Keyframe-based CPG for Stable Gait Design and Online Transitions in Legged Robots

Scott Shaw[1] and Guillaume Sartoretti[2]

Abstract— **This work introduces a CPG model, which allows intuitive gait definition, as well as reactive gait transitions, while ensuring stability and forward progression online and in real-time. Specifically, we propose to rely on keyframes – discrete key leg configurations that can be sequenced into a gait – to define arbitrary legged gaits. We then introduce a new task-space central pattern generator (CPG), which relies on the well-known Kuramoto model for leg phase difference coupling, as well as on a new feedforward term to achieve timed phase coupling between legs for convergence to arbitrary gaits. We then show how this framework can naturally be extended to allow arbitrary gait transitions, by developing two stabilization techniques. First, we reason about the robot's predicted stability to disable specific oscillator updates during transition, while minimizing the resulting effect on forward locomotion. Second, we control the robot's body position within the grounded legs to ensure current and predicted stability, based on inexpensive forward prediction of the CPG model. We validate our approach by presenting simulation and experimental results on a hexapod robot following and transitioning among hexapedal and quadrupedal gaits, in a number of indoors and outdoors locomotion, and mobile manipulation scenarios.**

I. Introduction

Legged animals excel at navigating varying environments and tasks by effortlessly adjusting their gaits, e.g., walk, trot, gallop [1]. In particular, some animals are able to seamlessly transition to gaits which only use a fraction of their limbs, freeing up the others for manipulation tasks or to handle limb injuries [2], [3]. However, current robotics methods to replicate such animal-like behavior remain limited. Existing methods usually rely on computationally expensive optimization [4]. These methods can require up to a minute to compute one stable gait transition, which limits their implementation on robot. This work takes a completely different approach, and proposes a new method by which gaits can be

1) easily designed by developing a *keyframe*-based approach, where a gait is discretized into a set of important leg configurations ("checkpoints"), through which the robot must pass to complete a gait cycle, and
2) transitioned between online and in real-time, by relying on a reactive controller that ensures steady forward progression of the robot during transition, while ensuring stability by updating the body's position and limiting leg usage.

[1]S. Shaw is with Northeastern University, Boston, MA 02115. shaw.sc@northeastern.edu

[2]G. Sartoretti is with the department of Mechanical Engineering, National University of Singapore, 117575 Singapore. mpegas@nus.edu.sg

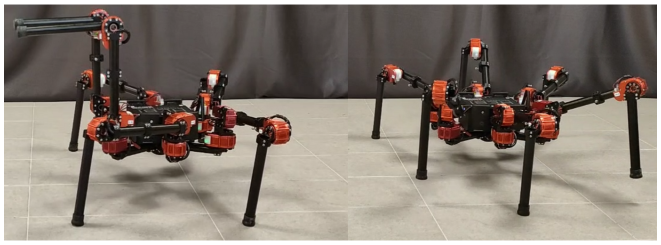
Fig. 1. Hexapod robot transitioning from an alternate tripod gait (right) to a quadrupedal gait that only uses the four hind legs for locomotion (left).

That is, our method generates gait transitions trajectory purely *reactively*, rather than through (expensive) global pre-planning [4], thus increasing flexibility and applicability to real-world scenarios, but at the cost of only producing locally-optimal gait transitions.

Our controller is built on top of a standard task-space central pattern generator (CPG) [5], [6], and ensures stable asymptotic convergence to any gait from any initial configuration for easier implementation on robot. The CPG oscillators generate end-effector trajectories for locomotion in Cartesian space, which can then be converted to joint angles via the robot's kinematics[1]. We extend this design by drawing inspiration from the well-known Kuramoto oscillator [7], [8], to create a novel CPG feedback term that ensures desired phase differences (defining the gait at hand) between legs are met.

Phase coupling is quick and effective [7], but is not sufficient to ensure gait convergence for general gaits, and in particular cannot easily handle gaits with non-constant phase differences during a gait cycle. In order to handle arbitrary gaits, we must account for both leg phases and timings, i.e., we need legs to each meet the correct phase at the correct time during a gait cycle. Based on this observation, we introduce a feedforward component to our Kuramoto-based oscillator model, which guarantees the target phase (and, as a result, phase differences between legs) is reached at the correct timing within the gait. By considering the interactions between oscillators, the Kuramoto feedback term works collectively with the feedforward term, which solely focuses on singular oscillators and their individual progression.

We then extend our keyframes-based CPG to produce stable gait transitions, by limiting leg movements and adapting the body's position. During transition, our approach enables/disables leg movements (i.e., CPG update) at each time step, by reasoning about the projected stability of the

[1]Throughout the paper, we refer to oscillators as legs/end effector positions interchangeably.

robot using a locally optimal, greedy approach. In doing so, we show that the feedback component of the CPG also helps minimize the duration legs remain stationary due to instability, to maximize steady forward progression during gait transition. Additionally, we constantly adapt the body's position within the grasp defined by the grounded legs (i.e., the support polygon) to ensure continued stability, by relying on an inexpensive prediction of our CPG model. In doing so, the feedforward and feedback components of our CPG collaboratively ensure the robot can accurately converge to a gait (feedforward), while maintaining a priority for quick forward movement (feedback). Our resulting approach enables reactive, online, and stable gait transitions at any point between two arbitrary gaits.

Our simulation results help shed light on the interactions between the feedforward and feedback components of our model in helping steady forward progress during gait transitions. We further compare our results to one gait transition reported in [4], showing that our reactive transitions are near-optimal in terms of leg liftoff, and discuss the differences between our approach in terms of forward progression. Finally, we present a number of experimental results on a hexapod robot transitioning between well-known hexapedal gaits on flat and (slightly) unstructed terrains, both indoors and outdoors. We further show results where the robot is tasked to transition to a quadrupedal gait, where two of the legs are unused for locomotion (and could be used for other tasks, e.g., mobile manipulation).

