

# State Estimation for Hybrid Locomotion of Driving-Stepping Quadrupeds

Mojtaba Hosseini

*Autonomous Intelligent Systems  
University of Bonn, Germany  
hosseini@ais.uni-bonn.de*

Diego Rodriguez

*Dexterity Inc.**diego.rodriguez@dexterity.ai*

Sven Behnke

*Autonomous Intelligent Systems  
University of Bonn, Germany  
behnke@cs.uni-bonn.de*

**Abstract**—Fast and versatile locomotion can be achieved with wheeled quadruped robots that drive quickly on flat terrain, but are also able to overcome challenging terrain by adapting their body pose and by making steps. In this paper, we present a state estimation approach for four-legged robots with non-steerable wheels that enables hybrid driving-stepping locomotion capabilities. We formulate a Kalman Filter (KF) for state estimation that integrates driven wheels into the filter equations and estimates the robot state (position and velocity) as well as the contribution of driving with wheels to the above state. Our estimation approach allows us to use the control framework of the Mini Cheetah quadruped robot with minor modifications. We tested our approach on this robot that we augmented with actively driven wheels in simulation and in the real world. The experimental results are available at <https://www.ais.uni-bonn.de/~hosseini/se-dsq>.

**Index Terms**—Quadruped robot, State estimation, Hybrid driving-stepping locomotion

## I. INTRODUCTION

Robots with legs are well suited for locomotion on irregular terrain. Walking robots are great for negotiating obstacles like stairs, but they have limited speed and are not very energy-efficient. In contrast, robots with wheels can move quickly and efficiently on flat terrain and have very low transportation costs—compared to their legged counterparts. They cannot overcome obstacles larger than the radius of their wheels, however. An environment with obstacles or height differences, such as curbs for pedestrians, imposes hard constraints that may be impossible for driving robots to overcome.

The above characteristics motivated us to combine the capabilities of wheeled and legged robots by using fast and efficient driving on flat terrain and articulated legs to adapt to the terrain and to overcome obstacles by making steps, as illustrated in the bottom part of Fig. 1. The resulting hybrid locomotion resolves the trade-off between efficiency and speed in systems with wheels and legs by reducing the number of leg swings, since driving is usually preferred over walking on flat terrains. At the same time, stepping is applied to handle steps and irregular terrain. This way, we increase the robot's locomotion speed and energy efficiency without compromising its versatility.

In this paper, we present a novel state estimation approach to enable dynamic hybrid driving-stepping locomotion for a wheeled quadruped robot that uses online Model Predictive

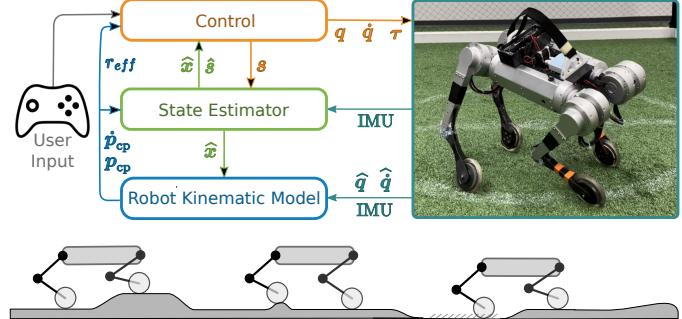


Fig. 1. Overview of the control framework (top). The measured joint position  $\hat{q}$  and velocity  $\hat{q}$ , and data from IMU are sent to the robot model kinematic module, which calculates the position  $p_{cp}$  and velocity  $\dot{p}_{cp}$  of each contact point as well as the effective radius  $r_{eff}$  of the end effector. The state estimation module uses  $p_{cp}$ ,  $\dot{p}_{cp}$ , the IMU data and the foot contact states  $s$  to calculate the robot state  $\hat{x}$  and the updated contact state  $\hat{s}$ . The user input includes the desired driving, stepping, and turning velocities which are used by the control module along with  $\hat{x}$  to calculate joint position  $q$ , velocity  $\dot{q}$ , and torque  $\tau$ . The wheel controller applies additional control inputs to the wheel joints. The bottom part illustrates the usefulness of our state estimation for hybrid wheeled-legged quadruped locomotion. The estimated robot height remains unchanged even if the terrain has elevation changes.

Control (MPC) and a Whole Body Controller (WBC), taking into account the additional wheels in terms of their shapes, masses, and inertia tensors. This approach (Fig. 1 top) integrates the wheels into the robot's kinematics model and updates the Kalman Filter's equations. In summary, this paper makes the following main contributions:

- 1) Integration of wheels into the kinematic model of a hybrid four-legged Mini Cheetah robot to determine the exact contact position and velocity,
- 2) Incorporation of the corresponding wheel kinematics into the state estimator of hybrid quadrupeds, and
- 3) Extending the controller to use the state estimator to drive and step simultaneously.

## II. RELATED WORK

### A. Hybrid Locomotion

Quadruped robots are complex systems that have received much attention in recent years. Some works also investigated hybrid wheeled-legged, driving-stepping locomotion. For example, Bjelonic et al. [1] and Klamt et al. [2], [3] present hybrid driving-stepping methods that use the wheels only

for driving while performing the stepping motions separately. Bellegarda and Byl [4] and Geilinger et al. [5] use Trajectory Optimization (TO) to precompute complex trajectories for hybrid locomotion platforms over a time horizon offline. They plan motion on flat terrain by solving a nonlinear programming problem. De Viragh et al. [6] extend [1] by computing robot base and wheel trajectories in a single optimization framework using linearized Zero Moment Point (ZMP) constraints. This approach is only suitable for low update rates, however, and no test on real platforms is performed. Bjelonic et al. [7] present an optimization framework that incorporates the additional degrees of freedom introduced by wheels into the motion generation. The optimization problem is split into end-effector and base trajectory planning, similar to Rebula et al. [8], to make locomotion planning more manageable for high-dimensional robots with wheels and legs. Bjelonic et al. [9] propose a method that assigns a utility value to each leg to allow the robot to switch between pure driving, static walking, and trotting. Klemm et al. [10] present the bipedal robot Ascento that drives quickly on wheels, balances on flat terrain, and overcomes obstacles by jumping. However, the dynamics of the leg links and motors are neglected, which is significant due to the weight of the wheels compared to the body.

