

Predictive Terrain Range Maps for Energy-Efficient Quadruped Navigation

Jialin Ma • Mingxuan Li • Yutong Zhou

Intelligent Mobility Lab, University of Science and Technology of China, Hefei, China

PUBLISHED

September 1, 2026

REGISTERED

September 8, 2026

RECORD TYPE

Preprint

ABSTRACT

Energy-efficient quadruped navigation requires knowing not only where obstacles are but also which feasible routes waste energy in rough terrain. Control-aligned future-range prediction estimates free range along the directions the robot's controller can actually execute, but it treats every free meter as equally cheap. This paper extends that representation into a predictive terrain-range map. Each control-aligned ray is annotated with an effort cost obtained by integrating a calibrated interface energy model, which relates terrain class, slope, and speed to electrical power, along the ray's predicted path. The result is a cost-weighted range field that a receding-horizon local planner consumes together with a consistent global cost-to-go, and a single explicit scalar λ sets the exchange rate between distance and energy. For exact route optimization, increasing λ cannot increase route energy or shorten the route. The approach is evaluated on 10 Isaac Lab terrain maps with graded effort cost (5 seeds, 25 goal pairs each) and on a 2.4 km mixed-terrain field loop with a Unitree Go2 and battery telemetry. With $\lambda = 1.0$, the planner reaches 1184 J/km, 13.3 percent below a distance-only planner (1365 J/km), with a mean arrival time of 218 s versus 248 s and 96.8 percent goal success versus 90.7 percent. A sweep over λ traces an energy-time frontier from 1412 J/km at 187 s to 1086 J/km at 289 s with diminishing returns beyond $\lambda = 2.0$, while energy-only routing is vulnerable to detour loops.

INDEX TERMS quadruped navigation, energy-efficient planning, terrain cost, control-aligned range prediction, cost of transport, Pareto trade-off, Unitree Go2

KAGG NOTE A local planner can trade distance for energy with one interpretable scalar if every control-aligned range estimate also carries the energy the robot would spend to use it.

01 Introduction

A quadruped robot that walks across a campus, a construction site, or a park does not only have to avoid collisions; it also has to decide whether to cross a gravel band, wade through a sand channel, or take the paved detour around them. These choices determine how far the robot can travel on one battery charge. Modern learned locomotion controllers can traverse all of these surfaces [1, 2, 3, 4, 5], which makes the question more pressing rather than less: once almost every surface is traversable, the navigation stack has to rank traversable options by their cost, not merely separate them from obstacles.

Most local navigation pipelines for legged robots represent the world as free space versus occupied space, sometimes augmented with a traversability or risk score. A particularly effective recent representation is control-aligned future range: FutureRay predicts, for each direction the robot's own controller can actually execute over a short horizon, how much free range lies ahead along that direction [6]. Because the rays are aligned with executable commands rather than with a fixed angular grid, obstacle queries are directly usable by the controller and the planner does not have to reconcile geometric free space with dynamic feasibility. However, FutureRay treats each free meter as equally cheap. A meter of loose sand and a meter of packed earth receive the same score, so a planner built on it minimizes distance or time and is blind to energy.

This paper addresses that gap. The specific question is how to attach an energy cost to control-aligned range estimates so that a local planner can trade route length and arrival time against electrical energy in a controlled, interpretable way, without retraining the perception module for every trade-off. The approach, called a *predictive terrain-range map*, augments every control-aligned ray with an effort cost obtained by integrating a calibrated interface energy model along the ray's predicted path. The resulting cost-weighted range field is consumed by a receding-horizon local planner whose objective contains a single explicit scalar λ that sets the exchange rate between meters of route and joules of energy.

The contributions are as follows.

1. A *cost-weighted range field* that extends control-aligned future-range prediction [6] with a per-ray effort cost, computed from the predicted path, terrain class, slope, and terrain-limited speed.
2. A calibrated *interface energy model* for a quadruped that maps terrain class, slope, and commanded speed to electrical power, together with a least-squares calibration procedure from battery telemetry.
3. A local planner with a consistent global cost-to-go in which one dimensionless scalar λ trades distance against energy, with a monotonicity result showing that, for exact route optimization, increasing λ never increases route energy and never shortens the route.

4. An evaluation on 10 Isaac Lab terrain maps with graded effort cost and on a 2.4 km mixed-terrain field loop with a Unitree Go2 and battery telemetry, including the energy-time frontier over λ , an ablation of each component, and an analysis of recorded failure cases.

With $\lambda = 1.0$, the planner reaches 1184 J/km, 13.3% below a distance-only planner, with 96.8% goal success. The frontier obtained by sweeping λ shows diminishing returns: most of the energy saving is obtained between $\lambda = 0$ and $\lambda = 2.0$, and pushing further mostly lengthens arrival time.

