2014 IEEE International Conference on Robotics & Automation (ICRA)
Hong Kong Convention and Exhibition Center
May 31 - June 7, 2014. Hong Kong, China

Towards Automatic Discovery of Agile Gaits for Quadrupedal Robots

Christian Gehring*/†, Stelian Coros†, Marco Hutter*,
Michael Bloesch*, Péter Fankhauser*, Markus A. Hoepflinger* and Roland Siegwart*
*Autonomous Systems Lab, ETH Zurich, †Disney Research Zurich, Switzerland (gehrinch@ethz.ch)

***Abstract*— Developing control methods that allow legged robots to move with skill and agility remains one of the grand challenges in robotics. In order to achieve this ambitious goal, legged robots must possess a wide repertoire of motor skills. A scalable control architecture that can represent a variety of gaits in a unified manner is therefore desirable. Inspired by the motor learning principles observed in nature, we use an optimization approach to automatically discover and fine-tune parameters for agile gaits. The success of our approach is due to the controller parameterization we employ, which is compact yet flexible, therefore lending itself well to learning through repetition. We use our method to implement a flying trot, a bound and a pronking gait for StarlETH, a fully autonomous quadrupedal robot.**

I. INTRODUCTION

Humans and animals move through their environments with grace and agility. A factor key to this ability is the option of employing different types of motions in order to trade-off energy efficiency, the load placed on the musculoskeletal system, risk of injury, maneuverability, tolerance to sources of noise and movement speed. In order for legged robots to match the level of skill of living creatures, it is therefore crucial that we develop flexible control methods that can generate a rich variety of motions.

In this paper we build on a control framework that we recently introduced [1], and we perform experiments on StarlETH [2], an autonomous, medium dog-sized quadrupedal robot that uses electrical motors for actuation. We previously demonstrated two main gaits, a walk and a trot, and showed that transitions between them can also be achieved. Here, we aim to further enrich the repertoire of agile gaits for quadrupedal robots. We extend our control method to allow it to compactly represent a wide variety of gaits in a unified framework, and we apply direct policy search methods to eliminate the need for the tedious process of manually tuning control parameters. For this we use a stochastic optimization approach that is inspired by the the ability of humans and animals to fine-tune motor skills through practice and repetition.

We restrict our attention to quadrupedal locomotion characterized by periodic leg motions. These gaits, such as the walk, trot or gallop, are commonly described by the footfall patterns that result from inter-limb coordination [3], [4]. While the study of quadrupedal gaits and gait transitions is still an active area of research in biology and biomechanics,

This research was supported by the Swiss National Science Foundation through the National Centre of Competence in Research Robotics.

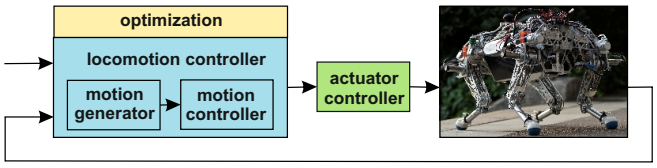

Fig. 1. Control scheme for quadrupedal locomotion

the insights currently available provide a good source of inspiration for quadrupedal robots. In particular, we implement a new gait pattern description that was recently proposed and validated on real-world data collected from different types of animals [5]. This parameterization is compact, can represent symmetric and asymmetric gaits, and lends itself well to learning through repetition. We use a simulation environment in order to efficiently optimize controller parameters for a variety of agile gaits, and we show that the flying trot, bound and pronking gaits work equally well when applied directly to StarlETH, without needing further parameter tuning.

A. Related Work

There are a number of quadrupedal robots that are capable of various dynamic gaits and gait transitions. Raibert's seminal work on controlling a one-legged robot was successfully applied to a hydraulic quadruped, allowing it to implement various gaits, such as trotting, pacing, bounding and pronking [6]. The control parameters for these gaits were manually tuned. Similar principles were also applied to BigDog, allowing it to achieve a robust trotting gait [7]. An asymmetric gait, a transverse-gallop-type bounce, was shown by the electric planar quadruped Scamper [8]. The eight joints of the articulated legs were individually controlled depending on the eight possible states of the gait. Each joint can either rotate freely, or it can track a desired joint angle or joint speed in order to lift the legs, to maintain the trunk posture horizontally, or to accelerate the trunk vertically. This control approach is thus tailored to the implemented gait, and does not easily generalize to other locomotion patterns. A neural oscillator based controller in combination with a reflex mechanism resulted in a trotting, bounding and pronking gait for the planar quadruped Patrush [9]. The parameters in each neural oscillator and its network were manually determined in simulation. The Tekken robot [10], as well as the Cheetah-Cub [11] were also able to demonstrate several dynamic gaits based on central pattern generators.

