2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Terrain-Adaptive Planning and Control of Complex Motions for Walking Excavators

Edo Jelavic, Yannick Berdou, Dominic Jud, Simon Kerscher and Marco Hutter

***Abstract*— This article presents a planning and control pipeline for legged-wheeled (hybrid) machines. It consists of a Trajectory Optimization based planner that computes references for end-effectors and joints. The references are tracked using a whole-body controller based on a hierarchical optimization approach. Our controller is capable of performing terrain adaptive whole-body control. Furthermore, it computes both torque and position/velocity references, depending on the actuator capabilities. We perform experiments on a Menzi Muck M545, a full size 31 Degrees of Freedom (DoF) walking excavator with five limbs: four wheeled legs and an arm. We show motions that require full-body coordination executed in realistic conditions. To the best of our knowledge, this is the first work that shows the execution of whole-body motions on a full size walking excavator, using all DoFs for locomotion.**

I. INTRODUCTION

Robotic platforms are becoming more sophisticated and required to operate in increasingly challenging environments. One example are hybrid legged-wheeled[1] systems that offer versatility in a variety of terrain. They combine the agility of legged systems with the speed of wheeled systems in flat terrain. However, planning and control algorithms suitable for hybrid systems remain somewhat underdeveloped compared to their purely legged or wheeled counterparts. To this end, we present a motion planner capable of producing robust, terrain-aware motion plans, suitable for execution on real platforms. Planning with entirely accurate terrain information is not possible, due to the imperfect sensor data and spatial map discretization. Furthermore, too many details and exact geometry (e.g., vegetation roughness, mud) can still render Trajectory Optimization (TO) very hard. Hence, it is desirable to be able to plan on simplified geometry and retain robustness at the execution time. We tackle the robustness problem by designing the terrain-adaptive controller suitable for machines that are not fully torque-controllable. Our framework has been validated on a hydraulic walking excavator [1] in both simulations and experimentally[2].

A. Related Work

Motion planning and control for legged systems, and for hybrid systems in particular, are nowadays typically done using TO based approaches due to their favorable scaling with the system dimension. Often, planning algorithms run in a receding horizon fashion to maximize robustness, which is beneficial for deployment on the real hardware. Such an architecture has been able to produce various motions for legged robots with point feet (e.g. [2], [3], [4]). Despite similarities with legged systems, hybrid systems have received less attention from the robotic community. So far, algorithms for hybrid systems have been developed mainly by the aerospace community. Typically, in extraterrestrial missions, a robot is required to be statically stable while traversing over uneven terrain; an example of such a work can be found in [5], [6] or [7]. However, these systems are usually teleoperated in purely reactive mode. There is no planning or whole-body control deployed on the robot.

Recently, several authors have investigated the application of TO to whole-body planning and control for hybrid robots. In [8], [9], *ANYmal* solves an Nonlinear Program (NLP) with box constraints on the end-effector position over a prediction horizon of $0.85\,\mathrm{s}$ - $2\,\mathrm{s}$. While motions on the real system look quite impressive, the planning update rates achieved (about $200\,\mathrm{Hz}$) are chiefly thanks to the short prediction

Fig. 1. ***Top:*** A walking excavator has five limbs; an arm, and four legs with a wheel at the end of each leg. They usually operate in muddy and slippery conditions, which requires motion plans with large stability margin. ***Bottom:*** HEAP balancing on three legs while performing a stepping maneuver. A video demonstrating all the motions accompanies this submission.

This work was supported in part by the Swiss National Science Foundation (SNF), the National Centre of Competence in Research Digital Fabrication and CTI project 18213.2

All authors are with the Robotic Systems Lab, ETH Zurich, 8092 Zurich, Switzerland, contact: edo.jelavic@mavt.ethz.ch

[1]Legged-wheeled systems are referred to as *hybrid* systems in further text

[2]https://youtu.be/7QU1UmnMy1Q

horizon and linear inequalities (box constraint) that can be used to enforce end-effector range of motion. In contrast, the kinematics of a walking excavator (absence of the knee joint) does not permit such a simplification, and one has to solve for the joint positions as well (see [10]). This renders the NLP considerably harder, which then results in a decreased Real-time (RT) factor. In [11], *ANYmal* performs hybrid stepping and driving motions generated by solving a Quadratic program (QP) with a prediction horizon of $2\,\mathrm{s}$. The authors introduce a simplified Zero Moment Point (ZMP) criterion that allows for planning at $50\,\mathrm{Hz}$ while the torque computation runs at $400\,\mathrm{Hz}$. Similar to [11], *Robosimian* [12] shows impressive dynamic driving motions computed for a $2\,\mathrm{s}$ planning horizon. However, both robots perform motions only in simulation and are typically deployed in less harsh environments compared to HEAP, see Fig. 1.

Momaro [13] and more recently, *Centauro* [14] both perform stepping and driving maneuvers over uneven terrain. Contrary to the approaches outlined above, *Momaro* and *Centauro* do not blindly locomote, but rather maintain a surroundings map that is used to decide when to drive and when to step. However, neither robot performs whole-body motion planning, and transitions between driving and motions are handcrafted. In contrast, our planning and control pipeline computes and executes whole-body plans, which is a more scalable approach for robots with many DoFs.

We propose an approach that decouples the problem into sequential offline motion planning and online tracking phases. Few previous works have approached the problem in this way. In [15], a plan involving dynamic motions is computed for a robot with point feet. The motion is tracked using a hierarchical Whole Body Controller (WBC) [16]. In [17], whole body plans are computed in a simplified 2D scenario and then extended to 3D with results only being shown in simulation. *Skaterbots* [18], use a general framework that allows for generation of different hybrid motions. While demonstrated motions are quite challenging and look smooth, they have only been demonstrated on small robots in laboratory conditions. In its present form, it remains an open question of how the planning and control algorithm would transfer to a large scale robot with a high degree of modeling and environment uncertainty.

Typically, whole-body control is associated with torque-controlled robots and there is rich literature covering the topic [19], [20], [2], [8], [9]. However, many existing machines do not feature actuators for high accuracy torque control (e.g. large friction, delays) or do not have all actuators torque-controllable. Furthermore, models of the system dynamics are only available with limited accuracy. In contrast, we develop a control framework that can handle a mixture of torque-controlled and position/velocity controlled DoFs.

