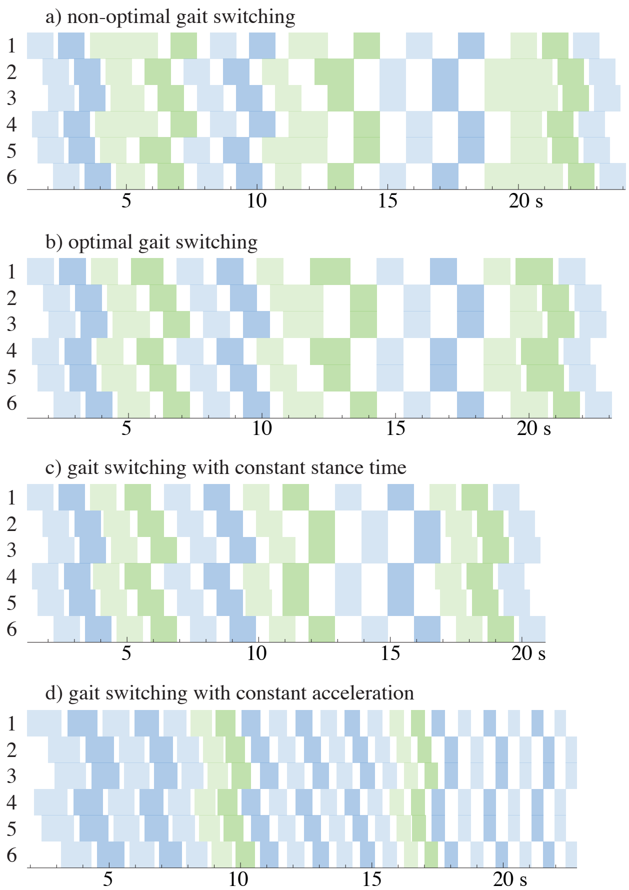

Fig. 8. Various real walking experiments executed using the Zebro robot. The color/gray bars represent legs in stance and the transparent areas represent leg swing. Transition steps are indicated by the green shades of color/solid outlines. Subfigure a) non-optimal gait switch for the transitions $\{1\} \prec \{4\} \prec \{5\} \prec \{2\} \prec \{3\} \prec \{6\} \to \{5,2\} \prec \{3,6\} \prec \{1,4\} \to \{2,3,6\} \prec \{1,4,5\}$. Non-dimensional standard deviation for transition 1 is $(\bar{\sigma})_1 = 0.57$, for transition 2 is $(\bar{\sigma})_2 = 0.45$, and for transition 3 is $(\bar{\sigma})_3 = 0.80$. Subfigure b) optimal gait transitions with fixed τ_{f} for the gaits $\{1\} \prec \{4\} \prec \{5\} \prec \{2\} \prec \{3\} \prec \{6\} \to \{1,4\} \prec \{5,2\} \prec \{3,6\} \to \{1,4,5\} \prec \{2,3,6\}$. $(\bar{\sigma})_1 = 0.14$, $(\bar{\sigma})_2 = 0.33$, and $(\bar{\sigma})_1 = 0.19$. Subfigure c) optimal gait switch with transitions with variable τ_{f}. $(\bar{\sigma})_1 = (\bar{\sigma})_2 = (\bar{\sigma})_3 = 0$. Subfigure d) gait transitions with constant acceleration.

C. Variable velocity

Variable velocity can be achieved by scaling the time τ. As presented earlier, the actuator reference trajectories q_{ref} are generated by the following equation:

$$q_{\mathrm{ref}}(\tau) = g(p, \theta_{\mathrm{ref}}(\tau, S(\tau))) \tag{34}$$

By introducing a "time modulating" function $\alpha : \mathbb{R} \to \mathbb{R}$ we obtain a new reference phase generator:

$$q_{\mathrm{ref}}(\tau) = g(p, \theta_{\mathrm{ref}}(\alpha(\tau), S(\alpha(\tau)))) \tag{35}$$

A constant accelerating robot can be obtained by choosing $\alpha(\tau) = a\tau$ where a is the desired acceleration. Taking into account the minimum time required for a leg to swing, gait switching can be automatically inferred for each resulting forward velocity. Figure 8.d) illustrates a hexapod robot that is constantly accelerating and doing gait transitions for a hexapod robot.