In this work, a hybrid driving-stepping Mini Cheetah quadruped robot with actively driven wheels is introduced and used for experiments in simulation and with real hardware.

### B. State Estimation

State estimation plays a crucial role in this type of system. Robots can localize themselves in their environment using visual or GPS-based methods [11], [12], [13]. However, for a legged robot that lacks perceptual units and maps of the environment, state estimation relies on data from Inertial Measurement Units (IMUs) and the robot's interaction with the environment over multiple intermittent ground contacts.

Lin et al. [14] fuse IMU measurements with a leg-based odometer for a hexapod robot. The state of the robot is calculated based on the assumption that three feet of the hexapod robot are constantly in contact with a flat surface. Their method is limited to walking and running on three legs and is therefore not applicable to gaits such as trotting where the robot is underactuated.

Blösch et al. [15] present an Extended Kalman Filter (EKF) method that fuses kinematic encoder data with IMU measurements by including the absolute position of all feet in the filter state to accurately capture the uncertainties associated with ground contacts. The resulting filter simultaneously estimates the pose of the main body and the positions of the footholds without making assumptions about the shape of the terrain. This approach is only tested in simulation, though.

The aforementioned contact-based methods require accurate knowledge of the contact state of each leg and therefore need dedicated sensors on each foot to detect ground contact. However, the use of accurate contact sensors increases hardware design complexity and associated costs. Our proposed

approach does not require such sensory capabilities, making it suitable for a broader range of robotic platforms.

Bledt et al. [16] present a probabilistic contact estimation algorithm that uses a KF to fuse precomputed probabilities: the contact probability from the gait swing phase, foot height, and foot force. Similarly, Camurri et al. [17] estimate the probability of reliable contact and detect foot impacts using internal force sensing. This information is then used to improve the estimation of the kinematic-inertial state of the robot base, resulting in performance comparable to systems with foot contact sensors.

The state estimation approach proposed in this work is inspired by Katz et al. [18], who estimate the position and orientation of a Mini Cheetah quadruped robot for an inertial coordinate system without a perceptual unit. Instead, information derived from ground contact points is incorporated into the state estimation equations. In our approach, measurements from an IMU and joint encoders are fused to account for driving movements. We estimate the 6D robot pose and the position of all four end-effectors using a KF. The fusion is based on the contact states given by the robot's gait sequence.

## III. ROBOT KINEMATIC MODEL

Accurate kinematic modeling of the robot plays a key role in state estimation.

The Mini Cheetah quadruped has a rubber ball at the end of each leg as an end effector. The size and shape of these rubber balls are small. Therefore, they have not been previously considered in robot kinematics, but were modeled as single ground contact points [7], [9], [19], [1], [20]. In this work, however, the wheels replacing the rubber balls have a radius of 5 cm, which is non-negligible compared to the shank length of 19 cm. Consequently, we can no longer consider the ground contact points to be in the center of the end effector.

We define a ground contact point as shown in Fig. 2. The touchdown position of the end effector on the arbitrary horizontal ground is defined as  $\mathbf{p}_{cp}$ . We assume that the ground surface is locally flat. The possible errors between the actual contact point and  $\mathbf{p}_{cp}$  are considered negligible in this work.

We model the wheel of the Mini Cheetah robot as shown in Fig. 2, with radius  $a_{end}$  and width  $b_{end}$ . Assuming that  $b_{end} \leq a_{end}$ , the touchdown position  $\mathbf{p}_{cp}$  in the relative coordinate system is defined as:

$$\mathbf{p}_{cp} = \begin{bmatrix} L_3 S_{23} + L_2 S_2 \\ L_1 C_1 + L_3 (S_1 C_{23}) + L_2 C_2 S_1 \\ L_1 S_1 - L_3 (C_1 C_{23}) - L_2 C_1 C_2 \end{bmatrix} - \underbrace{\begin{bmatrix} 0 \\ -r S_1 \\ r C_1 + b_{end} \end{bmatrix}}_{\text{end effector terms}}, \quad (1)$$

where  $r = a_{end} - b_{end}$  gives the length of the last link of the kinematic chain,  $S_i$  and  $C_i$  are abbreviations for  $\sin q_i$  and  $\cos q_i$ , respectively,  $C_{23} = C_2 C_3 - S_2 S_3$  and  $S_{23} = S_2 C_3 + C_2 S_3$ . The end effector terms in the equation are the result of an additional end effector geometry in the kinematics that defines the offset of the contact point  $\mathbf{p}_{cp}$  from the center of the end effector.

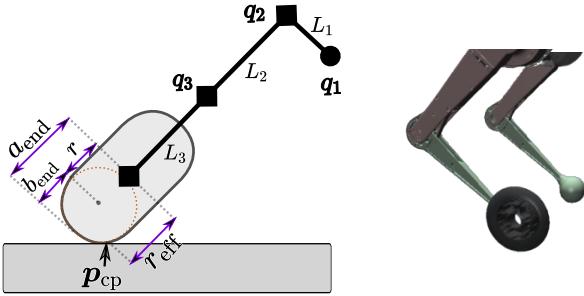


Fig. 2. Ground contact point modeling. On the right side, a comparison between the rubber ball and the wheel end effector is shown. On the left, the joints  $q_i$  are shown as circles or rectangles and the links are labeled  $L_i$ . The radii  $a_{\text{end}}$  and  $b_{\text{end}}$  define the geometry of the wheel end effector, and  $p_{\text{cp}}$  is the ground contact position at the effective radius  $r_{\text{eff}}$ .

The kinematic contribution of the leg to the touch-down velocity in the world frame is given as:

$$\dot{p}_{\text{cp}_k} = \mathbf{J}\dot{q}, \quad (2)$$

where  $\mathbf{J}$  is the Jacobian matrix of the kinematic chain of the leg including the end effector geometry. The contribution of the added terms for the end effector into the Jacobian matrix  $\mathbf{J}$  is given as:

$$\mathbf{J}_w = \begin{bmatrix} 0 & 0 & 0 \\ rC_1 & 0 & 0 \\ rS_1 & 0 & 0 \end{bmatrix}. \quad (3)$$

We have addressed the kinematic singularity at joint space by soft-stopping the knee joint before the leg is fully stretched. We define  $r_{\text{eff}}$  as the effective radius of the end effector (see Eq. 2). This radius is employed by the state estimator and the wheel controller to accurately convert linear velocities and forces into joint velocities and torques, and vice versa. Eq. 2 is formulated as a function of the hip angle  $q_1$  in the world coordinate:

$$r_{\text{eff}} = a_{\text{end}} - b_{\text{end}} \sin(q_1 + \phi), \quad (4)$$

where  $\phi$  is the roll angle of the robot base.

