2017 IEEE International Conference on Robotics and Automation (ICRA)
Singapore, May 29 - June 3, 2017

Quadrupedal Locomotion using Trajectory Optimization and Hierarchical Whole Body Control*

Christian Gehring[1], C. Dario Bellicoso[1], Péter Fankhauser[1], Stelian Coros[2], Marco Hutter[1]

Abstract— **Quadrupedal locomotion can be described as a constrained optimization problem that is very hard to solve due to the high dimensional, nonlinear and non-smooth system dynamics. In this paper, we propose a formulation that can be solved within few seconds using sequential quadratic programming. This method considers only a simplified model that just sufficiently represents the system dynamics. The output is a very coarse plan, which can be accurately and robustly followed on a real system using hierarchical whole-body control combined with inverted pendulum-based reactive stepping. Using the fully torque controllable quadrupedal robot ANYmal, we present successful experiments for walking, trotting, and gait transitions even under substantial external disturbances.**

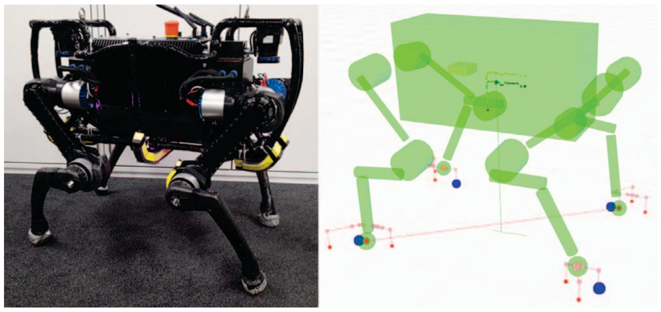


Fig. 1: Our testbed ANYmal, a fully torque-controllable quadrupedal robot, and its planned motion for two trot cycles.

I. Introduction

Quadrupedal robots have great potential as mobile machines to support us in difficult terrain with steps and gaps which are unnegotiable obstacles for most of the state-of-the art mobile robots. Progress in research and industry enabled to build powerful quadrupedal robots in recent years [1], [2], [3]. However, the capabilities of the existing machines are still very limited compared to the skills of their biological counterparts. A main reason is the rather low performance of the control systems. It remains a challenge to design control algorithms for legged locomotion, as this typically involves to define and solve a high-dimensional, nonlinear, and non-smooth optimization problem.

In this work, we tackle the problem of motion generation and control for fully torque-controllable systems, such as the quadruped robot ANYmal shown in Fig. 1. Our goal is to generate different gaits and gait transitions for the quadruped to enable robust walking.

To control legged locomotion in unstructured environments, we would ideally like a high-fidelity model that can generate long-horizon, full-body motion plans in real time. Many model predictive control approaches [4], [5] and trajectory optimization techniques [6], [7] have been suggested for humanoids and other legged creatures. However, most - if not all - of these algorithms are not feasible to run in real time on an onboard computer of a legged robot. For instance, Koenemann et al. [8] enabled real-time model predictive control based on differential dynamic programming (DDP) for the HRP-2 humanoid robot, however, their planner was running on an offboard 12-core machine. Similarly, the motion planner developed for the same robot in [9] using a multiple shooting method was executed offline on a 4-core machine. Another approach based on DDP presented in [10] finds motions for the quadruped robot HyQ within a minute and is able to optimize a simpler manipulation task online. While the latter three methods employ a model of the robot dynamics including forces, our approach solves the motion planning problem with a simple model and a direct collocation scheme within seconds on the onboard computer. The main difference and novelty of our proposed method lies in the combination of a simple, and thus computationally efficient, planning and robust tracking method. In [8], only a PD joint position controller was used to track the optimized balancing motions because the optimized state-feedback gains could not be applied due to transmission bandwidth problems between the computers, and a virtual model controller with PD joint control was applied in [10] instead of the feedback gains to avoid linearization issues. Our proposed tracking controller can compensate for modeling errors induced by the simplifications used for the planning by adapting the motion based on state measurements. Hence, online re-planning is not absolutely necessary and the controller can even cope with large perturbations. We employ a hierarchical inverse dynamics approach and exploit the torque-controllability of our robot instead of applying an inverse kinematics position-based method as in [9].

An alternative for motion planning are sampling-based optimization techniques like genetic algorithms [11], CMA-ES [12] and reinforcement learning [13], which have been used to search for motion primitives of parameterized controllers. In our previous work [14], we however showed that such a method using CMA-ES fails to find gait transitions, as many explored solutions failed to generate stable locomotion. Moreover, these methods are slow to generate motion plans for different high-level commands.

Consequently, we propose a different control architec-

*This work was supported in part by the Swiss National Science Foundation (SNF) through the National Centre of Competence in Research Robotics.

[1] Authors are with the Robotic Systems Lab, ETH Zurich, Switzerland
[2] Author is with the Robotics Institute, Carnegie Mellon University, USA.

ture. We use a fast trajectory optimization technique to create long-horizon plans that, through a simplified dynamics model, capture salient features of the robot's motion (e.g. center of mass and feet trajectories). As errors are introduced due to modelling approximations or unperceived events such as pushes, we employ a reactive feedback strategy that continuously adjusts the location of the footsteps and appropriately adjusts the tracked motion plan such that it remains synchronized with the state of the robot. Moreover, a hierarchical whole-body controller tracks the motion plan while considering the robot's dynamics based on actual measurements.