02 Related Work

2.1 Learned locomotion on rough terrain

Reinforcement learning has made robust blind and perceptive locomotion routine. Hwangbo et al. trained agile skills in simulation with a learned actuator network and transferred them to ANYmal [1]; Lee et al. extended this to challenging natural terrain using a teacher-student scheme with proprioceptive history [2]; and Miki et al. fused exteroceptive height samples with proprioception through a belief encoder that learns when to distrust the map [3]. Rudin et al. showed that massively parallel simulation trains such policies in minutes [4], and RMA adapts online to unknown terrain and payload through an estimated extrinsics vector [5]. Vision-based controllers now climb stairs and gaps from egocentric depth [7], and parkour policies demonstrate highly dynamic traversal of obstacles [8, 9, 10]. Walk These Ways exposes gait parameters such as frequency and foot clearance as command inputs, allowing behavior to be tuned at deployment [11]. This work treats such a controller as a fixed lower layer that tracks velocity commands at 50 Hz; it relies on the controller's ability to cross gravel and sand and focuses on deciding *whether* the robot should do so.

2.2 Energy and cost of transport in legged systems

Energy appears in the locomotion literature mainly as a training objective. Fu et al. showed that minimizing energy consumption alone, under a velocity-tracking task, is sufficient for natural gaits and gait transitions to emerge [12]. Yang et al. learned gait transitions that reduce cost of transport across speeds [13]. These works shape *how* the robot walks over a given surface; they do not decide *where* it walks. The present method is complementary: the energy model is calibrated on the closed-loop behavior of whatever controller is deployed, so a more efficient gait simply produces smaller coefficients. On deformable ground, Li et al. developed a terradynamics model of legged locomotion on granular media in which penetration resistance depends on intrusion depth and leg geometry [14]. That analysis motivates treating the foot-terrain interface, not just slope, as the main determinant of locomotion energy on sand and gravel, and it explains why an uncalibrated, slope-only model misranks such surfaces.

2.3 Navigation representations for legged robots

Legged navigation stacks typically build a robot-centric elevation map [15] and plan over it with sampling-based or learned planners. Hoeller et al. learned a compact state representation and a navigation policy for cluttered and dynamic scenes [16]; Rudin et al. trained locomotion and local navigation end-to-end [17]; and Lee et al. deployed learned navigation and locomotion for wheeled-legged robots over kilometer-scale missions [18]. Fu et al. coupled a vision-based planner with proprioceptive

feedback so that the planner respects what the walking policy can achieve [19]. Imperative planners such as iPlanner [20] and ViPlanner [21] learn to output paths by differentiating through a cost map that can include semantic traversal costs; ViPlanner in particular assigns class-dependent costs to semantic terrain labels. These costs are hand-specified and dimensionless, so a change of the trade-off requires retraining. In contrast, the cost used here has physical units derived from telemetry, and the trade-off is changed at run time by one scalar.

2.4 Control-aligned range prediction

FutureRay [6] is the direct starting point of this work. It predicts future free range along the robot's own predicted control-aligned directions: each ray corresponds to a candidate command, its direction follows the trajectory the controller is expected to produce, and its length is the distance the robot could travel along that trajectory before reaching an obstacle. Aligning obstacle queries with executable motions removes the mismatch between a geometric free-space map and what the controller can actually do, and it yields a compact, fixed-size input for a local planner. FutureRay, however, scores rays purely by free range and progress, so it cannot distinguish cheap from expensive free space. The present work keeps FutureRay's ray parameterization and range estimates unchanged and adds a second channel per ray: the electrical energy needed to use that range. A related idea of predicting collision-free motion primitives directly from sensing appears in high-speed quadrotor flight [22], and sampling of command-space trajectories underlies the classic dynamic window approach [23] and model predictive path integral control [24]; the cost-weighted range field can be viewed as supplying such samplers with a physically meaningful per-trajectory energy term.

2.5 Traversability and terrain-cost learning

A large body of work learns traversability or terrain cost. BADGR learns event probabilities such as collision and bumpiness from self-supervised experience [25]. Castro et al. learn off-road costmaps from proprioceptive roughness signals [26], Frey et al. adapt visual traversability online during deployment [27], and TerraPN learns surface costs online from robot-terrain interaction [28]. GA-Nav segments outdoor terrain into navigability classes for planning [29]. Risk-aware approaches such as STEP [30] and EVORA [31] model the uncertainty of traversability estimates and plan with tail-risk measures. These methods produce a cost or risk that is either dimensionless or tied to vehicle vibration, slip, or failure probability. The predictive terrain-range map differs in two respects: its cost is electrical energy per meter, calibrated against battery telemetry, and it is attached to control-aligned rays rather than to map cells, so that the energy estimate already reflects the trajectory and terrain-limited speed that the controller would realize.

2.6 Multi-objective route planning

Weighted-sum scalarization is the most common way to combine competing objectives [32]. Its known limitation is that it recovers only the supported (convex-hull) part of a Pareto front, and uniformly spaced weights do not yield uniformly spaced front points [33]. The global layer of the planner uses an A* or Dijkstra search [34] over a weighted-sum edge cost. The analysis in Section 4 makes the weighted-sum property explicit for the distance-energy pair, and the experiments report the frontier

actually traced by the closed-loop system rather than assuming it.