Scout II, a quadrupedal robot with four actuated hip joints and telescopic legs with passive springs [12], was able to bound stably by simply switching torque values at the hip during support or flight phases in an open loop fashion [13].

Another bounding controller based on Raibert's approach used three control actions depending on the leg states: stance-retraction, stance-brake and flight, detected by measuring joint angles and spring lengths. This resulted in a stable running gait even without having any coupling between the leg pairs [14]. Scout II was also the first quadrupedal robot that was able to perform a rotary gallop fully autonomously [15]. The gallop was initiated from the bounding gait by changing the touchdown angles of the legs and was controlled in a similar fashion, but with coupling between the legs. The pronking and galloping of the planar quadruped KOLT [16] was controlled by a closed loop method that regulated the thrust through the control of the hip liftoff speed. A second closed loop approach regulated the energy added in each hop based on a model of the actuator system. A third approach based on fuzzy control was capable of learning the leg touchdown angles and leg thrusts required to track the desired running height and velocity of a quadruped in only one stride [17]. A recently introduced robust walking trot controller which combines an analytic trajectory generator with active compliance control was implemented on the more complex and powerful quadruped HyQ [18].

The shortcomings of these control approaches are that they are ultimately tailored to only one or two gaits and mostly require manually tuning. In computer graphics, parameter optimization for motion synthesis is well known and has been applied recently to find the parameters of a muscle-based control method for simulated bipeds [19]. As we show in this paper, our control framework and its parameterization, coupled with a similar automated parameter optimization step, are able to produce a wider array of gaits in a unified manner. Furthermore, we demonstrate that our approach is directly applicable to real robots.

B. Contribution

We employ a cascading control architecture, as depicted in Fig. 1, to decouple the problem of locomotion control from the specific details of the hardware platform that it is applied to. A model-based approach is used to control the motion of the legged multi-body systems. We make use of domain-specific knowledge, such as pre-defined foot-fall patterns and templates, in order to address the problem of motion generation and control, and we apply optimization techniques to find optimal values for the various parameters of our locomotion controllers. We build on the control scheme we introduced in [1], where we demonstrated robust walking and trotting in the presence of external perturbations. In order to achieve this, many control parameters had to be manually tuned. We found this to be a relatively straightforward process for the static walk, as it is a slower gait that maximizes stability, and for the trot, where the center of mass typically remains above the line of support. In contrast, manually finding optimal parameters for more dynamic and agile gaits, becomes tedious if not impossible. We address this challenge by applying direct policy search methods to automatically fine-tune controller parameters.

The remainder of this paper is structured as follows:

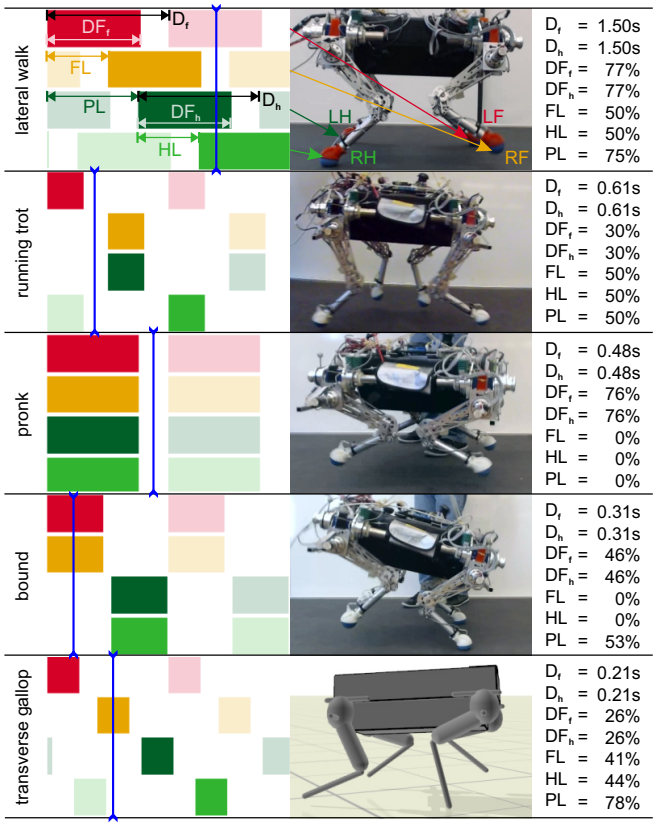

Fig. 2. Gait diagrams of various gaits using the APS parameterization are shown together with snapshots of StarlETH and a simulated version of it. Note that the cycle durations are scaled.