The trajectory optimization method we use is based on the work of Megaro et al. [15], where it was successfully demonstrated on 3D-printable legged robots. However, the motion plans were tracked using stiff, position-controlled actuators, whereas we employ our control scheme on a fully torque-controllable robot. Moreover, the robots that were used to execute the planned motions had low center-of-mass and wide area of support, which made them significantly more stable than our quadrupedal robot platform. We show the effectiveness of our control system by enabling a 30kg quadruped to walk, trot and transition between these two gaits.

II. Motion Planning

We transcribe the motion generation problem to a nonlinear program

$$\min \sum_j \alpha_j f_j(\mathbf{p}) \quad \text{s.t.} \quad \mathbf{g}(\mathbf{p}) = 0, \quad \mathbf{h}(\mathbf{p}) \quad \leq 0 \tag{1}$$

by discretizing the state trajectories of the robot over a finite horizon. The high-level motion goals (traveling distance) and physical constraints are encoded in the objective function as well as in the equality constraints $\mathbf{h}(\mathbf{p})$ and inequality constraints $\mathbf{g}(\mathbf{p})$. We formulate the objective function as weighted sum of individual objectives $f_j(\mathbf{p})$ with weight α_j,

To keep the number of optimization variables small, we only search for state variables on position-level and approximate the velocities and accelerations through finite-differences. We approximate the dynamics and predefine the contact schedule to create a problem that is fast to solve with modern sparse solvers. To further speed up the optimization, we introduce auxiliary variables which allow us to formulate the constraints in a linear form. The remaining nonlinear equality constraints are included in the objective functions as least-squares problem, which accelerates the convergence of most off-the-shelf solvers in our experience.

For each sample point i, we search for the following parameter vector:

$$\mathbf{p}_i = \left[\mathbf{q}_i, \mathbf{x}_i, \mathbf{e}_{1,i}, \ldots, \mathbf{e}_{4,i}, w_{1,i}, \ldots, w_{4,i}\right], \tag{2}$$

where $\mathbf{q}_i = \left[{}_P\mathbf{r}_{PB}, \mathbf{R}_{PB}, \varphi_1, \ldots, \varphi_{12}\right]$ is the state of the robot including the position ${}_P\mathbf{r}_{PB}$ and orientation $\mathbf{R}_{PB}$ of the base of the robot and the joint positions φ. The auxiliary variable $\mathbf{x}_i = \left[x_{i,x}, x_{i,y}, x_{i,z}\right]$ represents the COM position of the robot, whereas $\mathbf{e}_{k,i} = \left[e_{i,x}, e_{i,y}\right]$ corresponds to the k-th end-effector position. The height of the foot position is defined by the terrain during the stance phase and by a predefined foot clearance trajectory during swing phase. In addition, each end-effector has a corresponding scalar weight $w_{k,i}$ for defining the stability margin.

The complete motion plan is presented by N parameter vectors $\mathbf{p}_i$ over a finite horizon, the contact flags $\left[c_{1,1}, \ldots, c_{4,N}\right]$, which indicate whether the k-th end-effector is in contact with the ground ($c_{k,i} = 1$) or not ($c_{k,i} = 0$), and the time t_i of each sample point i.

The end-effector and COM positions are consistent with the state of the robot, if the following constraints are met:

$$\begin{aligned} \mathbf{r}_{\mathrm{COM}}(\mathbf{q}_i) - \mathbf{x}_i &= 0 \\ \mathbf{r}_{\mathrm{EE}_k}(\mathbf{q}_i) - \mathbf{e}_{k,i} &= 0, \quad \forall k, \end{aligned} \tag{3}$$

where $\mathbf{r}_{\mathrm{COM}}(.)$ and $\mathbf{r}_{\mathrm{EE}_k}(.)$ are the forward kinematics function. Due to the nonlinearity of the forward kinematics, these constraints are formulated as least-squares objective and are added to the objective function with a high weighting term.

The physical constraints are enforced by preventing any motion of the grounded end-effectors with

$$(\mathbf{e}_{k,i-1} - \mathbf{e}_{k,i})c_{k,i} = 0 \quad \forall k, 2 \leq i \leq N, \tag{4}$$

by minimizing the accelerations to ensure temporal consistency formulated as objective

$$\sum_{i=2}^{N-1} \left\| \frac{\frac{\mathbf{q}_i - \mathbf{q}_{i-1}}{t_{i-1} - t_i} - \frac{\mathbf{q}_{i+1} - \mathbf{q}_i}{t_{i+1} - t_i}}{t_{i+1} - t_{i-1}} \right\|^2, \tag{5}$$

and by limiting the joint velocities to avoid actuator saturations by

$$-\dot{\varphi}_{max} \leq \frac{\varphi_{k,i+1} - \varphi_{k,i}}{t_{i+1} - t_i} \leq \dot{\varphi}_{max}, \forall k, 1 \leq i \leq N-1, \tag{6}$$

where $\dot{\varphi}_{max} = 10\,\mathrm{rad/s}$ is the maximal speed of the actuator. Furthermore, joint limits are included as boundary constraints $\varphi_{k,min} \leq \varphi_{k,i} \leq \varphi_{k,max}$ in the optimization problem.