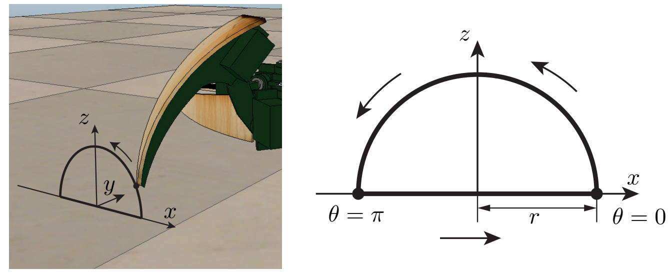

Fig. 9. A parameterized trajectory for the foot end-effector of the 3 degree of freedom per leg hexapod robot available in V-Rep.

VII. Simulation and Experimental Results

In this paper we utilize the robots Zebro and RQuad, which are morphologically identical to RHex [25], for experimental validation and the V-Rep software [26] for physics simulation, illustrated in Figure 3. The physical robots have a single motor per leg, and as such the dimensions of the vectors q_{ref} and θ match. For simulation we utilized a 23 degree-of-freedom hexapod robot in the V-Rep simulation environment, resulting in $q_{\mathrm{ref}} \in \mathbb{R}^{18}$ and $\theta \in \mathbb{T}^6$.

A. Simulation on a 3 DOF per leg hexapod robot

We have applied the work presented in this paper to a 3 degree of freedom per leg hexapod robot present in the V-Rep simulation environment. The map g that translates the abstract phase θ into reference trajectories of the end effector, using the parameters $p = \{r\}$, is written as (see Figure 9):

$$g(\theta) = \begin{bmatrix} x_{\mathrm{ref}} \\ y_{\mathrm{ref}} \\ z_{\mathrm{ref}} \end{bmatrix} = \begin{cases} \begin{bmatrix} r\cos(\theta) \\ 0 \\ r\sin(\theta) \end{bmatrix} & \text{if } 0 \leqslant \theta \bmod 2\pi < \pi \\ \begin{bmatrix} \frac{2r}{\pi}((\theta - \pi) \bmod 2\pi) - r \\ 0 \\ 0 \end{bmatrix} & \text{otherwise} \end{cases}$$

The abstract phase θ is obtained using (29) with $\theta_{\mathrm{t}} = 0$ and $\theta_{\mathrm{l}} = \pi$. The simulation results from V-Rep are illustrated on the left side of Figure 10. The simulated controller implements an inverse kinematics module to track the reference trajectories of the feet end-effectors in the local reference frame of the body, resulting in forward motion. In Figure 10 the gait transitions are highlighted by the solid blue bars. A constant acceleration "trend" is seen although the average velocity is not exactly linear. This is due to the complex ground interactions and possible slip happening in the simulation. It is also noticeable that different gaits result in difference oscillating patterns in the pitch-yaw-roll directions. For example, the tripod gait results in less yaw drift than the quadruped gait, due to its symmetry.

B. Experiments on a hexapod robot

The morphology of RHex/Zebro is such that each leg is directly mounted onto a motor. Therefore, one can match the abstract leg phase directly to the leg shaft angle. In this situation, the function g_{zebro}, defined in (30), is simply $g_{\text{zebro}}(p, \theta) := \theta$ for straight-line motion[6].

For the Zebro and RQuad robots the reference trajectory tracker block from Figure 6 is a simple PID phase tracker:

$$u(\tau) = K_P(\theta_{\text{ref}}(\tau) - \theta(\tau)) + K_D(\dot{\theta}_{\text{ref}}(\tau) - \dot{\theta}(\tau)) \\ + K_I \int_{\tau_0}^{\tau} (\theta_{\text{ref}}(s) - \theta(s))ds,$$

where $\theta(\tau)$ represents the leg shaft angles. Since these robots do not have leg touch sensors, we consider that the touchdown and lift-off events fire as a function of the leg angle. In practice this works well, allowing the robot to locomote without the need of touch sensors. In Figure 6 the phases $\theta(\tau)$ and $\dot{\theta}(\tau)$ are fed back to the controller in three locations: in the reference trajectory tracker, to update the input signals; in the max-plus gait scheduler, to keep track of when the leg touchdown and lift off actually occur; and in the supervisory controller, to trigger gait switching when necessary. Figure 7 illustrates an experiment executed in the RQuad robot where leg 1 was prevented from touching down. Since the touchdown t_1 event for leg 1 does not occur, all other events depending on t_1 are automatically postponed in time, resulting in the reference phases illustrated by the dashed lines. Once leg 1 is released and its touchdown event occurs, the motion of the other legs continues as normal. In practice, the max-plus gait generator prevents the robot from tripping due to lack of support if one or more legs are held back during their swing. As such it guarantees that a desired number of legs are in stance at all times. If one of the legs never touches down, then this information can be fed to the supervisory controller, which can switch gaits or take other recovery actions.

