1 Introduction

Most locomotion control strategies are developed for flat terrain. We explore the use of reinforcement learning to develop motor skills for the highly dynamic traversal of terrains having sequences of gaps, walls, and steps. Results are demonstrated using simulations of a 21-link planar dog and a 7-link planar biped. Our approach is characterized by: non-parametric representation of the value function and the control policy; value iteration using batched positive-TD updates; localized epsilon-greedy exploration; and an optimized state distance metric. The policies are progressively improved using repeated iterations of epsilon-greedy exploration and value iteration.

2 Related Work

Significant progress has been made over the past decades in designing and learning control strategies for dynamic bipedal and quadrupedal gaits. One common approach for control consists of supplying short-term or finite-horizon goals for the motion using a mix of constraints and objectives, together with knowledge of the equations of motion (EOM). Solutions to such model-predictive control problems can then be computed by solving a quadratic program, or a finite-horizon trajectory optimization, e.g., [2]. Alternatively, explicit control policies can be developed that leverage simplified models in order to compute the actions required to help achieve desired goals without assuming full knowledge of the EOM, e.g., [1, 5]. These methods include the use of foot-placement feedback laws, inverted pendulum models, and virtual model control. Lastly, reinforcement learning methods, including policy search, have also proved to be effective, e.g., [3]. A general limitation of previous work is that it most often targets locomotion skills for flat terrain.

3 Method

We apply reinforcement learning towards highly dynamic gaits capable of traversing obstacle sequences. Our principal contribution lies in showing how non-parametric value function approximation and policy approximation can be used to learn challenging terrain-adaptive locomotion skills. An overview of the skill learning pipeline is given in Figure 2. The control policies are learned in a progressive fashion through the continued accumulation of transition tuples, \( T = \{(s_i, a_i, r_i, s'_i)\} \) during exploration. All available tuples are then used by value iteration stages to compute a set of policy tuples \( T_\pi \), which then serve as a non-parametric approximation of the policy.

A state \( s_i \) consists of a combination of features of the state of the quadruped or biped, as well as features that describe the next two upcoming obstacles. The state descriptors for the dog and biped are 10 and 11 dimensional, respectively, including the terrain descriptors. An action \( a_i \) describes the control needed to complete a full bound or running step, as represented using parameterized finite-state machines. Within a state, control is computed using control abstractions that include: foot placement, PD controllers, and desired leg forces that are realized via the Jacobian transpose [1, 5]. The actions...
for the dog and biped are 28 and 14 dimensional, respectively. The reward function for our terrain traversal tasks is given by a function $R(s,a,s')$ reflecting the desirability of a state transition. It consists of rewards for preventing a fall, maintaining a desired velocity, and minimizing expended effort. The goal of a policy is then to maximize the cumulative reward.

To collect an initial set of tuples, a policy that makes uniform random choices among a small fixed set of provided actions is used for bootstrapping. Further iterations of exploration and value iteration are then repeated to produce successive improvements in the control policy. Tuples are collected by using $\epsilon$-greedy that alternates between introducing perturbations to locally optimal actions and selecting an action from a random tuple in $T_2$. Perturbations are sampled from a distribution that favours changes correlated with nearby optimal actions.

Our non-parametric value function approximation [4] is based on kNN interpolation with a Gaussian kernel. This approach relies heavily on having a good distance metric to find the most relevant nearby states for a query state. Starting with a uniform distance metric, an improved linear weighting for the various features is learned via a derivative-free optimization method applied to the space of possible weightings.

Given a set of transition tuples obtained during exploration, approximate value iteration is used to compute a policy that is optimal with respect to the observed tuples. While RL problems with discrete actions spaces can explicitly maximize over all possible actions, this is not possible in the continuous action setting. Instead, we apply iterative batch processing of updates based on positive temporal difference updates (similar to [6]), as summarized in Algorithm 1. For each transition tuple, the temporal difference measures the net benefit of taking the tuple transition as compared to the current value function estimate for the state (line 9). For ‘winning’ tuples having $\delta > 0$, the value function is adapted and the tuple is tagged as contributing towards the optimal policy. Value iteration stops when the maximum value function adaptation of the current iteration is less than a given threshold. The final policy is defined by the winning tuples, $T_2 : \{T_i | \Pi_i = true\}$.

### Algorithm 1 Positive temporal difference value iteration

1: input $\mathcal{T} = \{(s_i,a_i,r_i,s'_i)\}$: state transition tuples
2: input $\alpha$: learning rate; $\gamma$: discount rate
3: output $v_i$: value functions at $s_i$
4: output $\Pi_i = \{true/false\}$: optimal policy tuple flag
5: $v_i \leftarrow 0$ for all $i$
6: $\Pi_i \leftarrow false$ for all $i$
7: while not converged do
8:   for all $i$, do
9:     $\delta = r_i + \gamma \hat{V}(s'_i) - \hat{V}(s_i)$
10:    if $\delta > 0$ then
11:        $v_i \leftarrow v_i + \alpha \delta$
12:        $\Pi_i \leftarrow true$
13:    else
14:        $v_i \leftarrow v_i + \alpha(\hat{V}(s_i) - v_i)$
15:        $\Pi_i \leftarrow false$
16:    end if
17: end for
18: end while

### 5 Future Work

Much remains to be investigated. We wish to develop control policies that generalize to broader classes of terrain. It should be possible to also use the non-parametric model for motion prediction, thereby enabling better motion planning. Strategies need to be developed for good control policies based on a fixed budget of transition tuples. We also wish to bootstrap the learning with more naively parameterized actions.

### References