To ensure stable walking, we require the plan to meet a *dynamic stability criterion*, which keeps the approximated center of pressure (COP) within the support polygon at all times. The relation between the COM and the COP can be obtained by using the model of an inverted pendulum [16], [15]:

$$\mathbf{r}_{\mathrm{COP,i}} = \mathbf{x}_i - \frac{x_{z,i}\ddot{\mathbf{x}}_i}{\ddot{x}_{z,i} + g}, \tag{7}$$

where the gravitational acceleration is denoted by $g = 9.81\,\mathrm{m/s^2}$. The COP lies automatically within the support polygon if it is equal to the weighted average of the grounded end-effector positions, which we can achieve by adding the constraint:

$$\begin{aligned} &\sum_{k=1}^{4} c_{k,i} w_{k,i} \mathbf{e}_{k,i} - \mathbf{r}_{\mathrm{COP,i}} = 0, \quad \forall i, \\ &\sum_{k=1}^{4} c_{k,i} w_{k,i} - 1 = 0, \quad w_{min} \leq w_{k,i} \leq 1, \quad \forall i. \end{aligned} \tag{8}$$

By modulating the weights w_k, the COP can be shifted within the support polygon. We let the optimizer select the appropriate weights. By introducing a lower boundary w_{min} on the weight, a safety margin to the boundary of the support polygon can be realized.

The motion tasks are defined by

$$\begin{aligned} \mathbf{x}_e - \mathbf{x}_s &= \Delta \mathbf{x}, \\ \psi(\mathbf{q}_e) - \psi(\mathbf{q}_s) &= \Delta \psi, \end{aligned} \tag{9}$$

where $\Delta \mathbf{x}$ is the desired traveling distance between sample point s and target point e, and $\Delta\psi$ is the desired turning angle which is obtained from the robot pose by $\psi(.)$.

A periodic motion is encoded in the problem by constraining the joint positions at the last sample point with the start sample point p of the periodic motion:

$$\varphi_{k,N} - \varphi_{k,p} = 0, \quad 1 \leq k \leq 12. \tag{10}$$

Because multiple solutions can potentially exist for the aforementioned problem, we additionally add a regularizer term to the objective function.

We minimize the constrained optimization problem with sequential quadratic programming (SQP) using OOQP [17] to obtain the search direction as in [15]. To compute the Jacobians of the end-effectors, which are needed to compute the gradient of the objective function, we make use of the C++ library RBDL [18] and compute the Hessians according to [19].

An example of a resulting motion plan is shown in Fig. 1, where each point indicates a sample point either of the end-effector positions (red), COM trajectory (green) or base motion (yellow).

III. Motion Execution

The motion plan determines the desired joint position trajectories of the legs of the robot, but also the motion of the torso, the COM and the feet. We can select different trajectories for tracking to achieve the same motion since the information is redundant. However, model errors and perturbations can make a difference in practice. The pose of the robot can diverge from the planned path substantially, for instance, when the robot is pushed.

The benefit of tracking joint trajectories is that the joint angles are invariant to the robot's pose. If the robot is simply yawed, all Cartesian quantities, like the COM positions, which are expressed in a fixed frame (expressed in the plan's inertial frame), need to be rotated accordingly.

Vice versa, if the actual ground is at a different location than assumed during the planning phase, high-gain joint position control of the stance legs will lead to a wrong torso position and can create large ground reaction forces, which can cause slippage in worst case. To avoid the latter problem, the position control loop should be rather closed on whole-body level than on joint-level. By using a whole-body controller with joint torque regulation, the physical constraints can be considered based on actual measurements to safely track the base motion.

State feedback control is essential to successfully execute the motion plans on a real system. Contact sensing coupled with a reactive behavior helps to guarantee that the feet touch the ground at the end of the swing phase. A simple balance controller based on velocity feedback of the torso can stabilize the robot through stepping as showed in [20] and in our previous work [14]. These reactive behaviors for the swing foot define a desired set-point in Cartesian space. For this reason, we choose the base motion and end-effector motions for tracking.

As a consequence of tracking Cartesian quantities, an alignment of the plan with the actual pose of the robot is needed. However, the pose of the robot cannot be measured directly and thus needs to be estimated. We employ an Extended Kalman filter that fuses leg kinematics with inertial measurements and contact sensing to estimate the pose and velocity of the robot's trunk with respect to an inertial frame [21].

A. Plan Alignment

The estimated position and yawing of the torso drift because they are not observable with the robot's limited perception. For this reason, the plan needs to be aligned at each control update. Our goal is to find the homogeneous transformation between the plan frame P and the robot's inertial frame I:

$$\mathbf{T}_{IP} = \begin{bmatrix} \mathbf{C}_{IP} & {}_I\mathbf{r}_{IP} \\ 0 & 1 \end{bmatrix} \text{ s. t. } \begin{bmatrix} {}_I\mathbf{r}_{IR} \\ 1 \end{bmatrix} = \mathbf{T}_{IP} \begin{bmatrix} {}_P\mathbf{r}_{PR} \\ 1 \end{bmatrix}. \tag{11}$$

All Cartesian quantities of the parameter vector $\mathbf{p}_i$ are expressed in the plan frame P, whereas, the robot's state is expressed in the robot's inertial frame I. We first compute the rotation matrix $\mathbf{C}_{IP}$ and compute the translational part ${}_I\mathbf{r}_{IP}$ in a second step by

$${}_I\mathbf{r}_{IP} = {}_I\mathbf{r}_{IR} - \mathbf{C}_{IP}\,{}_P\mathbf{r}_{PR}, \tag{12}$$

where we match a reference point R.

First of all, we want the terrain described in the plan frame to coincide with the terrain expressed in the robot's inertial frame. In previous work [14], we estimated the local terrain by fitting a plane through a history of stance feet locations and aligned the desired motion accordingly. This allowed the quadruped to walk and trot on slopes. For the planning phase, we assume level-ground, which is aligned with the x-y-plane of the plan frame. By aligning the plan frame with the estimated terrain, one rotational degree of freedom is left, namely the yaw angle around the normal of the plane. Since the ground planes should coincide, the reference point should also lie on the terrain. Hence, another two degrees of freedom need to be determined.