B. Contribution

We present a planning and control pipeline for legged wheeled machines. In particular, we focus on large scale robots with many DoFs such as walking excavators. Our contributions can be summarized as follows: We extend the motion planner introduced in our previous work [10] for execution on the real hardware. Besides, we design a tracking controller for executing challenging whole-body motions on a real machine. All of the motions shown are computed and executed using the same pipeline, i.e., the same planner and the same controller without any changes. We use an extended terrain-adaptive whole-body controller based on Hierarchical Optimization (HO) framework [16]. The controller handles both torque and position/velocity controllable DoFs in a single structure. Lastly, our framework has been demonstrated on a full-scale hydraulically-actuated excavator in a realistic environment. We show that the proposed approach can execute challenging motions despite the mud, high actuator friction, and limited model accuracy available for HEAP. To the best of our knowledge, this is the first time whole-body motions have been shown on a full size walking excavator.

II. PLANNING

In our previous work [10], we introduced a collocation [21] based planner that solves an optimal control problem. It produces kinematically consistent plans while respecting the non-holonomic rolling constraint for the wheels. In this article our planner has been extended with additional constraints and analytical costs, to make plans executable on real hardware.

A. Notation

Before introducing any equations, we introduce the notation used. Legs are denoted with LF (Left Front), RF, LH, RH (Right Hind). The inertial frame is denoted with I and B denotes the floating base of the robot. The left subscript indicates the frame in which the quantity is expressed, e.g., ${}_I\boldsymbol{r}_{IB}$ denotes the position of the base with respect to the inertial frame expressed in the inertial frame. For rotation, we use quaternions or rotation matrices, where $\boldsymbol{R}_{IB}$ ($\boldsymbol{q}_{IB}$) is a rotation of the base with respect to the inertial frame. We use $\boldsymbol{v}$ for linear velocities; ${}_I\boldsymbol{v}_{ee}$ is a linear velocity of the end-effector expressed in the inertial frame. Angular velocity is denoted with $\boldsymbol{\omega}$, ${}_I\boldsymbol{\omega}_{IB}$ is the angular velocity of the base frame as seen from the inertial frame expressed in the inertial frame. Further examples using the same convention can be found in [22]. We denote desired quantities with the right superscript d, e.g., ${}_I\boldsymbol{v}_B^d$ is a desired base linear velocity, expressed in the inertial frame. Right subscript denotes a vector component, e.g. $\boldsymbol{v}_x$ is the x component of vector $\boldsymbol{v}$. Finally, the generalized coordinate vector $\boldsymbol{q}$ and the generalized velocity vector $\boldsymbol{u}$ are given with:

$$\boldsymbol{q} = \begin{bmatrix} {}_I\boldsymbol{r}_{IB} \\ \boldsymbol{q}_{IB} \\ \boldsymbol{q}_j \end{bmatrix} \in SE(3) \times \mathbb{R}^{n_j}, \boldsymbol{u} = \begin{bmatrix} {}_I\boldsymbol{v}_B \\ {}_B\boldsymbol{\omega}_{IB} \\ \dot{\boldsymbol{q}}_j \end{bmatrix} \in \mathbb{R}^{n_u} \quad (1)$$

where $n_u = 6 + n_j$ and n_j is the number of joints. $\boldsymbol{q}_j \in \mathbb{R}^{n_j}$ is the vector of joint coordinates.

B. Extensions

The planner uses a set of constraints, introduced in our previous work [10]. We describe the newly added planner components with the equations given below.

Base motion constraint: Box constraint on the base roll angle encourages the optimization to primarily use the arm for balancing and not to tilt the base. Large roll angles θ of the base shrink the support polygon and could lead to catastrophic falls. In addition, a limit on the angular velocity of the base discourages fast motions.

$$|\theta| \leq \theta_{max}, \;\; {}_B\boldsymbol{\omega}_{IB} \leq \boldsymbol{\omega}_{max} \tag{2}$$

End-effector motion constraint: We do not allow fast movement of the limbs. Fast movement of the limbs may cause the controller to request too much actuation, rendering the tracking problem infeasible (e.g. not enough oil flow from the hydraulic pump). The second issue that the planner does not account for inertial forces when limbs are swinging. Hence, high velocities can lead to poor base tracking performance.

$$|\boldsymbol{v}_{EE}| \leq \boldsymbol{v}_{EE,max} \tag{3}$$

Nominal posture terminal cost: The planner is encouraged to finish the motion in the nominal posture. This makes the planning the next maneuver easier since the nominal posture is stable and joint positions are far away from the bounds. We denote cost functions or cost terms with J.

$$J_{nominal} = (\boldsymbol{q}_T - \boldsymbol{q}_n)^T \boldsymbol{S} (\boldsymbol{q}_T - \boldsymbol{q}_n) \tag{4}$$

where $\boldsymbol{S}$ is the cost matrix and $\boldsymbol{q}_T$ and $\boldsymbol{q}_n$ are terminal and nominal joint positions, respectively.

Wheel velocity difference cost: Since HEAP cannot control each wheel's velocity individually, we encourage the planner to find the solutions where all wheels have similar velocities. In this way, it easier for controller to track the motion and the amount of slip is reduced.

$$J_{wheel,i,j} = \sum_{t=0}^{T} \|v_i^2 - v_j^2\|_2^2 \tag{5}$$

where v_i is the linear velocity magnitude of the i^{th} wheel and $(i,j) \in \{(LF,RF),(LH,RH)\}$. The effect of the cost on the wheel magnitude difference is illustrated in Figure 2.

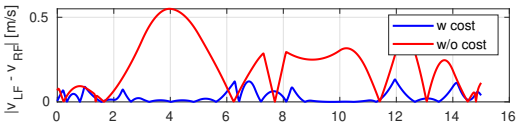


Fig. 2. Difference in velocity magnitude between LF and RF wheel during the sideways driving maneuver. The base command was to drive left. It can be seen that the introduction of the cost reduces speed difference by an order of magnitude over the trajectory duration. Motions that require less velocity difference are easier for the controller to track.

Similar to our findings, [23] also reports introducing costs in the optimization improves the quality of the planned motion. Unfortunately, improved quality comes with more computation time (about 2x decrease of the RT factor).

