Perceptive Locomotion through Whole-Body MPC and Optimal Region Selection

Thomas Corbères, Carlos Mastalli, Wolfgang Merkt, Jaehyun Shim, Ioannis Havoutis, Maurice Fallon, Nicolas Mansard, Thomas Flayols, Sethu Vijayakumar, Steve Tonneau

Perceptive Locomotion through Whole-Body MPC and Optimal Region Selection

Journal Title
XX(X):1–18

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Thomas Corbères[1], Carlos Mastalli[2], Wolfgang Merkt[3], Jaehyun Shim[1], Ioannis Havoutis[3], Maurice Fallon[3], Nicolas Mansard[4], Thomas Flayols[4], Sethu Vijayakumar[1], Steve Tonneau [1]

Abstract

Real-time synthesis of legged locomotion maneuvers in challenging industrial settings is still an open problem, requiring simultaneous determination of footsteps locations several steps ahead while generating whole-body motions close to the robot's limits. State estimation and perception errors impose the practical constraint of fast re-planning motions in a model predictive control (MPC) framework. We first observe that the computational limitation of perceptive locomotion pipelines lies in the combinatorics of contact surface selection. Re-planning contact locations on selected surfaces can be accomplished at MPC frequencies (50-100 Hz). Then, whole-body motion generation typically follows a reference trajectory for the robot base to facilitate convergence. We propose removing this constraint to robustly address unforeseen events such as contact slipping, by leveraging a state-of-the-art whole-body MPC (CROCCODYL). Our contributions are integrated into a complete framework for perceptive locomotion, validated under diverse terrain conditions, and demonstrated in challenging trials that push the robot's actuation limits, as well as in the ICRA 2023 quadruped challenge simulation.

Keywords

perceptive locomotion, model predictive control, contact planning, quadruped robots

1 Introduction

Reliable and autonomous locomotion for legged robots in arbitrary environments is a longstanding challenge. The hardware maturity of quadruped robots Hutter et al. (2016); Unitree (2021); Boston Dynamics (2016) motivates the development of a motion synthesis framework for applications including inspections in industrial areas Bellicoso et al. (2018). Synthesising motions in this context requires handling the issues of *contact decision* (where should the robot step?) and *Whole-Body Model Predictive Control* (WB-MPC) of the robot (what motion creates the contact?).

Each contact decision defines high-dimensional, non-linear geometric and dynamic constraints on the WB-MPC that prevent a trivial decoupling of the two issues: How to prove that a contact plan is valid without finding a feasible whole-body motion to achieve it? Even worse, the environments we consider comprise holes and gaps, introducing a *combinatorics* problem: On which contact surface(s) should the robot step?

1.1 State of the art

The mathematical complexity of the legged locomotion problem in arbitrary environments is such that an undesired decoupling between contact planning and whole-body control is required. Typically, a contact plan describing the contact locations is first computed, assumed to be feasible, and provided as input to a WB-MPC framework to generate whole-body motions along it. As a result, the contact decision must be made using an approximated robot model, under the uncertainty that results from imperfect perception and state estimation. The complexity of the approximated model has, unsurprisingly, a strong correlation with the versatility and computational efficiency of the proposed approach.

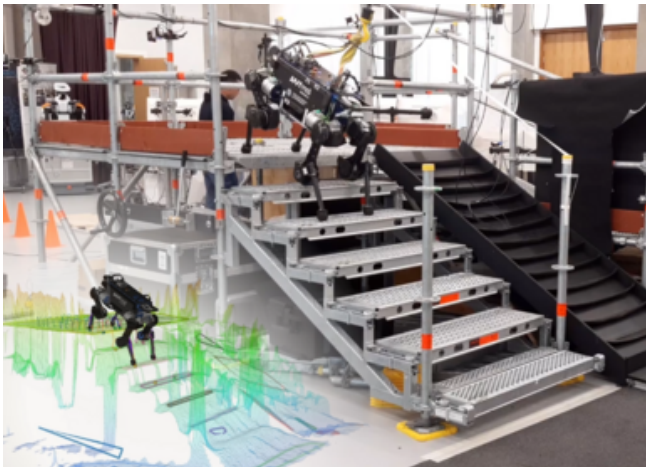


Figure 1. Industrial staircase descent with onboard perception. Video: https://youtu.be/bNVxTh0eixI.

[1]School of Informatics, University of Edinburgh, UK
[2]School of Eng. and Physical Sciences, Heriot-Watt University, UK
[3]Oxford Robotics Institute, University of Oxford, UK
[4] LAAS-CNRS, France

Corresponding author:
Thomas Corbères
Email: t.corberes@sms.ed.ac.uk

1.1.1 Offline contact planning with full kinematics Early approaches to contact planning are robust and complete, because they integrate a whole-body kinematic model in the planning phase, with quasi-static feasibility guarantees. These *contact-before-motion* approaches Bretl (2006); Hauser et al. (2008); Escande et al. (2009) mix graph-based search with contact-posture sampling to explicitly tackle the problem's combinatorics. Whole-body feasibility is explicitly checked before validating each new contact. The generality of these approaches is unmatched, but the associated computational cost is high (from minutes to hours), which prevents online re-planning. The computation time can be reduced to a few seconds by constraining the root's path Bouyarmane et al. (2009), and then by approximating the robot with low-dimensional abstractions Tonneau et al. (2018a); Murooka et al. (2021). These abstractions use hand-crafted heuristics for collision avoidance and geometry Short and Bandyopadhyay (2018), while dynamic feasibility is often asserted using centroidal dynamics (e.g., Tonneau et al. (2018b); Fernbach et al. (2020)). Such approximations proved unreliable for the most challenging scenarios (such as car egress motions Tonneau et al. (2018a)), mostly because of the difficulty of approximating collision avoidance constraints. Still, most "2.5D" environments (which can be accurately represented with a heightmap) composed of quasi-flat contact surfaces (i.e., a friction cone containing a gravity vector) can be handled with such constraints Tonneau et al. (2018a). Unsurprisingly, these environments, including rubles, stepping stones and staircases, are the application targets for most contributions in the literature.

1.1.2 Online contact planning with reduced dynamics The combination of reduced dynamic models and simplified collision constraints makes optimisation-based techniques tractable. Optimal control is attractive as it allows us to find solutions robust to uncertainties through the minimisation of selected criteria. Sampling-based approaches, instead, only look at feasibility. To model the combinatorics, the first approach is to relax the problem by modelling the discrete (boolean) variables that represent the contact decisions with continuous variables, resulting in a formulation that can be readily solved by off-the-shelf nonlinear programming (NLP) solvers Mordatch et al. (2012); Winkler et al. (2018). However, there is no guarantee that the system's dynamic constraints will be satisfied even with reduced models, with contacts potentially planned where no surfaces exist Song et al. (2021). Alternatively, Linear Complementary Constraints (LCP) can be used to accurately model contact constraints, but they are notoriously difficult to handle by NLP solvers Posa et al. (2014). In both cases learning initial guesses can help the solver converge to a feasible solution Melon et al. (2020, 2021). The second approach is to explicitly handle combinatorics using Mixed-Integer Programming (MIP). MIP solvers tackle contact planning problems for bipeds Deits and Tedrake (2014) and quadrupeds Aceituno-Cabezas et al. (2018), provided that the underlying optimisation problem is convex, which results in a conservative approximation of the dynamics Ponton et al. (2021). Monte Carlo Tree Search (MCTS) has recently been proposed as a promising alternative to MIP that could provide a relevant trade-off between exploration and exploitation Amatucci et al. (2022). In this work, we choose MIP for planning contacts as it has experimentally led to the most effective results in terms of computation time and reliability Risbourg et al. (2022).

1.1.3 Perceptive locomotion with instantaneous decisions Whether the environment is fully known or not impacts on the validity of a method. Reactive perceptive pipelines exist on the LittleDog robot Zucker et al. (2011); Kolter et al. (2008); Kalakrishnan et al. (2011) and have inspired further works Fankhauser et al. (2018a). However, they all require high-precision terrain pre-mapping and an external motion capture system. When the environment is not fully known, it is typically modelled as an elevation map by fusing depth sensor information within proprioceptive information Fankhauser et al. (2018b); Miki et al. (2022). Recent approaches propose to directly optimise the next contact position, the torso orientation and obstacle avoidance for the foot trajectory based on this input Jenelten et al. (2022); Grandia et al. (2020, 2023). The approaches share similarities with the framework we propose in terms of the model's proposition. However, their main difference is that they focus on planning the immediate contact location and posture, which is why we argue that a preview window of several steps ahead is required for the scenarios we consider. As discussed in Section 9, we believe that combining these approaches is a promising research avenue.

1.1.4 Whole-body predictive control relying on CoM motions Because of the nonlinearity induced by any changes to the contact plan, the WB-MPC rarely challenges the step locations, even though the approximations do not guarantee feasibility. The uncertainties resulting from state estimation and environment perception motivate frequent re-computation of the contact plan, which is not possible as their frequency is usually low (about 5 Hz in Song et al. (2021)). Furthermore, the WB-MPC is usually additionally constrained to track a reference trajectory for the Centre Of Mass (COM) or the base Carpentier and Mansard (2018); Mastalli et al. (2020a) to facilitate convergence, but we argue that this tracking is problematic when perturbations such as contact slipping occur. Our conclusion is that the use of reduced models for contact planning necessarily leads to errors in the WB-MPC that result in slipping contacts. In the current state of the art, reduced models appear necessary for satisfying real-time constraints. Mitigating this issue involves allowing to adapt a contact plan at a higher frequency. However, it involves writing a WB-MPC that robustly accommodates these errors and gives as much freedom as possible when following a contact plan.

1.2 Contribution

This paper extends our published conference paper Risbourg et al. (2022), where we propose a *contact repositioning* module between the contact surface planner and the MPC to adapt to the robot's estimated state and perceived environment uncertainties. To achieve this, we decouple the contact surface selection from the control, but not the computation of the contact position on that surface, which we instead update synchronously with the MPC output and updated state of the robot, at 50 Hz. As such, the contact

repositioning module is our primary contribution. This work is thus to be considered as an experimental and theoretical extension in the following manner.

We propose a novel, complete perceptive locomotion architecture comprised of the following features: terrain segmentation, real-time surface selection and footstep planning, free-collision foot-swing planning, and whole-body MPC which considers torque limits and generates local controllers. It relies on four technical contributions:

(i) a theoretical contribution to the decoupling between contact planning and WB-MPC,

(ii) an empirical demonstration of the added value of WB-MPC in perceptive locomotion,

(iii) a convex segmentation approach using onboard terrain elevation maps, and

(iv) exhaustive hardware trials on challenging terrains demonstrating increased capabilities for the ANYmal B robot.

Finally, we extend the plane segmentation algorithm Fallon and Antone (2019), which decomposes potential contact surfaces into a sequence of convex surfaces needed by our contact planner. We use the Visvalingam–Whyatt and Tess2 algorithms to correctly handle overlapping surfaces and conservatively reduce the complexity of the scene, leading to a more versatile and robust environment construction. Our framework results in state-of-the-art locomotion capabilities under a wide range of conditions, robust to strong perturbations including sliding contacts and missed steps during stair climbing.

The reader should note that we have incorporated a feedback WB-MPC Mastalli et al. (2022, 2023) into our framework, as opposed to the more conventional approach in our previous work, which involved MPC with reduced dynamics followed by a WBC running at a higher frequency. Constraining only the end-effector trajectories, instead of the CoM motions, allows the WB-MPC to freely accommodate substantial perturbations and perception errors, and to maximise the robot's capabilities by optimising its posture. While a subset of our experimental results are shared by both papers, their contributions are orthogonal.

2 Architecture overview

An overview of the locomotion pipeline is presented in Fig. 2. Walkable surfaces are described via convex planes extracted from the terrain elevation map at 1 Hz. Given the current state of the robot (position / orientation of the base, position of active contacts) and a desired velocity (joystick input), a mixed-integer program is used to select the convex surfaces on which the next 6 or 8 steps will occur, depending on the gait used. A new surface plan is computed at the beginning of each new phase, which corresponds to approximately 3-5 Hz in our experiments. Given the next contact surfaces, the trajectory of each end-effector is updated at 50 Hz, before each iteration of the whole-body MPC. The control policy is then sent to the robot with a Riccati gain controller. State estimation is performed onboard at 400 Hz by fusing inertial sensors from IMU and odometry provided by the ANYbotics software Bloesch et al. (2012). Finally, a LIDAR is used to correct the drift of these measurements by analysing fixed points in the environment.