Section II includes relevant background information about CPGs, gaits, gait transitions, and previous work. Section III explains our approach to keyframe-based gait definition and design. Section IV presents our CPG controller in detail. Section V describes how locomotion and gait transitions are stabilized. Section VI presents our experiential results gather from simulation and on robot. Section VII includes a summary of our methods and outlines possible future works.

II. Background

For as long as researchers have been working on robotics, they have been attempting to replicate locomotion in nature by mimicking animal-like motion. In animals, the best characterized spinal network for locomotion is the central pattern generator (CPG). Across species of animals, CPGs are networks of spinal neurons that produce rhythmic patterns, without the need of any rythmic input, for use in a number of fundamental tasks such as walking, flying, and swimming [9], [10]. While biological CPGs are still an active area of research, the robotics community have started formulating mathematical CPG models to develop controllers for articulated robot [10]. In robotics, CPGs are often defined as a set of coupled oscillators with assigned parameters for frequencies, amplitudes, weights, offsets, etc. This model allows coupling of the key robot articulations/joints, and has the ability to be easily adjusted through relevant parameters [11], [12]. CPG's have been used to control quadruped [7], hexapod [4], [6], [13], octopod [13], and snake-like robots [14], [15].

Most animals use a variety of gaits during locomotion depending on the terrain in their environment, their speed, their body, etc. Moreover, these gaits are defined and have been optimized during the long processes of evolution and natural selection [16]. Replicating this level of optimization on a robot is an ongoing research challenge. There are many methods of defining gaits for use in legged robots. A common method is to define a gait relative to the coupling of the CPG. [6], [11] use a coupling matrix directly acting upon each component of the CPG, while [7] relies on a matrix of angular phase differences in order to define gaits. Finally, gaits can also be described by their associated *footfall pattern* of the robot, which defines when each leg is grounded during a gait cycle [4].

Gait transitions are finite motions which allow a robot to switch between gaits [17]. A common method to perform gait transition involves adjusting internal CPG parameters which define the coupling configuration [7], [11]. This method relies on the CPG's natural ability to converge to arbitrary limit cycles, but does not reason about the applicability of said transition on a robot (and, in particular, about stability during transition). Another method would be to overlay two leg trajectory patterns at the same time such that legs are switched from one gait to another one-by-one [17].

Previous work has used optimization based transitions to develop footfall patterns for a known transition between two gaits [4]. They separate the transition into the optimization of transitional footfall patterns, describing which leg is grounded at which time during the transition, before running a second optimization that yields exact joint trajectories and body poses during the transition to meet these footfall patterns and ensure stability. [4] both aims at maximizing forward progression and minimizing ground contact changes over a given, fixed gait transition period. Additionally, this method is always able to find a stable, optimal transition between gaits and intended to create smooth transitional motions, but does so at the cost of computation times, which can reach one minute and are therefore usually performed offline and ahead of robot implementation. As such, these transitions have limited application in situations which require an unpredicted gait transition on short notic, which could result in a loss of stability, e.g., in response to leg failure or unexpected variations in terrain.

III. Keyframe-Based Gait Definition and Design

During a gait cycle we can view each legs in one of two states - *stance* and *swing*. Legs in stance are grounded, while swing legs are lifted off the ground, and the transition between these states occurs whenever legs make/break ground contact. The CPG model described in Section IV) controls the progression of the *phase* of the legs through one gait cycle, often represented as their angular position on a (usually circular) limit cycle (colored dots in Fig. 2). In this work, this limit cycle is directly defined in the Cartesian space, and the phase of legs also represents the position of their end-effector in the world. Legs in stance (with phase in the lower half-plane) are responsible for producing motion by translating the body (animal or robot) in the desired direction of motion. On the other hand, swing legs (with phase in the upper half-plane, so end-effector position above-ground) are responsible for progressing the gait cycle, i.e., they are advancing so they can once again be used to produce motion.

We propose using *keyframes* to easily define arbitrary gaits. We also extend this framework to enable stable, online gait transitions by performing iterative adjustments in Section V. A keyframe-based gait cycle requires progression through a set of timestamped ”checkpoints” (keyframes), which capture discretized, timestamped leg configurations. One keyframe is defined at each point of the gait where legs change ground contacts. At all times, the CPG will advance toward the next *target* keyframe within the defined gait. Progressing through every keyframe once is defined as a *gait cycle* (i.e., one complete period of the oscillators). Specifically, keyframes are represented as N-dimensional vectors of desired leg phases (associated with a vector of timestamps for each keyframe), and are combined together to build a *keyframe matrix* that fully defines a gait. In this work, for ease of presentation, we assume that the keyframe timings are implicitly defined, such that keyframes are uniformly distributed over a gait cycle. The most simple example can be seen in the definition of a simple alternate tripod (K_T) gait for a hexapod robot:

$$K_T = \begin{bmatrix} 0 & \pi & \pi & 0 & 0 & \pi \\ \pi & 0 & 0 & \pi & \pi & 0 \end{bmatrix} \quad (1)$$

where the phase differences between legs in different tripods is π at all times, which can be calculated as the difference between different legs on the same row (keyframe). Ground contacts only change at two times and therefore only two keyframes are required to fully define the gait. Gaits such as a wave gait (K_W) or tetrapod gait (K_{TT}) will require more keyframes to define and will be less intuitive to design due to them having non-constant phase differences between legs:

