Multi-controller multi-objective locomotion planning for legged robots

Martim Brandão, Maurice Fallon, Ioannis Havoutis

***Abstract*— Different legged robot locomotion controllers offer different advantages; from speed of motion to energy, computational demand, safety and others. In this paper we propose a method for planning locomotion with multiple controllers and sub-planners, explicitly considering the multi-objective nature of the legged locomotion planning problem. The planner first obtains body paths extended with a choice of controller or sub-planner, and then fills the gaps by sub-planning. The method leads to paths with a mix of static and dynamic walking which only plan footsteps where necessary. We show that our approach is faster than pure footstep planning methods both in computation (2x) and mission time (1.4x), and safer than pure dynamic-walking methods. In addition, we propose two methods for aggregating the multiple objectives in search-based planning and reach desirable trade-offs without weight tuning. We show that they reach desirable Pareto-optimal solutions up to 8x faster than fairly-tuned traditional weighted-sum methods. Our conclusions are drawn from a combination of planning, physics simulation, and real robot experiments.**

I. Introduction

Legged robots have the potential to be deployed in a wide range of real-world domains due to their high versatility. Such systems can offer unmatched mobility over complex terrain as they don't require (smooth) continuous support and naturally handle environments built with humans in mind. One key strength of modern legged robots, similar to their biological counterparts, is that they can change and adapt their gait to suit the environment at hand. For example, the ANYmal quadruped robot can utilize a trotting gait to swiftly traverse flat areas, use a slower walking gait to cross more challenging terrain, and use a planning-focused controller to carefully pick footholds over, for example, gaps, stepping stones and stairs.

In this paper we present an approach to legged locomotion planning using multiple controllers and objectives. Much of the work on legged locomotion has focused on developing single controllers and planners. We focus on planning locomotion when multiple controllers and sub-planners are available. The contributions of our work are the following:

- We propose and evaluate a legged locomotion planning architecture that plans the use of controllers and complex sub-planners together with robot trajectories to increase mission efficiency and safety.
- We propose and evaluate the use of utopian and lexicographic cost aggregation methods on multi-objective settings, that lead to faster convergence to desirable Pareto-optimal solutions without weight tuning
- We show the method can plan safe, fast, versatile quadruped robot locomotion over realistic environments with gaps, bumps, obstacles, steps, and computationally challenging environments such as stepping-stones.

This work was supported by the UKRI/EPSRC ORCA Hub [EP/R026173/1], Robust Legged Locomotion [EP/S002383/1] and the EU H2020 Project THING. It was conducted as part of ANYmal Research, a community to advance legged robotics. The authors are with the Oxford Robotics Institute, University of Oxford, UK.

Fig. 1: A testing facility used for fire-fighter training, as a mock-up of an oil rig. Among other scenarios, we evaluate our method on this environment both in simulation and the actual location. The bottom row highlights different parts of a 25m long locomotion plan where 3 different modes of locomotion are required. The accompanying video presents the full experiment.

We demonstrate our approach on the ANYmal quadruped robot [1] with a combination of simulated and real world examples, and focus on our goal to deploy the system in an offshore facility similar to the one depicted in Fig.1.

II. Related work

Legged locomotion planning has traditionally been tackled with search, sampling and optimization-based methods. Most approaches require planning full-body or contact motion to execute on a single controller, which might be unnecessary in certain terrain conditions; [2], [3] compute footstep positions, [4], [5] searches over rough body paths and selects locally-best footholds, [6] uses hierarchical trajectory optimization to plan dynamic motion and contact sequences, while [7] jointly optimizes body motion and foothold locations.

In this paper we instead focus on planning locomotion when multiple controllers and sub-planners are available, which we call "controllers", "gaits" or "modes" interchangeably. The problem is related to planning with a library of motion primitives: where the choice of primitive is discrete

but each primitive is parameterized in continuous space. [8] used a search-based planner over a discrete set of primitives which were randomly perturbed from a continuous space. [9] applied a sampling-based planner with motion primitive projections to the same problem. Another related approach is adaptive dimensionality planning [10], where a single graph search is made on a hybrid state space with multiple low-dimensional subspaces, a high-level subspace, and mappings between them. The approach has been applied to legged robots for automatically planning over crawling, bipedal and climbing motions [11], although currently reported results have low planning success rates.

When simulating a controller or executing a sub-planner within the planning stage is unfeasible or time consuming, it makes sense solve the multi-modal-planning problem hierarchically [12], [13]. In such an approach, the first hierarchy level plans only a guide path and a mode, and the second level executes the controllers or sub-planners over sections of that path. This is the approach we also adopt in this paper. Similarly to [13], we consider modes which are directly executable without further sub-planning depending on environment statistics (e.g. blind trotting on flat ground). In the manner of [12], we first plan a path over robot base motion and mode, followed by execution and sub-planning where appropriate. Compared to [12] we introduce feasibility constraints at mode transitions and explicitly consider the multi-objective nature of the problem. Another promising method is the use of mixed-integer optimization for such hybrid problems, as was used for joint-planning of contact order and position in [14] - although it requires potentially expensive or semi-supervised methods for decomposing the environment into convex shapes [15].

While the most common way to solve multi-objective problems in the robotics literature is through weighted-sums [16], [17], some efforts have been made for sound no-preference sampling-based planning [18] and Pareto-optimal planning [19], [20]. In this paper we introduce a new cost aggregation method for a typical optimization setting in robotics: of a ranked set of objectives which have to be optimized in order of preference (usually called hierarchical optimization). We also empirically evaluate the behavior of no-preference and ranked-preference multi-objective optimization over time on an anytime search-based planner and compare it to traditional weighted-sum methods.