3 Definitions and notations

In line with the notations used in our previous work Risbourg et al. (2022), the *robot state* is formally described by the:

- Centre Of Mass (COM) position, velocity and acceleration $\mathbf{c}$, $\dot{\mathbf{c}}$ and $\ddot{\mathbf{c}}$, each in $\mathbb{R}^3$;
- base transformation matrix in the world frame;
- 3D position of each end-effector in the world frame;
- gait, i.e., the list of effectors currently in contact, as well as the contacts to be activated and deactivated over the planning horizon.

The *horizon* n is defined as the number of future contact creations considered. In the case of the trotting gait, a horizon $n = 6$ describes three steps, as at each step two contacts are created simultaneously.

At the *Surface Selection* planning stage, motion is decomposed into *contact phases*. Each contact phase is associated with a number of feet in contact and one or more contacts are broken/created at each phase. In the case of a trotting gait with a horizon $n = 6$, it corresponds to three contact phases since two feet move at the same time. For a walking gait, the horizon $n = 8$ contains 8 contact phases since only one foot moves at a time.

The *environment* is the union of $m + 1$ disjoint quasi-flat* contact surfaces $\mathcal{S} = \bigcup_{i=0}^{m} \mathcal{S}^i$. Each set $\mathcal{S}^i$ is a polygon embedded in a 3D plane, i.e.,

$$\mathcal{S}^i := \{\mathbf{p} \in \mathbb{R}^3 | \mathbf{S}^i \mathbf{p} \leq \mathbf{s}^i\}\,, \tag{1}$$

where $\mathbf{S}^i \in \mathbb{R}^{h\times 3}$ and $\mathbf{s}^i \in \mathbb{R}^h$ are respectively a constant matrix and a vector defining the h half-space bounding surface.

The *contact plan* is described as a list of contact surfaces $\mathcal{S}_k^j \in \mathcal{S}, 1 \leq j \leq l$ with l being the total number of end-effectors and k being the k-th contact phase.

4 Perception

This section reviews the *Elevation map* and *Convex Plane Segmentation* components of the architecture (Fig. 2).

4.1 Sensors

Regarding the proprioceptive sensors, the state is estimated by fusing leg odometry and the Xsens MTi-100 IMU Bloesch et al. (2012). Additionally, a rotating Hokuyo UTM-30LX-EW lidar sensor is placed at the back of the robot to correct the drift of the state estimation with an iterative closest point (ICP) method Hutter et al. (2017). A single depth camera Intel RealSense D435, mounted at the front of the robot, extracts height information from the surrounding area into a point cloud. High-accuracy presets are used on the camera to prioritize the accuracy of the point cloud over speed. This

*ie such that its associated friction cone contains the gravity vector.

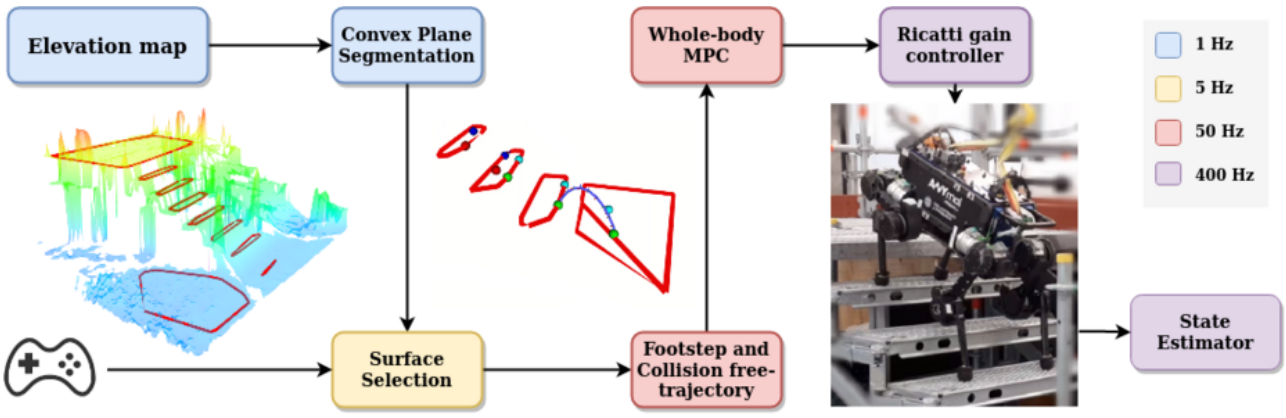


Figure 2. Overview of our perceptive locomotion pipeline. Around 1Hz, the perceptive elements *Elevation map* and *Convex Plane Segmentation* extract convex planes from the surrounding environment (Section 4). The *Surface Selection* block, running around 5Hz (depending on the gait chosen), chooses between them the next surfaces of contact (Section 5). At the frequency of the MPC, 50 Hz, **Footstep and Collision free-trajectory** elements generate the curve for each moving foot (Section 6). Finally, the *Whole-Body MPC* and **Riccati gain controller** synthesise the motion (Section 7).

increases the quality of the elevation map and no filters are needed to post-process the point clouds. However, this preset degrades the frequency in our setup to a range between 2 and 5 Hz for point cloud collection.

4.2 Contact surfaces extraction

The probabilistic and local mapping method in Fankhauser et al. (2018b) converts point cloud data into an elevation map locally around the robot's pose. Including proprioceptive localisation from kinematic and inertial measurements produces an estimate of the surroundings as a 2.5D heightmap. Potential surfaces are then extracted with the Plane-Seg algorithm Fallon and Antone (2019) by clustering planar points with similar normals. The planes are extracted without memory and therefore the surfaces can change suddenly with each height map update. The quality and consistency of the surfaces then depend entirely on the quality of the height map, as developed in Sec. 9.

4.3 Refinement and margin of safety

Post-processing of Plane-Seg planes is essential to ensure safe footstep decisions. We have added it for three reasons. First, the complexity of the contact surfaces (i.e., the number of points) has a significant impact on the computation time of the surface selection algorithm. We propose conservatively approximate surfaces with more than 8 vertices with an 8-vertex polygon. Filtering is also done to remove unreachable surfaces. For example, the plane extraction method could return overlapping surfaces when considering a staircase. Here, the ground surface is often detected below the steps and planning a footstep inside it will obviously result in failures. Finally, a safety margin is applied, around 4 cm on each surface, to avoid putting feet on the edges of the surfaces. This can be harmful in the event of estimation errors. This margin is also useful to avoid knee collisions of the knees with the environment, which are not explicitly accounted for.

4.3.1 Vertices number reduction The number of vertices is first reduced using the conservative Visvalingam–Whyatt line simplification algorithm Visvalingam and Whyatt (1993). It eliminates progressively the points from a line that forms the smallest area with its closest neighbours as described in pseudo-code 1. No hyperparameters other than the final number of points (which is 8) are needed. Fig. 3a shows an example of this reduction.

Algorithm 1 Pseudo-code for Visvalingam–Whyatt line simplification

List of n vertices : $L = [P_0 \dots P_n]$
while len$(L) > n_{\max}$ **do**
  for $i \in [0, len(\mathrm{L}) - 2]$ **do**
    Compute area of $[P_i, P_{i+1}, P_{i+2}]$
  end for
  Remove P_{i+1} corresponding to the smallest area.
end while
return L

4.3.2 Safety margins On this updated contour, an inner and an outer margin are computed parallel to each edge as shown in Fig. 3b. The inner margin allows the robot to step into a safe area. The outer margin artificially increases the size of the obstacle to prevent the end-effector from getting too close of the obstacle. This is done to avoid collisions while computing swing-foot trajectories. These two margins are used to avoid stepping on the corner of an obstacle due to state estimation errors and for collision avoidance with the body.

4.3.3 Convex decomposition Starting from the lowest surface, the overlapping surfaces are removed and a convex decomposition is performed on the resulting contour using the Tess2 algorithm azrafe7 (2013) as shown in Fig. 4a. Smaller areas under 0.03 m^2 are deleted. Algorithm 2 gives the pseudo-code of this post-processing. In addition, a rectangle is added below the robot's position, at the estimated height of the feet. This is to ensure that there is always a surface under the robot's feet if the elevation map has not been built. This surface is treated differently in the process to avoid overlapping it with real obstacles but has been removed from the pseudo-code for clarity.

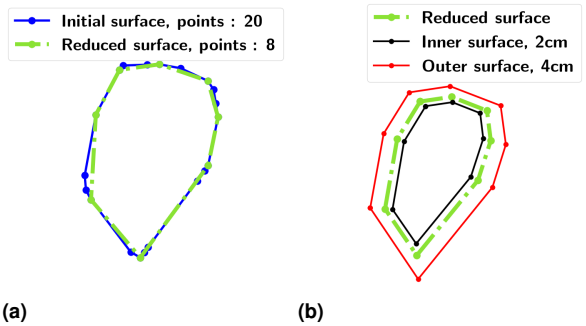


Figure 3. Reduction to an 8-vertex polygon using the Visvalingam–Whyatt algorithm on an initial 20-vertex polygon on the left 3a. Applying inner and outer margins to an 8-vertex polygon 3b.

Algorithm 2 Pseudo-code for surface processing

for all surfaces **do**
  Reduce nb of points with Visvalingam's algorithm.
  Compute inner and outer contour.
end for
$L_i \leftarrow$ List of surfaces in ascending order of height.
$L_f \leftarrow$ Empty list.
while L_i is not empty **do**
  Get first surface Sf and remove it from L_i.
  $L_o \leftarrow$ Empty list.
  for all surface in L_i **do**
    if surface.outer intersect with S_f.inner **then**
      Increment L_o with outer contour.
    end if
  end for
  if L_o is not empty **then**
    Convex Decomposition between S_f.inner and L_o.
    Remove small areas.
    Add remaining surfaces in L_f.
  else
    Add S_f.inner in L_f
  end if
end while
return L_f

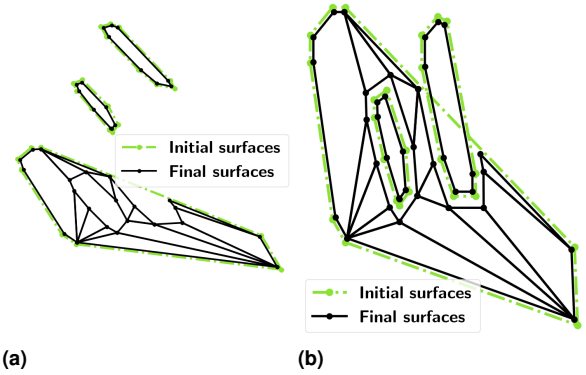


Figure 4. Example of surface processing and convex decomposition. These two figures represent the same 3D scene with 2 air overlapping the ground surface. On the right Fig. 4b, the scene is viewed from a top perspective.

5 Surfaces selection

Given the current state of the robot (active contacts and position, COM location), the environment given as a union of non-intersecting surfaces, as well as a desired target velocity for the robot and a desired gait, our *Surface Selection* (Fig. 2 - yellow) algorithm computes a feasible contact plan, composed of n contact surfaces that the robot should step on for the planning horizon (Sec. 3). We set $n = 8$ for a walking gait and $n = 6$ for a trotting gait in our experiments. The surface selection algorithm is executed between 3 and 5 Hz. This means that the contact plan is updated before each new step is made by the robot. The frequency depends on the gait and each optimisation starts at the beginning of each step.

The algorithm is implemented as a Mixed-Integer Program (MIP) Deits and Tedrake (2015). We use the SL1M formulation of this algorithm Song et al. (2021), with adaptations to better match the desired robot behaviour. These adaptations are in Risbourg et al. (2022) and are described here for completeness. In this previous work, the number of contacts optimised was set to 4 for the Solo robot Grimminger et al. (2020). Here the ANYmal's robot dynamics are slower and the timing between each new contact is longer[†]. This gives more time for computing the contact plan, allowing us to increase the planning horizon. We refer the reader to Risbourg et al. (2022) for empirical justifications for these choices.