$$K_W = \begin{bmatrix} 0 & \frac{7\pi}{5} & \frac{6\pi}{5} & \frac{9\pi}{5} & \frac{8\pi}{5} & \pi \\ \pi & \frac{8\pi}{5} & \frac{7\pi}{5} & 0 & \frac{9\pi}{5} & \frac{6\pi}{5} \\ \frac{6\pi}{5} & \frac{9\pi}{5} & \frac{8\pi}{5} & \pi & 0 & \frac{7\pi}{5} \\ \frac{7\pi}{5} & 0 & \frac{9\pi}{5} & \frac{6\pi}{5} & \pi & \frac{8\pi}{5} \\ \frac{8\pi}{5} & \pi & 0 & \frac{7\pi}{5} & \frac{6\pi}{5} & \frac{9\pi}{5} \\ \frac{9\pi}{5} & \frac{6\pi}{5} & \pi & \frac{8\pi}{5} & \frac{7\pi}{5} & 0 \end{bmatrix} \quad (2)$$

$$K_{TT} = \begin{bmatrix} 0 & \pi & \frac{3\pi}{2} & 0 & \pi & \frac{3\pi}{2} \\ \pi & \frac{3\pi}{2} & 0 & \pi & \frac{3\pi}{2} & 0 \\ \frac{3\pi}{2} & 0 & \pi & \frac{3\pi}{2} & 0 & \pi \end{bmatrix}. \quad (3)$$

We can also define gaits which do not use all available legs on the robot by defining certain legs to remain in swing phase throughout the entirety of the gait cycle. We can define a hind-legged quadruped gait (K_H) which uses the four hind legs or more simply, a simple quadruped gait (K_Q) which keeps the two middle legs raised:

$$K_Q = \begin{bmatrix} 0 & \frac{4\pi}{3} & \frac{\pi}{2} & \frac{\pi}{2} & \pi & \frac{5\pi}{3} \\ \pi & \frac{5\pi}{3} & \frac{\pi}{2} & \frac{\pi}{2} & \frac{4\pi}{3} & 0 \\ \frac{4\pi}{3} & 0 & \frac{\pi}{2} & \frac{\pi}{2} & \frac{5\pi}{3} & \pi \\ \frac{5\pi}{3} & \pi & \frac{\pi}{2} & \frac{\pi}{2} & 0 & \frac{4\pi}{3} \end{bmatrix}. \quad (4)$$

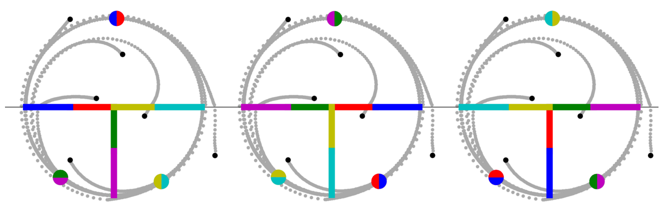
Fig. 2. The three distinct keyframes that define the tetrapod gait, as well as CPG convergence to the tetrapod gait from random initial positions. Oscillators (legs) phases are represented by colored circles (each color is a separate oscillators - circles have more than one color when multiple oscillators share the same phase). Given that we are working with a task-space CPG, the oscillator phase directly represents the end effector position in the world. Keyframes are represented by colored lines (matching the oscillators' colors), which define the desired phase of the oscillators at that instant during the gait cycle. Oscillators which are below the x-axis of the plot are grounded (in stance phase), while those above are in the air (in swing phase). The oscillators rotate around the limit cycle counterclockwise.

IV. Kuramoto-Inspired CPG Model

We introduce a novel CPG based on the Hopf model [18], which utilizes a feedforward component to set individual oscillators' angular velocities and guarantee coupling, as well as a feedback component extended from the Kuramoto model to help maximize convergence speed and forward progression by accelerating legs in swing phase. The Kuramoto model describes the dynamics of a set of N phase oscillators, θ_i with angular velocities $\omega_i(t)$ [12]. The goal of these oscillators is to converge and rotate given a formation (i.e., a set of desired phase differences):

$$\begin{cases} \omega_i(t) = -\overbrace{\sum_{j=1}^{N} \lambda \sin(\theta_j - \theta_i - \Delta\phi_{ij}) \mathbb{1}_{y(t)\geq 0}}^{\text{feedback}} + \overbrace{\frac{\kappa_i - \theta_i}{P}}^{\text{feedforward}} \\ \dot{x}_i(t) = \alpha(\mu - r_i^2)x_i + \omega_i(t) y_i \\ \dot{y}_i(t) = \beta(\mu - r_i^2)y_i - \omega_i(t) x_i, \end{cases} \quad (5)$$

where the first equation is adapted from the Kuramoto model to calculate the angular velocities of our CPG oscillators and reach the desired phase differences defined by the current keyframe. There, the first term, the Kuramoto feedback, is only applied to swing legs and provably ensures asymptotic convergence of the legs to the desired phase differences defined by the current target keyframe. $\omega_i(t)$ is the velocity of the ith oscillator ($1 \leq i \leq N$) at time t, and N is the number of legs/oscillators. λ is a constant, which affects the strength of the adjustment to $\omega_i(t)$ and θ_i is the angular phase of oscillator i. $\Delta\phi_{ij}$ represents the desired phase differences between oscillators i and j (i.e., $\theta_j - \theta_i$, with $\theta_i = \arctan(y_i)(t)/x_i(t)))$ in the target keyframe (see Eqs.(6)-(7) below). $\mathbb{1}_{y(t)\geq 0}$ controls the adjustment such that this term is only applied during swing phase.