Section II summarizes the control strategy we use, and introduces a new approach to compactly describe the gait patterns in order to facilitate the process of automatically discovering agile gaits. Our optimization technique is explained in detail in Sect. III. Experiments and their results are summarized in Sect. IV. The paper concludes with a discussion in Sect. V.

II. CONTROLLER AND PARAMETERIZATION

Our control framework is functionally divided into a motion generator and a motion controller, both of which present a set of parameters that may need to be individually adapted for each gait.

A. Motion Generation

Animal gaits are characterized by their footfall patterns, which encode, for each leg, the relative timing of the swing and stance phases within a stride. Abourachid et al. [20] proposed an interesting approach to analyzing quadrupedal gaits by considering the morphological and functional similarity between the back and hind limb pairs, the sequence of activations in the spinal network, and the succession of limbs used to sense the environment. The proposed anteroposterior sequence of movement (APS) theory results in a gait parameterization with a reduced number of variables that can represent both symmetrical and asymmetrical gaits.

Figure 2 illustrates several gait patterns (left), together with snapshots of the robot (right) corresponding to specific moments in time, as indicated by the blue vertical lines in

the gait diagrams. The colored horizontal bars indicate the stance phases of the left fore (LF), right fore (RF), left hind (LH), and right hind (RH) leg from top to bottom. The cycle duration D_f corresponds to the period between two consecutive footfalls of the left fore (LF) foot. The duty factor of the fore feet DF_f defines the period of the stance phases of both fore limbs. The fore lag FL determines the timing of the touch-down of the second fore foot (RF) relative to the touch-down of the first one (LF), expressed as a percentage of the cycle duration. The start of the stance phase of the ipsilateral hind foot (LH) is given by the pair lag PL, also expressed as a percentage of the cycle duration. To define the touch-down of the second hind foot (RH), the hind lag HL is introduced, describing the timing with respect to the start of the stance phase of the first hind limb (LH). Finally, the duty factor of the hind legs DF_h completes the parameterization. Steady-state locomotion is sufficiently described by the variables introduced so far, but a recent analysis of gait transitions in dogs [5] revealed the need for an independent cycle duration parameter, D_h, for the hind limbs. The hind lag HL and hind duty factor DF_h are then specified as percentages of it. We note that for steady-state gaits the cycle duration for the hind limbs is constrained to be equal to that of the fore limbs.

The benefit of the gait parameterization we use is twofold. First, the type of gait we want to implement can be specified intuitively: by definition, all symmetrical gaits need a fore lag and hind lag equal to 50%; a lateral walk can be simply distinguished from the diagonal walk based on whether the pair lag is larger or smaller than 50%, while the trot and pace are characterized by a pair lag equal to 50% and 100%, respectively; the pronk is characterized by having all time lags equal to zero, and, if the pair lag is additionally varied, a bound emerges; the transverse and rotary gallops are characterized by a fore and hind lag both smaller than 50%. Second, the compact representation of the gait pattern simplifies the optimization problem we need to solve in order to automatically fine-tune gaits, and it is easy to further restrict the search space to a set of desirable gaits by introducing meaningful parameter boundaries, as discussed above.

The swing and stance phases of the legs are used to determine when the role of each leg is to move towards the next foot-hold, and when it should be used to support the body weight. In order to compute appropriate foot-holds for the swing legs, we use an inverted pendulum-based foot placement model that includes a parametrized foot height trajectory. The vertical position and pitch of the main body are given by two parametrized hip height trajectories, fore and hind. We use radial basis function interpolation to represent these height trajectories, with additional constraints that they enforce periodic motions (both at the position and velocity levels) for the hips, or that they have zero velocity at the start and end, for the swing foot.

The position of the center of mass (CoM) is particularly important during locomotion, and it should be actively controlled in order to remain above the support polygon. We represent the target position for the COM as a time-varying, weighted average of the feet positions, and optimize the three parameters that control its motion (as detailed in [1]) in order to ensure that it is smooth. The output of the Motion Trajectory module consists of desired foot positions for each foot, as well as the desired orientation, position, velocity and angular velocity of the main body.