Figure 10 on the right illustrates a constant acceleration experiment on the Zebro robot. As in the simulation results in V-Rep, a similar velocity trend is found for the Zebro robot, here "less linear" as in the case of simulation. Once more we attribute these results to the complex interactions of the robot with the terrain.

VIII. Conclusions

This paper presents a discrete-event modeling approach for leg phases in walking robots. We have shown that modeling each foot's interaction with the ground via switching max-plus linear systems presents a feasible alternative to the traditional CPG approach for motion control in legged locomotion. In our approach it is not necessary to solve a differential equation online, as in the general case of CPGs, resulting in a very simple implementation. By translating the resulting discrete-event time schedules into piecewise constant phase velocities, our methodology can be directly applied to any phase-controlled legged system. This has been demonstrated in two types of platforms with different morphologies and different number of degrees of freedom per leg. The compact representation of the class of walking gaits presented in this paper simplifies the synthesis of supervisory controllers for legged locomotion and provides guarantees about safe transitions. Furthermore by introducing "time modulation" functions in the continuous time scheduler constant acceleration/deceleration on multi-legged robots is achieved.

Max-plus linear systems for modeling discrete-event ground interactions present in legged locomotion opens a new door of opportunities for further research. We are currently investigating instant gait transitions (without waiting for a cycle to finish), and the modeling of more general gaits.

[6]For turning we use the parameter p to introduce offsets in the reference phases of the legs that either increase or decrease the sweep angles of the right or left leg groups during stance, creating a differential in the ground distance traversed that results in turning.

References

[1] P. Holmes, R. Full, D. Koditschek, and J. Guckenheimer, "The dynamics of legged locomotion: models, analyses, and challenges," *SIAM Review*, vol. 48, no. 2, pp. 207–304, 2006.

[2] M. Vukobratovic and B. Borovac, "Zero-moment point — thirty five years of its life," *International Journal Of Humanoid Robotics*, vol. 01, no. 01, pp. 157–173, 2004.

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

[4] H. Nagashino, Y. Nomura, and Y. Kinouchi, "A neural network model for quadruped gait generation and transitions," in *Neurocomputing*, 2001, pp. 1469–1475.

[5] S. Inagaki, H. Yuasa, and T. Arai, "CPG model for autonomous decentralized multi-legged robot system - generation and transition of oscillation patterns and dynamics of oscillators," in *Robotics and Autonomous Systems*. Nagoya Univ, Sch Engn, Suematsu Lab, Dept Elect Mech Engn,Chikusa Ku, Nagoya, Aichi 4648603, Japan, 2003, pp. 171–179.

[6] S. Inagaki, H. Yuasa, T. Suzuki, and T. Arai, "Wave CPG model for autonomous decentralized multi-legged robot: Gait generation and walking speed control," in *Robotics and Autonomous Systems*. Nagoya Univ, Grad Sch Engn, Dept Mech Sci & Engn, Sub Dept Mechatron,Suematsu Lab,Chikusa Ku, Nagoya, Aichi 4648603, Japan, 2006, pp. 118–126.

[7] X. Zhang, H. Zheng, and L. Chen, "Gait transition for a quadrupedal robot by replacing the gait matrix of a central pattern generator model," *Advanced Robotics*, vol. 20, no. 7, pp. 849–866, 2006.

[8] B. Li, Y. Li, and X. Rong, "Gait generation and transitions of quadruped robot based on Wilson-Cowan weakly neural networks," in *Proceedings of the IEEE International Conference on Robotics and Biomimetics*, Tianjin, China, 2010, pp. 19–24.

[9] S. Aoi, T. Yamashita, A. Ichikawa, and K. Tsuchiya, "Hysteresis in gait transition induced by changing waist joint stiffness of a quadruped robot driven by nonlinear oscillators with phase resetting," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Taipei, Taiwan, 2010, pp. 1915–1920.

[10] S. Daun-Gruhn and T. I. Toth, "An inter-segmental network model and its use in elucidating gait-switches in the stick insect," *Journal of Computational Neuroscience*, vol. 31, no. 1, pp. 43–60, 2011.

[11] C. P. Santos and V. Matos, "Gait transition and modulation in a quadruped robot: A brainstem-like modulation approach," *Robotics and Autonomous Systems*, vol. 59, no. 9, pp. 620–634, 2011.

[12] G. C. Haynes and A. A. Rizzi, "Gaits and gait transitions for legged robots," in *Proceedings of the IEEE Conference on Robotics and Automation*, Orlando, Florida, USA, 2006, pp. 1117–1122.