The second term is a feedforward component, which calculates the angular velocities required by each oscillator to reach their phase in the target keyframe, κ_i, given the duration, P, until that target keyframe. It is necessary to rely on this feedforward term to ensure convergence, because the Kuramoto feedback term couples phase differences only asymptotically, and thus is indifferent to keyframe timings, which is not sufficient for our task-space CPG. That is, the Kuramoto model alone can achieve convergence to an alternate tripod gait because the phase differences between the legs are constant throughout the duration of the gait, i.e., the timing of the convergence does not matter, only the phase differences of the legs. However, a tetrapod gait (shown in Fig. 2) requires more than phase difference coupling to achieve convergence, since it has non-constant phase differences during a gait cycle. There, the Kuramoto feedback alone would ensure convergence to the correct phase differences, but most likely not at the right timing, which would offset the leg positions and disturb the timings of ground contacts.

The second and third equations in Eq.(5) make up a common two-state Hopf oscillator [11], [18], which gives us the horizontal and vertical components of the velocity in the phase space (here, Cartesian coordinates, similar to [6]). Whereas the task-space CPG in [6] is defined in 3 dimensions, we here ignore the 3rd dimension for simplicity, and let x and y evolve on a plane, parallel to the direction of locomotion, whose origin is at given points in the robot's body frame (i.e., the *grasp map*). However, note that our approach naturally extends to the full 3D, cylindrical CPG defined in [6]. The final joint angles of the robot are obtained from inverse kinematics. α and β are positive constants that control the speed of convergence to the limit cycle [11]. $\sqrt{\mu}$ is the amplitude of oscillations and $r = \sqrt{x^2 + y^2}$.

To integrate our keyframe-based gait definition with our Kuramoto-based CPG, we require θ and $\Delta\phi$ to use in Eq.(5). The θ of the legs is defined as $\arctan(y_i(t)/x_i(t))$. $\Delta\phi$ is a matrix whose elements are calculated by:

$$\Delta\phi_{ij} = \theta_j - \theta_i. \tag{6}$$

Accordingly, the $\Delta\phi$ matrix is constructed as follows:

$$\Delta\phi = \begin{bmatrix} \theta_0 - \theta_0 & \theta_1 - \theta_0 & \dots & \theta_N - \theta_0 \\ \theta_0 - \theta_1 & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ \theta_0 - \theta_N & \dots & \dots & \theta_N - \theta_N \end{bmatrix}. \tag{7}$$

Our CPG model assumes a pre-defined gait cycle duration, which can be manually selected. We use this value along with the current time, t, to determine the current progression of the oscillators within each gait cycle. This information is used to identify when to update the target keyframe to continue the gait progression.

A. Constant Grounded Velocity

So far, our CPG model ensures smooth (mathematical) convergence to any arbitrary gait (albeit without any guarantee of stability yet). Our CPG already prioritizes forward progression by assigning a non-zero velocity to the grounded legs, but the horizontal component of this velocity (i.e., the end-effector velocity on the ground) is currently not constant and follows a sine wave on the ground. To ensure legs can make contact with the ground at arbitrary times relative to each other, as is common to most gaits except for alternate tripod, we need to ensure that grounded foot velocities are constant.

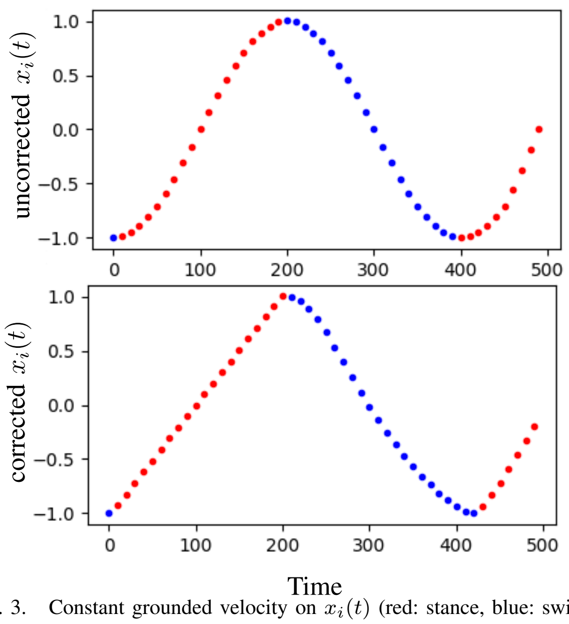

Fig. 3. Constant grounded velocity on $x_i(t)$ (red: stance, blue: swing).

We develop a function, $\psi(t)$, which corrects the angular velocity $\omega(t)$ to result in a constant velocity for all grounded feet. This is achieved by multiplying the desired constant velocity by the inverse of the current speed:

$$\psi(t) = \begin{cases} \frac{\gamma\omega(t)}{r \sin(\theta(t))} & y \leq 0 \\ \omega(t) & y > 0, \end{cases} \tag{8}$$

where γ is a normalization constant ($\gamma = 0.65$ in practice).

Looking at Fig. 3, we can observe the top plot of the uncorrected $x_i(t)$ which is driven directly by $\omega(t)$ whereas the bottom plot shows the $x_i(t)$ which is corrected by integrating our new definition of $\psi(t)$, allowing us to establish constant grounded velocity. It is important to note that our approach for assigning constant velocity to the grounded feet is exclusive to a task-space CPG (but could likely be extended to joint-space CPGs by reasoning about the robot's kinematics).

V. Stability during Gait Transition

This section presents our method of performing quick and versatile gait transitions and propose two solutions which when used together, not only improve the stability of gait transitions, but also the general locomotive stability of gaits. First, we propose to limit the movements of individual legs such that only leg movements which do not hinder the stability of the robot are permitted. Second, to ensure body stability we propose a simple reactive controller to translate the robot's body (with respect to grounded legs) in the plane parallel to the ground by predicting the CPG model.