We first describe how the potential contact surfaces are pre-filtered to improve the algorithm's computational performance without loss of generality. We then provide the mathematical formulation of the Surface Selection algorithm. We conclude this section with the details of the cost function used in this optimisation problem.

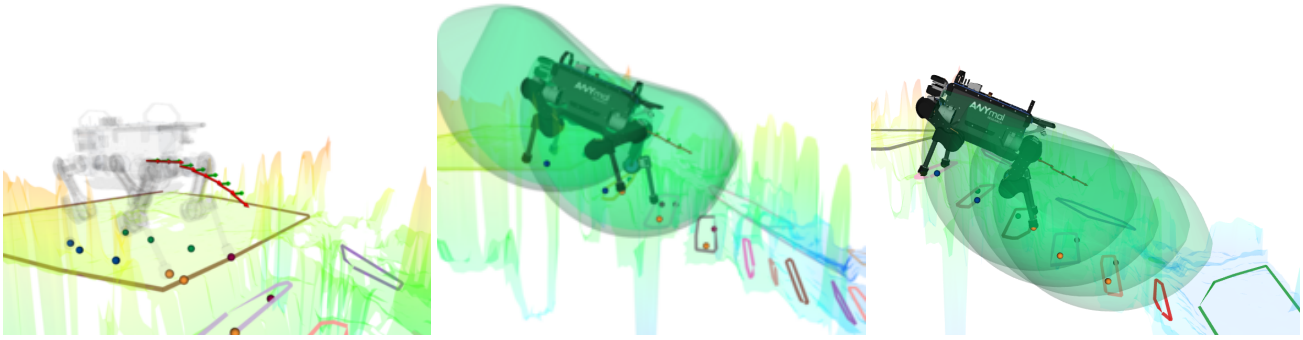
5.1 Pre-selection

We first pre-filter the number of contact surfaces using the robot's range of motion (ROM) to reduce the combinatorics. To do so, a CoM trajectory is extrapolated from the joystick velocity command (Fig. 5a) over the contact phases. The position of each state and the yaw angle (orientation around the z-axis) are integrated from the linear and angular (yaw only) desired velocity. The roll (orientation around the x-axis) and pitch (orientation around the y-axis) angles of the guide are computed as the average slope of the terrain around the robot position, given by solving (Fig. 5a)

$$\min_{\mathbf{n}} \quad \|\mathbf{An} - \mathbf{B}\|^2$$
$$\text{with} \quad \forall\, i,j \in N_x \times N_y, \quad \mathbf{A}[i+jN_x,:] = [x_i, y_j, 1.],$$
$$\mathbf{B}[i+jN_x] = \text{elevation}(x_i, y_j),$$

where x_i and y_j are respectively the positions on the x and y axis around the robot's position with a user-defined resolution of $(N_x, N_y) \in \mathbb{N}^{+2}$. The elevation function gives the terrain height at the position (x_i, y_j) and can be obtained directly from the heightmap or by evaluating the convex plane corresponding to this 2D position. $\mathbf{n} = [a, b, c] \in \mathbb{R}^3$

[†]The duration of a step was set to 160 ms on Solo whereas it is set to 600 ms for a walking gait and 300 ms for a trotting gait on ANYmal.



(a) (b) (c)

Figure 5. (a) CoM extrapolation along the horizon. (b) ROMs of the 4 effectors for the current state. (c) ROMs along the horizon for the front right foot.

is the vector optimised to form the plane of equation $ax + by - z + c = 0$. The experiments use 10×10 as resolution. For each extrapolated state $\mathbf{c}_j^*$ (8 states for a walking gait as $n = 8$ and 3 for a trotting gait as $n = 6$ but two feet contacts are created at each phase), the 6D configuration is given by:

$$\mathbf{c}_j^* = \begin{bmatrix} x_0 + \int_0^{t_j} (v_x^* \cos(\dot{\psi}^* t) + v_y^* \sin(\dot{\psi}^* t))dt \\ y_0 - \int_0^{t_j} (v_x^* \sin(\dot{\psi}^* t) - v_y^* \cos(\dot{\psi}^* t))dt \\ ax_j + by_j + c + h_{\text{ref}} \\ \arctan(b) \\ -\arctan(a) \\ \psi_0 + \psi^* t_j \end{bmatrix}, \quad (3)$$

where the state is described using a 3D position $[x_j, y_j, z_j]$ and 3 Euler angles. The current extrapolated positions x_j and y_j are used to compute the configuration height. v_x^*, v_y^* are the x-y linear velocities and $\dot{\psi}^*$ the yaw angular velocity coming from the joystick input. $h_{\text{ref}} = 0.48$ is the robot's nominal height. t_j is the step duration, manually defined for each gait.

For each $\mathbf{c}_j^*$, the ROM of each moving leg is intersected with the surfaces. The ROM is approximated by a convex set and represented by the green surfaces in Fig. 5b. Only the surfaces that intersect this ROM are selected. The distance between two convex sets is computed efficiently with the GJK algorithm Gilbert et al. (1988) from the PINOCCHIO and HPP-FCL libraries Carpentier et al. (2019). In our experiments, this usually reduces the number of potential surfaces from 20 to 3 for each moving foot on average. This significantly reduces combinatorics (from about 20^n to 3^n possible combinations).

5.2 Surface Selection algorithm as a MIP

The surface selection module computes a contact plan for the robot that satisfies linearized kinematic constraints Tonneau et al. (2018b); P.-B. (2006). In the following, we recall the mathematical formulation of the problem as a mixed-integer program (MIP). This MIP outputs the contact surfaces selected. It also computes the 3D locations of footsteps as a by-product. However, they are discarded as the target footsteps will be adapted at a higher frequency by the footstep planner.

5.2.1 Contact constraint representation The perception pipeline provides a set of $m + 1 \in \mathbb{N}^+$ disjoint quasi-flat surfaces, which define the environment $\mathcal{S}$, and the motion is decomposed into n contact phases (Sec. 3), as shown in Fig. 6. To constrain a point $\mathbf{p} \in \mathbb{R}^3$ to be in contact, we must write:

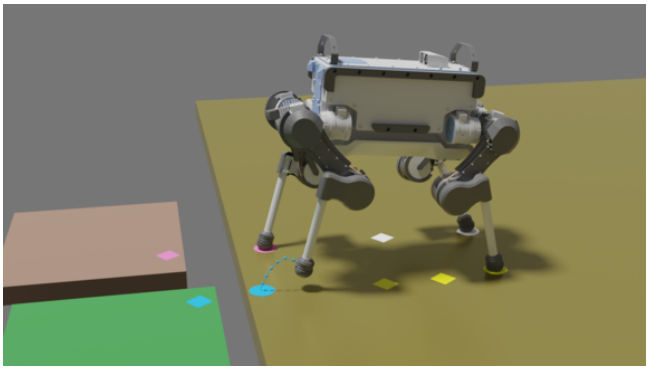


Figure 6. Environment $\mathcal{S}$ with 3 contact surfaces for a walking gait (1 foot moving at a time). Circles: initial position; squares: next steps locations.

$$\exists\, i, \mathbf{S}^i\mathbf{p} \leq \mathbf{s}^i \Leftrightarrow \mathbf{S}^0\mathbf{p} \leq \mathbf{s}^0 \vee \cdots \vee \mathbf{S}^m\mathbf{p} \leq \mathbf{s}^m. \quad (4)$$

The *or* constraint is classically expressed as an integer constraint using the Big-M formulation Lofberg (2022) as follows. We introduce a vector of binary variables $\mathbf{a} = [a_0, \ldots, a_m] \in \{0, 1\}^{m+1}$ and a sufficiently large constant $M \in \mathbb{R}^+, M >> 0$. (4) is equivalently rewritten as:

$$\forall i, \mathbf{S}^i\mathbf{p} \leq \mathbf{s}^i + M(1 - a_i); \sum_{i=0}^{m} a_i = 1. \quad (5)$$

Under this formulation if $a_i = 1$, then $\mathbf{p}$ belongs to $\mathcal{S}^i$ (and is thus in contact). Instead, if $a_i = 0$ for a sufficiently large M then the constraint $\mathbf{S}^i\mathbf{p} \leq \mathbf{s}^i + M(1 - a_i)$ will be satisfied for any value of $\mathbf{p}$, in other words, the constraint is inactive. $\sum_{i=0}^{n} a_i = 1$ implies that $\exists i, a_i = 1$. The obtained behaviour is thus the desired one: if (5) is true then $\mathbf{p}$ is in contact with a surface.

5.2.2 Problem formulation The final MIP problem is obtained by combining contact and surface constraints as follows. For simplicity, and without loss of generality, we assume that the candidate contact surfaces are the same for

each step.

$$\begin{aligned} \textbf{find} \quad & \mathbf{P}, \mathbf{A} = [\mathbf{a}_0, \cdots, \mathbf{a}_m] \in \{0,1\}^{(m+1)\times n} \\ \textbf{min} \quad & l(\mathbf{P}) \\ \text{s.t.} \quad & \mathbf{KP} \leq \mathbf{k}, \\ & \mathbf{P} \in \mathcal{C}, \\ & \forall j \in \{0, \ldots, n-1\} : \\ & \quad \forall i, \mathbf{S}^i \mathbf{p}_j \leq \mathbf{s}^i + M(1 - a_i^j); \sum_{i=0}^{m} a_i^j = 1, \end{aligned} \tag{6}$$

where $\mathbf{P} = [\mathbf{p}_0 \ \ldots \ \mathbf{p}_n] \in \mathbb{R}^{3\times n}$ is the vector comprising the variable n next foot positions (6 or 8 in our experiments); $\mathbf{a}^j = [a_0^j, \ldots, a_m^j]$ is the vector of binary variables associated with the j-th optimised foot position; $l(\mathbf{P})$ is an objective function; $\mathcal{C}$ is a set of user-defined convex constraints (in our case, constraints on initial contact positions); $\mathbf{K}$ and $\mathbf{k}$ are constant matrix and vector representing the linearised kinematic constraints on the position of each effector with respect to the others P.-B. (2006); Tonneau et al. (2018b); Winkler et al. (2018).

5.2.3 Cost function details As mentioned in Risbourg et al. (2022), two quadratic costs are used in the problem. The first attempts at regularising the footstep locations using Raibert's heuristic. Since the optimisation is triggered at the beginning of each step (between 3 and 5 Hz), and not at each time step (50 Hz), an approximation of the Raibert heuristic is applied. The idea is to interpolate the base position given the desired velocity and place the foot accordingly to the estimated hip position. The second term penalises the distance between the hip and the foot location. This aims at penalising solutions close to reaching the robot's kinematic limits.

$$l(\mathbf{P}) = \sum_{j}^{n} w_1 \|\mathbf{p}_j - \mathbf{p}_j^*\|_{x,y}^2 + w_2 \|\mathbf{p}_j - \mathbf{p}_{\text{hip},j}^*\|^2, \tag{7}$$

where $\mathbf{p}_j^*$ is the extrapolated foot position taking into account the linear and angular reference velocity at the corresponding contact phase. The weights w_1 and w_2 are set to 1 and 0.5. This penalisation only considers the ℓ^2 norm on the x- and y-axis. Similarly, $\mathbf{p}_{\text{hip},j}^*$ is the extrapolated hip position from the desired velocity. The computation of the reference foot location is detailed in Sec. 6.

6 Foot trajectory generation

Our *Footstep and Collision free-trajectory* (Fig. 2 - red) algorithms compute the end-effector trajectory as a Bezier curve given the current robot state (COM position/orientation), the next moving foot within the horizon of the MPC, the contact plan obtained by the Surface Planner, the timings of contact depending on the gait chosen, as well as a desired target velocity for the robot. At the MPC frequency, set to 50 Hz, a first quadratic program (QP) computes the next foot location and a second QP calculates its trajectory during the flight. Their formulation differs from our previous work Risbourg et al. (2022) as the base velocity is not optimised here. The total computation time for the foot trajectory is negligeable in the pipeline, as evidenced by Table 2.