[13] G. C. Haynes, F. R. Cohen, and D. E. Koditschek, "Gait transitions for quasi-static hexapedal locomotion on level ground," in *Proceedings of the International Symposium of Robotics Research*, Lucerne, Switzerland, 2009, pp. 105–121.

[14] R. A. Cuninghame-Green, *Minimax Algebra*, ser. Lecture Notes in Economics and Mathematical Systems. Springer-Verlag, 1979, vol. 166.

[15] F. Baccelli, G. Cohen, G. Olsder, and J. Quadrat, *Synchronization and Linearity: an Algebra for Discrete Event Systems*. Wiley, 1992.

[16] B. Heidergott, G. Olsder, and J. van der Woude, *Max Plus at Work: Modeling and Analysis of Synchronized Systems*. Kluwer, 2006.

[17] J. G. Braker, "Max-algebra modelling and analysis of time-table dependent transportation networks," in *Proceedings of the First European Control Conference*, Grenoble, France, 1991, pp. 1831–1836.

[18] B. Heidergott and R. de Vries, "Towards a (max,+) control theory for public transportation networks," *Discrete Event Dynamic Systems*, vol. 11, no. 4, pp. 371–398, 2001.

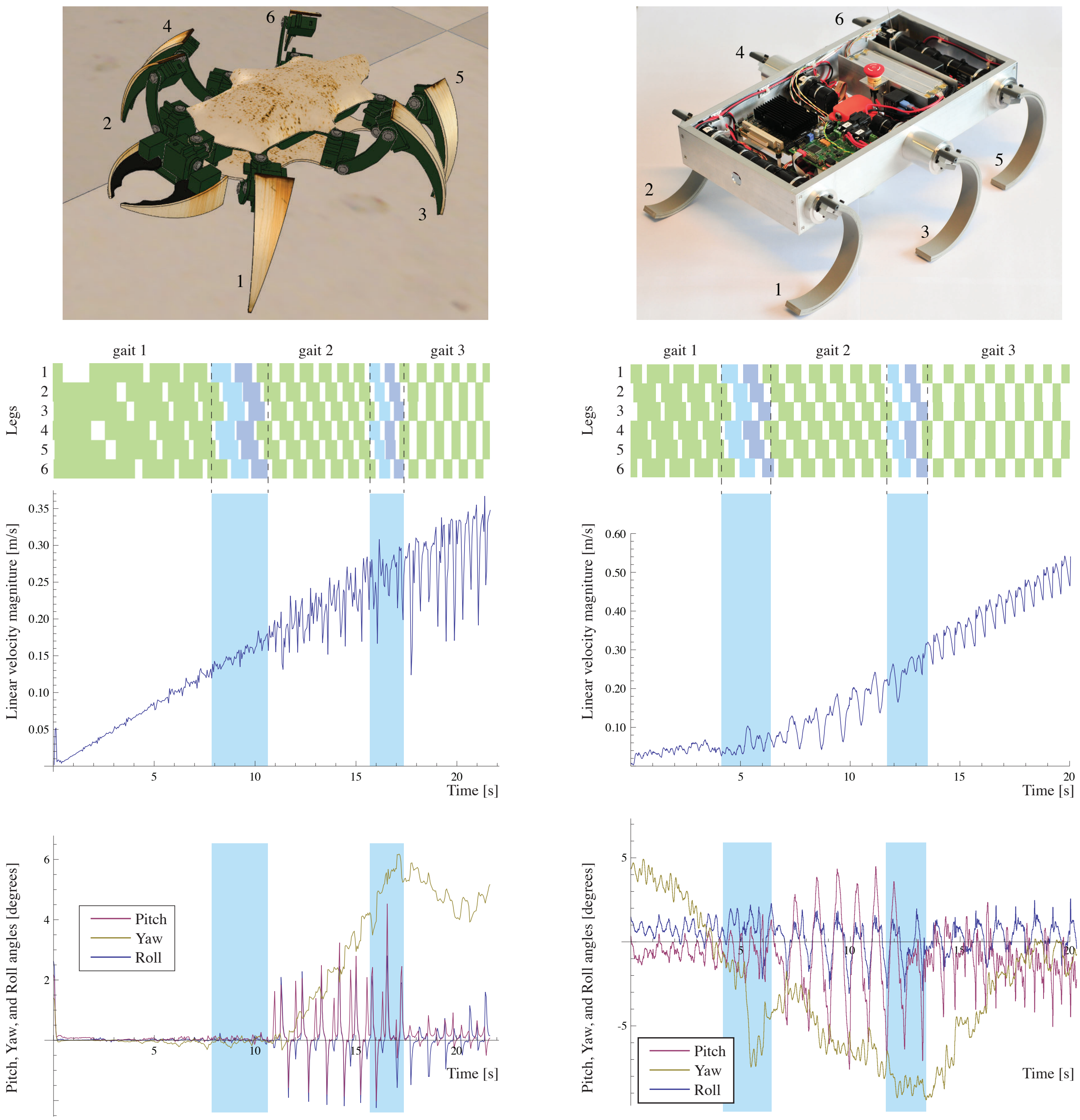

Fig. 10. Simulation and experiment results for a constant acceleration gait switching. Illustrated on the left are the simulations results obtained using the V-Rep physics simulators. On the right the experimental results obtained with the Zebro robot are illustrated.

[19] B. Heidergott, “A characterisation of (max,+)-linear queueing systems,” *Queueing Systems: Theory and Applications*, vol. 35, no. 1–4, pp. 237–262, 2000.
[20] S. Gaubert and J. Mairesse, “Task resource models and (max,+) automata,” in *Idempotency*, J. Gunawardena, Ed. Cambridge University Press, 1998.
[21] B. Bede and H. Nobuhara, “A novel max-plus algebra based wavelet transform and its applications in image processing,” in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, San Antonio, Texas, USA, 2009, pp. 2585 –2588.
[22] R. Full and D. Koditschek, “Templates and anchors: neuromechanical hypotheses of legged locomotion on land,” *Journal of Experimental Biology*, vol. 202, no. 23, pp. 3325–3332, 1999.
[23] S. Grillner, “Control of locomotion in bipeds, tetrapods, and fish,” *Comprehensive Physiology*, vol. Supplement 2: Handbook of Physiology, The Nervous System, Motor Control, pp. 1179–1236, 2011.