III. CONTROL

The proposed control system is based on HO [16]. HO controller essentially implements an inverse dynamics algorithm [24]. Such a control scheme allows HEAP to be force controlled. Despite operating in purely tracking mode, force control can be beneficial since the machine can adapt to unperceived terrain changes (e.g., bumps or puddles not captured in the planning phase). The schematic of the proposed whole-body controller is shown in Fig. 3. Fig. 4 shows the excavator and all the DoFs.

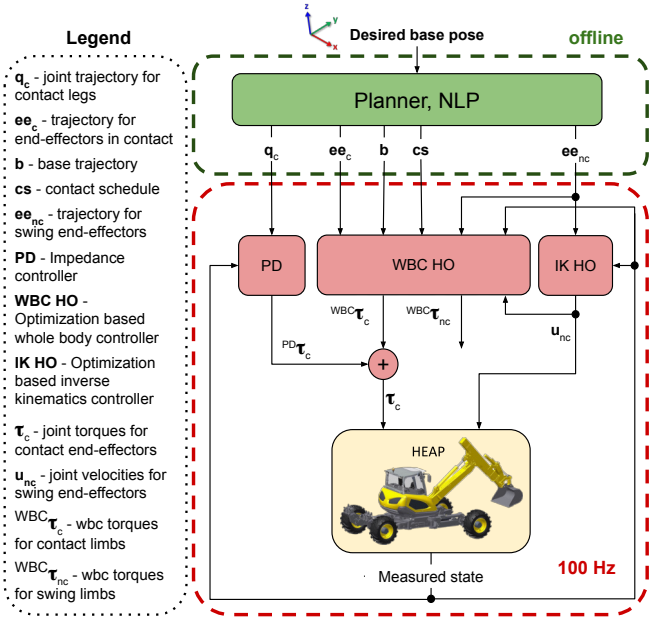


Fig. 3. Structure of the proposed planning and control system. The planner (green color) sends the desired references to the controller (red color). The WBC computes desired torques for both contact and swing end-effectors. In addition to the WBC, we run an impedance controller for the stance end-effectors and the IK controller for the swing end-effectors. Impedance controller torques are added to the WBC torques and sent to the contact limbs. Velocities computed from IK are sent to the low-level controller. Note how the WBC torques computed for swing limbs ${}^{WBC}\boldsymbol{\tau}_{nc}$ are not used. While the planner runs offline, the proposed controller runs at 100 Hz. The limiting factor in achieved frequency is the bandwidth of the CAN bus.

A. Whole-Body Control

The WBC is the core controller of our framework. In addition, we extended it with two more controllers (see Fig. 3) in order to compute references for both dynamically (torque) controlled joints and kinematically (position/velocity) controlled joints. The control system receives operational space references from the planner and computes optimal generalized accelerations $\dot{\boldsymbol{u}}^*$ and contact forces $\boldsymbol{\lambda}^*$, i.e. the solution vector $\boldsymbol{x}^*$ looks like: $\boldsymbol{x}^* = [\dot{\boldsymbol{u}}^{*T} \boldsymbol{\lambda}^{*T}]^T \in \mathbb{R}^{n_u+3n_c}$, where n_c is the number of end-effectors that are in contact.

WBC solves a series of QP problems in a prioritized order. A solution of a QP with lower priority is computed in the nullspace of the QP with higher priority. For each QP, inequality constraints $\boldsymbol{A}_i\boldsymbol{x} \leq \boldsymbol{b}_i$ and equality constraints $\boldsymbol{C}_i\boldsymbol{x} = \boldsymbol{d}_i$ define a task T_i where index i denotes the task priority. Matrices $\boldsymbol{A}$ and $\boldsymbol{C}$, together with vectors $\boldsymbol{b}$ and $\boldsymbol{d}$ are task-dependent and have to be specified by the user. By adding tasks T_i and assigning their priorities in a meaningful way, one can shape the behavior of the robot (see [16]).

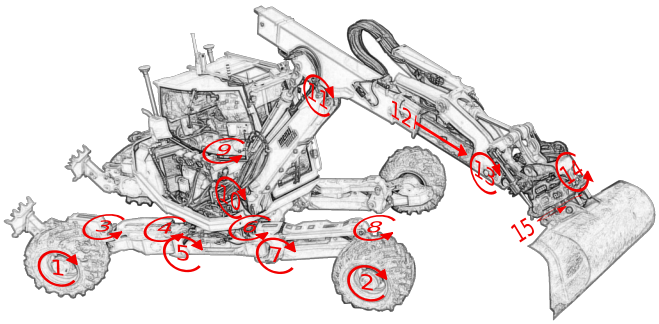


Fig. 4. Schematic of HEAP with all joints and their axes. Joint notations on RF and RH legs are omitted for the sake of clarity. They have the same DoF's as the LF and LH leg that are shown. Joints 1, 2, 9, 14 and 15 cannot be torque controlled. Joint 12 has a very high and nonlinear static friction which makes the torque control hard. The rest of the joints can be torque controlled.

B. Tasks

The tasks used in WBC with their respective priorities are specified in Table I. Before providing the formulation details, we explain how the priorities for the tracking tasks are chosen.

The base tracking task is split into sub-tasks with different priorities to reduce the number of DoFs in the optimization. Unconstrained optimization variables can have a detrimental effect on the quality of the motion (bang-bang solutions) and may produce unnecessary motions. Looking at the kinematics of the excavator in Fig. 4, it can be seen that roll, pitch and height of the base can be controlled using Flexion/Extension (FE) joints (joints number 5 and 7 in Fig. 4); other joints have a minor contribution. This holds even for the case when one leg is in the air. Hence, the base tracking task is split into terrain adaptive posture tracking (roll, pitch and height that are influenced by FE joints) and 2D pose tracking (x, y and yaw influenced by Abduction/Adduction (AA) and steer joints). Exploiting the hierarchical task setup to achieve posture adaptation has been reported in [19]. We exploit hierarchical task setup to realize the terrain adaptive behavior. Prioritizing the adaptive posture tracking tasks in the HO over the 2D pose tracking tasks achieves the desired goal and the machine adapts to the terrain (see Section IV-A). In addition, the HO is more constrained which helps finding steadier motions.