6.1 Foot position optimisation

We use an alternative to Raibert's heuristic Raibert (1986) as it is commonly done on quadruped robots Kim et al. (2019); Di Carlo et al. (2018); Léziart et al. (2021). In our case, we are simply interested in coherent foot locations according to the base position, reference velocity and gait period. The measured velocity is not part of this heuristic, making it a feed-forward strategy. Instead, our MPC plans the CoM trajectory (Sec. 7). A target 2D position of the foot $\mathbf{p}^*$ is computed as follows:

$$\mathbf{p}^* = \mathbf{p}_{\text{hip}} + \frac{T_s}{2}\mathbf{v}_{ref} + \sqrt{\frac{h}{g}}\mathbf{v}_{ref} \times \omega_{ref}, \tag{8}$$

where T_s is the stance phase time extracted from the phase-based gait pattern, $\mathbf{v}_{\text{base}}$, and $\mathbf{v}_{\text{ref}}$ are respectively the current base velocity and the reference velocity commanded by the user. Finally, h, g and ω_{ref} are respectively the nominal height of the robot, the gravity constant and the reference angular velocity around the z-axis. The estimated hip position at contact $\mathbf{p}_{\text{hip}}$ accounts for the accumulated delay in the estimation of the base position, since the moving average is used to reject oscillations of the same period as gait, as discussed in section 8.7. The final position $\mathbf{p}$ is the closest to $\mathbf{p}^*$ that lies on the selected surface $\mathcal{S}$ under the locally-approximated kinematic constraints $\mathcal{K}$:

$$\min_{\mathbf{p}} \quad \|\mathbf{p} - \mathbf{p}^*\|^2 \tag{9a}$$

$$\text{s.t.} \quad S\mathbf{p} \leq s, \tag{9b}$$

$$\mathbf{p} \in \mathcal{K}. \tag{9c}$$

The difference with our MIP optimisation lies in the 50 Hz computation frequency. As a result, the footstep is constantly updated relative the base position.

6.2 Collision free trajectory

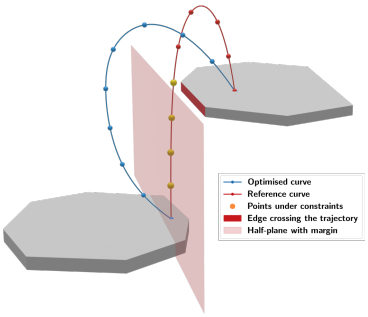


Figure 7. Adaptation of the end-effector trajectory to climb a step. The active constraint corresponds to the half-plane crossed by the reference trajectory -red curve-. Both curves are discretised to perform the optimisation and the orange points highlight the ones under constraints.

A collision-free trajectory $\mathbf{p}(t) : \mathbb{R} \to \mathbb{R}^3$ connects the current foot location to the optimised location. $\mathbf{p}(t)$ aims at

following a reference trajectory while avoiding obstacles and keeping computational time low as it is parameterized using a Bezier curve of degree d on the Bernstein basis $(B_i)_{i\leq d}$:

$$\mathbf{p}(t) = \sum_{i=0}^{d} B_i^d(\frac{t}{T})\mathbf{P}_i, \tag{10}$$

where $\mathbf{P} = \begin{bmatrix} P_0 & \dots & P_d \end{bmatrix}^T$ are the $d+1$ control points and T is the total time of the trajectory, empirically set to 600 ms.

6.2.1 Reference trajectory The trajectory $\mathbf{p}_{ref}(t) : \mathbb{R} \rightarrow \mathbb{R}^3$ is composed of a degree 6 polynomial curve of degree 6 on the z-axis and a degree 5 one on the x and y axes. All of which have with the first 3 control points fixed to ensure the continuity in position, velocity and acceleration of the curve from the current state. End velocity and acceleration are set to 0 to avoid slippage at the end position. Additionally, the height at $\frac{T}{2}$ is fixed on the z-axis, constraining all degrees of freedom.

6.2.2 Collision avoidance The trajectory $\mathbf{p}(t_k)$ is a 3D curve of degree $d = 7$ that follows $\mathbf{p}_{ref}(t)$ while avoiding collisions. To do so, we enforce collision avoidance along the foot trajectory. This is achievable iteratively Campana et al. (2016) by solving a sequence of QPs and adding constraints where collisions occur. For computational efficiency, we empirically identify the points likely to be in a collision and add collision constraints to them (Fig. 7 - Yellow dots). These $n_c + 1$ points $\mathbf{p}(t_k) = \mathbf{A}_k\mathbf{P}, \forall k \in [0, \dots, n_c]$ are linearly defined by Bezier's control points, with $\mathbf{A}_k \in \mathbb{R}^{3\times(d+1)}$. We choose the active constraint to be the half-space traversed by the reference curve (Fig. 7 - pink half-space). We thus write $\forall k, \mathbf{S}_m^i\mathbf{p}(t_k) \geq \mathbf{s}_m^i$, where $\mathbf{S}_m^i \in \mathbb{R}^3$ and $\mathbf{s}_m^i \in \mathbb{R}$ define what constitutes the current half-space. All the collected constraints are then stacked into a single matrix and vector $\mathbf{G}$ and $\mathbf{h}$, leading to the QP:

$$\min_{\mathbf{p}} \quad \sum_{k=0}^{n_c} \|\mathbf{p}(t_k) - \mathbf{p}_{\text{ref}}(t_k)\|^2 \tag{11a}$$

$$\text{s.t.} \quad \mathbf{p}(0) = \mathbf{p}_{\text{ref}}(0), \qquad \mathbf{p}(T) = \mathbf{p}_{\text{ref}}(T), \tag{11b}$$

$$\dot{\mathbf{p}}(0) = \dot{\mathbf{p}}_{\text{ref}}(0), \qquad \dot{\mathbf{p}}(T) = \mathbf{0}_{\mathbb{R}^3}, \tag{11c}$$

$$\ddot{\mathbf{p}}(0) = \ddot{\mathbf{p}}_{\text{ref}}(0), \qquad \ddot{\mathbf{p}}(T) = \mathbf{0}_{\mathbb{R}^3}, \tag{11d}$$

$$\mathbf{GP} \leq \mathbf{h}. \tag{11e}$$

This approach can be expanded to avoid local obstacles along the trajectory by imposing constraints that depend on the environment's heightmap. Currently, our trajectory avoidance is based only on obstacles perceived as walkable surfaces, which has been found to work effectively in many scenarios. Nevertheless, if an obstacle is along the trajectory but not detected as such, it will not be avoided. This highlights the importance of including height-map information when available.

7 Motion Generation

Given the current state of the robot (6D position and velocity), the next contact sequence and the end-effector trajectories, our *Whole-Body MPC* (WB-MPC) and *Riccati gain controller* elements (Fig. 2 - red and violet) generates the motion of the legged robot. The WB-MPC optimises a trajectory at 50 Hz and the Riccati gain controller applies it at 400 Hz.

7.1 Optimal control formulation

The legged robot generates motions through a model predictive controller that relies on the robot's full-body dynamics. The formulation is based on two previous papers Mastalli et al. (2022, 2023). We solve the optimal control (OC) problems using Crocoddyl's advanced solvers Mastalli et al. (2020a). Our pipeline relies on two different OC formulations based on forward and inverse dynamics. Both yield similar performances, demonstrating our method's generality. Both formulations can be written as Mastalli et al. (2022, 2023):

$$\min_{\{\mathbf{q},\mathbf{v}\},\{\boldsymbol{\tau}\text{or}\{\dot{\mathbf{v}},\boldsymbol{\lambda}\}} \ell_N(\mathbf{x}_N) + \sum_{k=0}^{N-1} \int_{t_k}^{t_{k+1}} \ell_k(\mathbf{q}_k, \mathbf{v}_k, \boldsymbol{\lambda}_k, \boldsymbol{\tau}_k)\, dt \tag{12a}$$

$$\text{s.t.} \quad \mathbf{q}_{k+1} = \mathbf{q}_k \oplus \int_{t_k}^{t_{k+1}} \mathbf{v}_{k+1}\, dt,$$

$$\mathbf{v}_{k+1} = \mathbf{v}_k + \int_{t_k}^{t_{k+1}} \dot{\mathbf{v}}_k\, dt,$$

$$\begin{bmatrix} \dot{\mathbf{v}}_k \\ -\boldsymbol{\lambda}_k \end{bmatrix} = \begin{bmatrix} \mathbf{M}_k & \mathbf{J}_c^\top \\ \mathbf{J}_c \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{\tau}_b \\ -\mathbf{a}_c, \end{bmatrix} \quad \text{(forward dyn.)}$$

$$\text{or}$$

$$\text{ID}(\mathbf{q}_k, \mathbf{v}_k, \dot{\mathbf{v}}_k, \boldsymbol{\lambda}_k) = \mathbf{0}, \quad \text{(inverse dyn.)}$$

where the robot state $\mathbf{x} = (\mathbf{q}, \mathbf{v})$ contains the generalized position and velocity vectors. More precisely, $\mathbf{q} \in \mathbb{SE}(3) \times \mathbb{R}^{n_j}$ with the generalized velocity lying in the tangent space $\mathbf{v} \in \mathfrak{se}(3) \times \mathbb{R}^{n_j}$ where n_j is the number of articulated joints and $\oplus$ denotes the integration operator in $\mathbb{SE}(3)$ inspired by Blanco (2010) and used in Mastalli et al. (2020a).

For the forward-dynamics formulation, the input command is $\mathbf{u} = \{\boldsymbol{\tau}\}$ corresponds to the joint torques. k, $\mathbf{M}_k$ and $\mathbf{J}_c$ corresponds to the discrete-time t_k, the generalised mass matrix and the contact Jacobian, respectively. $\boldsymbol{\tau}_b$ includes the joint torque command, Coriolis and gravitation terms and $\mathbf{a}_c$ is the desired acceleration which results from the rigid contact constraint (contact point velocity is null) and includes the Baumgarte stabilisation term Baumgarte (1972). The impulse dynamics described in more detail in Mastalli et al. (2022) which allow velocity changes at impact have also been omitted from the formulation. This contact dynamics formulation comes from the application of the Gauss principle of least constraint P.-B. (2006), as described in detail in Budhiraja et al. (2018). By handling this constraint in the backward pass, the decision variables can be condensed with forward dynamics Mastalli et al. (2022); Budhiraja et al. (2018); Mastalli et al. (2020a) and contact forces can be removed. This drastically reduces the number of decision variables.

Regarding the inverse-dynamics formulation, where $\mathbf{u} = \{\dot{\mathbf{v}}, \boldsymbol{\lambda}\}$, the contact forces and the acceleration become decision variables of the problem Mastalli et al. (2023); Erez

and Todorov (2012). We can perform efficient factorizations by applying nullspace parametrization as proposed in Mastalli et al. (2023). Furthermore, an alternative inverse dynamic equation that allows us to remove joint torques from the control vector is possible. Both strategies reduce the size of the problem needed to enable MPC applications.

Table 1. Details of the costs used in the OCP

Name	Formulation	Cost
State bound	$\mathbf{x}_{\min} \leq \mathbf{x} \leq \mathbf{x}_{\max}$	10^3
Base orientation reg.	$\log(\mathbf{p}_{\text{base}}^{-1})$	$[0_{\mathbb{R}^3}\ 10^2\ 10^2\ 0]$
Base velocity reg.	$\|\dot{\mathbf{p}}_{\mathbf{base}}\|$	10
Joint position reg.	$\|\mathbf{q}_{\mathbf{a}k} - \mathbf{q}_{\mathbf{a}\text{ref}}\|$	0.01
Joint velocity reg.	$\|\dot{\mathbf{q}}_{\mathbf{a}k}\|$	1
Torque reg.	$\|\mathbf{u}_k\|$	0.1
Force reg.	$\|\boldsymbol{\lambda}_{\mathcal{C}_k}\|$	1
Friction cone	$C\boldsymbol{\lambda}_{\mathcal{C}_k} \leq c$	10
Feet position track.	$\log(\mathbf{p}_{\mathcal{G}_k}^{-1}\mathbf{p}_{\text{ref},\mathcal{G}_k})$	10^6
Feet velocity track.	$\dot{\mathbf{p}}_{\text{ref},\mathcal{G}_k} - \dot{\mathbf{p}}_{\mathcal{G}_k}$	10^4

Regularisation terms included in the cost function ℓ are summarized in Table 1. These different formulations are transcribed into quadratic terms, where $\mathbf{q}_\mathbf{a}, \dot{\mathbf{q}}_\mathbf{a} \in \mathbb{R}^{12}$ are respectively the actuated joint positions and velocities. $\mathbf{p}_{\text{base}} \in \mathbb{SE}(3)$ describes the base pose in the world frame. $\mathbf{p}_{\mathcal{G}_k} \ominus \mathbf{p}_{\text{ref},\mathcal{G}_k} = \mathbf{p}_{\mathcal{G}_k}^{-1}\mathbf{p}_{\text{ref},\mathcal{G}_k}$ represents the pose error in $\mathbb{SE}(3)$ of the feet relative to the reference position. It is penalised by considering it in its tangent space Blanco (2010) with the log function that maps $\mathbb{SE}(3)$ to $\mathfrak{se}(3)$. Feet position and velocity tracking error are expressed in the robot's inertial frame. The base position is not penalised whereas its rotation is penalised around the x- and y-axis only, corresponding to the roll and pitch angles. $\mathbf{C}$ and $\mathbf{c}$ are the matrix and vector of the linearised friction cone. $\mathcal{C}_k$ represents the set of feet in contact with the ground and inversely $\mathcal{G}_k$ is the set of feet in the swing phase for the node k. Costs are set to zero if the corresponding feet are in the contact phase.