The heading direction and the reference point should be attached to the robot in order to compensate for large tracking errors. Fig. 2 illustrates two ways to align the motion plan (green). The base-centered approach matches the planned base position (blue) with the measured base position (black), whereas the feet-centered method (red) matches the planned and measured average of the feet positions.

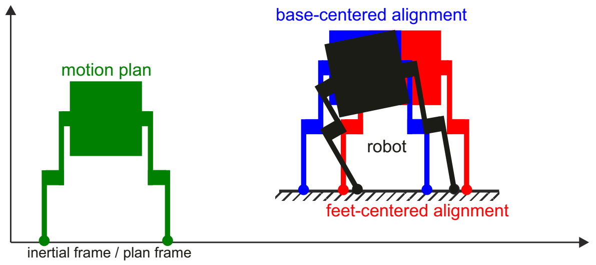


Fig. 2: The 2D drawings show the different alignment strategies of the motion plan (green): base-centered (blue) and feet-centered (red). If the planned motion expressed in plan frame is not aligned with the measured state of the robot (black) expressed in inertial frame, the robot will fall.

The heading direction can be be defined by the vector that goes through the middle of the fore and hind hips in the base-centered method, or through the middle of the fore and hind feet in the feet-centered approach.

A large push at the torso favors a base-centered approach because then the next footholds will still be in the leg's reachable workspace, but since the alignment needs to be updated at each control step due to the drifting state, the tracking error of the torso motion would be set to zero at each step. The feet-centered approach would not suffer from this and would match better the support polygon of the actual robot with the one of the plan, which is important to guarantee stability. We use a mixed approach to benefit from both methods.

As the motion of the swing legs can differ substantially from the planned ones, especially when the stepping control to maintain balance is active, only the positions of the currently grounded feet and the positions of the swing feet at lift-off event are used to compute the heading direction and reference point. This assumes that the drift is relatively low, which is valid for our system. The disadvantage of this method is that the reference point and the heading direction jump every time when a new foothold is established. To smooth out this effect, we employ a simple first order filter for the position and use spherical linear interpolation for the rotation. To get the corresponding reference point in the plan frame, the actual lift-off times are stored and the positions are looked up in the plan accordingly.

B. *Swing Motion Tracking*

The desired motion of the swing legs are generated by following the position and velocity of an interpolated quintic Hermite spline that goes through the measured lift-off position and the desired foothold position expressed in the robot's inertial frame. The spline needs to be fitted at each control update because the desired foothold changes due to plan alignment and balance control.

The balance controller based on an inverted pendulum model predicts the next foothold by considering the measured and desired velocity of the torso [14]. The intermediate foot positions from the plan are corrected with the linearly interpolated feedback. At lift-off the feedback control has no influence while it is fully active towards the end of the swing phase.

Since the feet-centered plan alignment is adapting too slowly in case of large pushes, we choose the base-centered approach for determining the swing foot positions.

The desired joint positions and velocities are computed from the desired foot position and linear velocity using inverse kinematics and damped least-squares pseudo inverse [22], respectively.

C. *Stance Motion Tracking*

For the motion of the torso, which lives in SE(3), we use third order Hermite splines with Hamiltonian unit quaternions as proposed in [23]. The spline is fitted through the base positions and evaluated in plan frame. The resulting position is transformed using Eq. 11 at each control step using the feet-centered plan alignment while the absolute velocities are only rotated by $\mathbf{C}_{IP}$.

In our previous work [14], we used a contact force distribution with a virtual model controller together with Jacobian transpose control to regulate the pose of the robot. The contact force distribution solved a quadratic problem to find contact forces, which result in a desired net force and torque at the robot's torso to generate the desired base motion. For different gaits, different weights were required to balance the gravity compensation against the tracking of the center of mass to ensure stability. This is unfortunately a problem for gait transitions and we therefore select a hierarchical technique in this work. The hierarchical approach solves the physical constraints on a higher priority than the tracking task such that one set of gains can be used for all gaits.

We use a hierarchical inverse dynamics controller based on the the whole body control framework used in [24]. In contrast to [24], the optimization parameters of the quadratic problems are selected as desired accelerations $\dot{\mathbf{u}}$ and contact forces $\boldsymbol{\lambda}$ similar to the approach in [25], which is faster to generate the joint torques $\boldsymbol{\tau}$.

The following tasks are solved with decreasing priority:

a) Dynamic consistency: The equations of motion can be written in the form

$$\begin{bmatrix}\mathbf{M}_b\\ \mathbf{M}_j\end{bmatrix}\dot{\mathbf{u}} - \begin{bmatrix}\mathbf{h}_b\\ \mathbf{h}_j\end{bmatrix} - \begin{bmatrix}\mathbf{J}_{C,b}^{\mathsf{T}}\\ \mathbf{J}_{C,j}^{\mathsf{T}}\end{bmatrix}\boldsymbol{\lambda} = \begin{bmatrix}0\\ \mathbf{S}_j^{\mathsf{T}}\end{bmatrix}\boldsymbol{\tau}, \tag{13}$$

where the mass matrix is denoted by $\mathbf{M}$, the nonlinear effects by $\mathbf{h}$, the support Jacobian by $\mathbf{J}_{C,b}$, and the selection matrix by $\mathbf{S}_j$. The top six equations of Eq. 13, indicated by subscript b, define the motion of the base.