A. Keyframe-Based Gait Transitions

We have the ability to design complex gaits using keyframes and will now explore how keyframe-defined gaits are used to enable gait transitions. Gait transitions are initiated by changing the current *target* keyframe (from the current gait) to a keyframe on the destination gait. Previous

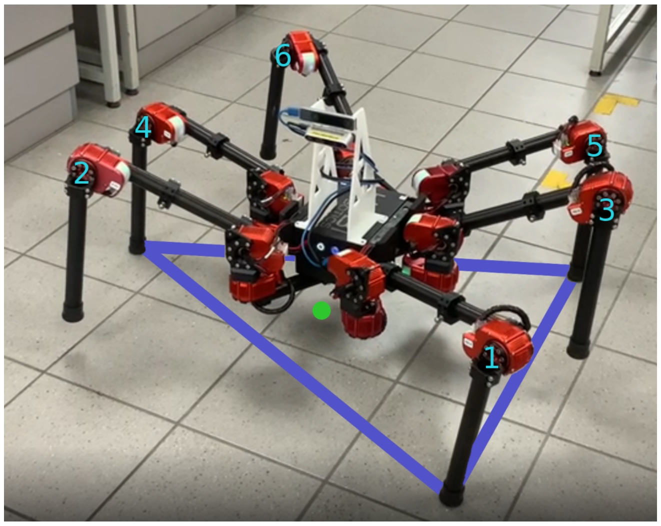

Fig. 4. Center of robot body projected down along gravity (green) inside support polygon (blue) during tripod gait.

works such as [4] transition to an arbitrary keyframe within a gait (e.g., the first keyframe). Although our approach can handle such request, we may generally prefer to reduce the transition distance to maximize both stability and forward locomotion. In this work, we propose to transition to the closest point in joint space (by Euclidean distance) if it is stable, otherwise transition to the closest keyframe in the desired gait. However, transitioning to the closest point minimizes distance travelled, but may not be stable. Thus, we use the keyframes of the destination gait instead in such cases, ensuring we can always transition to a stable keyframe. The duration of a gait transition is user defined and, to achieve the best results possible, a reasonable duration should be defined (we have found $\frac{\text{gait cycle time}}{\text{number of keyframes}}$ to result in a fair estimate in practice).

B. Selectively Enabling Leg Movements

By simply aiming to reach a target keyframe on the destination gait, a transition could cause a grounded leg which is providing stability to be lifted, thus impacting balance and preventing the safe transition. In this work, we focus on ensuring the static stability of the robot, that is, keeping the robot's center of mass projected down by gravity within the *support polygon*, the region defined by the convex hull formed by the grounded contact points (see Fig. 4). We simply score stability using a binary value, which indicates whether (1) or not (0) the robot is statically stable, using the robot's current configuration and assuming a flat, level ground.

Stability scores of the predicted CPG are checked at each time step, by predicting whether the current CPG update (dx,dy) would keep the robot stable - by looking only a single time step forward we are optimizing locally, allowing for reactive CPG updates. Predicting this type of stability longer into the future will be the subject of future works, as this is highly nontrivial in the presence of hybrid contacts with the environment. If the stability score (defined above) indicates the robot is predicted to be stable at the next time step, then all oscillators are updated. Otherwise, we propose to search for and allow the maximum number of oscillators to be updated, while ensuring predicted stability. Specifically, we discretize the updates of legs into binary vectors which can then be individually checked for stability score and used in the CPG update to enable/disable the update of each oscillator, based on their entry in this vector – legs which are represented with a 1 in the vector are updated and those represented with a 0 are not. We create a matrix, M, of 2^N binary vectors of length N, that capture all possible binary vectors:

$$\underbrace{\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}}_{\text{Discretized leg movement arrangements}} . \tag{9}$$

We then search through every possibility in descending order of the number of legs used, at most exhaustively, and stop at the first option that results in a stable update (i.e., we greedily find a stable configuration update with maximal number of legs used, i.e., minimal *leg stoppage*). This yields the binary vector, $\vec{\rho}$, associated with the stable permutation found. $\dot{x}(t)$ and $\dot{y}(t)$ are then element-wise multiplied by this vector to only enable the appropriate leg updates:

$$\begin{cases} \dot{x}(t) = \dot{x}(t) \circ \vec{\rho} \\ \dot{y}(t) = \dot{y}(t) \circ \vec{\rho}. \end{cases} \tag{10}$$

If the only stable update results in no leg movement, we instead force the update of all legs currently in swing phase. The intuition here is to allow these legs to lower to the ground, as this will eventually help regain stability and unlock the situation. Moreover, we note that the inclusion of this fallback strategy intuitively allows us to guarantee the *completeness* of our reactive gait transitions. In practice, we observed that standard gait transitions rarely (nearly never) need to use this fallback strategy.

C. Body Translation

In addition to selectively enabling the robot's leg movements, we also take advantage of planar body translations to further support robot stability during transitions (and locomotion). Body translation may not be necessary for simple gaits and transitions, but is a fundamental part of developing gait transitions to less stable gaits, such as those which do not use all of the robot's legs or gaits which enact drastic changes to the robot's support polygon. We translate the robot body in the grasp frame along the x and y axis (i.e., parallel to the ground). We determine the robot's center of mass using the Pinocchio library [19].