The contact point velocity  $\dot{p}_{\text{cp}_k}$  given in (2) takes into account the robot kinematics and the shape and size of the end effector. The robot's pitch velocity  $\dot{\theta}$ , hip velocity  $\dot{q}_2$ , and knee velocity  $\dot{q}_3$ , as well as the robot's roll velocity  $\dot{\phi}$  and hip roll velocity  $\dot{q}_1$  directly result in additional angular velocity of the end effector (i.e., wheels) along the  $x$  and  $y$  axes, respectively. These rotations result in a velocity  $\dot{p}_{\text{cp}_w}$  for the contact point, in addition to the current rotational velocity  $\dot{q}_4$  of the wheel read by the encoders. We define  $\dot{p}_{\text{cp}_w}$  as the contribution of the end effector to the global velocity of the contact point:

$$\dot{p}_{\text{cp}_w} = \left[ (\dot{\theta}C_1 + \dot{q}_2 + \dot{q}_3 + \dot{q}_4)r_{\text{eff}} \quad (\dot{\phi} + \dot{q}_1)b_{\text{end}} \quad 0 \right]^T. \quad (5)$$

Finally, the comprehensive contact point velocity  $\dot{p}_{\text{cp}}$  is:

$$\dot{p}_{\text{cp}} = \dot{p}_{\text{cp}_k} + \dot{p}_{\text{cp}_w}. \quad (6)$$

The state estimator makes use of the  $\dot{p}_{\text{cp}}$ , which accurately formulates the velocity at the contact position in world coordinates by taking into account the kinematic shape of the end

effector, the velocity feedback from the wheel encoder, and the velocity terms induced by other joints to the end effector.

#### IV. STATE ESTIMATION

The task of the state estimation module is to estimate the position, velocity, orientation, and angular velocity of the robot base in the world coordinate system. Ideally, a state estimator for a quadruped robot should be robust to different types of terrain, gaits, locomotion modes, and travel speeds. It should also take into account the fact that the motions are periodic over an interval and that the system interacts with the environment through multiple intermittent ground contacts.

The control of our four-legged robot uses the residuals between the desired and estimated robot states and therefore relies on accurate robot state estimation. The controller aims to reduce the aforementioned residual and brings the robot state close to the desired one. However, if the estimated state is incorrect, the controller will generate inappropriate control outputs, resulting in an undesirable body posture that could cause the robot to fall.

The legs in contact with the ground are the reference points for the state estimation. The estimator relies on these reference points and solves for the position and velocity of the body using forward kinematics and Jacobian matrices of the legs in contact. However, due to obstacles and disturbances, a leg that is supposed to be in contact may actually be in a swinging state, which clearly invalidates the estimation process because the contact condition cannot be trusted.

Our approach for state estimation does not rely on contact or ground reaction force sensors. Instead, we use the expected ground contacts defined with a periodic phase-based state planner for different gaits such as Trot, Walk, and Pronk [16]. We change the above-expected contacts to zero when the legs are extended close to their kinematic limits to avoid updating the robot state for legs in singular configurations.

For our state estimation, we use a KF that combines the end effector contact positions (2) and the comprehensive contact velocities (6), the angular velocity of the wheels, and the linear acceleration values from the IMU to obtain a robot position estimate. Furthermore, we obtain the orientation estimate directly from the IMU hardware.

##### A. Contact Trust Modeling

At each instant, each leg of the robot assumes the contact state  $s$  with one of two distinct states: contact or swing. The expected contact state  $\hat{s}$  for each leg is determined periodically from the gait at each time step during the control cycle. In reality, however, the legs are unable to follow the desired commands accurately due to disturbances and surface height differences, which results in a difference between  $\hat{s}$  and  $s$ . Therefore, simply using the contact locations in state estimation is not applicable and leads to large inaccuracies in the estimated robot state. For this reason, motivated from the phase-based probabilistic model for the expectation of contact given by Bledt et al. [16], we present contact trust modeling

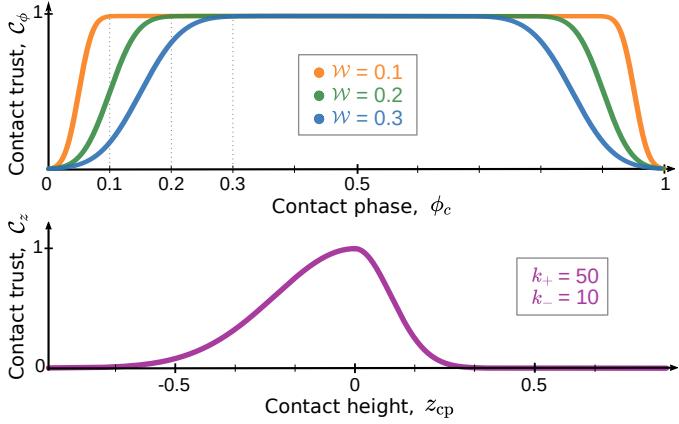


Fig. 3. Contact Trust Modeling. Top: The phase-based contact trust  $C_\phi$  is shown for various values of  $\mathcal{W}$  defining the mistrust window at the beginning and the end of the contact phase  $\phi_c$ . Below: The contact trust  $C_z$  as a function of contact height  $z_{cp}$  is presented for distrust gains  $k_+$  and  $k_-$ . Lower distrust gain for negative height differences results in relatively higher trust outputs.

that computes a confidence value for the legs expected to be in contact.