Finally, there is also a connection of this work to multi-task learning [21] and hierarchical reinforcement learning [22], [23]. For example, [23] learns a high-level policy neural network which selects lower-level "skills" that are committed to for a certain amount of steps. This is a similar hierarchical setup to ours, the difference being that the high-level (and potentially low-level) policies are learned instead of solved through classical planning methods.

III. Multi-controller multi-objective planner

A. Problem statement

Let $\mathcal{S}$ be the state space of a locomotion planning problem and $\mathcal{S}_{\text{free}} \subset \mathcal{S}$ be the set of states free of collision with obstacles. We define a set of robot locomotion controllers $\mathcal{M} = \{m_1, m_2, ..., m_M\}$, where the term "controller" is used in a loose sense to represent anything from a periodic-gait locomotion controller (e.g. a velocity-based trotting controller), requiring just desired direction input, to a more complex footstep or whole-body sub-planner used together with a joint-level controller.

The goal in this paper is to obtain optimal paths (i.e. sequences of states) $\pi = s_1, ..., s_n$, where $s_i \in (\mathcal{S}_{\text{free}} \times \mathcal{M}) \forall i$. The path π simultaneously minimizes a set of cost functions

$$\underset{\pi}{\text{minimize}} \quad (\ c_1(\pi)\ ,\ ...\ ,\ c_K(\pi)\ ), \tag{1}$$

where

$$c_k(\pi) = \sum\nolimits_{i=1}^{n-1} c_k(s_i, s_{i+1}), \tag{2}$$

and s_n is constrained to be the goal state $s_n = s_{\text{goal}}$. Additionally, $c_k(s_i, s_{i+1})$ can be infinite in case of invalid states such as those in collision, so in practice our problem is that of constrained multi-objective optimization. The typical approach to the multi-objective problem (1) is to optimize a weighted-sum of the cost functions, i.e. $\sum_{k=1}^{K} w_k c_k(\pi)$ where $w_k \in \mathbb{R}$, although in this paper (Section III-D) we will also look into alternative aggregation methods.

As mentioned previously, the motivation for including a choice of controller in the search space comes from the observation that different controllers may have different advantages depending on terrain features at hand, and so should be the object of planning. Importantly, this will allow us to save computational resources by running a complex sub-planner only when required, and to reduce mission time by running faster dynamic controllers, for example a walking or running trot, wherever possible and optimal.

B. State spaces and controllers

In this paper we consider a high-level search space $\mathcal{S}^{\text{xyzt}} = \{(x, y, z, \theta) | (x, y, z) \in \mathbb{R}^3, \theta \in SO(2)\}$, and a set of controllers $\mathcal{M} = \{m_{\text{Walking}}, m_{\text{Trotting}}, m_{\text{FootPlan}}\}$ where:

1) m_{Walking} refers to a statically stable walking gait controller without terrain geometry information [24]
2) m_{Trotting} refers to a dynamic velocity-based trotting controller without terrain geometry information [24]
3) m_{FootPlan} refers to a footstep planner which computes a sequence of statically stable footsteps based on a 3D map of the environment, which are later passed to and executed by a walking controller. This planner is explained in detail next.

Each path section where $m = m_{\text{FootPlan}}$ will be further refined by a footstep planner involving the same search-based planning method as the high-level planner. The search space of this planner is $\mathcal{S}^{\text{stance}} = \mathbb{R}^{3 \times L} \times \{1, ..., V\}$, where L is the number of limbs, and where the last index $1, ..., V$ indicates the node in a gait transition graph G with V nodes. For simplicity, we use a fixed cyclic graph that moves one limb at a time (i.e. $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 1 \rightarrow 2 \rightarrow ...$), though we could also include other transitions, such as a trotting or bounding cycle. State expansion in A* involves moving the next limb(s) in G across a $X \times Y$ grid, while the height of

the contact is obtained automatically by projection onto the environment model, a heightmap in our case.

For both high-level and footstep planning we use a simplified robot model to estimate state and transition feasibility. Inspired by [25], we approximate the robot by a set of spheres for the body of the robot, that need to be collision-free, and a set of spheres at the contact points, that need to be partly in collision. During footstep planning in $\mathcal{S}^{\text{stance}}$ the positions of the contact spheres are taken directly from the state's limb positions. During high-level planning in $\mathcal{S}^{\text{xyzt}}$ they are fixed with respect to the base of the robot, and set according to the nominal posture associated with controller m of state s. We consider a state feasible if all contact spheres partially collide with the environment and the body spheres do not, and if distances between contact positions are within some feasible interval with respect to the limb's workspace. Additionally, we consider constraints on mode transitions, as each controller can usually be started only when certain conditions are satisfied on the robot's state. For our particular choice of controllers this amounts to satisfying a feasible nominal posture at mode transitions, which means we enforce footstep planning trajectories to end in nominal stances or in close-to-goal stances where projection to nominal state is feasible (i.e. all feet partially collide with the environment model on a locally flat surface).

During execution, transitions between states in $\mathcal{S}^{\text{stance}}$ are made by square-interpolated trajectories on the moving end-effectors and QP-based optimization of the base pose [26].