B. Motion Control

The motion control formulation we use is based on the virtual model approach, coupled with a force distribution step that is formulated as a convex optimization problem with inequality constraints [1]. The virtual model controller regulates the position and orientation of the main body using a set of proportional and derivative gains, which generally depend on the gait that is used. For instance, due to the relatively long flight phase, the pitch motion trajectory is very important for the bound and needs to be carefully modulated, while for the trot, it is not as crucial. We therefore treat these gains as parameters that we can optimize over. The force distribution step is used to compute appropriate ground reaction forces for each stance leg, such that the resulting net force and torque acting on the main body have desirable values. To ensure that the resulting ground reaction forces are valid, their normal components cannot be negative (no pulling on the ground), and their tangential components are bounded as a function of the friction coefficient in order to ensure that the feet will not slip. More formally, we compute the desired ground reaction forces by solving the following quadratic program:

$$\text{minimize} \quad (A\boldsymbol{x}-\boldsymbol{b})^{\mathrm{T}} S (A\boldsymbol{x}-\boldsymbol{b}) + \boldsymbol{x}^{\mathrm{T}} W \boldsymbol{x} \quad \text{s. t.} \quad (1)$$

$$F_i^n \geq F_{\min}^n, \quad |F_i^t| \leq \mu F_i^n \quad \tau_{\min} \leq J^T \boldsymbol{F}_i \leq \tau_{\max} \quad (2)$$

where $\boldsymbol{x} = [\boldsymbol{F}_0 \dots \boldsymbol{F}_i \dots \boldsymbol{F}_n]^T$ represents the ground reaction forces that should be applied through the stance legs, and A and $\boldsymbol{b}$ encode the desired virtual forces as soft constraints as described in [1]. The first two sets of inequality constraints ensure the validity of the resulting ground reaction forces, and, for this work, we additionally append joint torque limits to the constraints by means of the contact Jacobian J. This new constraint further improves the performance of our controller when applied to the real robot, as it explicitly takes into account some of its actuator limitations. The different components of the net force and torque acting on the main body can be weighted differently by the diagonal matrix S to specify their priority, and the regularizing matrix W ensures that the effort needed to achieve the control objectives is also taken into account. The elements in the S and W matrices are also optimized independently for each gait.

III. OPTIMIZATION

Inspired by the motor learning principles observed in nature, we implemented a direct policy search method that fine-tunes controllers by repeating the same task with slight parameter variations. The success of our method is due to the compact, yet flexible parameter space defined by the locomotion controllers. In addition, our control strategy is

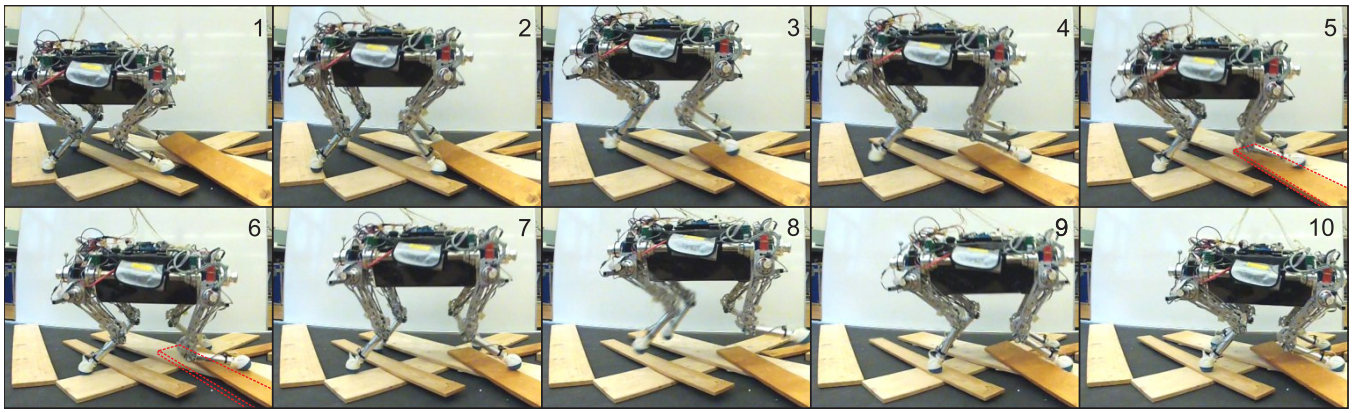

Fig. 3. StarlETH performs a pronk fully power and computation autonomously while dealing with unperceived obstacles. The start (1,6), middle (3,8) and end (5,10) of two strides are shown in the sequence. The controller can robustly cope with a slipping plank emphasized by the red dotted curves in picture 5 and 6. Note that the yellow strings attached to the body only served for safety reasons and did not support the robot.

