Multi-controller 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. 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 it is faster than pure footstep planning methods both in computation (2x) and mission time (1.4x), and safer than pure dynamic-walking methods.

I. INTRODUCTION

Legged robots 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 automatically plan locomotion, both in terms of body paths and the choice of gaits/controllers/sub-planners at each point in time. We demonstrate our approach on the ANYmal quadruped robot [1] in the oil rig facility shown in Fig.1. Our work is similar to [2], [3], in that we first plan a path over robot base motion and mode, followed by execution and sub-planning where appropriate. However, we also introduce feasibility constraints at mode transitions and explicitly consider multiple objectives through no-preference aggregation.

II. MULTI-CONTROLLER MULTI-OBJECTIVE PLANNER

We plan in a a high-level search space \( S^{\text{GZT}} \times M \) where \( S^{\text{GZT}} = \{(x, y, z, \theta) \mid (x, y, z) \in \mathbb{R}^3, \theta \in SO(2)\} \), and \( M = \{m_{\text{Walking}}, m_{\text{Trotting}}, m_{\text{FootPlan}}\} \). The set \( M \) corresponds to a choice of controller: a “blind” reactive walking controller, a similar but trotting-based controller, and a vision-based controller that uses A* search on a map to plan footstep placements. We use A* search for both searches (high-level and footsteps), with the same cost functions related to expected energy consumption and traversability.

The energy cost is given by
\[
c_1(s_i, s_{i+1}) = \Psi \Delta t(s_i, s_{i+1}),
\]
where \( \Psi \) 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 assume each controller executes trajectories at a constant characteristic velocity \( v(m) = \)

![Image](image_url)

Fig. 1: Testing facility. The bottom rows highlight different parts of a 25m long locomotion plan where 3 different modes of locomotion are required.

\((v_x, v_y, v_z)\) 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:

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

where \( t_{\text{FootPlan}}^i \) 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:

\[
\Delta t(s_i, s_{i+1}) = \frac{d_x}{v_x} + \frac{d_y}{v_y} + \frac{d_z}{v_z} + \Delta t_{\text{comp}}.
\]

Regarding the traversability cost \( c_2(s_i, s_{i+1}) \) we use that from [4]: a weighted sum of roughness and slope heuristics averaged over the contact regions, as implemented in [5].

We then aggregate the two costs through a no-preference (a.k.a. utopian) aggregation scheme as in [6], to promote balanced trade-offs between the two objectives:

\[
c_{\text{utopia}}(s_i, s_{i+1}) = c + \sum_{k=1}^{K} c_k(s_i, s_{i+1}) - \frac{z_{k}^{*}}{z_{k}^{\text{up}}} - \frac{z_{k}^{*}}{z_{k}^{\text{low}}},
\]

where \( z_{k}^{*} \) and \( z_{k}^{\text{up}} \) are the lower and upper bounds of the state transition costs, respectively. \( K \) is the number of cost functions (two in this case). This provides a balanced cost aggregation free of weight tuning.

*Oxford Robotics Institute, University of Oxford, UK. This work was supported by the EPSRC ORCA Hub [EP/R026173/1]. Conducted as part of ANYmal Research, a community to advance legged robotics.
Fig. 2: Planned trajectory executed on the real robot. Left to right and top to bottom: footsteps over slab, then fast trotting towards goal.

Fig. 3: Multi-controller paths. Coloring of the map is based on traversability (red is less traversable). Coloring of paths is based on chosen controller (blue for walking, purple for trotting, red for footprint plans).

TABLE I: Mission performance, simulation

<table>
<thead>
<tr>
<th>Method</th>
<th>Failures</th>
<th>Mission time (min:sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trotting</td>
<td>2</td>
<td>1:05</td>
</tr>
<tr>
<td>Walking</td>
<td>0</td>
<td>5:21</td>
</tr>
<tr>
<td>FootPlan</td>
<td>0</td>
<td>4:25</td>
</tr>
<tr>
<td>Ours</td>
<td>0</td>
<td>3:09</td>
</tr>
</tbody>
</table>

III. RESULTS

We started by using our planner to obtain paths that cross an industrial floor model with obstacles and gaps on the floor. The environment requires the robot to carefully place footsteps when close to such gaps to avoid failure. We executed the paths in physics simulation and measured mission time and safety (the number of times that contacts entered a slipping state in Gazebo). Table I shows the results, compared to single-controller plans. The table shows that our method was the fastest feasible method when compared to pure-walking, pure-trotting or pure-footstep-planning methods. This is because of a mixed use of fast trotting gaits in gap-free areas, and careful placement around gaps. From our experiments, our method is also computationally faster (up to 2x) than a pure footstep planner.

Finally we used the planner to obtain a path which crosses a real-world oil rig site used for firefighting practice. The scenario required locomotion over a slab (0.2m high wall) and on horizontal uneven ground. The planner returned a path involving the use of a walking controller close to the slab, a footstep plan over the slab, and a trotting controller on horizontal ground as seen in Fig. 3. The robot successfully executed the plan in the real scenario, as shown in Fig. 2.

IV. DISCUSSION

We presented a multi-controller search-based locomotion planner which considers multiple objectives without weight tuning. The method is computationally fast, 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.

A current limitation of the method is the requirement of predicting computation time of a sub-planner, which breaks the possibility of using the method in strict anytime-fashion. It is still an open problem to design an architecture that provides guarantees strict anytime planning of the whole pipeline - which is a direction we plan to pursue. Other research directions include learning cost and feasibility functions from experience.

REFERENCES