The contact phase variable  $\phi_c \in [0, 1)$  is defined over the period of the ground contact trajectory as:

$$\phi_c = \frac{t - t_c}{T_c}, \quad (7)$$

which is a linear function of time  $t$ , where  $t_c$  and  $T_c$  are the start time and duration of the contact phase. Phase-based contact trust (shown in the top part of Fig. 3) is formulated as<sup>1</sup>:

$$C_\phi = \frac{\hat{s}}{2} \left[ \text{erf} \left( 4 \frac{\phi}{\mathcal{W}} - 2 \right) + \text{erf} \left( 4 \frac{1-\phi}{\mathcal{W}} - 2 \right) \right], \quad (8)$$

where  $\hat{s}$  chooses the value 1 and 0 for the expected contact or swing, respectively,  $\mathcal{W}$  is the mistrust phase window at the beginning and end of the contact trajectory, and erf is the sigmoid error function [21].

The contact trust as a function of foot height (shown in the bottom part of Fig. 3) is defined as:

$$C_z = \begin{cases} \exp(-k_+ * z_{cp}^2), & \text{for } z_{cp} \geq 0 \\ \exp(-k_- * z_{cp}^2), & \text{for } z_{cp} < 0 \end{cases} \quad (9)$$

where  $z_{cp}$  is the contact height and  $k_+$  and  $k_-$  are the distrust gains for positive and negative contact height values, respectively. Higher gains lead to lower contact trust for the given foot height. By setting  $k_- < k_+$ , the lower ground surfaces produce more trustworthy contacts compared to the higher surfaces.

### B. State Estimator

To achieve hybrid driving-stepping locomotion, the state estimator should accommodate the wheels that can roll along the  $x$ -axis. The KF is able to perform state estimation for robots with legs using the robot measurements and exploiting

our contact trust modeling to update its covariance matrices accordingly.

We extended the state estimator of the Mini Cheetah robot [18] to include the wheels in the estimation process accurately. In the following, we first define the KF state vector and then further explain the KF model and equations. The notation  $k|k-1$  stands for estimation at time  $k$  considering observations up to and including time  $k-1$ . The identity matrix  $I_3$  and the zero matrix  $O_3$  are of size  $3 \times 3$ , and  $I_{12}$  and  $O_{12}$  are of size  $12 \times 12$ .

**1) Filter State:** The state vector of the filter must be chosen such that the corresponding prediction and measurement equations can be specified consistently. In our approach, we compose the state vector  $\mathbf{x}$  of the quadruped robot from the position and velocity of the robot base, the contribution of driving to the position and velocity of the robot base, and the contact point positions for all legs relative to the body, which take into account the kinematics of the legs:

$$\mathbf{x} := (p \ p \dot{p} \ p_w \ \dot{p}_w \ p_{cp}^1 \dots p_{cp}^4), \quad (10)$$

where all terms of  $\mathbf{x}$  are in the world frame,  $p$  and  $\dot{p}$  are the global position and velocity of the robot,  $p_w$  and  $\dot{p}_w$  are the contribution of driving with the wheels to the position and velocity of the robot, and  $p_{cp}^i$  is the ground contact position of the leg  $i$  relative to the body (Fig. 2).

Although the robot dynamics are complex, the above state vector allows us to use a simple linear KF. The inclusion of foot contact positions in the filter state, similar to [15], as well as the contribution of driving with wheels into the estimated position and velocity, are key points in our filter design that allow for a simple and consistent representation of the model equations. The KF is able to simultaneously correct for the position of the foot contacts, the pose of the main body, and the contribution of driving.

**2) Noise Covariance Update Gain:** When a foot lands on a height difference or at the beginning and end of the contact phase, we should avoid updating the filter for that contact point. Therefore, we formulate a trust function that takes into account the contact trust values  $C_\phi$  and  $C_z$  from (8) and (9), respectively, as:

$$\underset{3 \times 3}{C} = C_\phi \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & C_z \end{bmatrix}. \quad (11)$$

This formulation allows the legs to step on height differences without affecting the estimated robot height. If the robot steps on a height difference with one or two legs, the estimation avoids considering them in the filter update by assigning a lower trust to the corresponding legs. However, if multiple legs step on a height difference, the estimator recognizes this height difference as a new ground level and updates the estimated height accordingly. When the robot steps on negative ground heights, the corresponding legs get stretched and may reach their kinematic limits, which is not ideal. Therefore, for negative height differences, we configure a smaller  $k_-$  gain in (9), which favors lower ground heights in the estimation,

<sup>1</sup>Live graph: <https://www.desmos.com/calculator/djke61nx8n>

to compute the estimated height of the robot based on the lower surfaces in order to avoid over-extension of the legs. The trust information in (11) is used to define the noise covariance update gain as:

$$\xi_{12 \times 12} = \mathbf{I}_{12} + \kappa \left( \mathbf{I}_{12} - \text{diag}(\mathbb{C}^1 \ \mathbb{C}^2 \ \mathbb{C}^3 \ \mathbb{C}^4) \right), \quad (12)$$

where  $\kappa$  is a hand-tuned value defined as a high suspect number and  $\mathbb{C}^i$  is the contact trust value of  $i$ -th leg in (11).

$\xi$  encodes the confidence value for all legs (in contact or swing) and is used at each prediction and correction step of the KF to update the process and measurement noise covariance matrices such that only the trustworthy legs contribute to the estimation process.

3) *Prediction Model*: We formulate prediction equations to propagate the state from one time step to the next. We use the IMU acceleration measurements in the prediction model of the sensor fusion method (similar to [15], [22]) defined as:

$$\mathbf{u} = \mathbf{R}(\mathbf{a}_{\text{IMU}} - \mathbf{g}), \quad (13)$$

where  $\mathbf{a}_{\text{IMU}}$  is the measured linear acceleration in the body frame which is rotated by  $\mathbf{R}$  to obtain the acceleration in the global coordinate system. The standard prediction equations for the KF are as follows:

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_k \hat{\mathbf{x}}_{k-1} + \mathbf{B}_k \mathbf{u}_k, \quad (14)$$

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k \mathbf{P}_{k-1} \mathbf{F}_k^\top + \mathbf{Q}_k, \quad (15)$$

where  $k$  is the time step,  $\hat{\mathbf{x}}$  is the estimate of  $\mathbf{x}$ ,  $\mathbf{P}$  is the covariance matrix of the state vector, and  $\mathbf{F}$  is the state transition matrix, which applies the system dynamics as:

$$\mathbf{F}_{24 \times 24} = \begin{bmatrix} \mathbf{I}_3 & \Delta t \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & \Delta t \mathbf{I}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 \\ & & \mathbf{0}_{12} & & \mathbf{I}_{12} \end{bmatrix}. \quad (16)$$