7.2 Riccati-gain controller

OC formulations with reduced-order dynamics require an instantaneous whole-body controller for tracking computed forces and maintaining robot balance Mastalli et al. (2020b); Léziart et al. (2021); Kim et al. (2019). Such instantaneous controllers may compete with the MPC policy and not necessarily generate the motion predicted by them Léziart et al. (2022). At a higher cost in computing time, which has a non-negligible influence, many benefits can appear when including whole-body dynamics in the OC problem, e.g., imposing the joint effort limits. Another benefit is to derive local feedback controllers from optimisation principles. This control policy looks like this:

$$\boldsymbol{\tau}_d = \boldsymbol{\tau}_{ff} + \mathbf{K}(\mathbf{x} \ominus \mathbf{x}^*), \tag{13a}$$

$$= -\mathbf{Q_{uu}}^{-1}\mathbf{Q_u} - \mathbf{Q_{uu}}^{-1}\mathbf{Q_{ux}}(\mathbf{x} \ominus \mathbf{x}^*), \tag{13b}$$

where $\ominus$ denotes the $\mathbb{SE}(3)$ error. $\mathbf{Q_{uu}}$, $\mathbf{Q_u}$ and $\mathbf{Q_{ux}}$ are respectively the partial derivatives of the value function for the joint-effort command and the state Mastalli et al. (2022). While the MPC runs every $0.02\,\text{s}$ ($50\,\text{Hz}$), the discretisation is set to $0.01\,\text{s}$. Instead, the low-level Riccati-gain controller runs at $400\,\text{Hz}$, and an interpolation step is necessary. For this, the feedback term is computed by using an interpolation of the reference state $\mathbf{x}^*$ by integrated the dynamic with the contact model dynamic 12. The optimal effort command is then provided at $400\,\text{Hz}$ in addition to a joint impedance controller based on the joint command and velocity obtained from $\mathbf{x}^*$.

8 Results

8.1 Implementation

Communication between each module is achieved via a low-latency ROS communication layer (TCP, no-delay) Quigley (2009). Three onboard computers (`Intel(R) Core(TM) i7-5600U CPU @ 2.6GHz`) share the main tasks of locomotion, perception and estimation. Point-cloud framework, heightmap generator, state estimation and Riccati-gain controller are performed onboard. We used an additional computer (`Intel(R) Core(TM) i5-8365U CPU @ 1.60GHz`) to extract the segmented planes from the heightmap, post-process the extracted surfaces and run the mixed-integer program in a different thread. A final computer (`Intel(R) Core(TM) i9-9900KF CPU @ 3.60GHz`) is used to run the MPC which sends the plan to the Riccati-gain lower controller at $50\,\text{Hz}$.

8.2 Experiments

The pipeline has been tested in various scenarios, first with onboard perception and then with a model of the environment. This is to emphasize the motion generation part and break away from the perception constraints. During all experiments, the user commands the robot's velocity with a joystick. Some of the experiments (*1.3*, *1.4* and *2.1*) were conducted using the inverse dynamic formulation and presented in the related paper Mastalli et al. (2023) as evidence of the formulation's effectiveness. We present these experiments again in this paper to specifically highlight the planning aspect. We release the environments tested for this paper in a public repository (`https://github.com/thomascbrs/walkgen-environments`). The source code for our planner is available as an open source package (`https://github.com/loco-3d/sl1m.git`) and we will release the rest of our code upon acceptance of the paper.

8.2.1 Using onboard perception

We evaluated our complete pipeline on three major scenarios, listed below.

- *Experiment 1.1* (Fig. 1): The first experiment is a 5-minute experiment, representative of the type of environment our perceptive locomotion pipeline

Figure 8. Screenshots of different experiments highlighting our architecture's capabilities. The first three rows of results were obtained using the onboard camera and a complete perception pipeline for active obstacle detection. Rows (d), (e), and (f) were performed without perception setup but instead used a pre-computed model of the environment to overcome the perception system limitations and test the controller's limits. The last two rows were obtained through simulation to further test our pipeline in challenging scenarios. Video accessible at https://youtu.be/bNVxTh0eixI.

enables. The robot starts at the bottom of an industrial staircase with 7 steps, each 17 cm high and 29 cm deep. The average terrain slope is 30 degrees. Once

at the top, the robot makes a U-turn on the industrial platform and performs its descent.

- *Experiment 1.2* (Fig. 8-a): The second scenario corresponds to two platforms of 1 by 1 meters connected by a piece of wood placed diagonally to the right of the assembly. Once the robot is on the first platform, we manually remove the ground from the contact surface list to prevent the robot from moving forward. A 20x30 cm block was then added to the left of the assembly, and after a few seconds, the platform is detected and the robot moved forward. This demonstrates the pipeline's reactive capabilities.
- *Experiment 1.3* (Fig. 8-b): It is the same configuration as the previous experiment with 2 pieces of wood connecting the two platforms. This experiment highlights the accurate execution of footstep plans.
- *Experiment 1.4* (Fig. 8-c): Using the robot onboard perception, the final experiment aims to mix various terrains types and heights. Since the metallic plate was not fixed, it slid at the end of the experiment, showing the stability of the controller.

8.2.2 Using a model of the environment To further evaluate the locomotion capabilities of our pipeline, we conducted additional experiments in which the contact surfaces are known a priori, and not given by Plane-Seg. The robot's pose with respect to the world frame is still estimated using the onboard sensors (LIDAR and proprioceptive sensors) and in general, the rest of the framework remains the same.

- *Experiment 2.1* (Fig. 8-d) : The first experiment was to climb stairs with 2 missing steps. We removed steps 2 and 6 of the stairs. It corresponds to climbing a slope of 30 degrees with two gaps of 34 cm. This pushes the robot to its kinematic and the actuation limits. This, therefore, highlights the benefit of taking into account the entire robot model to plan the motion and adapt the posture. This justifies our approach, as discussed in Sec. 8.5.
- *Experiment 2.2* (Fig. 8-e): The second experiment is similar and corresponds to descenting stairs with 1 missing step (step 6). It corresponds to crossing a slope of 30 degrees with a gap of $34\,\mathrm{cm}$.
- *Experiment 2.3* (Fig. 8-f) : The last experiment was conducted on a platform of size $1\,\mathrm{m} \times 1\,\mathrm{m}$ and $38\,\mathrm{cm}$ in height. It is higher than the robot's height in its nominal position. It is the only experiment where we had to increase the safety margin around the obstacle (outer margin in Sec. 4) to $12\,\mathrm{cm}$ since the robot's shoulders are at the same height as the obstacle and collide with it.

8.2.3 Simulation experiments A final set of experiments were run in the PyBullet simulator Coumans and Bai (2016–2021) to show more dynamic gaits such as trotting. This was done to illustrate the diversity of terrains the framework can target. The setup and communication protocol are identical to the one used on the hardware and described previously in Sec. 8.1.

- *Experiment 3.1* (Fig. 8-g): The first experiment corresponds to a set of stepping stones of different sizes and heights crossed with a trotting gait.
- *Experiment 3.2* (Fig. 8-h): The second experiment involves the ICRA 2023 Quadruped Challenge, which comprises a parkour task featuring a range of obstacles, including pallets, inclined ramps, rounded rubber ramps, and a stack of plastic crates. This challenge serves as a valuable benchmark for evaluating our pipeline capabilities. For most parts, our pipeline navigates parkour using a dynamic trot. For some of them, we had to marginally modify some parameters. The wooden pallets with gaps between the wooden slats imply many potential surfaces for each contact, up to 12, even in the pre-selection section. Hence, we had to reduce the surface planner's horizon to optimise the next 4 contact sequences (and not 6 as used previously). Step height need to be increased by up to $25\,\mathrm{cm}$ on flat terrain with wood panels of $20\,\mathrm{cm}$. It would be ideal to include height map directly in trajectory optimization. This is rather than solely relying on surfaces for obstacle avoidance. as discussed previously in Sec. 6. Furthermore, when dealing with a stack of plastic crates, adapting the gait to a walking gait (in which only one foot moves at a time) is more stable and enables safe passage across the terrain, compared to a more dynamic trotting gait.

8.3 Evaluation of the perception pipeline

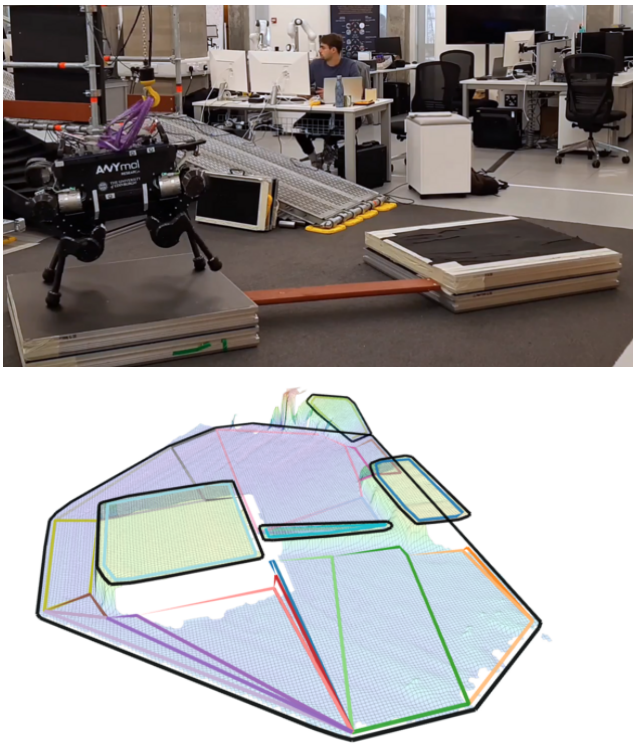


Figure 9. Perception output during experiment 1-(ii) before ground removal. Black surfaces: initial convex surfaces; coloured surfaces: filtered.

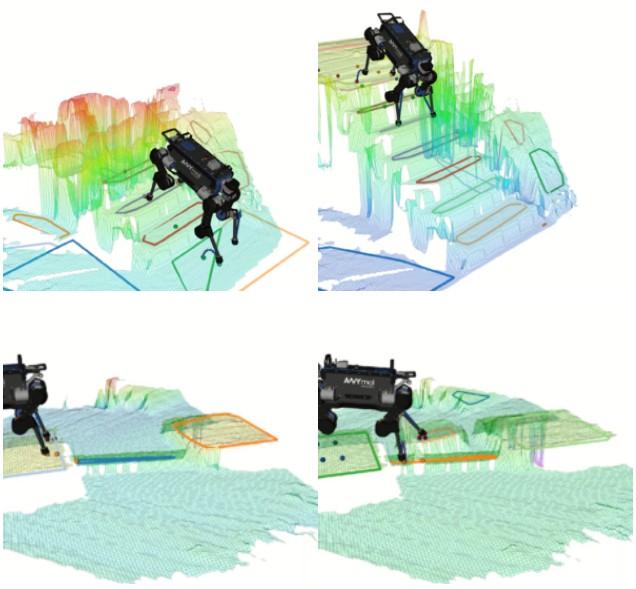


Figure 10. Height map and resulting surfaces during experiment 1-(i) (first row) and experiment 1-(ii) (second row). (https://youtu.be/bNVxTh0eixI).