[24] E. Klavins and D. Koditschek, “Phase regulation of decentralized cyclic robotic systems,” *International Journal of Robotics Research*, vol. 21, no. 3, pp. 257–275, 2002.
[25] U. Saranli, M. Buehler, and D. E. Koditschek, “RHex: a simple and highly mobile hexapod robot,” *International Journal of Robotics Research*, vol. 20, no. 7, pp. 616–631, 2001.
[26] Capellia Robotics, http://www.v-rep.eu.
[27] J. L. Peterson, *Petri Net Theory and the Modeling of Systems.* Englewood Cliffs, New Jersey: Prentice Hall, Inc., 1981.
[28] B. Giffler, “Mathematical solution of production planning and scheduling problems,” IBM ASDD, Tech. Rep., Oct. 1960.
[29] R. Cuninghame-Green, “Describing industrial processes with interference and approximating their steady-state behaviour,” *Operational Research Quarterly*, vol. 13, no. 1, pp. 95–100, 1962.
[30] G. Cohen, D. Dubois, J. Quadrat, and M. Viot, “A linear-system-theoretic view of discrete-event processes and its use for performance evaluation in manufacturing,” *IEEE Transactions on Automatic Control*, vol. 30, no. 3, pp. 210–220, 1985.
[31] S. Gaubert and M. Plus, “Methods and applications of (max,+) linear algebra,” in *Proceedings of the Symposium on Theoretical Aspects of Computer Science*, Aachen, Germany, 1997, pp. 261–282.
[32] G. Lopes, B. Kersbergen, B. De Schutter, T. van den Boom, and R. Babuška, “Synchronization of a class of cyclic discrete-event systems describing legged locomotion,” *Submitted. Available at http://arxiv.org/abs/1212.5525*, 2011.
[33] G. Lopes, T. van den Boom, B. De Schutter, and R. Babuška, “Modeling and control of legged locomotion via switching max-plus systems,” in *Proceedings of the International Workshop on Discrete Event Systems*, Berlin, Germany, 2010, pp. 392–397.
[34] M. Hildebrand, “Symmetrical gaits of horses,” *Science*, vol. 150, no. 3697, pp. 701–708, 1965.
[35] G. Lopes, B. Kersbergen, T. van den Boom, B. De Schutter, and R. Babuška, “On the eigenstructure of a class of max-plus linear systems,” in *Proceedings of the IEEE Conference on Decision and Control*, Orlando, Florida, 2011, pp. 1823–1828.
[36] G. C. Haynes and A. A. Rizzi, “Gait regulation and feedback on a robotic climbing hexapod,” in *Proceedings of the Robotics: Science and Systems*, Philadelphia, USA, 2006.