The matrix  $\mathbf{B}_{24 \times 3} = [\mathbf{0}_3 \ \Delta t \mathbf{I}_3 \ \mathbf{0}_3 \ \dots \ \mathbf{0}_3]^\top$  is defined to update the velocity of the body from the acceleration vector  $\mathbf{u}$ , and  $\mathbf{Q}$  is the covariance matrix of the process noise defined as:

$$\mathbf{Q}_{24 \times 24} = \begin{bmatrix} \omega_p \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \omega_p \mathbf{I}_3 & \mathbf{0}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \omega_{p_w} \mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \omega_{p_w} \mathbf{I}_3 \\ & & \mathbf{0}_{12} & & \omega_{p_{cp}} \xi \mathbf{I}_{12} \end{bmatrix}, \quad (17)$$

where the  $\omega$  parameters are the process noise values for the corresponding variables in the state vector  $\mathbf{x}$ . The white noise terms  $\omega_{p_{cp}}$  account for some degree of slippage of the feet, and  $\xi$  sets the noise parameter of the swinging feet to a very large value, allowing the corresponding foot to shift and reset its position estimate when it makes contact with the ground again. This process deals with the contact changes required to execute a gait.

4) *Correction Model*: Knowing that the prediction in the hybrid locomotion scheme is likely to contain inaccuracies, we can use available measurements to correct the prediction and obtain a more educated estimate for the position. Since the wheels are integrated into the hybrid quadrupedal robot, the measurement vector  $\mathbf{z}$  contains the contact position and kinematic contribution, as well as the driving contribution to the contact point velocity for all legs as follows:

$$\mathbf{z} := \left( \mathbf{p}_{cp}^1 \dots \mathbf{p}_{cp}^4 \ \dot{\mathbf{p}}_{cp_k}^1 \dots \dot{\mathbf{p}}_{cp_k}^4 \ \dot{\mathbf{p}}_{cp_w}^1 \dots \dot{\mathbf{p}}_{cp_w}^4 \right), \quad (18)$$

where all terms of  $\mathbf{z}$  are in the global frame,  $\mathbf{p}_{cp}^i$  is the measured contact position of leg  $i$  relative to the body,  $\dot{\mathbf{p}}_{cp_k}^i$  and  $\dot{\mathbf{p}}_{cp_w}^i$  are the measured linear contact point velocity from kinematics and driving from Eq. (2) and (5), respectively.

The separate inclusion of the kinematic and driving contributions in the measurement vector  $\mathbf{z}$  plays a key role in the estimation to correctly capture the effects of stepping and driving on the position of the hybrid quadruped robot. The correction equations of the standard KF are formulated as follows:

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k|k-1}, \quad (19)$$

$$\mathbf{S}_k = \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^\top + \mathbf{R}_k, \quad (20)$$

$$\mathbf{K}_k = \mathbf{P}_{k|k-1} \mathbf{H}_k^\top \mathbf{S}_k^{-1}, \quad (21)$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \tilde{\mathbf{y}}_k, \quad (22)$$

$$\mathbf{P}_{k|k} = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1}, \quad (23)$$

where  $\tilde{\mathbf{y}}$  is the measurement residual (or innovation),  $\mathbf{S}$  is the covariance of the residual (or innovation covariance), and  $\mathbf{K}$  is the Kalman gain.

$\mathbf{R}$  is the measurement noise covariance matrix that encodes the confidence we place in the accuracy of the measurements. It is defined as follows:

$$\mathbf{R}_{36 \times 36} = \xi \begin{bmatrix} \nu_{p_{cp}} \mathbf{I}_{12} & \mathbf{0}_{12} & \mathbf{0}_{12} \\ \mathbf{0}_{12} & \nu_{\dot{p}_{cp_k}} \mathbf{I}_{12} & \mathbf{0}_{12} \\ \mathbf{0}_{12} & \mathbf{0}_{12} & \nu_{\dot{p}_{cp_w}} \mathbf{I}_{12} \end{bmatrix}, \quad (24)$$

where the  $\nu$  parameters are the measurement noise values for the corresponding variables in  $\mathbf{z}$ . The noise covariance update gain  $\xi$  is applied to the measurement confidence. Similar to the process noise covariance,  $\xi$  sets the noise parameter of the untrusted feet to a very large value when they are not in contact with the ground, which prevents the corresponding measurements from participating in the estimation process. Hence, we can rotate the wheels at different speeds during the swing phase without affecting the estimation process.

We construct the observation matrix  $\mathbf{H}$  with the following Jacobian:

$$\mathbf{H}_{36 \times 24} = \begin{bmatrix} \mathbf{I}_3 & \mathbf{0}_3 & -\mathbf{I}_3 & \mathbf{0}_3 & & -\mathbf{I}_{12} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0}_3 & \mathbf{I}_3 & \mathbf{0}_3 & -\mathbf{I}_3 & & \mathbf{0}_{12} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{0}_3 & \mathbf{I}_3 & & \mathbf{0}_{12} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}. \quad (25)$$

## V. CONTROL

The control method of our quadruped robot is based on Model Predictive and Whole-Body Controllers (MPC and WBC), which rely on accurate state estimation of the robot. Both MPC and WBC directly use the residual between the desired and estimated robot positions, and aim to reduce this residual and bring the robot close to the desired state. The MPC generates reaction force commands that are enhanced with the WBC to achieve accurate and fast joint control and refined reaction forces [19].

### A. Gait Controller

Our state estimator estimates the robot's position and velocity and separately includes the contribution of driving with wheels to the robot's estimated position and velocity. This separation allows us to use the Mini Cheetah control scheme with minor modifications and execute arbitrary gaits while simultaneously driving with wheels. By taking the current velocity of the robot without the contribution of driving as:

$$\hat{v}_g = \dot{p} - \dot{p}_w, \quad (26)$$

we update the footstep planner equations in [19]:

$$p_{\text{target}} = p_{\text{symmetry}} + p_{\text{centrifugal}}, \quad (27)$$

$$p_{\text{symmetry}} = \frac{t_{\text{stance}}}{2} \hat{v}_g + g_v \left( \hat{v}_g - v_g^{\text{cmd}} \right), \quad (28)$$

$$p_{\text{centrifugal}} = \frac{h}{2g} \dot{p} \times \omega, \quad (29)$$

where  $t_{\text{stance}}$  is the duration of the contact phase,  $\dot{p}$  and  $\omega$  are the linear and angular velocities of the robot at time step  $k$ ,  $g_v$  is the velocity feedback gain, and  $v_g^{\text{cmd}}$  is the stepping velocity command.