In this section, we evaluate the perception pipeline, from the heightmap to the resulting surfaces filtered and reshaped for security margins. Fig. 9 shows the scene during experiment *1.1*, before the manual removal of the ground. The necessity of the filtering routine, described in Sec. 4 is highlighted in this example since the convex planes extracted from the height map overlap. Additionally, we note that the security margin is applied inside the obstacles to avoid walking on the edges. This is due in part to estimation errors, which are around evaluated to 3-4 cm. More height maps and resulting potential surfaces are shown in Fig 10, taken from experiments *1.1* and *1.2*. It is interesting to note the evolution of the terrain estimation around the robot during these experiments. When climbing or descending stairs, the robot's camera only identifies the next 1 or 2 stairs ahead. During the second experiment *1.2*, where an obstacle was added in front of the robot, it took several iterations of the probabilistic algorithm to update the heightmap with the detected object and consequently 2 to 3 seconds to obtain a feasible surface to walk on.

8.4 Computation performance

Table 2 presents the computation time statistics for each module of the pipeline during experiment *1.1*, i.e., climbing up and down stairs with active perception.

8.4.1 Surface Processing It takes 90 ms on average to post-process the incoming surfaces from perception with Algorithm 2 presented in Sec. 4. In comparison of the plane's update frequency, which is between 0.5 Hz and 1 Hz with an non-optimised code, it represents roughly an increase of 5-10%.

8.4.2 Surface Selection Surface selection, as described in Sec. 5 encompasses the pre-selection step to reduce the number of potential surfaces considered by the mixed-integer program. It takes around 120 ms to find the next surface of contact in this experiment. Pre-selection reduces the potential surfaces from 23 to 3 for each foot. During this experiment, we optimise over 8 contact phases, which correspond to 8-foot positions optimised with a walking gait (1 contact is created/broken for each contact phase). The mixed-integer optimisation starts at the beginning of each upcoming foot trajectory and the next surface information needs to be received before the beginning of the upcoming cycle. In this experiment, the foot trajectory is 600 ms, which is enough for the maximum time taken. However, for a trotting gait, the foot trajectory is set to 300 ms. We must reduce the number of foot locations to 6, thus allowing 3 contact phases (2-foot locations optimised for each contact phase) to ensure a safe margin regarding the computing time. We note that the computing time of mixed-integer optimisation depends heavily on the number of potential surfaces for each contact and the number of contact locations optimised. For a detailed analysis of the MIP computation times, we refer the reader to Song et al. (2021).

Table 2. Computing time

Name	Mean	Min	Max
Surface Processing			
Number of surfaces processed	**23.71**	8.	45.
Surface processing [ms]	**88.89**	19.49	164.22
Surface Selection			
Number of potential surfaces	**3.08**	1.	9.
Pre-selection [ms]	**17.16**	5.47	47.46
MIP [ms]	**98.62**	32.95	253.15
Foot Trajectory			
Foot location [ms]	**0.14**	0.08	0.74
Foot trajectory [ms]	**0.082**	0.007	0.33
Motion Generation (MPC)			
Solve time [ms]	**9.29**	8.02	13.52
Total time [ms]	**13.27**	8.76	18.50

8.4.3 Foot trajectory The foot trajectory encompasses the optimisation of the footstep inside the attributed surface and the end-effector curve optimisation as described in Sec. 6. It represents only 1% of every MPC step.

8.4.4 Motion generation (MPC) On average the MPC step takes around 13 ms, including the update of the optimal control problem and solving this latter with 1 iteration which takes around 9 ms. This is below the maximum 20 ms time required for an MPC running at 50 Hz.

8.5 Analysis of the whole-body MPC

In scenarios involving climbing steps, the torque limit is reached on at least one actuator in each of these experiments. It is in this case that the whole-body MPC becomes

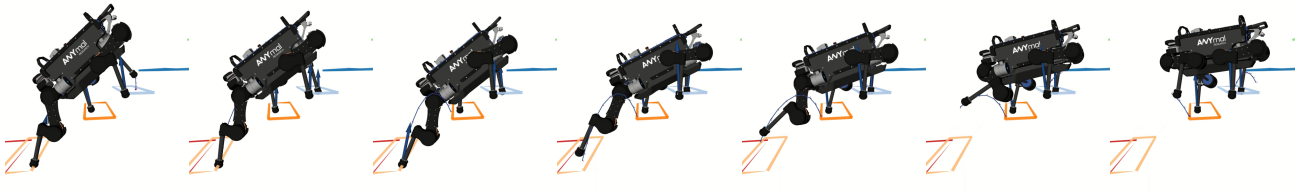


Figure 11. Posture adjustment (experiment *2.1*). The hind right leg reaches the torque limits on both HFE (hip flexion/extension) and KFE (knee flexion/extension) when crossing the gap. This corresponds to the peak torque at 52s. The whole-body MPC adjusts the body posture to compensate. The body leans forward and lowers as much as possible, reaching the kinematic limit of the hind leg to reduce torque on it. (`https://youtu.be/bNVxTh0eixI`).

crucial as it adjusts body posture to reduce joint torques. Experiment *2.1* is a representative case. First, we observe that the COM motion leans forward and close to the feet, almost in contact with the stairs, at the moment of giving the last motion to cross the gap (Fig. 11). The torque limit is reached on both joints of the hind right leg during this motion, as shown in Fig. 12. We analyse the following quantities: torques, angular position and angular velocities at the HFE joint (hip flexion/extension) and the KFE joint (knee flexion/extension). The HAA joints (hip abduction/abduction) are less prone to reach torque limits. While crossing the gap, we can observe two torque peaks that correspond exactly to the robot configuration shown in Fig. 12. In this instance, the overshoot of the torque command above the joint limits (a few Nm) can be attributed to two main reasons. First, the constraints on torque limits can be violated (Sec. 7) as the Riccati controller guarantees joint limits within a neighbourhood.

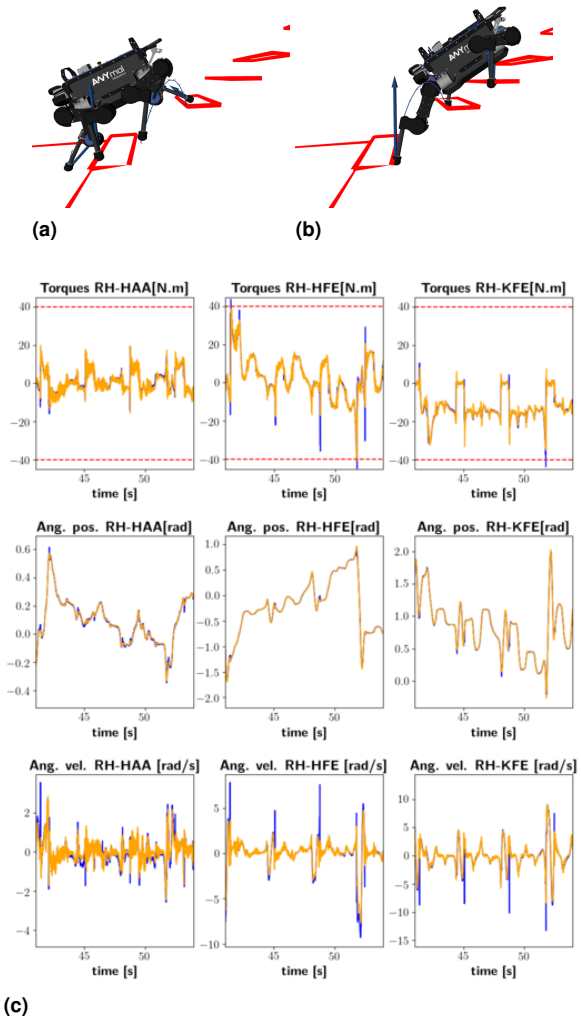


Figure 12. Torque, angular position and velocities of the joints HAA (hip abduction/abduction), HFE (hip flexion/extension) and KFE (knee flexion/extension) of the hind right leg during experiment *2.1*. Blue quantities are computed by the state feedback controller and orange ones are measured on the robot. The maximum torque limit is approximately $40\,\mathrm{N\,m}$ and is represented by red dashed curves. Each actuator can reach 12 $\mathrm{m.s^{-1}}$. Two torque peaks reaching the boundaries can be observed around the $40\,\mathrm{s}$ and $52\,\mathrm{s}$ which correspond to the robot configurations shown above.

8.6 *Evaluation of the collision-free foot trajectory*

The end-effector trajectory is re-computed at $50\,\mathrm{Hz}$ in order to get robust tracking. Some minor modifications were necessary when transitioning from simulation to hardware. An offset of $-1\,\mathrm{cm}$ has been added on the z-axis to accommodate for perception errors, reaching 2/3 cm at the end of our longest experiments (*1.1*), and ensuring contact creation occurs. A finer control of the feet's trajectory according to the contact detection was not necessary in our case, as the whole-body MPC is robust to state estimation errors (Sec. 8.8). Additionally, the swing-foot trajectory and footstep are no longer updated once 70% of the flight phase has passed in order to avoid a sliding contact after a sudden change in the target position. Finally, the end-effector velocity feedback is not taken into account in the optimisation due to large uncertainties about the end-effector velocities. The foot velocity is therefore assumed to be tracked properly and the initial velocity while re-computing the curve is taken from the previous control cycle.

Fig. 13 shows a swing-foot trajectory while the robot crosses an obstacle of $20\,\mathrm{cm}$ and while climbs the stairs during experiment *1.1*. We observe the state estimation errors on the foot position when landing on the ground, resulting in the foot slightly bouncing.

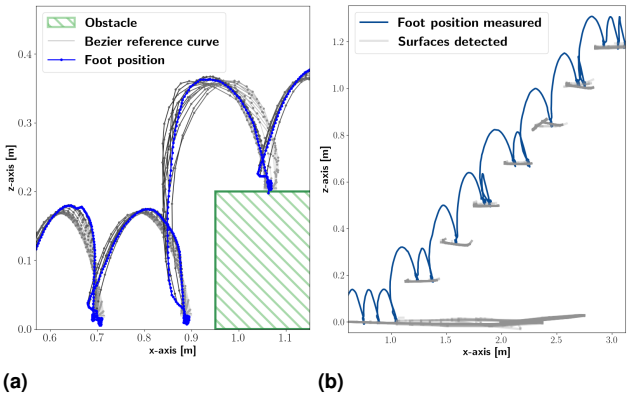


Figure 13. Side view of the front-left foot's trajectory while crossing a $20\,\mathrm{cm}$ obstacle (left) and the staircase (right) using a walking gait pattern at a reference velocity of 0.1 $\mathrm{m.s}^{-1}$. The grey curves on the left graph are reference Bezier curves computed during each control cycle. Their transparency indicates the prediction horizon. More transparent curves represent further predictions for the future. Grey areas on the right correspond to the cumulative position of surfaces detected by the camera.

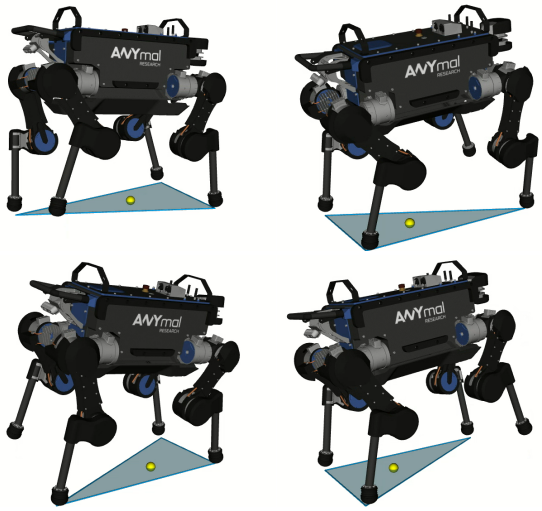


Figure 14. Screenshots of the motion resulting from our OCP regularization setup for a walking gait pattern. The blue area between represents the support polygon and the yellow circle is the Centre Of Pressure (COP) applied.