designed to robustly handle external perturbations, various sources of sensor or actuator noise and modeling errors. This has two major implications. First, the search space we define is relatively easy to explore, as many different parameter configurations result in successful roll-outs (i.e. robot does not fall), and can thus be assigned a meaningful score in order to guide the optimization process towards optimal parameter values. Second, the robustness to modeling errors allows us to perform the parameter optimization in simulation, and then apply exactly the same controllers on the real robot. As we typically require thousands of roll-outs, many of which fail, this is much faster and safer than performing optimizations directly on the physical robot. Additionally, running the optimization in simulation allows us to improve the convergence speed, as we can incrementally increase the difficulty of the problem we solve. We generally start by running simulations where we assume ideal sensor and actuator models, and, as we find solutions that work under these settings, we progressively increase the sensor noise, and decrease the joint position and torque limits in order to match the constraints of the robot.

A. Objectives

As our goal is to reduce the need for manual parameter tuning, we want to define a simple and intuitive objective function which should be minimized. One possible approach is to use kinematic data captured from animals in order to measure the quality of the motions, as was done for simulated quadrupeds [21]. However, as the morphology of animals is typically significantly different than that of robots, this may not always be a feasible option. We therefore formulate the optimization problem in a generic form:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum w_i \cdot C_i(\boldsymbol{\theta}, \boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\tau}), \tag{3}$$

where we seek to find an optimal parameter set $\boldsymbol{\theta}^*$ by minimizing a weighted sum of cost terms C_i, which depend on the state trajectory of the robot $(\boldsymbol{q}, \dot{\boldsymbol{q}})$ and the actuation signals $\boldsymbol{\tau} \in \mathbb{R}^{N_t}$. A variety of cost terms can be defined (see [22] for a comprehensive survey). For instance, to promote energetically efficient gaits, the collision losses, positive mechanical work, or the square of joint torques [23] can be minimized. To ensure that the resulting motions are smooth, we additionally minimize the change of joint torques from one control step to the next as $\sum_{i=1}^{N_t} \int_0^T \dot{\tau}_i(t)\,\mathrm{d}t$. In order to ensure that the trivial solution, standing, is avoided, we also penalize the error between the measured and desired foot apex heights.

We keep the influence of the energy term low, but we incrementally amplify the hardware constraints instead. By reducing $\sum_{i=1}^{N_t} \int_0^T (\varphi_{\max} - \max\{\dot{\varphi}_i(t), \varphi_{\max}\})^2\,\mathrm{d}t$, we penalize the joint velocities as soon as they surpass 80% of the motor limits, which is $8\,\mathrm{rad/s}$. We similarly penalize the mechanical peak power over $150\,\mathrm{W}$. If the maximum allowed torques get exceeded, we clamp them before applying them to the simulated robot model. Joint position limits are taken into account in a similar manner, resulting in a simple, but helpful criterion for penalizing falls and other failure modes. Other, more sophisticated measures of stability of a gait could also be used, although some of them may not be well defined if the system becomes unstable, e.g. the zero moment point [24], they may not be well-suited for dynamic gaits, or they may be prohibitively expensive to evaluate.

Although our goal is to use cost functions that are as simple and generic as possible, we can add new, gait-specific cost terms in order to shape the resulting controllers. For instance, the motion of a pronking gait could be optimized to control the height of the hops by adding $\frac{1}{N}\sum_{i=1}^{N}(h^* - \max(h(t)))^2$ to the optimization objective. This term compares the target height h^* with the apex height of the main body for each of the N strides.

B. Implementation

We apply a direct policy search method based on the Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES) [25], a state-of-the-art stochastic optimization technique. This method can effectively deal with discontinuities in the objective function, noise and local optima in the search landscape. We use the freely available implementation of Hansen [25] and select the number of sampled solutions for each generation $\lambda = 4 + 3\ln(N)$ to be dependent on the number of parameters N as suggested. We handle upper and

lower bounds in the parameter space using a combination of rejection sampling and a penalty term that is added to the cost function. We run the optimization method in simulation in order to avoid the tedious robot initialization procedure and to minimize the risk of damaging the hardware. Each roll-out involves a multi-body dynamics simulation, which we run using the Open Dynamics Engine [26]. The simulations are used to execute several strides using each parameter setting, and they take several seconds of computation time.