Wheel and base 2D pose (x, y and yaw) are most influenced by the same set of joints. These are steering (numbers 1 and 2 in Fig. 4) and AA joints (numbers 4, 5, 6 and 8). For this reason, the base and wheel 2D pose tracking tasks should have the same priority. Otherwise, the DoFs may be used to satisfy one task perfectly, and then the lower priority task could be rendered infeasible. It is worth observing that given the fixed positions of the wheels, the position of the base is largely determined. There is little space to move the base without violating the wheel rolling constraint, which means that accurate wheel tracking implies accurate base tracking. For this reason, we tune the wheel tracking gains to be higher than the base for the 2D pose tracking tasks.

TABLE I
WBC TASK SETUP. SMALLER NUMBER INDICATES HIGHER PRIORITY.

Priority	Task
1	Floating base equations of motion
1	Joint limits
1	Friction cone
1	Wheel rolling constraint
2	Base orientation (*pitch, roll*)
2	Base translation (*height*)
3	Base orientation (*yaw*)
3	Base translation (*lateral, longitudinal*)
3	Ground leg orientation (*yaw*)
3	Ground leg translation (*lateral, longitudinal*)
4	Swing limb orientation
4	Swing limb translation
5	Joint acceleration minimization
5	Contact force minimization

Floating base equations of motion: We require that computed solution $\boldsymbol{x}^*$ satisfies rigid body Equations of Motion (EoM). For details on the EoM see [8], [24] or [25].

Joint limits: This task implements box constraints on position, velocity and torque joint limits, i.e.

$$\boldsymbol{q}_{j,lower} \leq \frac{1}{2}\ddot{\boldsymbol{q}}_j \Delta t^2 + \dot{\boldsymbol{q}}_{j,t}\Delta t + \boldsymbol{q}_{j,t} \leq \boldsymbol{q}_{j,upper} \quad (6)$$

$$\dot{\boldsymbol{q}}_{j,lower} \leq \ddot{\boldsymbol{q}}_j \Delta t + \dot{\boldsymbol{q}}_{j,t} \leq \dot{\boldsymbol{q}}_{j,upper} \quad (7)$$

$$\boldsymbol{\tau}_{j,lower} \leq \boldsymbol{\tau} \leq \boldsymbol{\tau}_{upper} \quad (8)$$

Since the optimization has accelerations as decision variables, we need to integrate one time step to enforce the position and bounds limits. Δt denotes time step, $\boldsymbol{q}_{j,t}$ are the joint positions and $\dot{\boldsymbol{q}}_{j,t}$ joint velocities at time step t, respectively.

Friction Cone: The friction cone is approximated by a friction pyramid (to have linear constraints). Details about the implementation of the constraint can be found in [26].

Wheel rolling constraint: Wheel rolling constraint ensures that there is no lateral movement of the wheel, i.e., the wheel can only move in the longitudinal direction. The derivation of the rolling constraint can be found in [8].

Base tracking: We use a standard formulation that be commonly found for floating base robots, e.g., [26], [16]. The equations are given below:

$$[\boldsymbol{J}_{B,r} \;\; \boldsymbol{0}_{3\times 3n_c}]\boldsymbol{x} = \boldsymbol{R}_{WI}\big({}_I\ddot{\boldsymbol{r}}^d_{IB} + \boldsymbol{K}_p({}_I\boldsymbol{r}^d_{IB} - {}_I\,\boldsymbol{r}_{IB}) + \boldsymbol{K}_d({}_I\dot{\boldsymbol{r}}^d_{IB} - {}_I\,\dot{\boldsymbol{r}}_{IB})\big) - \dot{\boldsymbol{J}}_{B,r}\boldsymbol{u} \quad (9)$$

$$[\boldsymbol{J}_{B,q} \;\; \boldsymbol{0}_{3\times 3n_c}]\boldsymbol{x} = \boldsymbol{R}_{WI}\big({}_I\boldsymbol{\alpha}^d_{IB} + \boldsymbol{K}_p(\boldsymbol{q}^d_{BI} \boxminus \boldsymbol{q}_{BI}) + \boldsymbol{K}_d({}_I\boldsymbol{\omega}^d_{IB} - {}_I\,\boldsymbol{\omega}_{IB})\big) - \dot{\boldsymbol{J}}_{B,q}\boldsymbol{u} \quad (10)$$

where $\boldsymbol{J}_{B,r}$ is a translational Jacobian and $\boldsymbol{J}_{B,q}$ si a rotational Jacobian. The desired angular acceleration of the base is denoted with ${}_I\boldsymbol{\alpha}^d_{IB}$. Operator $\boxminus : SO(3) \to \mathbb{R}^3$ is defined in [27].

Ground leg translation: With steerable wheels, one can control both lateral and longitudinal translation of the wheel. We close the control loop over the longitudinal error (distance to W^d along the e_x axis in Fig. 5a) and lateral error

(distance to W^d along the e_y in the same Figure) resulting in the equality constraint:

$$\pi_j\big([\boldsymbol{J}_{W,r}\ \ \boldsymbol{0}_{3\times 3n_c}]\boldsymbol{x}\big) = \pi_j\big(\boldsymbol{R}_{WI}\big({}_I\ddot{\boldsymbol{r}}^d_{IW} + \boldsymbol{K}_p({}_I\boldsymbol{r}^d_{IW} - {}_I\,\boldsymbol{r}_{IW}) + \boldsymbol{K}_d({}_I\dot{\boldsymbol{r}}^d_{IW} - {}_I\,\dot{\boldsymbol{r}}_{IW})\big) - \dot{\boldsymbol{J}}_{W,r}\boldsymbol{u}\big) \quad (11)$$

where $\boldsymbol{J}_{W,r}$ is a translational Jacobian in wheel frame W (see Figure 5b). The operator $\pi_j(\cdot), j \in \{x, y\}$ is the projection onto the wheel longitudinal axis e_x (red color in Figure 5a) and lateral axis e_y (green color).