The most straightforward approach would be to translate the robot body reactively (i.e., pure feedback control). However, we experimentally observed that this is not very effective because of the delay between identifying an instability and moving the body accordingly - stability is often already lost by the time the body adjusts and it can then be difficult or impossible to recover then. In this work, we propose

Algorithm 1: CPG Update // O($2^N N$)

```
// Stable Leg Arrangement: O($2^N N$)
for arrangement in M do
    Check stability score for arrangement
    if Stable arrangement then
        Update CPG according to this arrangement
    if No stable arrangement exists then
        Update CPG by lowering legs in swing phase
// Translate Body: O(N)
Calculate intersection of support polygons
Get body translation from PD controller
Translate body by $\Delta x_{body}$ and $\Delta y_{body}$
```

to play the current CPG model forward in time, which can be done very inexpensively, to predict how the body should translate to maintain stability at all times, both during normal locomotion and gait transitions. CPG prediction is ran until the next keyframe, at which point we calculate the intersection of the current support polygon and the one at that keyframe, and translate the body towards the center of mass of the intersection of these two hulls. This choice will ensure that the robot remains stable at all times, because the furthest we can translate the body to is the edge of the current polygon (in the case the polygons have very limited overlap or is a line). Additionally, if the polygons do not overlap, we translate toward the center of mass of the sampled polygon. We then use a simple PD controller to apply our body translation.

In practice, to allow the body enough time to re-position, especially for complex gaits where successive support polygons vary greatly, we introduce pauses in the CPG update at keyframe changes.

D. Complexity

The complexity of our update algorithm with both the exhaustive leg search and forward-sampled body translation is $O(2^N N)$ as showed in Algorithm 1.

Although the worst case complexity looks to be quite poor, the effective complexity is significantly better, as the average number of arrangements checked through the exhaustive search of M is very low in most realistic cases. We gathered data from every transition between tripod, wave, tetrapod, quadruped, and hind-legged gaits over 1500 CPG updates following the start of the transition. This allows for five full gait cycles following the transition period for the CPG to converge. On average 96.6 percent of searches were solved on the initial arrangement, 98.87 percent looked at five or fewer, and 99.87 percent looked at ten or fewer. The worst case run time seen during testing in simulation was 0.04 seconds using both stabilization methods.

VI. Experiments

In this section, we describe the hardware and simulation software used for our experiments. We present our experimental results and highlight the effectiveness of our approach in various gait transitions.

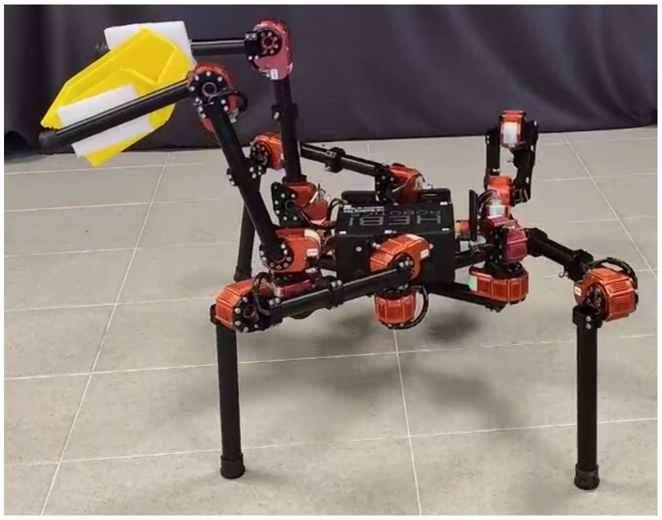

Fig. 5. Hexapod robot walking using a hind-legged quadruped gait to free its front limbs for the manipulation of a container.

A. Experimental Hardware and Setup

In order to test our methods, we conducted experiments in both simulation and on hardware with a HEBI X-Series modular hexapod composed of 18 actuators (HEBI Daisy). Each leg of the robot has three joints, two proximal joints acting as a shoulder (or hip), and a single distal joint acting as an elbow or knee. The robot has a nominal stance of 1.1m x 1.1m and has a mass of 21kg. For simulation experiments, we use the ROS/Gazebo simulator to gather results. We run our tests with our hexapod traversing an empty world (no obstacles or uneven terrain).

B. Simulation Results

To measure the effectiveness of our approach, we report the number of liftoffs and the amount of leg stoppage throughout a number of gait transitions. These data sets were gathered in simulation using some of the common gaits listed in Section III, including tripod (K_T), wave (K_W), tetrapod ($K_T T$), quadrupedal (K_Q), and hind-legged (K_H) gaits. The results gathered in simulation during our experiments can be seen in tables I and II.

The number of foot liftoffs during transitions can be used as an indicator for speed of transition and of the forward progression during transitions. Most approaches try to minimize the number of foot liftoffs during gait transition, but more (useful) liftoffs implies more gait cycles and thus more progression (as shown in some of the results in [4]). Looking at our results in Table I, we can see that the transitions with the highest number of foot liftoffs (three) are those which start from a tripod gait. The other transitions have only a single foot liftoff with the exception of the transition to the hind-legged gait. This is not surprising because of the difficulty of this transition - the hind-legged gait requires a great amount of adjustment to eventually converge. We also note that our approach does not optimize for forward progression by allowing additional gait cycles during transition, but instead tries to minimize the *travel*

Foot Liftoffs During Gait Transitions					
Gait Transitions	→T	→W	→TT	→H	→Q
T	-	3	3	3	3
W	1	-	1	2	1
TT	1	1	-	2	1

TABLE I

Number of foot liftoffs during gait transitions. The left column shows the initial gait (defined in Section III). Every other column shows the ending gait above as well as the number of liftoffs below.