C. Objective functions

We consider the simultaneous optimization of multiple conflicting objectives:

1) Energy cost:

$$c_{\text{ener}}(s_i, s_{i+1}) = \Psi \Delta t(s_i, s_{i+1}), \tag{3}$$

where Ψ represents electrical power and $\Delta t(s_i, s_{i+1})$ is the time required to execute the state transition. To approximate this duration we use assume each controller executes trajectories at a constant characteristic velocity $\boldsymbol{v}(m) = (v_x, v_y, v_z)$ defined on the reference frame of the robot's body. Additionally, we assume each controller may require extra time for computation while the robot is standing still:

$$\Delta t_{\text{comp}}(s_i, s_{i+1}) = \begin{cases} 0, & m_i = m_{\text{Walking}} \\ 0, & m_i = m_{\text{Trotting}} \\ t_{\text{FootPlan}}, & m_i = m_{\text{FootPlan}}, \end{cases} \tag{4}$$

where t_{FootPlan} is empirically measured over a set of toy problems. Finally, let (d_x, d_y, d_z) be the distance covered by the COM between s_i and s_{i+1}, seen from the robot's reference frame at s_i. We estimate the total duration of a state transition by a Manhattan-distance upper bound:

$$\Delta t(s_i, s_{i+1}) = \frac{d_x}{v_x} + \frac{d_x}{v_y} + \frac{d_x}{v_z} + \Delta t_{\text{comp}}. \tag{5}$$

2) Feasibility cost (expected success probability):

$$c_{\text{feasibility}}(s_i, s_{i+1}) = 1 - P(S(s_i, s_{i+1})F(s_i, s_{i+1})), \tag{6}$$

where $P(F(s_i, s_{i+1}))$ represents the probability that both states are statically feasible (i.e. reachable and providing enough contact area), and $S(s_i, s_{i+1})$ represents successful motion execution on those states.

For the experiments in this paper we model these quantities by an average over both states

$$P(S(s_i, s_{i+1})F(s_i, s_{i+1})) := \frac{P(S(s_i)F(s_i)) + P(S(s_{i+1})F(s_{i+1}))}{2}, \tag{7}$$

and then for each state we compute,

$$P(S(s)F(s)) = P(S(s)|F(s))P(F(s)) \tag{8}$$

i.e. the probability of motion around a state being successful given that the state is statically feasible, and the probability of a state being statically feasible, which we define as

$$P(S(s)|F(s)) = \begin{cases} \Pi_{i \in \text{limbs}} T_i R_i, & m = m_{\text{Walking}} \\ \Pi_{i \in \text{limbs}} (T_i R_i)^2, & m = m_{\text{Trotting}} \\ \sigma_{\text{FootPlan}}, & m = m_{\text{FootPlan}}, \end{cases} \tag{9}$$

$$P(F(s)) = \begin{cases} \Pi_{i \in \text{limbs}} \frac{\rho(l_i)}{\rho_{\max}}, & s \text{ reachable} \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

where R_i and T_i are average map roughness and traversability heuristics at contact i (averaged over a local radius around the contact point), defined as in [27]. We assume that motion from the footstep planner has constant success rate σ_{FootPlan} independent of ground geometry as long as stances are statically stable. Lastly, $\rho(l_i)$ is point density (number of points per area) at each contact point l_i and $\rho_{\max}$ is the ideal density given by map resolution.

3) WiFi signal strength, for remote and shared-autonomy operations:

$$c_{\text{wifi}}(s_i, s_{i+1}) = N_{\text{walls}}(\boldsymbol{x}_{\text{teleoperator}}, \boldsymbol{x}(s_{i+1})), \tag{11}$$

where the function $N_{\text{walls}}(\boldsymbol{x}_1, \boldsymbol{x}_2)$ involves performing ray-casting between $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ and counting the number of intersections with map geometry. The function is the environment-dependent term of the model of WiFi signal strength of [28]. The objective is to provide paths that keep signal strength high, by preferring to keep line-of-sight visibility of the communication station and thus avoiding delays in communication with the operating station. The same function would make sense in scenarios where it is desirable to visualize the robot from an external camera (e.g. security camera in an industrial building).

D. Multi-objective search

Although the typical approach to the multi-objective problem (1) is to optimize a weighted-sum of the cost functions, i.e. $\sum_{k=1}^{K} w_k c_k(\pi)$ where $w_k \in \mathbb{R}$, in this paper we look into alternative methods that avoid the need of tuning w_k in the case of no-preference or lexicographic optimization, as well as taking into account the bounds of each cost function.

In cases where we care equally about all cost functions we use the no-preference method from the multi-objective optimization literature [29] and solve a new approximate optimization problem which minimizes the scalar cost:

$$c^{\text{utopia}}(\pi) = \sum_{i=1}^{n-1} \left(\epsilon + \sum_{k=1}^{K} \frac{c_k(s_i, s_{i+1}) - z_k^{**}}{z_k^{\text{nad}} - z_k^{**}} \right), \tag{12}$$

where z_k^{**} and z_k^{nad} are respectively lower and upper bounds[1] on $c_k(s_i, s_{i+1})$:

$$z_k^{**} = \min_{a,b \in \mathcal{S}} c_k(a,b), \qquad z_k^{\text{nad}} = \max_{a,b \in \mathcal{S}} c_k(a,b), \tag{13}$$

and ϵ is a small constant that promotes few-state-transition paths and avoids issues of zero-cost transitions in A*. In this approach, we are therefore normalizing the cost functions such as to obtain scalar transition-costs of ϵ when all costs are minimum, and $K + \epsilon$ when all are maximum. In practice the upper and lower bounds can be estimated by inspection of the cost functions (3) (6) (11) by respectively computing the maximum and minimum possible values of each function across all modes. All cost functions we use are bounded with known bounds for each mode and therefore we only have to take maximum and minimum values over modes. Bounds on d come from environment discretization, $\boldsymbol{v}$ and Δt are known, bounds on T and R are $[0, 1]$, and we assume $N_{\text{walls}} \in [0, 5]$ given the geometry and scale of our experimental scenarios[2].