IV. RESULTS

Before running the optimized controllers on StarlETH, each gait was first examined in simulation to determine its ability to robustly handle external perturbations and sensor noise. Figure 2 shows several snapshots of our experiments, along with the gait parameters that were found by the optimization process. Our results, however, are best seen in the accompanying video[1].

A. Running Trot and Pronk

We previously presented a running trot with $DF = 40\%$, whose parameters were tuned by hand. It was very difficult to manually set the controller parameters and consequently, the resulting flight phase was barely visible [1]. As a first optimization task we therefore aimed to increase the duration of the flight phase. To restrict the search space to trotting gaits with a flight phase, we fixed the time lags ($FL = HL = 0\%$, $PL = 50\%$) and duty factors ($DF_f = DF_h = 30\%$), but let the optimizer find the optimal cycle durations ($D_f = D_h$) in addition to the remaining controller parameters. We added cost terms rewarding, for each stride, the maximum height of the body's trajectory, the maximum height of the foot above ground, longer stride durations and the ability to match a desired forward speed that was progressively increased from 0 to $0.25\,\mathrm{m/s}$ within 10 strides. A total of 18 parameters were optimized altogether, and the flight phase was increased by 76% as compared to the initial set of parameters we provided. The cost function depicted in Fig. 4 shows a rugged cost landscape and illustrates the difficulty of finding good solutions manually. The parameter optimization step took about two hours to generate 1000 roll-outs on a single core of a modern PC.

The second objective was to automatically discover a robust pronking gait. We set all time lags to 0% and optimized the duty factors, the cycle duration, and the remaining controller parameters. The optimized pronk was successfully executed by StarlETH on variable terrain, as illustrated by the sequence of images in Fig. 3. For this experiment, StarlETH was fully autonomous, as all power and computation was onboard. As shown also in the accompanying video, the gaits produced with our framework are very robust to unperceived obstacles. This is due to the underlying locomotion controller that adjusts the location of foot steps and optimally modulates the ground reaction forces applied by the stance legs, as described in detail in Gehring et al. [1]. The optimization framework we propose fine-tunes the parameters that control these two main behaviors, enusring that they allow the robot to succesfully recover from perturbations.

[1] http://youtu.be/Tj1wreifYhU

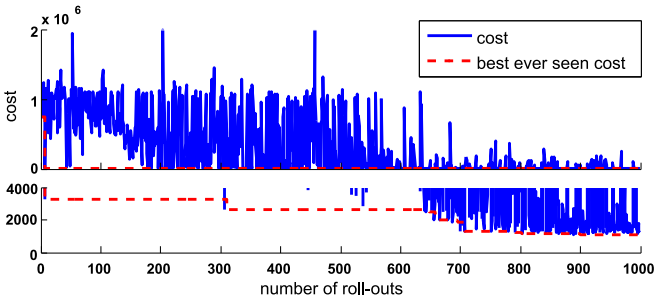

Fig. 4. The evolution of the cost of the roll-outs of a running trot displayed by the blue solid line indicates a rugged cost landscape. The lower graphic shows a zoomed view of the upper to better visualize the evolution of the best ever seen cost (red dashed) that was updated when the CMA-ES updated.

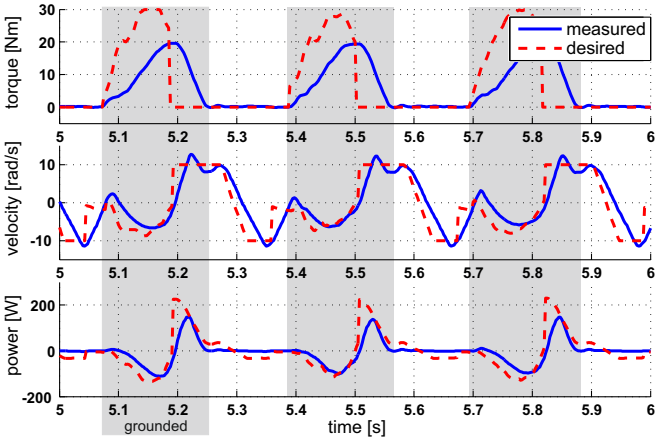

Fig. 5. Desired and measured torque, motor velocity and corresponding power of the left fore knee of a bounding gait are shown.