Percentage of Leg Stoppage During Gait Transitions					
Gait Transitions	λ				
	0	**1**	**2**	**5**	**25**
T→W	0.29	0.29	0.29	0.29	0.29
T→TT	0	0	0	0	0
T→H	0.76	0.76	0.76	0.67	0.57
T→Q	1.24	1.24	1.24	1.14	1.05
TT→T	0.57	0.57	0.57	0.57	0.57
TT→W	0	0	0	0	0
TT→H	2.14	2.14	1.86	1.86	1.43
TT→Q	6.86	5.71	4.57	3.14	1.43

TABLE II

Average percentage of leg stoppage during a gait transition due to instability. The transitions performed are listed in the left column. Increasing values of λ are used to show the effect of the feedback component on reducing leg stoppage.

of the legs during the transition, while keeping the robot's progression as steady as possible.

The amount of leg stoppage is important to observe to ensure we are prioritizing forward movement (i.e., we want to stop legs – disable their oscillator update – as infrequently as possible). For our results we specifically look at leg stoppage during gait transitions. In Table II, we show the effect of varying the value of λ (the strength of the feedback term of the CPG) on the average percentage of leg stoppage during gait transitions. During most gait transitions, leg stoppage is infrequent and therefore the effect of the feedback term is limited. More difficult gait transitions result in great amount of leg stoppage, where the effect of lambda can be observed more easily. In particular during the quadruped transition, nearly 7 percent of the total leg updates were stopped during the transition, but the amount of leg stoppage can be greatly reduced with high values of lambda. With a lambda of 25 the percentage of leg stoppage was reduced from 6.86 percent to 1.43 percent.

Our gait transitions rely on a user-defined period to execute. However, our method also naturally supports transition periods that are too short to enable a complete transition to the target keyframe, because of the ability of the feedforward component of the CPG to couple to the target gait even after a partial transition. Table III) reports the true, complete transition duration required by our approach if an unreasonably small duration period is input, showing that our approach can still adapt and provide a reasonable, self-timed transition.

Finally, we compare the result of our reactive gait transition to previous works. Specifically, since code is not available for this prior work, we focus on the specific gait transition detailed in Fig. 2 of [4]. We recreate an identical scenario by designing the same quadrupedal and pentapedal

Unreasonably Short Transition Times and Coupling					
Gait Transitions	→T	→W	→TT	→H	→Q
T	-	0.458	0.354	1.154	0.667
TT	0.529	0.523	-	0.579	0.579

TABLE III

True transition duration (in seconds) until total coupling to the target gait, given an unreasonably short transition time of 0.001s (i.e., one time step,

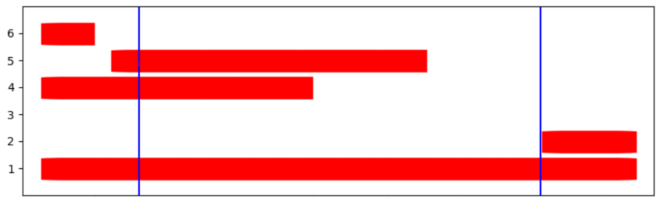

Fig. 6. Footfalls patterns during a gait transition from a quadrupedal gait (left) to a pentapedal gait (right), to be compared with Figure 2 from [4]. The vertical axis are the leg numbers which correspond to Fig. 4. The period of the gait transition is displayed by the vertical blue lines.

gaits, and perform a gait transition between them at the same exact times during the gaits using our methods.

When comparing the figure from [4] to Fig. 6, we first note that our transition period is shorter and has fewer changes in ground contacts. [4] optimizes for forward progression while minimizing ground contacts changes (although the former seems to be weighted more heavily), given a user-defined gait transition period. Although the transition period is longer in [4]'s figure, the robot is progressing more than in Fig. 6. This is because we take a different approach, aimed at prioritizing short transition times and reaching the destination gait quickly (and with minimal leg movement) to resume normal locomotive progression. Despite the reactive nature of our method, we are still able to closely match the performance of [4] in this case, which uses global optimization, demonstrating the high quality of the resulting transition trajectories of our approach.

C. Experimental Results

Our on-robot demos were run both indoors and outdoors and can be seen here: https://bit.ly/CDC2022-KuramotoCPG

We performed on-robot experiments using common hexapedal gaits (alternate tripod, tetrapod, and wave gaits) both indoors on a static surface and outdoors while moving over varied terrains. Transitioning between these gaits indoors in a controlled environment is results is perfectly stable transitions. We further tried to make the transitions between them more challenging by performing them at (manually selected) inopportune times within the gait cycle, but observed complete stability throughout our experiments nonetheless. Although body translation is not necessary to stabilize these gaits, we note that it has the benefit of continually adjusting the body position closer to the center of the support polygon (i.e., the ideal stable position), which would likely make the robot more robust to outside disturbances.

We can take advantage of these gaits' varied stabilities to locomote over concrete, grass, and a small grassy incline, using a different gait each time the terrain becomes increasingly challenging. Alternate tripod is used on the flat concrete surface, tetrapod on the grass, and wave gait on the grassy incline. These gait transitions display the advantages of adapting gaits based on the stability required from the surrounding environment. This demo (which can be seen in our video playlist) is for visualization only, and is presented to showcase the potential usage of gait transitions for real-life tasks that involve varying terrains.

The most challenging gaits to perform and transition to are those which use a subset of the robot's legs. To further stress-test our method, we performed transitions from a tripod to a hind-legged gait, which only uses the hind four legs for locomotion. During these experiments, the body translation our model employs saves the robot from losing stability and falling forward. We additionally observe seldom restriction of leg movements during transition, to prevent further instability. Although we were able to achieve stability during this gait transition, we were limited in the speed of the locomotion due to the body requiring significant time between leg liftoffs to adjust its position ahead of the next leg's movement. We were also able to take advantage of the front limbs being freed during this gait and for mobile object manipulation (see Fig. 5 and in our playlist).