We also consider cases where a preference ranking exists, e.g. maximize feasibility first, then minimize energy consumption within the space that keeps feasibility unchanged. Based on the no-preference method, we propose to use a hierarchical optimization approach on the normalized costs using the common 10^k rule for scalarization [16], [17]:

$$c^{\text{lexiSoft}}(\pi) = \sum_{i=1}^{n-1} \left(\epsilon + \sum_{k=1}^{K} 10^k \frac{c_k(s_i, s_{i+1}) - z_k^{**}}{z_k^{\text{nad}} - z_k^{**}} \right), \tag{14}$$

We will call this function *lexiSoft* (short for "soft lexicographic" cost aggregation).

In the experimental section we compare these two options to more traditional cost-aggregation schemes through weighted-sums. To make the comparison fair, in particular in order to obtain the same lower-bound of cost values in an automatic fashion, we use the following normalization strategy:

$$c^{\text{normalized}}(\pi) = \sum_{i=1}^{n-1} \left(\epsilon + \sum_{k=1}^{K} \frac{c_k(s_i, s_{i+1})}{z_k^{**}} - 1 \right), \tag{15}$$

so that the costs of state transitions at their minimum will be equal to ϵ for all methods.

E. Heuristic

Following [30], we compute heuristics as lower bounds of cost-per-Euclidean-distance (i.e. cost-of-transport), multiplied by Euclidean distance to the goal state s_{goal}. Since ϵ is the lower bound of all cost-aggregation methods we consider, the lower bound cost-per-distance is $\frac{\epsilon}{\delta}$ where δ is the largest side of discretized cells. Our heuristic cost-to-goal is therefore

$$h(s_i, s_{\text{goal}}) = \frac{||s_i - s_{\text{goal}}||\epsilon}{\delta}. \tag{16}$$

IV. Results

A. Experimental setup

In the first part of this section we analyze the performance of our planning method with a simulated model of the ANYmal robot, on the set of simulated environments described next. Then in Section IV-E we validate our approach with an experiment with the real robot.

Limb reachability checks were implemented by upper and lower bounds on distances between each limb and the body, as well as distances between (left/right, fore/hind, diagonal) limbs as per the default values used within the robot's software. We evaluated our locomotion planning method on several simulated scenarios, shown in Fig. 4:

- "Steps & barrier". This scenario was manually designed to showcase the advantages of each controller and to involve geometry relevant to our cost functions (a bump, steps, and a barrier that lowers line-of-sight to a communication station).
- "Stepping stones". This scenario tests the feasibility of the footstep planner.
- "Industrial floor". This is 3D model of an industrial floor used in the ARGOS challenge for mobile robotics [31].
- "Oil rig". This is a point cloud of a mock-up oil rig shown in Fig. 1, acquired at the Fire Service College, Moreton-in-Marsh, UK.

We used Anytime Repairing A* [32] as implemented in the SBPL library [33] to solve all search problems.

We analyze the performance of utopian- and lexiSoft-aggregated costs and compare them to the traditional weighted-cost baseline. For lexiSoft, we prioritize the feasibility cost, so the contribution of feasibility costs is 10 times greater than that of energy. $\epsilon = 0.1$ is the same for all cost aggregation methods. We used a grid of $X \times Y = 7 \times 7$ cells around nominal contact positions for state expansion in S^{stance}, we obtained $t_{\text{FootPlan}} = 10\text{s}$ experimentally from our scenarios and set $\sigma_{\text{FootPlan}} = 1$.

[1] Note that we have chosen this notation instead of, for example, $\overline{z}_k$ and $\underline{z}_k$, to be consistent with the notation in [29].

[2] This means that in our experiments we are assuming a maximum of 5 walls between the robot and the WiFi router, which is appropriate for the scale of the missions we consider, e.g. Fig. 4.

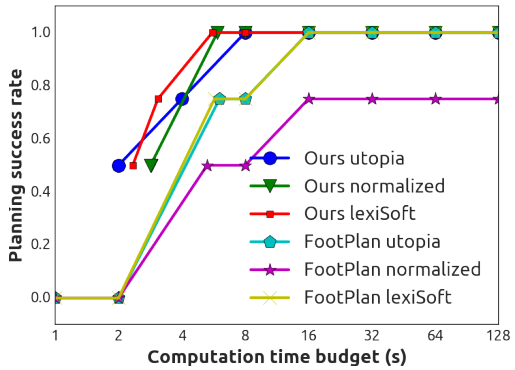


Fig. 2: Success rate of planning multi-controller (ours) and footstep-only (FootPlan) paths, for different computation time budgets and cost aggregation functions.