Eq. (28) allows the controller to execute a gait without the influence of driving with wheels, and (29) applies a centrifugal foot offset that accounts for the total robot velocity (driving and walking together), resulting in a well-balanced robot when driving with wheels and simultaneously turning.

Since the wheels are symmetric, we assume constant inertia for the rotating wheels and update the dynamic model of the robot by including the mass and inertia of the wheels in the lower leg linkage and relying on the WBC to correctly compute the joint control commands.

### B. Wheel Controller

While the robot is driving, the feet in the contact phase may not follow the desired rolling velocity due to interference or inadequate control along the rolling direction of the wheels, resulting in foot errors. We add a corrective velocity along the rolling direction to each foot during the contact phase, which reduces the foot position error and thus increases the control performance. The corrective velocity is defined for the  $i$ -th foot as follows:

$$\dot{p}_{\text{corrective}}^i = [k_p \ 0 \ 0]^T \odot p_{\text{error}}^i, \quad (30)$$

where  $k_p$  is the configured p-gain for the rolling direction,  $\odot$  is the element-wise multiplication operator, and  $p_{\text{error}}^i$  is

the filtered position error between the current and desired foot positions.

The wheel joints are controlled by a DI controller that follows the input velocities. However, the low resolution of the encoders (84 pulses per revolution) does not allow perfect velocity control, resulting in unwanted wheel motion and reducing the effectiveness of the controller. We solve this problem by converting the reaction forces from the WBC to the wheel joint torque  $\tau$  and passing them as feed-forward torque commands to the joint controller, which makes the wheels work against the reaction forces. The  $i$ -th wheel angular velocity  $\dot{q}$  and torque  $\tau$  are given by:

$$\tau = [r_{\text{eff}}^i \ 0 \ 0]^T \cdot \mathbf{R} f^i, \quad (31)$$

$$\dot{q} = \left[ \frac{1}{r_{\text{eff}}^i} \ 0 \ 0 \right]^T \cdot \mathbf{R} \left( \dot{p}_{\text{cmd}}^i + \dot{p}_{\text{corrective}}^i \right), \quad (32)$$

where  $r_{\text{eff}}$  is the effective radius of the wheel in (4),  $\mathbf{R}$  is a rotation matrix for conversion from global to the body reference frame,  $f^i$  is the current reaction force from WBC, and  $\dot{p}_{\text{cmd}}^i$  is the commanded foot velocity from the footstep planner [19].

Since the wheel joint velocities are set according to the foot velocities in (32), the tracking of swings along the rolling direction is greatly improved. If the wheel collides with a surface during the swing, it will continue to turn and help the foot bypass small height differences without additional swings.

## VI. EXPERIMENTAL RESULTS

We extended the Mini Cheetah's hardware to a hybrid quadruped by designing and fabricating relatively small wheels and enhanced shank links. These wheels can generate a maximum torque of 2.1 Nm and a speed of 2150 rpm, theoretically allowing the robot to reach up to 40 km/h. We use ODrive<sup>2</sup> BLDC drivers with custom firmware to control the speed and torque of the wheels with a low-resolution encoder (Hall sensor with 84 pulses per revolution). The modifications increase the weight of each leg by 0.39 kg, bringing the total weight of the extended robot to 12.5 kg (from its original weight of 10 kg). We transferred the open-source software version of the above robot to the ROS ecosystem and employed MuJoCo as a multibody simulator [23] that can accurately simulate the contact dynamics and rolling friction of the additional wheels, as well as the joints and corresponding rotor dynamics.

We intend to use the resulting system to study hybrid locomotion and associated control strategies. It will also serve as a research platform for other desired areas of hybrid mobile robotics, including navigation, path planning in complex terrain, and the application of learning-based control algorithms. The video showing the evaluation and performance of the real and simulated robot is publicly available<sup>3</sup>.

Hybrid driving is achieved by the robot exhibiting a natural driving style and stepping. The controller steers the robot along

<sup>2</sup><https://odriverobotics.com>

<sup>3</sup><https://www.ais.uni-bonn.de/~hosseini/se-dsq>

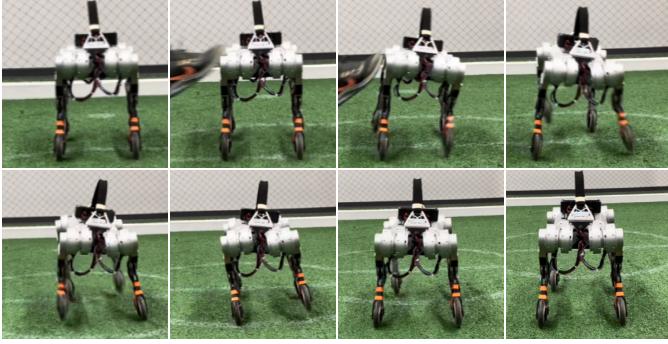


Fig. 4. Hybrid driving-stepping performance. This figure shows the robot performing forward hybrid driving-stepping with a trot gait (the robot is held in the center of the image and the background shifts backward). When the robot is disturbed, it performs side steps to balance the robot and maintain its current driving state.

the rolling direction of the wheels and minimizes the length of swings along the  $x$  axis in the local frame. Our modified robot achieved a velocity of 20 km/h while driving with wheels. Fig. 4 demonstrates the hybrid driving-stepping, which allows the robot's agile stepping to take control of the perturbed robot, maintain its balance, and perform corrective steps to follow the commanded track as the robot continues driving.