Ground leg orientation: Since wheels introduce non-holonomic constraints, one has to control the yaw angle of the wheel to track the position. Equalities describing the ground leg orientation task are given by:

$$\pi_z\big([\boldsymbol{J}_{W,q}\ \ \boldsymbol{0}_{3\times 3n_c}]\boldsymbol{x}\big) = \pi_z\big(\boldsymbol{K}_p\boldsymbol{R}_{WI}(\boldsymbol{q}^d_{WI} \boxminus \boldsymbol{q}_{WI}) + \boldsymbol{K}_d\boldsymbol{R}_{WI}({}_I\boldsymbol{\omega}^d_{IW} - {}_I\,\boldsymbol{\omega}_{IW}) + \boldsymbol{K}_{p,e}(\dot{q}^{ref}_{j,w})\boldsymbol{e} - \dot{\boldsymbol{J}}_{W,q}\boldsymbol{u}\big) \quad (12)$$

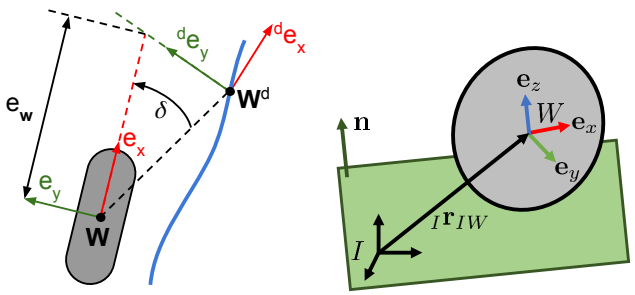
$$\boldsymbol{e} = [0\ 0\ sign(\dot{q}^{ref}_{j,w})\delta]^T \quad (13)$$

$$\delta = atan2({}_W\boldsymbol{r}_{WW^d,x},\ {}_W\boldsymbol{r}_{WW^d,y}) \quad (14)$$

The matrices $\boldsymbol{K}_P, \boldsymbol{K}_{p,e}$, denote proportional gains, and $\boldsymbol{K}_D$, derivative gains. All the matrices are diagonal and belong to a set of positive definite matrices $\mathbb{S}^3$. The orientational Jacobian expressed in the wheel frame W is given with $J_{W,q}$. Wheel joint velocity reference (joints number 1 and 2 in Figure 4) is denoted with $\dot{q}^{ref}_{j,w}$. We give more details about $\dot{q}^{ref}_{j,w}$ computation in Section III-E. Besides tracking the orientation given from the plan, we introduce an additional feedback term $\boldsymbol{K}_{p,e}\boldsymbol{e}$ over the desired position in the Equation 12. This term drives the wheel directly to the desired position (origin of $\boldsymbol{W}^d$ in Fig. 5a). The $sign(\cdot)$ function in Equation (13) is used to correctly handle the reverse driving. Such a control law is similar to the pure pursuit tracking algorithm [28], with the main difference that we do not impose moving along a circular arc towards the goal point. The pure pursuit algorithm computes a circular arc trajectory since it has been designed for wheeled robots with front steering when both front and rear wheel pairs have to point to the same center of rotation. However, because of the additional degree of freedom in the leg (the AA joints), HEAP's wheels do not have to point to the same center of rotation. The gain $\boldsymbol{K}_{p,e}(\dot{q}^{ref}_{j,w})$ is an adaptive gain on the $[q_{w,min}, q_{w,max}]$ interval. Such a practice is common in the vehicle control community [29].

Swing limb tracking: For swing limb motino tracking, we use the same formulation as for the base (see Equatinos (9) and (10). The base Jacobians J_B are replaced with limb end-effectors Jacobians $J_i, i \in \{W, A\}$, depending on whether we control the wheel frame W or the arm end-effector frame A. Desired and measured coordinates for the base are replaced with desired and measured coordinates for the limb end-effector.

Joint acceleration and contact force minimization: The task minimizes the joint accelerations by setting $\ddot{\boldsymbol{q}}_j = \boldsymbol{0}$. It also tries to minimize the contact forces to get rid of internal forces acting on the robot [26].



(a) Wheel top view (b) Wheel frame

Fig. 5. ***Left:*** Top view of a wheel tracking the desired trajectory (shown in blue color). The desired position and orientation of the wheel center at time t are denoted with a coordinate system $\boldsymbol{W}^d$. Coordinate system $\boldsymbol{W}$ denotes the current position and orientation. Wheel axes are shown in red and green colors. The yaw error δ from Equation (14) is denoted with δ in the Figure. Position error along the rolling direction is shown with a full black line and denoted with symbol e_w. ***Right:*** The wheel frame W is defined to lie on the origin of the wheel. The axis $\boldsymbol{e}_y$ is defined as the wheels rotation axis, and the axis $\boldsymbol{e}_x$ is defined as the direction perpendicular to $\boldsymbol{e}_y$ and the terrain normal $\boldsymbol{n}$. The wheel moves in the direction of the axis $\boldsymbol{e}_x$. Note that the wheel frame W does not rotate with the wheel.

C. Impedance control

For the end-effectors that are in contact, we add torques computed using the Proportional-Derivative (PD) controller ${}^{PD}\boldsymbol{\tau}_c = \boldsymbol{K}_p(\boldsymbol{q}^d_j - \boldsymbol{q}_j) + \boldsymbol{K}_d(\dot{\boldsymbol{q}}^d_j - \dot{\boldsymbol{q}}_j)$. $\boldsymbol{q}^d_j$ and $\dot{\boldsymbol{q}}^d_j$ are known from the planner (denoted $\boldsymbol{q}_c$ in Fig. 3). The PD action helps to combat the friction effects and model inaccuracies, which become prominent at low torques. E.g., if the arm is extended far in the front, contact forces at the hind legs become low, and there is less torque in the joints. At low torques, Coulomb friction can cause unwanted chattering. This issue is ameliorated by adding some torque computed from the PD controller. The net effect is that the legs are "stiffened up", i.e., the actuators operate in a higher frequency range where the friction effects are not as strong. Instead of PD control, a lead-lag controller could also be used.