VII. Conclusion

To create stable locomotion or free legs up for applications such as climbing or mobile manipulation, we introduced a new CPG model that can be used to easily define gaits and achieve reactive, online, and stable gait transitions. To these ends, we proposed a CPG composed of a feedforward term and Kuramoto-based feedback term, which cooperatively improve convergence to arbitrary gaits and prioritize forward progression and a keyframe-based gait design and transition mechanism that aims at streamlines the gait design process and supports stable, online gait transitions. Our CPG model relies on selectively enabling leg movements, while prioritizing forward progression, and predictively adjusting the body's position within the grasp defined by the legs.

Future work will investigate gradient-based methods that may allow us to search through the space of potential leg updates at a lesser cost than the current greedy exhaustive search, for more complex scenarios or robots with more legs. Additionally, we have noticed that $\omega(t)$ can ramp up near the end of a keyframe (e.g., if legs were stopped due to instability), and as a countermeasure, we have applied a bound on $\omega(t)$. Another solution may be to preemptively overshoot our target keyframe to avoid spikes in $\omega(t)$ by adding weights to the feedforward component of our CPG. Thirdly, we would also like to rely on more advanced proprioceptive sensors, such as torque sensors or IMU, to further improve general (dynamic) stability. Finally, we are also planning to explicitly investigate the use/extension of our model for cases where legs might break or go limp. There, we expect that more involved transition methods that can safely avoid self-collisions and/or more advanced gaits might be necessary, i.e., gaits with different duty factors per legs for cases where some legs need to pick up the task of another disabled limb, thus needing to contact the ground multiple times per gait cycle.

References

[1] K. Hashimoto, S. Kimura, N. Sakai, S. Hamamoto, A. Koizumi, X. Sun, T. Matsuzawa, T. Teramachi, Y. Yoshida, A. Imai, K. Kumagai, T. Matsubara, K. Yamaguchi, G. Ma, and A. Takanishi, "Warec-1 — a four-limbed robot having high locomotion ability with versatility in locomotion styles," pp. 172–178, 2017.

[2] A. Cully, J. Clune, D. Tarapore, and J.-B. Mouret, "Robots that can adapt like animals," *Nature*, vol. 521, no. 7553, p. 503–507, May 2015. [Online]. Available: http://dx.doi.org/10.1038/nature14422

[3] D. Owaki, H. Aonuma, Y. Sugimoto, and A. Ishiguro, "Leg amputation modifies coordinated activation of the middle leg muscles in the cricket gryllus bimaculatus," *Scientific Reports*, vol. 11, 01 2021.

[4] J. Whitman, S. Su, S. Coros, A. Ansari, and H. Choset, "Generating gaits for simultaneous locomotion and manipulation," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 2723–2729.

[5] S. Shaw, G. Sartoretti, J. Olkin, W. Paivine, and H. Choset, "Workspace cpg with body pose control for stable, directed vision during omnidirectional locomotion," pp. 6316–6322, 2019.

[6] G. Sartoretti, S. Shaw, K. Lam, N. Fan, M. Travers, and H. Choset, "Central pattern generator with inertial feedback for stable locomotion and climbing in unstructured terrain," pp. 1–5, 05 2018.

[7] C. Liu, Y. Chen, J. Zhang, and Q. Chen, "Cpg driven locomotion control of quadruped robot," in *2009 IEEE International Conference on Systems, Man and Cybernetics*, 2009, pp. 2368–2373.

[8] N. Chopra and M. W. Spong, "On exponential synchronization of kuramoto oscillators," *IEEE Transactions on Automatic Control*, vol. 54, no. 2, pp. 353–357, 2009.

[9] P. Guertin, "Central pattern generator for locomotion: Anatomical, physiological, and pathophysiological considerations," *Frontiers in Neurology*, vol. 3, p. 183, 2013. [Online]. Available: https://www.frontiersin.org/article/10.3389/fneur.2012.00183

[10] 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, robotics and Neuroscience. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0893608008000804

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

[12] A. Jadbabaie, N. Motee, and M. Barahona, "On the stability of the kuramoto model of coupled nonlinear oscillators," vol. 5, pp. 4296–4301 vol.5, 2004.

[13] S. Inagaki, H. Yuasa, T. Suzuki, and T. Arai, "Wave cpg model for autonomous decentralized multi-legged robot: Gait generation and walking speed control," *Robotics and Autonomous Systems*, vol. 54, no. 2, pp. 118–126, 2006, intelligent Autonomous Systems. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889005001508

[14] J. Whitman, F. Ruscelli, M. Travers, and H. Choset, "Shape-based compliant control with variable coordination centralization on a snake robot," pp. 5165–5170, 2016.

[15] A. J. Ijspeert and A. Crespi, "Online trajectory generation in an amphibious snake robot using a lamprey-like central pattern generator model," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, 2007, pp. 262–268.

[16] R. M. Alexander, "The gaits of bipedal and quadrupedal animals," *The International Journal of Robotics Research*, vol. 3, no. 2, pp. 49–59, 1984.

[17] G. C. Haynes and A. A. Rizzi, "Gaits and gait transitions for legged robots," pp. 1117–1122, 2006.

[18] X. Li, M. R. E. U. Shougat, S. Kennedy, C. Fendley, R. N. Dean, A. N. Beal, and E. Perkins, "A four-state adaptive hopf oscillator," *PLOS ONE*, vol. 16, no. 3, pp. 1–14, 03 2021. [Online]. Available: https://doi.org/10.1371/journal.pone.0249131

[19] J. Carpentier, F. Valenza, N. Mansard, *et al.*, "Pinocchio: fast forward and inverse dynamics for poly-articulated systems," https://stack-of-tasks.github.io/pinocchio, 2015–2021.