B. Bound and Transverse Gallop

Our third objective was to obtain a bounding gait, which has zero fore and hind lags by definition. We optimized for the cycle durations ($D_f = D_h$), the pair lag (PL) and the duty factors ($DF_f = DF_h$). Since bounding gaits are characterized by a significant pitch motion, the height trajectories of the fore and hind hips were parametrized with 5 evenly spaced control knots. The remaining optimization parameters are the same as for the pronk. The bound was successfully executed by StarlETH, allowing it to move freely in space and to even turn in place. The desired and measured torques, motor velocities and the corresponding mechanical power in the left fore knee are shown in Fig. 5. Note that due to the special setup of the knee joint of StarlETH, the torque can only be measured if the unilateral spring is loaded, i.e. if the leg is grounded, and during swing phases position control is used instead. The results indicate that the optimized gait is very close to exceeding the limitations of the hardware platform.

We further optimized the parameters for a transverse gallop. The benefit of the APS parameterization becomes apparent once again, as we only have to define the boundaries of the fore and hind lags ($< 50\%$) and the duty factors

($< 50\%$), thus still providing sufficient flexibility for the optimization process. A solution that satisfies the joint position limits was successfully found. However, the high torque and power supply demands prevented us from using this gait directly on StarlETH. Another optimized gait that worked in simulation but could not be directly applied to the robot was a pacing gait. In this case, StarlETH's reduced range of motion (due to the cabling setup), prevented the discovery of solutions that did not violate the joint limits of the physical robot.

V. CONCLUSION

Our approach of combining a robust, model-based controller with automatic tuning of control parameters resulted in the discovery of new agile gaits for the quadruped robot StarlETH. The optimization process was able to significantly improve the running trot, which we found very difficult to tune manually. Moreover, StarlETH was able to employ two new agile gaits, fully autonomously, untethered and in 3D space: a pronking and a bounding gait. Our direct policy search technique was also used to find parameters for a transverse gallop and a pace, both of which worked well in simulation. However, due to hardware limitations, they could not be executed on the robot. In order to provide a compact controller parameterization, we incorporated a new gait pattern description that is inspired by recent findings in biology. This description allows us to intuitively reduce the search space to a desired subset of gaits, while still giving the optimization routine sufficient freedom to discover agile and robust modes of locomotion.

Although we run the optimization process in simulation only, the controllers that satisfy the hardware limitations can be directly executed on the real robot. In the future however, we plan to further fine-tune them on-line in order to appropriately account for, and exploit, the dynamics of the real robot. We also plan to investigate a principled approach to controlling transitions between different gaits. The gait pattern description that we employ provides a promising step in this direction, as it was shown to be flexible enough to explain gait transitions in quadrupedal animals. Last, we believe that the approach we described here is flexible enough to concurrently optimize the morphology of the robot, in addition to appropriate control parameters, in order to provide design-guidelines for the next generation of legged robots.

REFERENCES

[1] C. Gehring, S. Coros, M. Hutter, H. M. A. Bloesch, Michael, and R. Siegwart, "Control of Dynamic Gaits for a Quadrupedal Robot," in *IEEE Int. Conf. on Robotics and Automation*, 2013.

[2] M. Hutter, C. Gehring, M. Bloesch, M. Hoepflinger, C. D. Remy, and R. Siegwart, "StarlETH: A compliant quadrupedal robot for fast, efficient, and versatile locomotion," in *Proc. of the Int. Conf. on Climbing and Walking Robots*, 2012.

[3] E. Muybridge, *Animals in motion*. London: Chapman & Hall, 1899.

[4] M. Hildebrand, "The Quadrupedal Gaits of Vertebrates," *BioScience*, vol. 39, no. 11, pp. 766–775, Dec. 1989.

[5] L. Maes and A. Abourachid, "Gait transitions and modular organization of mammal locomotion." *The Journal of experimental biology*, vol. 216, no. Pt 12, pp. 2257–65, Jun. 2013.

[6] M. H. Raibert, "Trotting, pacing and bounding by a quadruped robot." *Journal of biomechanics*, vol. 23 Suppl 1, pp. 79–98, Jan. 1990.

[7] M. Raibert, K. Blankespoor, G. Nelson, and R. Playter, "BigDog, the Rough-Terrain Quadruped Robot," in *Proceedings of the 17th IFAC World Congress*, M. J. Chung, Ed., vol. 17, 2008.

[8] J. Furusho, A. Sano, M. Sakaguchi, and E. Koizumi, "Realization of bounce gait in a quadruped robot with articular-joint-type legs," in *Proc. of IEEE Int. Conference on Robotics and Automation*, vol. 1, 1995.