D. Inverse Kinematics Control

The IK controller is also based on HO and computes the desired joint velocities $\dot{\boldsymbol{q}}_{nc}$. To make the swing limbs (including the arm) accurately follow a planned motion trajectory, we use joint velocity control as this can much better compensate for modeling errors and highly-dominant friction effects. Furthermore, some joints on our machine are simply not torque-controllable. One way to make the WBC and IK work together is to let the IK compute desired joint velocities and then add them as a constraint in the WBC. This constraint is merely another task T_i in the WBC whose priority should be lower than the joint limits task. The task can be formulated as an equality constraint:

$$\ddot{\boldsymbol{q}}_j\Delta t + \dot{\boldsymbol{q}}_{j,t} = \dot{\boldsymbol{q}}_{j,IK} \quad (15)$$

Such a task should only be added for non-contact limbs. Incorporating the IK into the WBC in this way ensures that solution $\boldsymbol{x}^*$ satisfies the EoM (highest priority task in the WBC). However, Equation 15 can introduce unwanted noise in the WBC which is why we do not add such a task.

We merely add the swing limb tracking task in the WBC controller (see Table I). The torques computed for the swing limb cannot be sent to the actuators (some are not torque-controllable, furthermore joint friction is prominent without any load). However, by enforcing the swing limb tracking task, the WBC becomes aware of the swing limb motion and accounts for inertial effects on the base. As a consequence, the IK controller does not deteriorate the base tracking because the WBC has already accounted for the swing limb motion. In this way, the two controllers (WBC and IK) can work together without conflicts. In addition, the inertial effects are minimized in the planning phase since the motion planner does not request any dynamic motions. In Fig. 3, note how the torques ${}^{WBC}\boldsymbol{\tau}_{nc}$ for swing limbs computed by the WBC are neglected and velocities from IK are used for control. Validation of our approach is shown in Fig. 6.

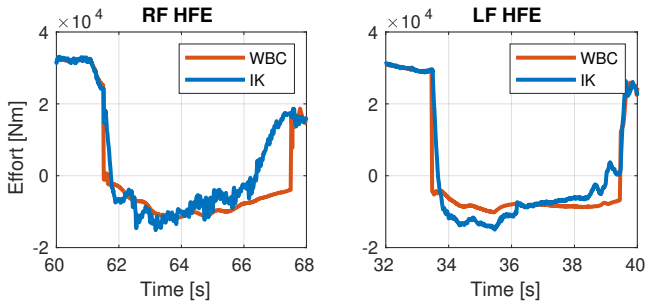


Fig. 6. Torques in LF and RF flexion joints during the swing phases while performing stepping experiment. It can be seen the torques for swing limb joints computed by the WBC do not substantially differ from the actual torques when tracking the velocities from the IK controller.

E. Wheel Speed Control

The turning speed of the wheels cannot be measured directly since wheel actuators do not have encoders or any sensors. Therefore it is not possible to close the control loop over the wheel joint (see Fig. 4) velocities. To address this problem, we close the loop over the wheel position. Furthermore, the wheel joints are not torque controllable either. Nonetheless, we still let the WBC include them in the optimization problem as if they were torque controllable. The same reasoning as for the swing limbs applies (see Section III-D).

The reference wheel speed $\dot{q}^{ref}_{j,w}$ is calculated using Equation 16. K_{ff} denotes the feed-forward gain multiplying the desired speed obtained from the planner, and K_p and K_i are proportional and integral gain, respectively. The reference current for the valves, is then calculated as $i^{ref}_v = l(\dot{q}^{ref}_{j,w})$. Note that function l is unknown and hard to identify.

$$\dot{q}^{ref}_{j,w} = K_{ff}\dot{q}^{d}_{j,w} + K_p e_w + K_i \int_0^t e_w(t)dt \qquad (16)$$

Where position error e_w (see Fig. 5a) is calculated as the distance from the measured wheel position along its rolling direction $\boldsymbol{e}_x$ to the plane spanned by the vectors $\boldsymbol{e}^d_y$ and $\boldsymbol{e}^d_z$ of the desired wheel frame.

$$e_w = \frac{{}_I\boldsymbol{e}^d_x \cdot ({}_I\boldsymbol{r}^d_{IW} - {}_I\boldsymbol{r}_{IW})}{{}_I\boldsymbol{e}^d_x \cdot {}_I\boldsymbol{e}_x}, \qquad (17)$$

where ${}_I\mathbf{r}_{IW}$ is the position of the wheel origin in the world frame and ${}_I\mathbf{e}_x$ is the rolling direction of the wheel in the world frame.

IV. RESULTS AND DISCUSSION

We perform experiments on HEAP, a full-size 12-ton walking excavator. HEAP is equipped with 12 torque-controllable and 4 position/velocity controllable joints in the chassis. The arm has 4 torque-controllable joints and 3 position/velocity controllable joints . The wheel joints in the chassis (numbers 1 and 2 in Figure 4) are not torque controllable, together with the cabin turn, shovel roll, and shovel yaw (numbers 9, 14 and 15). The planner is run offline on a laptop with Intel Xeon E3-1535M v5 2.90GHz processor. It is implemented using Ipopt NLP solver [30] which implements the primal-dual barrier method. Ipopt is interfaced from C++ using Ifopt [31]. Both the planner and the controller use Rigid Body Dynamics Library (RBDL) for rigid body algorithms [32]. For state estimation we rely on a two-state information filter [33] that fuses Real-Time Kinematic (RTK) Global Navigation Satellite System (GNSS) measurements with Inertial Measurement Unit (IMU) measurements from chassis and the cabin. The reader is encouraged to watch the video accompanying this article. In all of our experiments, we keep a human driver inside the cabin for safety.

Fig. 7. ***Left to Right***: HEAP driving forward over a small hill. The front leg adapts its height to the ground height, despite motion planner producing plans under the flat ground assumption.

A. Reactive controller behavior

Fig. 7 shows the terrain adaptive control behavior induced by our controller. Fig. 8 shows the tracking performance. The planner thinks that the ground is flat and comes up with a plan to move forward. However, there is a small hill in the way, and the controller adjusts the front leg position to keep the base upright. This behavior naturally emerges from the task prioritization given in our controller (see Section III-A). Without the torque control in the FE joints, such adaptation would be hard to achieve.