B. Evaluation of the multi-controller planner

In our first experiment we evaluated our multi-controller planning architecture. Fig. 2 shows a comparison of the success rate of our planner and a footstep-planner. The same footstep planner is used within our method whenever it is chosen at the first planning stage. Reported success rate is the percentage of problems for which a solution is found. Each problem corresponds to one scenario and a start/goal state. The distances between start and goal states were 5m, 8m, 5m and 16m, for the steps&barrier, stones, industrial, and oil rig scenarios respectively. Fig. 2 shows that, whatever the cost-aggregation method chosen, our method finds feasible trajectories with less computation than pure footstep planning. The times reported in Fig. 2 refer to planning only and do not include environment pre-processing time as this varies greatly across the different scenarios (i.e. the time to build a heightmap from a large point cloud in the case of the "oil rig" scenario, from a mesh in the "industrial floor" scenario, or to load it from a file in the other scenarios). Our method can solve 50% of the scenarios within 2 seconds. For computation budgets of 4 seconds and higher, our planner can solve all scenarios around twice as fast as a pure footstep planner even though they also include footstep planning regions as we will show later. The reason for this is the hierarchical nature of our planner, that quickly finds paths in a small state-space of position-and-controller, followed by footstep planning only in regions where that is necessary. On the pure footstep planning setting, the utopian and lexiSoft functions lead to higher success rates than normalized, which is likely due to a more uniform distribution of costs when compared to the normalized method. Because weighted-sum aggregation does not consider the upper bounds of the individual cost functions, the aggregated values in (15) can be very large when $z_k^{**} \ll z_k^{\text{nad}}$. This will lead to more exploration by A* when compared to utopia and lexiSoft. Interestingly, the figure suggests our method is more robust to the choice of cost-aggregation function, since success rates are more similar across functions when compared to a footstep planner. This could again be due to the hierarchical nature of the planner that reduces the length of the path over which footstep planning is run.

TABLE I: Mission performance, sim. industrial scenario

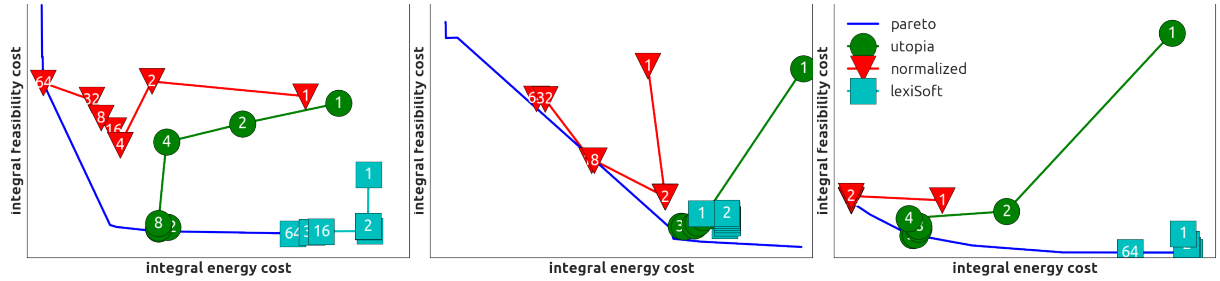
Method	Failures*	Mission time (min:sec)
Trotting	2	1:05
Walking	0	5:21
FootPlan	0	4:25
Ours (lexiSoft)	0	3:57
Ours (utopia)	0	3:09

*Number of times any contact enters a slipping state during execution.

We then executed the obtained plans in the industrial floor scenario, using the Gazebo physics simulator [34], to obtain mission time and a measure of plan safety. Mission time was measured as the time to go from start to goal state, and lack of safety was measured by the number of times that contacts entered a slipping state in Gazebo. Note that blind controllers execute paths successfully in simulation even if feet get briefly stuck in gaps (leading to slippage, high torques and velocity peaks), which would hardly be executable on the real robot. We use contact slipping in Gazebo as a proxy for failure. Table I shows the results on the industrial floor scenario. The blind trotting controller completed the mission in 1 minute but with failures (i.e. feet stuck in gaps), the blind walking controller successfully completed the mission in 5:20, and the executed footstep-only plan in 4:25. Our utopian method was the fastest feasible method (3:09) while lexiSoft, as it prioritizes safety, completed in 3:57 - still faster than a footstep-only plan due to slight use of trotting.

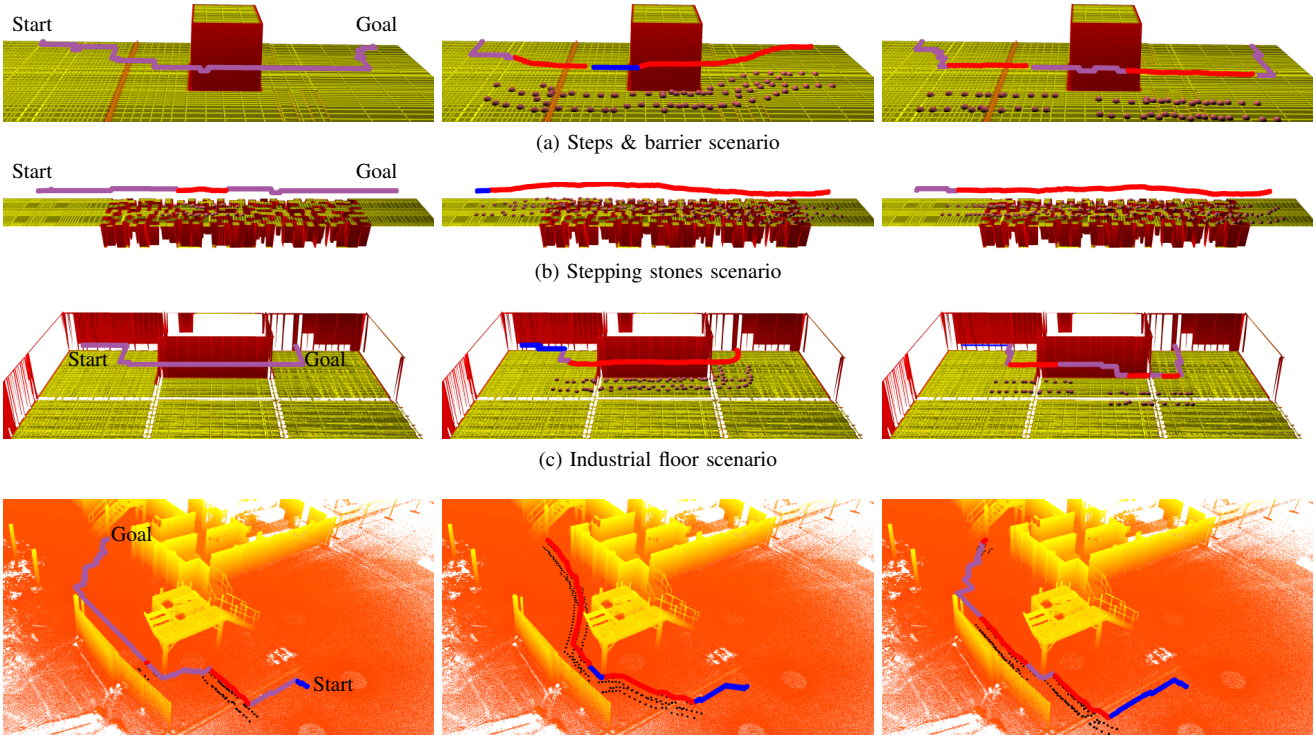
C. Analysis of multi-objective aggregation methods