As a highest priority task we define the following equality task to ensure dynamic consistency:

$$\begin{bmatrix}\mathbf{M}_b & \mathbf{J}_{C,b}^{\mathsf{T}}\end{bmatrix}\mathbf{x} = \mathbf{h}_b \tag{14}$$

with $\mathbf{x} = [\dot{\mathbf{u}}^{\mathsf{T}}, \boldsymbol{\lambda}^{\mathsf{T}}]^{\mathsf{T}}$.

b) Contact constraints: A second task constrains the contact forces by the unilateral and frictional constraints, which are approximated by friction pyramids. The constraints are set up as described in [24], but do not need to be mapped to joint torques since we optimize contact forces directly.

We additionally limit the norm of the force depending on the stride phase. We gradually increase the maximal force from zero at touch-down to a high value within 2% of the stance phase and decrease it towards the end of the stance phase. This avoids too large step inputs to the actuators.

c) Contact invariant motion: To generate consistent generalized accelerations, the linear accelerations of the grounded feet need to be zero. We add this equality task with a higher priority than the tracking task.

d) Base motion tracking: The desired position $\mathbf{r}^*_{IB}$, linear velocity $\dot{\mathbf{r}}^*_{IB}$ and acceleration $\ddot{\mathbf{r}}^*_{IB}$ of the base can be tracked by defining the task

$$\begin{bmatrix}\mathbf{J}_B & 0\end{bmatrix}\mathbf{x} = \ddot{\mathbf{r}}^*_{IB} + \mathbf{K}_D(\dot{\mathbf{r}}^*_{IB} - \dot{\mathbf{r}}_{IB}) + \mathbf{K}_P(\mathbf{r}^*_{IB} - \mathbf{r}_{IB}), \quad (15)$$

with $\mathbf{J}_B = \frac{\partial \mathbf{r}_{IB}}{\partial \mathbf{q}}$, and gain matrices $\mathbf{K}_D$ and $\mathbf{K}_P$. Note that the gains are tuned for the given robot and can differ for different directions depending on the morphology of the robot. For this reason all quantities are transformed to a so-called *control frame*, which is aligned with the base-centered reference frame and has its origin at the robot's inertial frame. More details about the control frame and how the desired orientation is tracked can be found in [14] and [24], respectively.

e) Swing leg motion tracking: As lowest priority task, the swing leg motion tracking is added to the optimization problem. On the current version of the robot, the joint positions of the swing legs are tracked by the motor drives of the actuators. For this reason, we compensate for the swing leg motion in the whole body controller. As a simple approximation, we set the joint accelerations of the swing legs equal to zero.

IV. Results

We implemented the complete framework on the onboard computer, of ANYmal, an Intel's NUC with 5th gen. Intel Core i5-5300U vPro processor (2.3 GHz Dual Core, 3 MB Cache), which updates the motion controller at 400 Hz. To verify the proposed method, we optimize motion plans for different gaits. The robot starts in a default standing configuration and executes a non-periodic gait followed by a periodic gait that is repeated. The motion plan is thus initialized with the current pose of the robot at the first sample point. An additional equality constraint on the end-effector, COM positions and base orientation ensures that the motion plan can be executed from the given start point.

A. Lateral walk

As a first experiment, we define the contact schedule according to Fig. 3a, which represents a lateral walking pattern. The motion plan has 16 sampling points for each walking cycle, which is the minimal number to realize that walking pattern. We add an extra sampling point at the end of the plan for the periodic constraint. The time is initialized with equal time steps resulting in a stride duration of 5 s. The goal of the robot is to walk forward by 0.2 m per stride.

We terminate the trajectory optimization after 6 iterations since the cost of the objective function has converged as

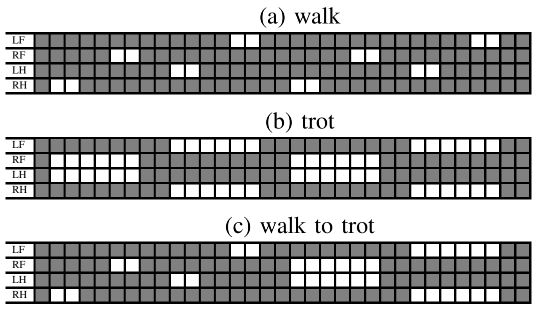


Fig. 3: The contact schedule defines for each sample point whether the four legs (LF, RF, LH, RH) are grounded (dark) or not (light).

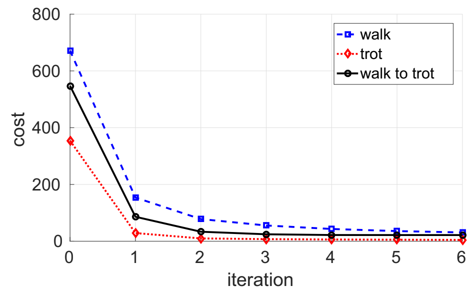


Fig. 4: Convergence of the objective function value for the different experiments.

shown in Fig. 4. The weights of the individual objectives and the resulting costs are listed in Tab. I. The smoothing objectives are minimizing the accelerations. We also added smoothing terms for the COM and end-effector motions, which are always smooth in reality.