The tracking accuracy of the state estimator is demonstrated in Fig. 5, which compares our contribution to the state estimator with previous work by Katz et al. [18]. We set the robot in the simulator to walk forward and turn left at the same time, and compared the true value received from the simulator with the estimated values for the robot's position and velocity. The previous method ignores the shape of the end effector assuming that its size is negligible. However, the rubber ball at the end of each leg has a radius of 2.5 cm, which is considerable compared to the 19 cm length of the shank link. The errors in the position and velocity of the legs resulting from the above assumption accumulate over time and reduce the accuracy of the estimate. The above errors become even more pronounced for wheels with a radius of 5 cm. Our method accurately formulates the position and velocity of the ground contact point, taking into account the shape of the wheel and the effect of other joint velocities, and the angular velocity of the robot on the velocity of the contact point, resulting in an accurate position and velocity estimates, as shown in Fig. 5.

We evaluated the effect of the trust formulation in Eq. (11) in a scenario where the robot places one or two legs on an obstacle with a height of 8 cm. Our approach successfully captures the difference and applies the distrust gains to modify the KF noise and process covariance matrices, resulting in a near-perfect estimate of the robot height. However, previous work has failed to find a suitable corresponding estimation. Fig. 6 shows this evaluation. The high oscillation range in the lower-left graph for the estimated height is the result of the trotting gait of the robot, where the front right leg periodically steps on the obstacle.

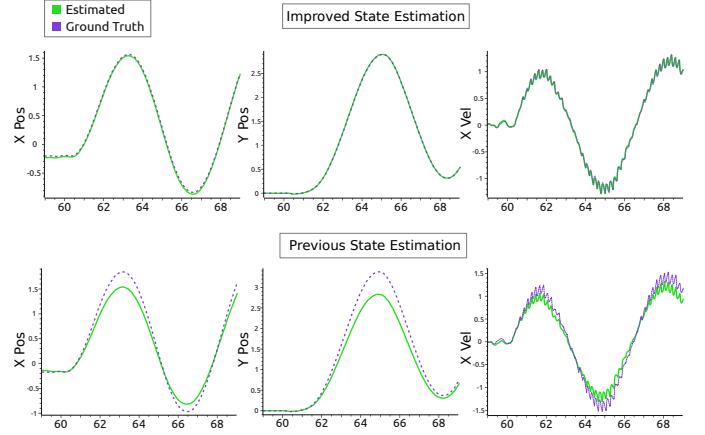


Fig. 5. State estimator evaluation graph. We compare the accuracy of state estimation in tracking with the true values (in the simulator) for our method enabled in the top row and the published version [18] for the Mini Cheetah with our method disabled in the bottom row. In this experiment, the robot walks forward and turns left at the same time. The shape of the end effector needs to be modeled, but in previous work, the center of the rubber ball was assumed to be the contact point and its shape was ignored. With our method, the exact ground contact point between the end effector and the ground is calculated, and the effect of the angular velocity of the body and the velocity of other joints on the velocity of the ground contact point is accurately calculated to achieve accurate position and velocity tracking for our robot.

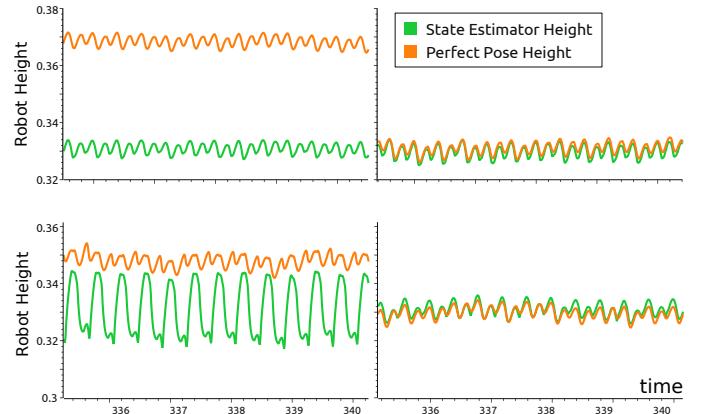


Fig. 6. Estimated height comparison. The green and orange graphs show the estimated and actual heights of the robot, respectively. The graphs in the top row refer to the situation where the robot steps with both front feet on an obstacle of height 0.08 m. The graphs in the left column show the state estimator output without applying the distrust gain formulation given in (11), and in the right column with applying the distrust gain. With our approach, the height of the robot remains unchanged even when stepping on the height difference.

We evaluated the effect of the wheel controller on the system, shown in Fig. 7. In this test, the robot executes a bounding gait, and the velocity of the front-right wheel is recorded over time. After time 307.6 s, the feed-forward torques generated in (31) are applied to the wheel and the wheels work against the reaction force along the  $x$  axis to reduce the unnecessary rotations, which results in more stable contact locations along the wheels' rolling direction.

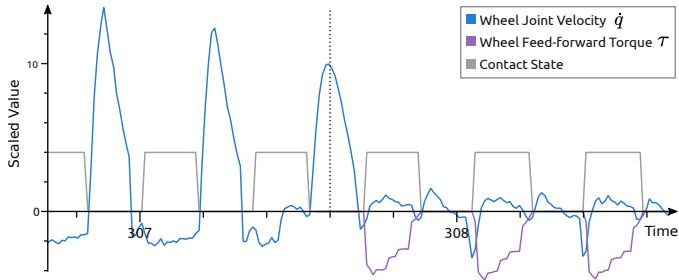


Fig. 7. The feedback effect of the wheel reaction force. This graph shows the velocity of the (front-right) wheel joint over time as the robot performs a bounding gait. Initially, the wheel is not receiving any feed-forward torque. After time 307.6 (the vertical dotted line), the wheel receives the torque generated in (31) to counteract the reaction forces applied by the controller, which results in fewer unwanted wheel joint rotations.

## VII. CONCLUSION

We incorporated wheels into the kinematic model and state estimator of the quadruped Mini Cheetah robot that we equipped with actively driven wheels, which enabled efficient and versatile hybrid driving-stepping locomotion. By maintaining a high travel speed with the help of wheels, the robot achieved an increased average locomotion speed. The effect of the end effector's large geometry was accurately captured to define precise contact points and comprehensive contact velocities. We formulated a KF for state estimation that includes the robot's position and velocity, as well as the contribution of driving with the wheels into the estimated state, which allowed us to use the control framework of the Mini Cheetah robot with minor modifications.