B. Driving motions

We show the ability of our proposed pipeline to generate driving motions in Fig. 9. The base is commanded to go forward and left. The optimization discovers the crab steering mode (frame 2), where all wheels point in the same direction. Upon reaching the goal pose (frame 3), we command the

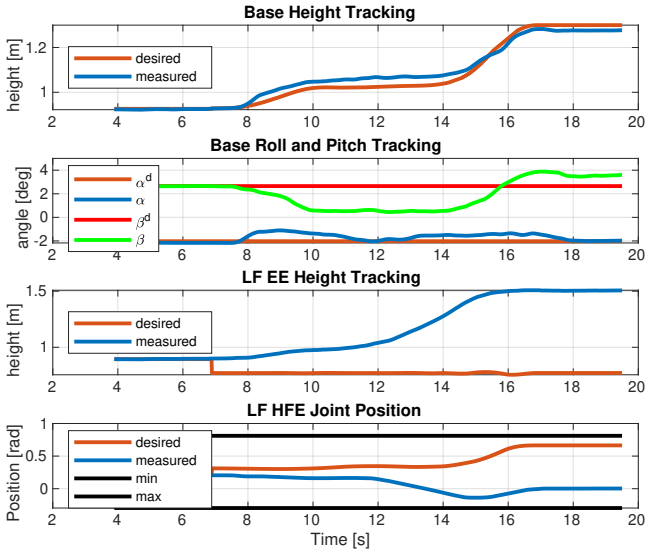


Fig. 8. Adaptive terrain behavior. Note how the base height, roll α and pitch β are tracked accurately over uneven terrain (see Fig. 7). To achieve the base tracking, the controller abandons tracking for the LF end-effector height and the FE joint position. Such a behavior is achieved through task prioritization.

machine to drive back to the place where it started. The same crab steering behavior emerges once more (frame 4), this time when driving backward. The motion is completed in frame 6. Note that the machine ends up in a configuration where the base is shifted to the right. This is because the planner receives only the base pose as a goal; the optimization discovers the actual configuration. Computing the plan for driving maneuvers takes about 1 s. Tracking performance is shown in Fig. 11c.

C. Stepping motions

To illustrate the whole-body capabilities of the controller, HEAP executes the stepping maneuver shown in Figure 10. In this experiment, we plan a stepping motion to turn the whole machine by 30° in the yaw direction. Motion is then tracked using our whole-body controller proposed in Section III. One can observe that the controller simultaneously coordinates the behavior of the boom, base, and all the legs. The boom is used as a counterweight and turns to the side opposite of the swing limb. Furthermore, the controller shifts the base to keep the Center of Mass (CoM) inside the stable region. In the end, one can observe a successfully reoriented base (snapshot 6). Thanks to the cost introduced in section II, the machine ends the motion in a stable configuration close to the nominal position. Hence, it is ready for re-planning and execution of another turn maneuver if necessary. Computing the plan for stepping motions takes about 10 s-15 s. Tracking performance in $x - y$ is shown in Fig. 11b and 11a.

V. CONCLUSION AND OUTLOOK

We extend the planning approach introduced in our previous work to make motion plans more robust and executable on a real platform. We design a terrain-adaptive controller that is capable of tracking the motion plans produced by our planner. The controller is based on a whole-body control framework and tailored for robots where not all joints may be torque-controllable, or the torque control may be difficult. Our framework has been experimentally validated through executing challenging motions on a full size walking excavator with 31 DoF. We hope to have advanced large machines towards the mobility level displayed by the best human operators. In the future, we would like to use the arm for locomotion through creating contacts with the ground and not merely balancing. The use of the fifth limb would enable stepping over obstacles, a maneuver commonly executed by humans. Furthermore, we would like to investigate what level of robustness is achievable by controlling as many joints as possible in position mode. Torque control is highly beneficial, however expensive since more cylinders have to be retrofitted with high-performance hydraulic valves.

REFERENCES

[1] "HEAP (Hydraulic Excavator for an Autonomous Purpose)," 2018. [Online]. Available: http://www.rsl.ethz.ch/robots-media/heap.html

[2] R. Grandia, F. Farshidian, R. Ranftl, and M. Hutter, "Feedback mpc for torque-controlled legged robots," *arXiv preprint arXiv:1905.06144*, 2019.

[3] C. D. Bellicoso, F. Jenelten, C. Gehring, and M. Hutter, "Dynamic locomotion through online nonlinear motion optimization for quadrupedal robots," *IEEE Robotics and Automation Letters*, 2018.

[4] F. Farshidian, E. Jelavic, A. Satapathy, M. Giftthaler, and J. Buchli, "Real-time motion planning of legged robots: A model predictive control approach," in *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE, 2017, pp. 577–584.

[5] P. Jarrault, C. Grand, and P. Bidaud, "Robust obstacle crossing of a wheel-legged mobile robot using minimax force distribution and self-reconfiguration," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 2753–2758.

[6] F. Cordes, A. Babu, and F. Kirchner, "Static force distribution and orientation control for a rover with an actively articulated suspension system," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 5219–5224.

[7] B. H. Wilcox, "Athlete: A limbed vehicle for solar system exploration," in *2012 IEEE Aerospace Conference*. IEEE, 2012, pp. 1–9.

[8] M. Bjelonic, C. D. Bellicoso, Y. de Viragh, D. Sako, F. D. Tresoldi, F. Jenelten, and M. Hutter, "Keep rollin'—whole-body motion control and planning for wheeled quadrupedal robots," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 2116–2123, 2019.

[9] 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," *arXiv preprint arXiv:1909.07193*, 2019.

[10] E. Jelavic and M. Hutter, "Whole-body motion planning for walking excavators," in *International Conference on Intelligent Robots and Systems (IROS 2019)*, 2019.

[11] 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*, vol. 4, no. 2, pp. 1633–1640, 2019.

[12] G. Bellegarda and K. Byl, "Trajectory optimization for a wheel-legged system for dynamic maneuvers that allow for wheel slip," in *under review for IEEE Conference on Decision and Control (CDC)*, 2019.