The generated motion plan was successfully executed on ANYmal. Based on experience, the relatively slow lateral walk is prone to fail if the dynamical stability margin is violated. Although the motion plan was very coarse, the generated trajectories were good enough for walking as shown in the supplementary video[1]. Fig. 5a shows the footprints of ANYmal with the center of mass position with respect to the robot's inertial frame. The robot was pushed to the side such that its left hind foot lost contact. Due to the plan alignment, the planned motion was adapted as illustrated in Fig. 5b such that the robot did not fall.

B. Trot

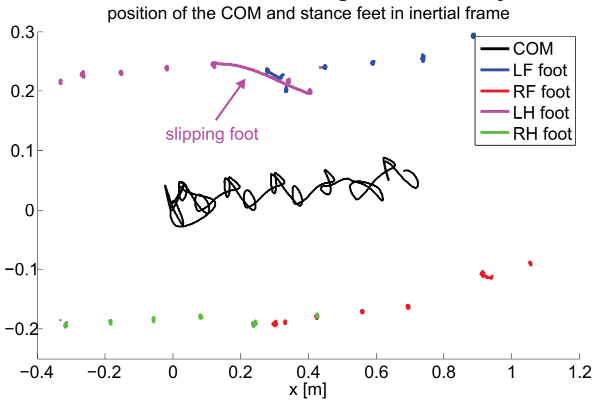
In a second experiment, we change the contact schedule to achieve a walking trot with stride duration of 0.8 s as shown in Fig. 3b. We use the exact same cost weights in the trajectory optimization and the exact same gains and prioritization in the whole body controller.

The robot successfully moves forward and can overcome a wooden plank that was not considered in the planning phase. Fig. 6 shows how the state-feedback controller adapts the pitch angle and foot height trajectories when the robot steps on the plank. Note that the foot heights are computed based

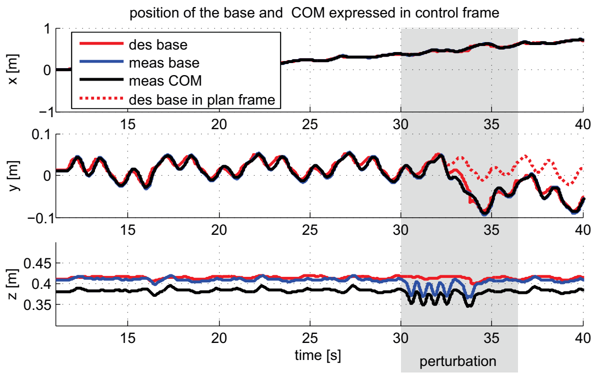
[1]https://youtu.be/LEgijRGu12I

Objective	Weight	Walk	Trot	Transition
Center of mass consistency	$1 \cdot 10^5$	0.41	0.01	0.12
End-effectors consistency	$1 \cdot 10^5$	0.23	0.01	0.09
Dynamic stability	$1 \cdot 10^3$	2.18	0.02	1.00
Traveling distance of 1st cycle	$1 \cdot 10^3$	11.95	2.91	15.79
Traveling distance of 2nd cycle	$3 \cdot 10^3$	5.71	0.25	1.64
Turning angle of 2nd cycle	$1 \cdot 10^3$	0.05	0.00	0.00
Smooth base motion	$1 \cdot 10^2$	5.57	0.04	1.12
Smooth joint motions	$1 \cdot 10^1$	3.08	1.26	1.46
Smooth center of mass motion	$1 \cdot 10^1$	0.40	0.001	0.10
Smooth end effector motions	$1 \cdot 10^1$	2.27	0.08	0.34

TABLE I: Cost and weights for the objectives.



(a) The visualization of the feet positions during stance show that the left hind foot lost contact.



(b) The plan is adapted based on the state feedback as can be seen by the reference signal of the y-position of the torso.

Fig. 5: The feed-back controller is able to recover from pushes at the torso while ANYmal is walking.

on the drifting base position. A shot of the planned motion can be seen on the right side of Fig. 1.

The trajectory optimization takes only about 6 to 8 seconds, which allows to quickly generate motion plans for different high-level commands, for instance, to turn in place. Fig. 7 shows the yawing angle of an experiment where we ask the robot to turn 20° per stride. We push the robot at the base as shown in the supplementary video to demonstrate the robustness of our approach. The pushes can be observed in Fig. 7 by the change of the horizontal position (x,y) of the robot's base.

C. Walk to trot transition

Automatic motion generation is especially useful for gait transitions where it is not obvious where the robot should step and how it should move its torso between the limit cycles. In a last experiment, we show that our approach is able to generate a transition from walk to trot with the given contact schedule as illustrated in Fig. 3c. We set the duration of the walking pattern to 4 s while the trotting phase should last 0.8 s. The proposed framework generates the foothold locations and achieves a transition as shown in the sequence of pictures in Fig. 8.

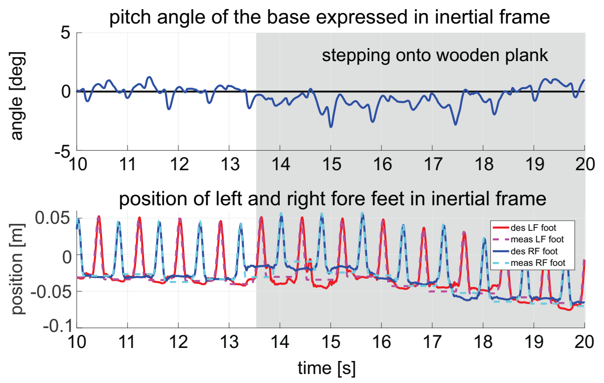


Fig. 6: The plot visualizes how the pitch angle and swing trajectories are adapted when ANYmal trots over an unperceived wooden plank with its front legs.