In another experiment we compared the performance of the different cost aggregation functions in an anytime planning scenario. The idea is to evaluate the Pareto-optimality of the different solutions as the computation time budget is increased, and empirically characterize the paths traced by the different methods in objective space. For simplicity of visualization and computation we solved the same problems as before on only two cost functions: energy and feasibility. We estimated the Pareto-curve of each problem by solving it for a large computation budget (128 seconds) over multiple times using different cost weights, i.e. $c_1 + w_2 c_2$ with $w_2 = 10^{i/2}, i = -6, -5, -4, ..., 6$, and different cost aggregation functions: utopia, normalized-sum, and sum. We then computed the epigraph of all the points (c_1, c_2) obtained from those experiments, which we plot in Fig. 3 as the Pareto-curve. We ran our method in the space $\mathcal{S}^{xyzt} \times \mathcal{M}$, without running the footstep planner, for different computation time budgets, and report the resulting cost values overlaid by computation time in Fig. 3. On the one hand, the figure shows that traditional weighted-sum aggregation can take up to 64 seconds to reach the Pareto curve in these problems, while it has a slight preference towards one of the costs (energy). This bias exists because the weighted-sum aggregation scheme does not take into account the upper bounds of the cost functions. On the other hand, utopia-aggregated costs move in the direction of the utopia point (i.e. lower left corner where both costs are low) and reach the Pareto curve within 4 to 8 seconds of ARA* repairing time. Our proposed lexiSoft method, which has an explicit preference for feasibility,



(a) Steps & barrier scenario (b) Stepping stones scenario (c) Industrial floor scenario

Fig. 3: Pareto curves and solutions using a sum, utopian, and lexicographic-based cost aggregation. Computation time budget (in seconds) is indicated on top of each point. Lower total costs (to the bottom right) are better, and axes go linearly from minimum to maximum attainable path cost values.



(a) Steps & barrier scenario

(b) Stepping stones scenario

(c) Industrial floor scenario

(d) Oil rig scenario

Fig. 4: Left to right: Multi-controller paths obtained with normalized weighted-sum, lexiSoft and utopia cost aggregation. Coloring of the map is based on traversability (red is less traversable) for the first 3 rows and based on height for the last row. Coloring of paths is based on chosen controller (blue for walking, purple for trotting, red for footstep plans).

shows the desirable property of quick convergence to the Pareto curve (within 2 seconds) and then moving along the curve towards energy reduction as more computation time is allowed - without visibly sacrificing feasibility. Since both utopian and normalized aggregations are attempts to automatically obtain no-preference solutions to the original multi-objective problem, without weight tuning, the figure shows that utopia-aggregation is more suited to that goal.

D. Qualitative analysis of paths

Next we show the actual paths obtained by our method on the different scenarios. Fig. 4 shows the planned COM paths, color-coded according to the chosen controller at each time, as well as footsteps planned within m_{FootPlan} sections. The figure shows that weighted-sum solutions involve more trotting, lexiSoft involve more footstep planning (which is expected as it prioritizes feasibility), and utopian solutions are more balanced. The latter leads to trotting only on flat ground and footstep planning over gaps and steps. Weighted-sum results, on the other hand, lead to trotting even over gaps and steps, which would lead to trips and falls as we will analyze later on. This is because of the large gain in energetic cost (i.e. trot has low energy per distance because it progresses very quickly) even if weights are fairly chosen, for the reasons discussed in the previous section (i.e. it does not consider the upper bounds of the cost functions). Fig. 4

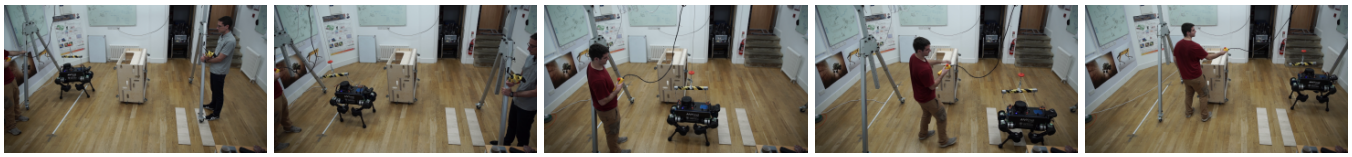


Fig. 5: Utopian trajectory executed on the real robot (same scenario as the simulated "Steps & barrier"). Left to right: footsteps over bump, fast trotting within line of sight, slow walking over the wooden planks, sideways trot to the goal pose.