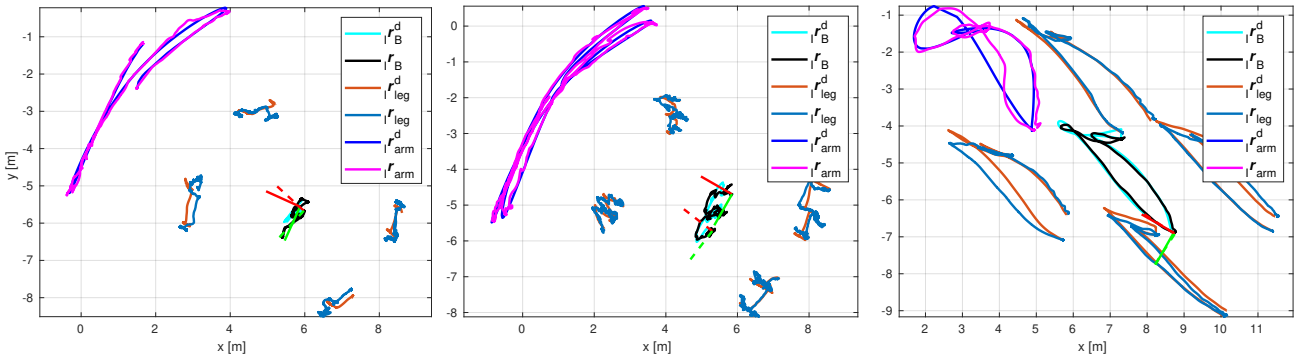
[13] T. Klamt and S. Behnke, "Planning hybrid driving-stepping locomotion on multiple levels of abstraction," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018.

[14] T. Klamt, D. Rodriguez, M. Schwarz, C. Lenz, D. Pavlichenko, D. Droeschel, and S. Behnke, "Supervised autonomous locomotion and manipulation for disaster response with a centaur-like robot," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018.

[15] A. W. Winkler, C. D. Bellicoso, M. Hutter, and J. Buchli, "Gait and trajectory optimization for legged systems through phase-based end-effector parameterization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1560–1567, 2018.

Fig. 9. ***Left to Right***: HEAP performs a driving maneuver. The base is commanded to move forward and left (frames 1-3). Subsequently, we command the machine to drive back to the starting point (frames 4-6). The planning and control pipeline naturally discovers the crab steering mode of the machine.

Fig. 10. ***Left to Right***: HEAP performs a stepping maneuver and reorients its base by 30°. The gait pattern requires it to lift the legs in the following order { LH, LF, RH, RF }. The machine cannot complete the maneuver without using the arm for balancing (e.g., in snapshot 3).



(a) Stepping turn (b) Stepping sideways (c) Driving

Fig. 11. ***Left:*** Tracking in $x-y$ plane for the base and end-effectors while performing turning maneuver shown in Figure 10. The coordinate frame (full lines) denotes starting pose while frame shown in dashed lines shows the end pose. The desired base pose is shown in cyan color, while the desired trajectory of each leg is shown in brown color. The desired trajectory for the arm is shown in blue color, while the actual trajectory is shown in the magenta color. ***Middle:*** Tracking performance for the stepping sideways maneuver (shown in video). The machine is commanded to step sideways two times by $0.5\,\mathrm{m}$ ***Right:*** Driving maneuver, shown in Figure 9. HEAP was commanded to drive front and right and then to return to the starting point.

[16] C. D. Bellicoso, C. Gehring, J. Hwangbo, P. Fankhauser, and M. Hutter, "Perception-less terrain adaptation through whole body control and hierarchical optimization," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2016.

[17] S. Medeiros, M. Bjelonic, E. Jelavic, R. Siegwart, A. Meggiolaro, M. Hutter, *et al.*, "Trajectory optimization for wheeled quadrupedal robots driving in challenging terrain," in *9th International Symposium on Adaptive Motion of Animals and Machines (AMAM 2019)*, 2019.

[18] 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, 2018.

[19] C. D. Bellicoso, K. Krämer, M. Stäuble, D. Sako, F. Jenelten, M. Bjelonic, and M. Hutter, "Alma-articulated locomotion and manipulation for a torque-controllable robot," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019.

[20] S. Fahmi, C. Mastalli, M. Focchi, and C. Semini, "Passive whole-body control for quadruped robots: Experimental validation over challenging terrain," *IEEE Robotics and Automation Letters*, 2019.

[21] C. R. Hargraves and S. W. Paris, "Direct trajectory optimization using nonlinear programming and collocation," *Journal of guidance, control, and dynamics*, vol. 10, no. 4, pp. 338–342, 1987.

[22] P. Furgale. Representing robot pose the good, the bad, and the ugly. [Online]. Available: http://paulfurgale.info/news/2014/6/9/representing-robot-pose-the-good-the-bad-and-the-ugly

[23] O. Melon, M. Geisert, D. Surovik, I. Havoutis, and M. Fallon, "Reliable trajectories for dynamic quadrupeds using analytical costs and learned initializations," *arXiv preprint arXiv:2002.06719*, 2020.

[24] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: modelling, planning and control*. Springer Science & Business Media, 2010.

[25] R. Featherstone, *Rigid body dynamics algorithms*. Springer, 2014.

[26] C. D. Bellicoso, F. Jenelten, P. Fankhauser, C. Gehring, J. Hwangbo, and M. Hutter, "Dynamic locomotion and whole-body control for quadrupedal robots," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 3359–3365.

[27] M. Bloesch, H. Sommer, T. Laidlow, M. Burri, G. Nuetzi, P. Fankhauser, D. Bellicoso, C. Gehring, S. Leutenegger, M. Hutter, *et al.*, "A primer on the differential calculus of 3d orientations," *arXiv preprint arXiv:1606.05285*, 2016.

[28] R. C. Coulter, "Implementation of the pure pursuit path tracking algorithm," Carnegie-Mellon UNIV Pittsburgh PA Robotics INST, Tech. Rep., 1992.

[29] Y. Kuwata, J. Teo, S. Karaman, G. Fiore, E. Frazzoli, and J. How, "Motion planning in complex environments using closed-loop prediction," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2008, p. 7166.

[30] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[31] A. Winkler, "Ifopt-a modern, light-weight, eigen-based c++ interface to nonlinear programming solvers ipopt and snopt," 2018.

[32] M. L. Felis, "Rbdl: an efficient rigid-body dynamics library using recursive algorithms," *Autonomous Robots*, 2017.

[33] M. Bloesch, M. Burri, H. Sommer, R. Siegwart, and M. Hutter, "The two-state implicit filter recursive estimation for mobile robots," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 573–580, 2017.