[9] H. Kimura, S. Akiyama, and K. Sakurama, "Realization of dynamic walking and running of the quadruped using neural oscillator," *Autonomous Robots*, vol. 258, pp. 247–258, 1999.

[10] Y. Fukuoka, H. Kimura, and a. H. Cohen, "Adaptive Dynamic Walking of a Quadruped Robot on Irregular Terrain Based on Biological Concepts," *The Int. Journal of Robotics Research*, vol. 22, no. 3-4, pp. 187–202, Mar. 2003.

[11] A. Sprowitz, A. Tuleu, M. Vespignani, M. Ajallooeian, E. Badri, and A. J. Ijspeert, "Towards dynamic trot gait locomotion: Design, control, and experiments with Cheetah-cub, a compliant quadruped robot," *The Int. Journal of Robotics Research*, vol. 32, no. 8, pp. 932–950, 2013.

[12] I. Poulakakis, J. A. Smith, and M. Buehler, "Modeling and Experiments of Untethered Quadrupedal Running with a Bounding Gait: The Scout II Robot," *The International Journal of Robotics Research*, vol. 24, no. 4, pp. 239–256, Apr. 2005.

[13] D. Papadopoulos and M. Buehler, "Stable running in a quadruped robot with compliant legs," in *Proc. of IEEE Int. Conference on Robotics and Automation*, vol. 1, 2000.

[14] I. Poulakakis, J. Smith, and M. Buchler, "Experimentally validated bounding models for the Scout II quadrupedal robot," *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp. 2595–2600 Vol.3, 2004.

[15] J. Smith and I. Poulakakis, "Rotary gallop in the untethered quadrupedal robot scout II," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 3, pp. 2556–2561, 2004.

[16] J. Estremera and K. J. Waldron, "Thrust Control, Stabilization and Energetics of a Quadruped Running Robot," *The International Journal of Robotics Research*, vol. 27, no. 10, pp. 1135–1151, Oct. 2008.

[17] D. Marhefka, D. Orin, J. Schmiedeler, and K. Waldron, "Intelligent control of quadruped gallops," *IEEE/ASME Transactions on Mechatronics*, vol. 8, no. 4, pp. 446–456, Dec. 2003.

[18] B. Ugurlu, I. Havoutis, C. Semini, and D. G. Caldwell, "Dynamic Trot-Walking with the Hydraulic Quadruped Robot - HyQ: Analytical Trajectory Generation and Active Compliance Control," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013.

[19] T. Geijtenbeek, M. van de Panne, and A. F. van der Stappen, "Flexible muscle-based locomotion for bipedal creatures," *ACM Transactions on Graphics*, vol. 32, no. 6, 2013.

[20] A. Abourachid, "A new way of analysing symmetrical and asymmetrical gaits inquadrupeds," *Comptes Rendus Biologies*, vol. 326, no. 7, pp. 625–630, Jul. 2003.

[21] S. Coros, A. Karpathy, B. Jones, L. Reveret, and M. van de Panne, "Locomotion skills for simulated quadrupeds," *ACM SIGGRAPH*, vol. 1, no. 212, p. 1, 2011.

[22] D. Gong, J. Yan, and G. Zuo, "A Review of Gait Optimization Based on Evolutionary Computation," *Applied Computational Intelligence and Soft Computing*, pp. 1–12, 2010.

[23] C. Remy, "Optimal exploitation of natural dynamics in legged locomotion," Diss., ETH Zurich, 2011.

[24] M. Vukobratovic and D. Juricic, "Contribution to the synthesis of biped gait," *IEEE Transact. on Biomedical Engineering*, no. 1, pp. 2–7, 1969.

[25] N. Hansen, "The CMA Evolution Strategy: A Comparing Review," in *Towards a New Evolutionary Computation*, ser. Studies in Fuzziness and Soft Computing, J. Lozano, P. Larraaga, I. Inza, and E. Bengoetxea, Eds. Springer Berlin Heidelberg, 2006, vol. 192, pp. 75–102.

[26] E. Drumwright, J. Hsu, N. Koenig, and D. Shell, "Extending open dynamics engine for robotics simulation," in *Simulation, Modeling, and Programming for Autonomous Robots*, ser. Lecture Notes in Computer Science, Ando, Noriaki et al., Ed. Springer Berlin Heidelberg, 2010, vol. 6472, pp. 38–50.