8.7 Evaluation of the velocity tracking and design choices

Not constraining the robot's pose and velocity to follow a reference trajectory has a significant impact on the COM trajectory, especially in a walking gait pattern when only one foot is lifted at a time (Fig. 14). Indeed, Fig. 15-(a) shows that the base trajectory during a forward walk oscillates in a range of $10\,\mathrm{cm}$ around the middle of the feet positions. However, constraining the base position and velocity would result in a fixed base orientation. This can be explained by the motion stability found in the optimization process. The centre of pressure is well located in the middle of the support polygon. The COM position is dragged inside this region resulting in this wave motion. Since it is a periodic motion, the state position has been filtered with a moving average for the walk period (Fig. 15), rejecting all frequencies in synchronisation

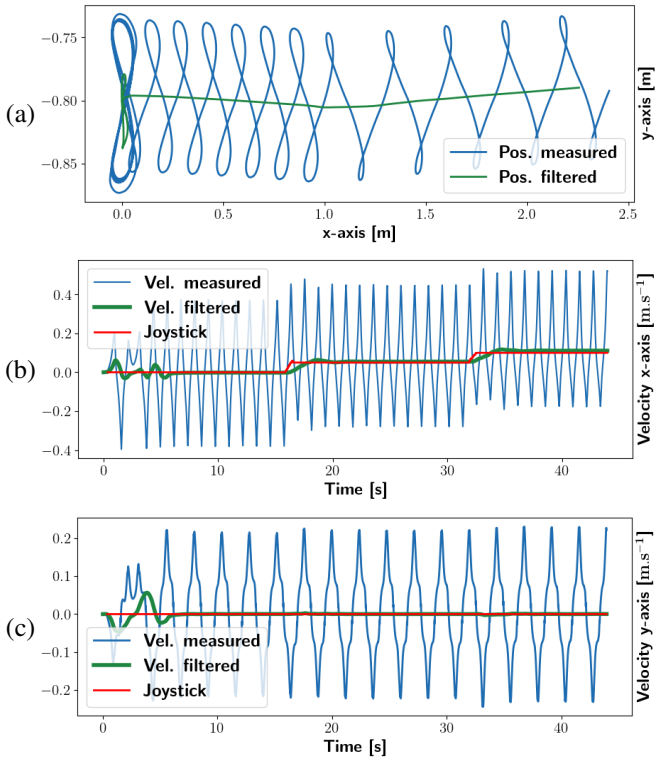


Figure 15. Base trajectory during a forward walking gait of period 2.4s. The joystick command is in 2 separate steps of $0.05m.s^{-1}$ and $0.1m.s^{-1}$ along the x-axis. To reject disturbances, the filter is a moving average on the walk period. (a) base position on the ground floor plane. (b) and (c) base velocity along respectively the x and y axes. Small disturbances at first in the filtered quantities are due to the filter initialisation phase.

with the gait. This behaviour does not appear with a trotting gait since two opposite legs are left at the same time and the resulting oscillation is much smaller.

8.8 Robustness of the pipeline

A crucial point is to understand the repeatability of our experiments and how the locomotion controller adapts to unforeseen events. We observed a significant difference in the reliability whether on-board perception was used or perfect knowledge of the environment was given. The climbing stairs scenario was reliably repeated ten times with active perception, with a 80% success rate, with errors mainly due to state estimation errors and drift. For climbing down the stairs the experiment was successfully repeated twice, but perception issues made the experiment difficult, as the camera position did not allow to clearly perceive all the stairs on the way down, resulting in unfeasible contact planning problems in some instances. In the absence of the perceptive part (experiments *2.1*, *2.2* and *2.3*), when the environment is perfectly known, the locomotion pipeline is stable and the experiments were successfully carried out on the first attempt and repeated twice.

An interesting point attributed to the whole-body MPC is the ability to recover to unplanned situations, as in experiment *2.2* when ANYmal misses a step during the descent. Similarly, at the end of experiment *1.4*, a metal plate

slips when walking on it. In both cases, the robot recovered properly (Fig. 16).

Finally, we refer the reader to our conference paper for an extensive ablation study on effect of the different elements of the pipeline on the success rate of the pipeline for the Solo robot Risbourg et al. (2022).

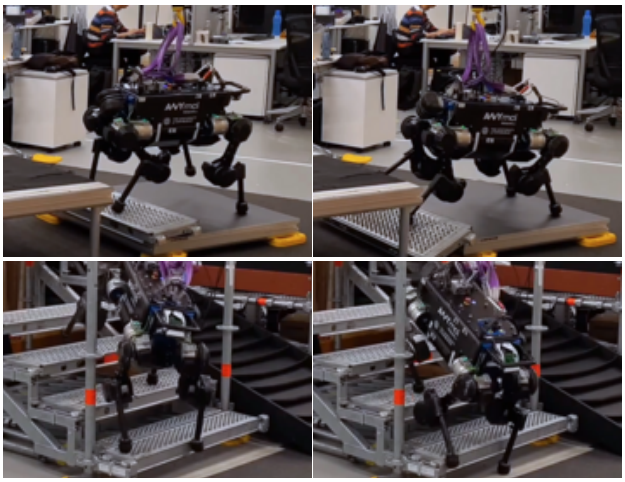


Figure 16. Screenshots of the robot recovering from an unplanned event. Top: the metal plate slips while the robot walks on it. Bottom: the two front feet slip and miss the intended step.

8.9 Comparison with the state of the art

The end-to-end implementation of frameworks similar to ours Grandia et al. (2023); Jenelten et al. (2022) are not publicly available at the time of submission, neither the associated benchmarks and the quantity of work and resources to re-implement them is not reasonable, which prevents a quantitative comparison of the approaches.

9 Discussion

In this paper, we propose a complete pipeline for perceptive quadrupedal locomotion; from onboard perception to locomotion generation and control. Our experimental results highlighted capabilities such as crossing challenging scenarios like climbing and descending industrial stairs or obstacle parkour with moving obstacles. Therefore, we have experimentally validated our architecture. This is based on the strong design assumption that a high-level planner, such as mixed-integer optimization, only selects the stepping contact surfaces. The utility of mixed-integer optimisation for design integration was partially validated in our previous work Risbourg et al. (2022). We used a different robot and integrated perception into our approach along with more complex experiments which reinforced interest in our decomposition.

9.1 Perception pipeline

We have demonstrated the efficacy of our perceptive pipeline based on the probabilistic terrain mapping method Fankhauser et al. (2018b) and the Plane-Seg algorithm Fallon and Antone (2019) which allow us to obtain satisfactory behaviour, in particular with our filtering work. Additionally, we demonstrated that our architecture can successfully navigate complex parkour terrains with moving objects or industrial staircases. Still, while our architecture is capable of navigating challenging scenarios when perception is removed, improving perception capabilities could enhance the overall robustness and reliability of the system. This would be particularly beneficial in scenarios such as the staircase where vision detection is restricted to only a few steps ahead due to our single depth camera with a 20% downward angle. Additionally, the algorithm used to extract convex planes is not incremental and does not take into account previously computed planes, making it vulnerable to faulty scans in some cases. The implementation of a short term memory for the plane decomposition system would contribute to the robustness of the approach and will be considered in future work.

9.2 Collision avoidance of the body

The trade-off proposed in our approach is to strongly regularise the OCP around a reference end-effector trajectory while avoiding obstacles, which proved to work in a wide range of environments. This occurs even if a collision with the environment is not specifically considered in our approach. This could be considered within the MPC but would require a dedicated study since it is a challenging nonlinear optimisation problem. The safety margin around obstacles was sufficient in almost all scenarios to avoid collisions, except for the $40\,\mathrm{cm}$ step in 2.3, where we had to increase these margins to $10\,\mathrm{cm}$. As mentioned before, our approach ensured that there was no collision in the end-effector trajectory. It could have been done inside the MPC with additional terms in the objective. However, this would increase complexity and possibly affect convergence rates. Knee collision is implicitly considered by the margin around obstacles. However, this could be addressed explicitely by planning the foot trajectory including the whole leg, as is the case in Zucker et al. (2011), although re-planning at $50\,\mathrm{Hz}$ might become unfeasible.

9.3 Comparison with the state of the art

Two recent works Grandia et al. (2023); Jenelten et al. (2022) present interesting similarities to our architecture. We can first observe the decomposed approach used in both cases with the foot location optimised outside the MPC. We have underlined the technical difficulties that would allow a objective comparison of the frameworks, although we would like to integrate the publicly available perception module proposed in Miki et al. (2022). However, this approach does not return a selection of potential candidates surfaces and only one candidate, such that further integration will be required to make the approach compatible with our framework.

9.4 Improvement points

As often occurs in model-based approaches to quadruped robots, the main limitation is the gait fixed beforehand. It would be interesting to replace the high-level planner with an acyclic planner to optimise the timing of contacts and the type of gait. This could lead to computing-time issues especially due to the increase in problem complexity. To take this further, it would be interesting to integrate this into a

global planner to achieve autonomous behaviour. Although we consider a stable walking gait necessary for the most challenging scenarios, we also acknowledge the potential benefits of implementing a trotting gait on the hardware. However, transitioning to a trotting gait, which works well in simulation, proved more challenging to implement on the robot. We believe that this limitation could be overcome with a faster controller and more integrative efforts. Finally, our foot placement is based on Raibert's heuristic, which is optimal on flat ground when considering an inverse linear pendulum model for the robot. We extend it in 3D which produces satisfying foot placement as we demonstrated in our experiment. Nevertheless, this remains a heuristic and does not ensure the position of feet is feasible in terms of torque power limit.

10 Conclusion

In this paper, we present a complete methodology for crossing challenging terrains; from terrain perception to locomotion generation and control. We have demonstrated our pipeline on various terrains, such as an industrial staircase or a parkour-type environment with a moving object in the scene. These experiments have allowed us to further validate our approach based on a sub-division of the global problem. First, a high-level planner formulated as a mixed-integer optimisation selects only the next surfaces of contact with a horizon of a few contacts (6 to 8 in our experiments). To achieve this, one must adapt the perception to extract relevant potential surfaces as convex planes. For motion generation, we rely on an efficient whole-body MPC and a linear state feedback controller. Collision-free trajectory and footstep adaptation on the high-level planner are optimised separately. The OCP problem is then strongly regularized around this reference end-effector trajectory while robot's posture is adapted by the MPC. This represents a wise choice of design to leverage whole-body optimisation capabilities. To move further, we would like to use a more complex high-level planner to optimise contact timing to cross even more challenging terrains, for example, dynamic motions that include jumping over gaps or obstacles.

Acknowledgements

This research was supported by (1) the European Commission under the Horizon 2020 project Memory of Motion (MEMMO, project ID: 780684) and (2) ROBOTEX 2.0 (Grants ROBOTEX ANR-10-EQPX-44-01 and TIRREX-ANR-21-ESRE-0015).

References

Aceituno-Cabezas B, Mastalli C, Dai H, Focchi M, Radulescu A, Caldwell DG, Cappelletto J, Grieco JC, Fernandez-Lopez G and Semini C (2018) Simultaneous contact, gait, and motion planning for robust multilegged locomotion via mixed-integer convex optimization. IEEE Robot. Automat. Lett. (RA-L) .

Amatucci L, Kim JH, Hwangbo J and Park HW (2022) Monte carlo tree search gait planner for non-gaited legged system control. In: IEEE Int. Conf. Rob. Autom. (ICRA). pp. 4701–4707. DOI: 10.1109/ICRA46639.2022.9812421.

azrafe7 (2013) hxGeomAlgo. URL https://github.com/azrafe7/hxGeomAlgo.

Baumgarte J (1972) Stabilization of constraints and integrals of motion in dynamical systems. Computer Methods in Applied Mechanics and Engineering 1: 1–16.

Bellicoso D, Bjelonic M, Wellhausen L, Holtmann K, Günther F, Tranzatto M, Fankhauser P and Hutter M (2018) Advances in real-world applications for legged robots. J. Field Robots. .

Blanco JL (2010) A tutorial on se(3) transformation parameterizations and on-manifold optimization. Technical report, University of Malaga.