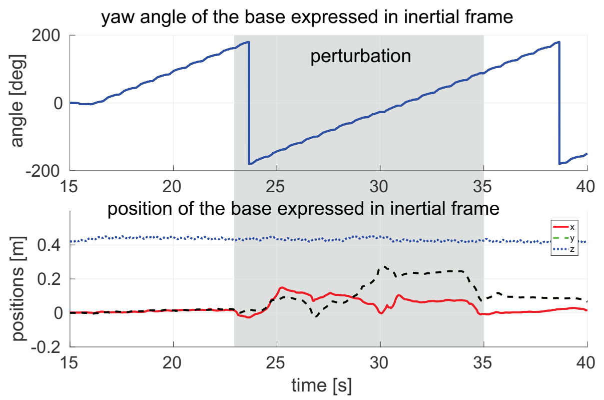


Fig. 7: The plot shows the yaw angle and position of ANYmal's torso while executing a motion plan that asks the robot to trot in a circle. The deviation of the x-y position visualize the external push.

V. Conclusion

In this paper we proposed a trajectory optimization formulation for quadrupedal locomotion, which is able to generate motion plans for a walk, a trot and a gait transition. The planning method simplifies the system extremely and outputs only a very coarse plan, but is able to do it within seconds compared to our previous sampling-based approach [14] that took hours. Due to the integration of simple state-feedback laws and a hierarchical whole body controller in the motion execution, the robot can successfully follow the motion plans, even in presence of perturbations. The experimental results show that motion optimization based on trajectory optimization is promising. In future, we want to search for gaits with full flight phases.

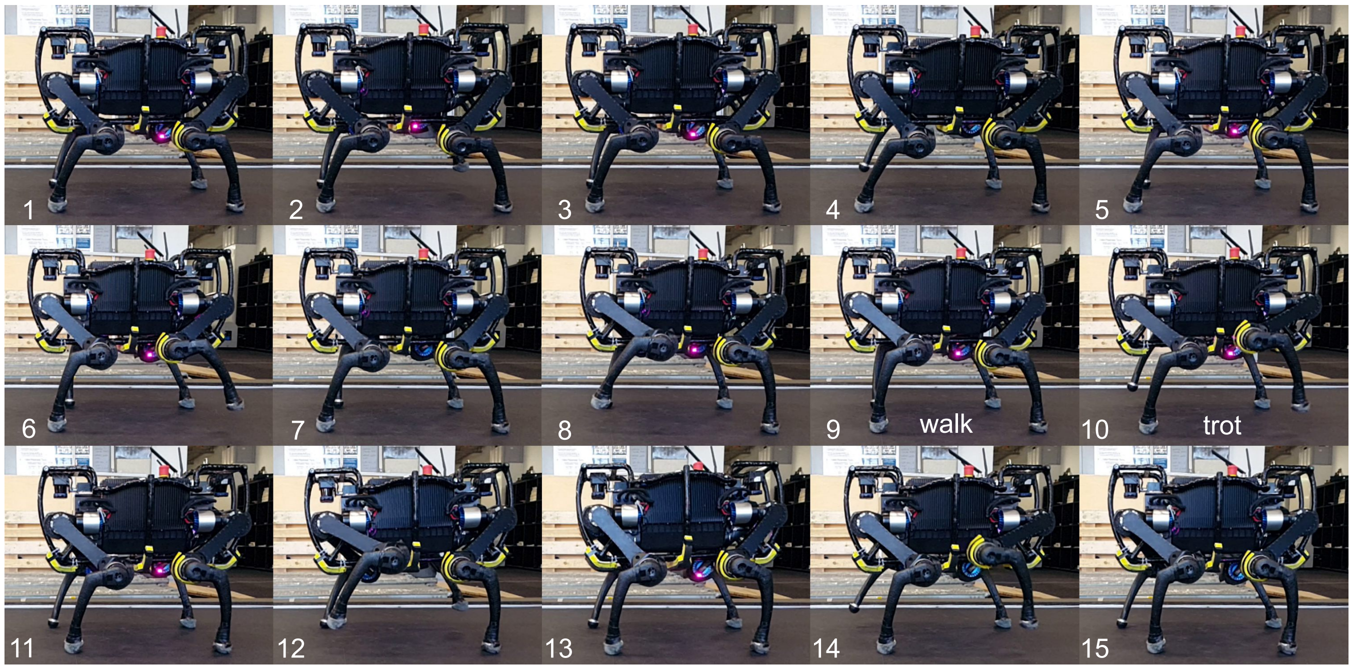


Fig. 8: The sequence shows a walking gait followed by two trotting cycles. The pictures were taken when a swing leg reaches its peak height and at touch-down event.

References

[1] M Raibert, K Blankespoor, G Nelson, and R Playter, "BigDog, the rough-terrain quadruped robot," in *Proceedings of the 17th World Congress*, 2008, pp. 10823–10825.

[2] S. Seok, A. Wang, M. Y. Chuah, D. J. Hyun, J. Lee, D. M. Otten, J. H. Lang, and S. Kim, "Design principles for energy-efficient legged locomotion and impl. on the mit cheetah robot," *IEEE/ASME Trans. on Mechatronics*, vol. 20, no. 3, pp. 1117–1129, June 2015.

[3] H. Hutter, C. Gehring, Jud D., A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, Bodie K., P. Fankhauser, M. Bloesch, R. Diethelm, S. Bachmann, A. Melzer, and M. Hoepflinger, "ANYmal - A Highly Mobile and Dynamic Quadrupedal Robot," *IEEE/RSJ Intenational Conference on Intelligent Robots and Systems*, 2016.