Fig. 6: Utopian trajectory executed on the real robot and oil rig scenario. Left to right and top to bottom: footsteps over slab, then fast trotting towards goal.

also shows that utopian paths are the only ones that respect WiFi signal preferences. Note that we assume the WiFi communication point to the robot is placed at the robot's initial position. This is a valid assumption as the real-world robot can start from a charging station that also serves as a communication point. In both the barrier and industrial floor scenarios, the figure shows that utopian trajectories keep a direct-line-of-sight with the starting point on the left of the images. Instead of moving on a straight path to the goal after overcoming an obstacle, utopian trajectories keep line-of-sight for longer and only move to the goal sideways at the end.

We also executed the utopian trajectory of the "oil rig" scenario within Gazebo. Fig. 1 shows the robot successfully executing the sequence of footsteps required to go over a 0.2m block, followed by trotting under the facility until the goal. The full results are also shown in the accompanying video for clarity.

E. Real robot experiments

Finally, to validate our planning approach we executed the utopian trajectory of the "oil rig" and "steps & barrier" scenarios on the real robot.

For the "steps & barrier" scenario, we arranged a barrier-obstacle, bump (2cm thick metal piece) and steps (wooden planks) in the laboratory in the same way as in Fig. 4. The robot successfully executed the path as seen in Fig. 5. It moved the front feet over the bump, trotted sideways, moved the hind feet over the bump, trotted forwards while the surface was flat, placed feet carefully on or between the wooden planks and then trotted sideways on flat ground. We refer the reader to the accompanying video for the whole experiment.

For the oil rig scenario, we tested at the actual location at the Fire Service College. We manually localized the robot with respect to the previously acquired global map which we used to generate the plan. We executed the plan open loop, i.e. without re-localizing against the global map. Fig. 6 shows that the robot successfully executed the plan: it first executed the footstep plan over a small 0.2m wall, and then trotted towards the goal. This demonstrates how our approach can be successfully deployed in realistic conditions.

V. Conclusion

We presented a multi-controller search-based locomotion planner which considers multiple objectives using utopian or lexicographic cost aggregation according to user preference. The method does not require manual tuning of optimization parameters, and has several desirable properties. It is computationally faster than a pure footstep planner using the same method and costs, reaches Pareto-optimal solutions in shorter times than weighted-cost-aggregation, leads to faster missions than pure-static-controllers and to safer missions than pure dynamic controllers. It achieves that by leveraging the advantages and disadvantages of each controller, e.g. trotting on flat ground for low energy per distance, carefully planning footsteps for feasibility. According to our experimental evaluation, our proposed ranked-preference cost aggregation scheme has the empirical advantage of quick convergence to Pareto-optimal solutions which satisfy the user's cost priorities, followed by a decrease in other costs, along the Pareto-curve, as more computation time is allowed, without parameter tuning.

A limitation of our approach is the requirement of predicting computation time of a sub-planner, which breaks the possibility of using the method in strict anytime-fashion as, for example footstep planning might go over the desired computation time budget. In this paper we rely on the empirical quick convergence of our sub-planner for short

paths (around 1 second in our problems). It is still an open problem to design an architecture that provides guarantees for strict anytime planning of the whole pipeline—which is a direction we plan to pursue.

Other future research directions include learning cost and feasibility functions from experience [2], including slippage predictions in the objectives [35], and analyzing Pareto-optimality in higher dimensions. We are also interested in comparing our hierarchical multi-controller planning approach to other hierarchical architectures, investigating the use of other cost aggregation methods, and learning map representations for faster multi-gait planning.

REFERENCES

[1] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch, R. Diethelm, S. Bachmann, A. Melzer, and M. Hoepflinger, "Anymal - a highly mobile and dynamic quadrupedal robot," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 38–44.

[2] M. Brandao, Y. M. Shiguematsu, K. Hashimoto, and A. Takanishi, "Material recognition cnns and hierarchical planning for biped robot locomotion on slippery terrain," in *16th IEEE-RAS International Conference on Humanoid Robots*, Nov 2016, pp. 81–88.

[3] R. Deits and R. Tedrake, "Footstep planning on uneven terrain with mixed-integer convex optimization," in *14th IEEE-RAS International Conference on Humanoid Robots*, Nov 2014, pp. 279–286.

[4] A. Winkler, C. Mastalli, I. Havoutis, M. Focchi, D. Caldwell, and C. Semini, "Planning and execution of dynamic whole-body locomotion for a hydraulic quadruped on challenging terrain," in *IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 5148–5154.

[5] M. Geisert, T. Yates, A. Orgen, P. Fernbach, and I. Havoutis, "Contact planning for the anymal quadruped robot using an acyclic reachability-based planner," in *Towards Autonomous Robotic Systems Conference (TAROS)*, 2019.

[6] C. Mastalli, I. Havoutis, M. Focchi, D. G. Caldwell, and C. Semini, "Hierarchical Planning of Dynamic Movements without Scheduled Contact Sequences," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016.

[7] C. Mastalli, M. Focchi, I. Havoutis, A. Radulescu, S. Calinon, J. Buchli, D. G. Caldwell, and C. Semini, "Trajectory and foothold optimization using low-dimensional models for rough terrain locomotion," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 1096–1103.

[8] C. M. Dellin and S. S. Srinivasa, "A framework for extreme locomotion planning," in *2012 IEEE International Conference on Robotics and Automation*, May 2012, pp. 989–996.

[9] K. Hauser, T. Bretl, K. Harada, and J.-C. Latombe, "Using motion primitives in probabilistic sample-based planning for humanoid robots," in *Algorithmic foundation of robotics VII*. Springer, 2008, pp. 507–522.

[10] K. Gochev, B. Cohen, J. Butzke, A. Safonova, and M. Likhachev, "Path planning with adaptive dimensionality," in *Fourth annual symposium on combinatorial search*, 2011.