Bloesch M, Hutter M, Hoepflinger M, Leutenegger S, Gehring C, Remy CD and Siegwart R (2012) State estimation for legged robots - consistent fusion of leg kinematics and IMU. In: Proceedings of Robotics: Science and Systems. Sydney, Australia. DOI:10.15607/RSS.2012.VIII.003.

Boston Dynamics (2016) Spot. https://www.bostondynamics.com/.

Bouyarmane K, Escande A, Lamiraux F and Kheddar A (2009) Potential field guide for humanoid multicontacts acyclic motion planning. In: IEEE Int. Conf. Rob. Autom. (ICRA). pp. 1165–1170. DOI:10.1109/ROBOT.2009.5152353.

Bretl T (2006) Motion planning of multi-limbed robots subject to equilibrium constraints: The free-climbing robot problem. The Int. J. of Rob. Res. (IJRR) 25.

Budhiraja R, Carpentier J, Mastalli C and Mansard N (2018) Differential dynamic programming for multi-phase rigid contact dynamics. In: IEEE Int. Conf. Hum. Rob. (ICHR). pp. 1–9. DOI:10.1109/HUMANOIDS.2018.8624925.

Campana M, Lamiraux F and Laumond JP (2016) A gradient-based path optimization method for motion planning. Advanced Robotics 30(17-18).

Carpentier J and Mansard N (2018) Multicontact Locomotion of Legged Robots. IEEE Trans. Robot. (T-RO) 34.

Carpentier J, Saurel G, Buondonno G, Mirabel J, Lamiraux F, Stasse O and Mansard N (2019) The pinocchio c++ library : A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives. In: IEEE Int. Sym. System Integration (SII). pp. 614–619. DOI:10.1109/SII.2019.8700380.

Coumans E and Bai Y (2016–2021) Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org.

Deits R and Tedrake R (2014) Footstep planning on uneven terrain with mixed-integer convex optimization. IEEE Int. Conf. Hum. Rob. (ICHR) .

Deits R and Tedrake R (2015) Efficient mixed-integer planning for uavs in cluttered environments. IEEE Int. Conf. Rob. Autom. (ICRA) .

Di Carlo J, Wensing PM, Katz B, Bledt G and Kim S (2018) Dynamic locomotion in the mit cheetah 3 through convex model-predictive control. In: IEEE/RSJ Int. Conf. Intell. Rob. Sys. (IROS). pp. 1–9. DOI:10.1109/IROS.2018.8594448.

Erez T and Todorov E (2012) Trajectory optimization for domains with contacts using inverse dynamics. IEEE/RSJ Int. Conf. Intell. Rob. Sys. (IROS) .

Escande A, Kheddar A, Miossec S and Garsault S (2009) Planning support contact-points for acyclic motions and experiments on hrp-2. In: Experimental Robotics. Springer, pp. 293–302.

Fallon M and Antone M (2019) Plane Seg – Robustly and Efficiently Extracting Contact Regions from Depth Data. URL

https://github.com/ori-drs/plane_seg.

Fankhauser P, Bjelonic M, Dario Bellicoso C, Miki T and Hutter M (2018a) Robust rough-terrain locomotion with a quadrupedal robot. In: IEEE Int. Conf. Rob. Autom. (ICRA). pp. 5761–5768. DOI:10.1109/ICRA.2018.8460731.

Fankhauser P, Bloesch M and Hutter M (2018b) Probabilistic terrain mapping for mobile robots with uncertain localization. IEEE Robot. Automat. Lett. (RA-L) 3. DOI:10.1109/LRA.2018.2849506.

Fernbach P, Tonneau S, Stasse O, Carpentier J and Taïx M (2020) C-CROC: Continuous and Convex Resolution of Centroidal Dynamic Trajectories for Legged Robots in Multicontact Scenarios. IEEE Trans. Robot. (T-RO) 36.

Gilbert E, Johnson D and Keerthi S (1988) A fast procedure for computing the distance between complex objects in three-dimensional space. IEEE Journal on Robotics and Automation 4(2): 193–203. DOI:10.1109/56.2083.

Grandia R, Jenelten F, Yang S, Farshidian F and Hutter M (2023) Perceptive locomotion through nonlinear model-predictive control. IEEE Trans. Robot. (T-RO) 39(5): 3402–3421. DOI:10.1109/TRO.2023.3275384.

Grandia R, Taylor AJ, Ames AD and Hutter M (2020) Multi-layered safety for legged robots via control barrier functions and model predictive control. CoRR abs/2011.00032.

Grimminger F, Meduri A, Khadiv M, Viereck J, Wüthrich M, Naveau M, Berenz V, Heim S, Widmaier F, Flayols T, Fiene J, Badri-Spröwitz A and Righetti L (2020) An open torque-controlled modular robot architecture for legged locomotion research. IEEE Robot. Automat. Lett. (RA-L) 5(2).

Hauser K, Bretl T, Latombe JC, Harada K and Wilcox B (2008) Motion planning for legged robots on varied terrain. The Int. J. of Rob. Res. (IJRR) 27(11-12).

Hutter M, Gehring C, Jud D, Lauber A, Bellicoso CD, Tsounis V, Hwangbo J, Bodie K, Fankhauser P, Bloesch M, Diethelm R, Bachmann S, Melzer A and Hoepflinger M (2016) Anymal - a highly mobile and dynamic quadrupedal robot. In: IEEE/RSJ Int. Conf. Intell. Rob. Sys. (IROS). pp. 38–44. DOI:10.1109/IROS.2016.7758092.

Hutter M, Gehring C, Lauber A, Gunther F, Bellicoso CD, Tsounis V, Fankhauser P, Diethelm R, Bachmann S, Bloesch M, Kolvenbach H, Bjelonic M, Isler L and Meyer K (2017) Anymal - toward legged robots for harsh environments. Advanced Robotics 31(17).

Jenelten F, Grandia R, Farshidian F and Hutter M (2022) TAMOLS: Terrain-aware motion optimization for legged systems. IEEE Trans. Robot. (T-RO) 38.

Kalakrishnan M, Buchli J, Pastor P, Mistry M and Schaal S (2011) Learning, planning, and control for quadruped locomotion over challenging terrain. The Int. J. of Rob. Res. (IJRR) 30.

Kim, Carlo JD, Katz B, Bledt G and Kim S (2019) Highly dynamic quadruped locomotion via whole-body impulse control and model predictive control. arXiv: 1909.06586 .

Kolter JZ, Rodgers MP and Ng AY (2008) A control architecture for quadruped locomotion over rough terrain. In: IEEE Int. Conf. Rob. Autom. (ICRA). pp. 811–818. DOI:10.1109/ROBOT.2008.4543305.

Léziart PA, Flayols T, Grimminger F, Mansard N and Souères P (2021) Implementation of a reactive walking controller for the new open-hardware quadruped solo-12. In: IEEE Int. Conf. Rob. Autom. (ICRA). pp. 5007–5013. DOI:10.1109/ICRA48506.2021.9561559.

Lofberg J (2022) Big-m and convex hulls. https://yalmip.github.io/tutorial/bigmandconvexhulls.

Léziart PA, Corbères T, Flayols T, Tonneau S, Mansard N and Souères P (2022) Improved Control Scheme for the Solo Quadruped and Experimental Comparison of Model Predictive Controllers. IEEE Robot. Automat. Lett. (RA-L) 7.

Mastalli C, Budhiraja R, Merkt W, Saurel G, Hammoud B, Naveau M, Carpentier J, Righetti L, Vijayakumar S and Mansard N (2020a) Crocoddyl: An efficient and versatile framework for multi-contact optimal control. In: IEEE Int. Conf. Rob. Autom. (ICRA). pp. 2536–2542. DOI:10.1109/ICRA40945.2020.9196673.

Mastalli C, Chhatoi SP, Corbères T, Tonneau S and Vijayakumar S (2023) Inverse-Dynamics MPC via Nullspace Resolution. IEEE Trans. Robot. (T-RO) .

Mastalli C, Havoutis I, Focchi M, Caldwell DG and Semini C (2020b) Motion planning for quadrupedal locomotion: Coupled planning, terrain mapping, and whole-body control. IEEE Trans. Robot. (T-RO) 36(6): 1635–1648. DOI:10.1109/TRO.2020.3003464.

Mastalli C, Merkt W, Xin G, Shim J, Mistry M, Havoutis I and Vijayakumar S (2022) Agile maneuvers in legged robots: a predictive control approach. DOI:10.48550/ARXIV.2203.07554. URL https://arxiv.org/abs/2203.07554.

Melon O, Geisert M, Surovik D, Havoutis I and Fallon M (2020) Reliable trajectories for dynamic quadrupeds using analytical costs and learned initializations. In: IEEE Int. Conf. Rob. Autom. (ICRA). pp. 1410–1416. DOI:10.1109/ICRA40945.2020.9196562.

Melon O, Orsolino R, Surovik D, Geisert M, Havoutis I and Fallon M (2021) Receding-horizon perceptive trajectory optimization for dynamic legged locomotion with learned initialization. In: IEEE Int. Conf. Rob. Autom. (ICRA). pp. 9805–9811. DOI:10.1109/ICRA48506.2021.9560794.

Miki T, Lee J, Hwangbo J, Wellhausen L, Koltun V and Hutter M (2022) Learning robust perceptive locomotion for quadrupedal robots in the wild. Science Robotics 7.

Mordatch I, Todorov E and Popovic Z (2012) Discovery of complex behaviors through contact-invariant optimization. ACM Trans. Graphics 31.

Murooka M, Kumagai I, Morisawa M, Kanehiro F and Kheddar A (2021) Humanoid Loco-Manipulation Planning Based on Graph Search and Reachability Maps. IEEE Robot. Automat. Lett. (RA-L) 6(2).

P-B W (2006) Holonomy and Nonholonomy in the Dynamics of Articulated Motion. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 411–425. DOI:10.1007/978-3-540-36119-0_20.

Ponton B, Khadiv M, Meduri A and Righetti L (2021) Efficient Multicontact Pattern Generation With Sequential Convex Approximations of the Centroidal Dynamics. IEEE Trans. Robot. (T-RO) 37.

Posa M, Cantu C and Tedrake R (2014) A direct method for trajectory optimization of rigid bodies through contact. The Int. J. of Rob. Res. (IJRR) 33.

Quigley M (2009) Ros: an open-source robot operating system. In: IEEE Int. Conf. on Rob. and Aut. (ICRA).

Raibert MH (1986) Legged robots that balance. MIT press.

Risbourg F, Corbères T, Léziart PA, Flayols T, Mansard N and Tonneau S (2022) Real-time footstep planning and control of the solo quadruped robot in 3d environments. In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 12950–12956. DOI:10.1109/IROS47612.2022.9981539.

Short A and Bandyopadhyay T (2018) Legged motion planning in complex three-dimensional environments. IEEE Robot. Automat. Lett. (RA-L) 3(1): 29–36. DOI:10.1109/LRA.2017.2728200.

Song D, Fernbach P, Flayols T, Del Prete A, Mansard N, Tonneau S and Kim YJ (2021) Solving Footstep Planning as a Feasibility Problem Using L1-Norm Minimization. IEEE Robot. Automat. Lett. (RA-L) 6.

Tonneau S, Del Prete A, Pettré J, Park C, Manocha D and Mansard N (2018a) An efficient acyclic contact planner for multiped robots. IEEE Trans. Robot. (T-RO) 34(3).

Tonneau S, Fernbach P, del Prete A, Pettré J and Mansard N (2018b) 2PAC: Two Point Attractors for Center of Mass Trajectories in Multi Contact Scenarios. ACM Trans. Graphics 37(5).

Unitree (2021) B1. https://www.unitree.com/b1/. Accessed: December 2022.

Visvalingam M and Whyatt JD (1993) Line generalisation by repeated elimination of points. Cartographic Journal 30: 46–51.

Winkler AW, Bellicoso DC, Hutter M and Buchli J (2018) Gait and trajectory optimization for legged systems through phase-based end-effector parameterization. IEEE Robot. Automat. Lett. (RA-L) 3.

Zucker M, Ratliff N, Stolle M, Chestnutt J, Bagnell JA, Atkeson CG and Kuffner J (2011) Optimization and learning for rough terrain legged locomotion. The Int. J. of Rob. Res. (IJRR) 30.