[4] Yuval Tassa, Tom Erez, and Emo Todorov, "Synthesis and Stabilization of Complex Behaviors through Online Trajectory Optimization," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2012.

[5] Joris Vaillant, Abderrahmane Kheddar, Adrien Escande, Kenji Kaneko, and Eiichi Yoshida, "Model Preview Control in Multi-Contact Motion Application to a Humanoid Robot," , no. Iros, pp. 4030–4035, 2014.

[6] Igor Mordatch, Emanuel Todorov, and Zoran Popović, "Discovery of complex behaviors through contact-invariant optimization," *ACM Transactions on Graphics*, vol. 31, no. 4, pp. 1–8, 2012.

[7] M. Posa, S. Kuindersma, and R. Tedrake, "Optimization and stabilization of trajectories for constrained dynamical systems," in *2016 IEEE Int. Conf. on Robotics and Automation*, May 2016, pp. 1366–1373.

[8] J. Koenemann, A. Del Prete, Y. Tassa, E. Todorov, O. Stasse, M. Bennewitz, and N. Mansard, "Whole-body model-predictive control applied to the hrp-2 humanoid," in *2015 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Sept 2015, pp. 3346–3351.

[9] J. Carpentier, S. Tonneau, M. Naveau, O. Stasse, and N. Mansard, "A versatile and efficient pattern generator for generalized legged locomotion," in *2016 IEEE Int. Conf. on Robotics and Automation (ICRA)*, May 2016, pp. 3555–3561.

[10] M. Neunert, F. Farshidian, A. W. Winkler, and J. Buchli, "Trajectory Optimization Through Contacts and Automatic Gait Discovery for Quadrupeds," *ArXiv e-prints*, July 2016.

[11] Karl Sims, "Evolving 3D Morphology and Behavior by Competition," *Artificial Life*, vol. 1, no. 4, pp. 353–372, jan 1994.

[12] Thomas Geijtenbeek, Michiel van de Panne, and a. Frank van der Stappen, "Flexible muscle-based locomotion for bipedal creatures," *ACM Transactions on Graphics*, vol. 32, no. 6, pp. 1–11, 2013.

[13] N. Kohl and P. Stone, "Policy gradient reinforcement learning for fast quadrupedal locomotion," in *Proceedings of IEEE Int. Conf. on Robotics and Automation*, April 2004, vol. 3, pp. 2619–2624 Vol.3.

[14] Christian Gehring, Stelian Coros, Marco Hutter, Carmine Dario Bellicoso, Huub Heijnen, Remo Diethelm, Michael Bloesch, Peter Fankhauser, Jemin Hwangbo, Mark Hoepflinger, and Roland Siegwart, "Practice Makes Perfect: An Optimization-Based Approach to Controlling Agile Motions for a Quadruped Robot," *IEEE Robotics & Automation Magazine*, vol. 23, no. 1, pp. 34–43, mar 2016.

[15] Vittorio Megaro, Bernhard Thomaszewski, Maurizio Nitti, Otmar Hilliges, Markus Gross, and Stelian Coros, "Interactive design of 3d-printable robotic creatures," *ACM Trans. Graph.*, vol. 34, no. 6, pp. 216:1–216:9, Oct. 2015.

[16] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, "Biped walking pattern generation by using preview control of zero-moment point," *2003 IEEE Int. Conf. on Robotics and Automation*, pp. 1620–1626, 2003.

[17] EM Gertz and SJ Wright, "Object-oriented software for quadratic programming," *ACM Transactions on Math. Software*, pp. 1–23, 2003.

[18] Martin Felis, "Rigid Body Dynamics Library," Available at `http://rbdl.bitbucket.org/`.

[19] Makoto Iwamura and Masafumi Nagao, "A method for computing the Hessian tensor of loop closing conditions in multibody systems," *Multibody System Dynamics*, vol. 30, no. 2, pp. 173–184, 2013.

[20] Marc H Raibert, *Legged robots that balance*, MIT Press, Cambridge, Mass., 1986.

[21] Michael Bloesch, Marco Hutter, Mark Hoepflinger, Stefan Leutenegger, Christian Gehring, C. David Remy, and Roland Siegwart, "State Estimation for Legged Robots-Consistent Fusion of Leg Kinematics and IMU.," *Proceedings of Robotics: Science and Systems*, 2012.

[22] Y. Nakamura and H. Hanafusa, "Inverse Kinematic Solutions With Singularity Robustness for Robot Manipulator Control," *Journ. of Dyn. Systems, Meas., and Control*, vol. 108, no. 3, pp. 163, 1986.

[23] Myoung-Jun Myung-Soo J S Kim and Sung Yong Shin, "A general construction scheme for unit quaternion curves with simple high order derivatives," *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pp. 369–376, 1995.

[24] C. Dario Bellicoso, Christian Gehring, Jemin Hwangbo, Péter Fankhauser, and Marco Hutter, "Perception-less terrain adaptation through whole body control and hierarchical optimization," in *Humanoids, 2015 IEEE-RAS 15th Int. Conf. on*, Nov 2016.

[25] Alexander Herzog, Nicholas Rotella, Sean Mason, Felix Grimminger, Stefan Schaal, and Ludovic Righetti, "Momentum control with hierarchical inverse dynamics on a torque-controlled humanoid," *Autonomous Robots*, vol. 40, no. 3, pp. 473–491, 2016.