[11] A. Dornbush, K. Vijayakumar, S. Bardapurkar, F. Islam, and M. Likhachev, "A single-planner approach to multi-modal humanoid mobility," in *2018 IEEE International Conference on Robotics and Automation*, May 2018.

[12] Y.-C. Lin and D. Berenson, "Humanoid navigation planning in large unstructured environments using traversability-based segmentation," in *Intelligent Robots and Systems (IROS), 2018 IEEE/RSJ International Conference on*, Oct 2018.

[13] M. X. Grey, A. D. Ames, and C. K. Liu, "Footstep and motion planning in semi-unstructured environments using randomized possibility graphs," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 4747–4753.

[14] B. Aceituno-Cabezas, C. Mastalli, H. Dai, M. Focchi, A. Radulescu, D. G. Caldwell, J. Cappelletto, J. C. Grieco, G. Fernández-López, and C. Semini, "Simultaneous contact, gait, and motion planning for robust multilegged locomotion via mixed-integer convex optimization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2531–2538, 2018.

[15] R. Deits and R. Tedrake, *Computing Large Convex Regions of Obstacle-Free Space Through Semidefinite Programming*. Cham: Springer International Publishing, 2015, pp. 109–124.

[16] S. Feng, E. Whitman, X. Xinjilefu, and C. G. Atkeson, "Optimization-based full body control for the darpa robotics challenge," *Journal of Field Robotics*, vol. 32, no. 2, pp. 293–312, 2015.

[17] S. Caron and A. Kheddar, "Multi-contact walking pattern generation based on model preview control of 3d com accelerations," in *Proceedings of the 2016 IEEE-RAS International Conference on Humanoid Robots*. IEEE, Nov. 2016. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01349880

[18] D. Yi, M. A. Goodrich, and K. D. Seppi, "Morrf*: Sampling-based multi-objective motion planning," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[19] A. Lavin, "A pareto optimal d* search algorithm for multiobjective path planning," *arXiv preprint arXiv:1511.00787*, 2015.

[20] S. Choudhury, C. M. Dellin, and S. S. Srinivasa, "Pareto-optimal search over configuration space beliefs for anytime motion planning," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*. IEEE, 2016, pp. 3742–3749.

[21] A. A. Rusu, S. G. Colmenarejo, C. Gulcehre, G. Desjardins, J. Kirkpatrick, R. Pascanu, V. Mnih, K. Kavukcuoglu, and R. Hadsell, "Policy distillation," *arXiv preprint arXiv:1511.06295*, 2015.

[22] K. Frans, J. Ho, X. Chen, P. Abbeel, and J. Schulman, "Meta learning shared hierarchies," *arXiv preprint arXiv:1710.09767*, 2017.

[23] C. Florensa, Y. Duan, and P. Abbeel, "Stochastic neural networks for hierarchical reinforcement learning," *arXiv preprint arXiv:1704.03012*, 2017.

[24] C. Gehring, S. Coros, M. Hutler, C. D. Bellicoso, H. Heijnen, R. Diethelm, M. Bloesch, P. Fankhauser, J. Hwangbo, M. Hoepflinger, and R. Siegwart, "Practice makes perfect: An optimization-based approach to controlling agile motions for a quadruped robot," *IEEE Robotics Automation Magazine*, vol. 23, no. 1, pp. 34–43, March 2016.

[25] S. Tonneau, A. D. Prete, J. Pettr, C. Park, D. Manocha, and N. Mansard, "An efficient acyclic contact planner for multiped robots," *IEEE Transactions on Robotics*, vol. 34, no. 3, pp. 586–601, June 2018.

[26] P. Fankhauser, C. D. Bellicoso, C. Gehring, R. Dubé, A. Gawel, and M. Hutter, "Free Gait An Architecture for the Versatile Control of Legged Robots," in *IEEE-RAS International Conference on Humanoid Robots*, 2016.

[27] M. Wermelinger, P. Fankhauser, R. Diethelm, P. Krüsi, R. Siegwart, and M. Hutter, "Navigation planning for legged robots in challenging terrain," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*. IEEE, 2016, pp. 1184–1189.

[28] H. K. Rath, S. Timmadasari, B. Panigrahi, and A. Simha, "Realistic indoor path loss modeling for regular wifi operations in india," in *Communications (NCC), 2017 Twenty-third National Conference on*. IEEE, 2017, pp. 1–6.

[29] K. Miettinen, "Introduction to multiobjective optimization: Noninteractive approaches," in *Multiobjective optimization*. Springer, 2008, pp. 1–26.

[30] M. Brandao, K. Hashimoto, J. Santos-Victor, and A. Takanishi, "Footstep planning for slippery and slanted terrain using human-inspired models," *IEEE Transactions on Robotics*, vol. 32, no. 4, pp. 868–879, Aug 2016.

[31] Argos challenge. [Online]. Available: http://www.argos-challenge.com/

[32] M. Likhachev, G. J. Gordon, and S. Thrun, "Ara*: Anytime a* with provable bounds on sub-optimality," in *Advances in Neural Information Processing Systems*, 2003, pp. 767–774.

[33] M. Likhachev. (2010) Search-based planning library. [Online]. Available: http://www.ros.org/wiki/sbpl

[34] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sendai, Japan, Sep 2004, pp. 2149–2154.

[35] M. Brandao, K. Hashimoto, J. Santos-Victor, and A. Takanishi, "Gait planning for biped locomotion on slippery terrain," in *14th IEEE-RAS International Conference on Humanoid Robots*, November 2014, pp. 303–308.