In the future, we will update the formulation of the MPC and the WBC to accurately perform simultaneous stepping and driving with the wheels. We will also reduce the robot's transportation cost by forcing the robot to purely drive and perform steps only to respond to disturbances or in response to rough terrain.

## ACKNOWLEDGMENTS

The authors thank Michael Schreiber for fabricating and assembling customized actuated wheels and for his assistance with experimental implementations, and AIS<sup>4</sup> Lab for the ROS packages used for this project.

## REFERENCES

- [1] M. Bjelonic, D. Bellicoso, Y. Viragh, D. Sako, F. Tresoldi, F. Jenelten, and M. Hutter, "Keep rollin' – Whole-body motion control and planning for wheeled quadrupedal robots," *IEEE Robotics and Automation Letters (RA-L)*, vol. 4, no. 2, pp. 2116–2123, 01 2019.
- [2] T. Klamt, D. Rodriguez, L. Bacchelliere, X. Chen, D. Chiaradia, T. Cichon, M. Gabardi, P. Guria, K. Holmquist, M. Kamedula, H. Karaoguz, N. Kashiri, A. Laurenzi, C. Lenz, D. Leonardis, E. Mingo, L. Muratore, D. Pavlichenko, F. Porcini, and S. Behnke, "Flexible disaster response of tomorrow: Final presentation and evaluation of the CENTAURO system," *IEEE Robotics and Automation Magazine*, vol. 26, no. 4, pp. 59–72, 10 2019.
- [3] T. Klamt and S. Behnke, "Planning hybrid driving-stepping locomotion on multiple levels of abstraction," in *International Conference on Robotics and Automation (ICRA)*, 2018.
- [4] G. Bellegarda and K. Byl, "Trajectory optimization for a wheel-legged system for dynamic maneuvers that allow for wheel slip," in *IEEE Conference on Decision and Control (CDC)*, 12 2019.
- [5] M. Geilinger, R. Poranne, R. Desai, B. Thomaszewski, and S. Coros, "Skaterbots: Optimization-based design and motion synthesis for robotic creatures with legs and wheels," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, p. 160, 07 2018.
- [6] Y. de Viragh, M. Bjelonic, C. D. Bellicoso, F. Jenelten, and M. Hutter, "Trajectory optimization for wheeled-legged quadrupedal robots using linearized ZMP constraints," *IEEE Robotics and Automation Letters (RA-L)*, vol. 4, no. 2, pp. 1633–1640, 2019.
- [7] M. Bjelonic, P. K. Sankar, C. D. Bellicoso, H. Vallery, and M. Hutter, "Rolling in the deep – Hybrid locomotion for wheeled-legged robots using online trajectory optimization," *IEEE Robotics and Automation Letters (RA-L)*, vol. 5, no. 2, pp. 3626–3633, 2020.
- [8] J. Rebula, P. Neuhaus, B. Bonnlander, M. Johnson, and J. Pratt, "A controller for the LittleDog quadruped walking on rough terrain," in *IEEE International Conference on Robotics and Automation (ICRA)*, 05 2007.
- [9] M. Bjelonic, R. Grandia, O. Harley, C. Galliard, S. Zimmermann, and M. Hutter, "Whole-body MPC and online gait sequence generation for wheeled-legged robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 7 2021.
- [10] V. Klemm, A. Morra, C. Salzmann, F. Tschopp, K. Bodie, L. Gulich, N. Küng, D. Mannhart, C. Pfister, M. Vierneisel, F. Weber, R. Deuber, and R. Siegwart, "Ascento: A two-wheeled jumping robot," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 7515–7521.
- [11] J. Cobano, J. Estremera, and P. Gonzalez-de Santos, "Location of legged robots in outdoor environments," *Robotics and Autonomous Systems*, vol. 56, no. 9, pp. 751–761, 09 2008.
- [12] F.-l. Tong and M. Q.-H. Meng, "Localization for legged robot with single low-resolution camera using genetic algorithm," in *IEEE International Conference on Integration Technology (ICIT)*, 2007.
- [13] S. P. N. Singh, P. J. Csonka, and K. J. Waldron, "Optical flow aided motion estimation for legged locomotion," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2006.
- [14] P.-C. Lin, H. Komsuoglu, and D. Koditschek, "Sensor data fusion for body state estimation in a hexapod robot with dynamical gaits," *IEEE Transactions on Robotics (T-RO)*, vol. 22, no. 5, pp. 932–943, 2006.
- [15] M. Blösch, M. Hutter, M. Höpflinger, S. Leutenegger, C. Gehring, C. Remy, and R. Siegwart, "State estimation for legged robots - consistent fusion of leg kinematics and IMU," in *Robotics: Science and Systems (RSS)*, 2012.
- [16] G. Bledt, P. Wensing, S. Ingersoll, and S. Kim, "Contact model fusion for event-based locomotion in unstructured terrains," in *International Conference on Robotics and Automation (ICRA)*, 05 2018.
- [17] M. Camurri, M. Fallon, S. Bazeille, A. Radulescu, V. Barasuol, D. Caldwell, and C. Semini, "Probabilistic contact estimation and impact detection for state estimation of quadruped robots," *IEEE Robotics and Automation Letters (RA-L)*, vol. PP, 01 2017.
- [18] B. Katz, J. D. Carlo, and S. Kim, "Mini Cheetah: A platform for pushing the limits of dynamic quadruped control," *International Conference on Robotics and Automation (ICRA)*, 05 2019.
- [19] D. Kim, J. Carlo, B. Katz, G. Bledt, and S. Kim, "Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control," *CoRR*, vol. abs/1909.06586, 09 2019.
- [20] J. Carlo, P. Wensing, B. Katz, G. Bledt, and S. Kim, "Dynamic locomotion in the MIT Cheetah 3 through convex model-predictive control," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 10 2018.
- [21] L. Andrews, *Special Functions of Mathematics for Engineers*. SPIE Optical Engineering Press, 1998.
- [22] J. Ma, M. Bajracharya, S. Susca, L. Matthies, and M. Malchano, "Real-time pose estimation of a dynamic quadruped in GPS-denied environments for 24-hour operation," *International Journal of Robotics Research (IJRR)*, vol. 35, no. 6, pp. 631–653, 07 2015.
- [23] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033.

<sup>4</sup>Autonomous Intelligent Systems: <https://ais.uni-bonn.de>