2.7 Simulation for legged navigation

Large-scale GPU simulation [35] and its successor frameworks [36] allow thousands of parallel locomotion and navigation episodes. Domain randomization [37] is the usual tool for bridging the reality gap, but simulator rankings do not always predict physical rankings [38]. For that reason, this study evaluates both in simulation, where terrain cost can be graded systematically, and on hardware with measured battery energy.

03 Problem Formulation

3.1 Robot, commands, and terrain

The robot state at planning step t is the planar pose $x_t = (p_t, \psi_t) \in \mathbb{R}^2 \times \mathbb{S}^1$ with position p_t and heading ψ_t . The locomotion controller accepts body-frame velocity commands $u = (v, \omega)$, with forward speed $v \in [0, v_{\max}]$ and yaw rate $\omega \in [-\omega_{\max}, \omega_{\max}]$. On the Go2 the command limits are set to $v_{\max} = 1.5$ m/s and $\omega_{\max} = 1.2$ rad/s, well inside the platform's capability, so that the controller tracks commands reliably on loose terrain. The controller runs at 50 Hz (20 ms period); the local planner runs at 10 Hz.

The terrain is described by a robot-centric elevation map $h(q)$ over positions $q \in \mathbb{R}^2$ and a terrain-class posterior $z(q) \in \Delta^{C-1}$ over C surface classes (in this study: firm ground, gravel, sand, and vegetated or mixed ground). Both are maintained at 0.1 m resolution over an 8 m by 8 m window from onboard LiDAR. The local slope along a heading ψ at q is $\beta(q, \psi) = \arctan(\nabla h(q)^\top [\cos \psi, \sin \psi]^\top)$.

3.2 Control-aligned rays

Following FutureRay [6], a finite set of K candidate commands $\{u_k\}_{k=1}^K$ is defined over the command space. For each u_k , a short-horizon motion model predicts the planar path the controller would follow over horizon T ,

$$\gamma_k(s) = \Phi(x_t, u_k; s), \quad s \in [0, s_k^{\max}], \quad (1)$$

parameterized by arc length s , where $s_k^{\max} = v_k T$. The future free range $r_k \in [0, r_{\max}]$ is the arc length along γ_k at which the robot's footprint would first intersect an obstacle, capped at $r_{\max}$. The set $\mathcal{R}_t = \{(u_k, \gamma_k, r_k)\}_{k=1}^K$ is the FutureRay output and is used unchanged.

3.3 Objectives

For an executed route π from start p_s to goal p_g , let $L(\pi)$ be its length, $E(\pi)$ the electrical energy drawn from the battery, and $\tau(\pi)$ its duration. The energy per kilometer is $\bar{E}(\pi) = E(\pi)/L(\pi)$ in J/km. The navigation problem is to reach p_g without collision while trading L (and through it τ) against E . Because L and E are in different units, the trade-off is made through a nominal specific energy ε_0 (J/m), defined as the calibrated specific energy of the robot walking on flat firm ground at its cruise speed. The scalarized objective is

$$J_\lambda(\pi) = L(\pi) + \lambda \frac{E(\pi)}{\varepsilon_0}, \quad \lambda \in [0, \infty). \quad (2)$$

The second term is the energy of the route expressed as an equivalent number of meters walked on flat ground, so λ is dimensionless: $\lambda = 1$ means that one meter of detour is accepted if it saves at least the energy of one meter of flat walking. The limiting cases $\lambda = 0$ (distance only) and $\lambda \rightarrow \infty$ (energy only, with distance as tie-breaker) are the two baselines used in Section 5.

04 Method

The pipeline has four stages: (i) FutureRay produces control-aligned rays and free ranges; (ii) the interface energy model assigns an effort cost to each ray, producing the cost-weighted range field; (iii) a global layer computes a λ -consistent cost-to-go over a coarse graph; and (iv) the local planner scores rays and sends the selected command to the locomotion controller.

4.1 Interface energy model

The model predicts the electrical power drawn by the robot while walking at speed v on terrain with class posterior z and slope β . It is written per unit distance, as a specific energy ε in J/m:

$$\varepsilon(v; z, \beta) = \frac{P_0}{v} + mg(c_r(z) + g_\kappa(\beta)) + c_d(z)v, \quad (3)$$

where P_0 is the standing power of computation, sensing, and posture holding (W); m is the robot mass including payload and g the gravitational acceleration; $c_r(z) = \sum_{c=1}^C z_c c_{r,c}$ is a dimensionless interface resistance coefficient, the class-weighted average of per-class coefficients $c_{r,c}$; $c_d(z) = \sum_c z_c c_{d,c}$ (J s/m²) captures speed-dependent losses such as increased stance forces and slip at higher speed; and

$$g_\kappa(\beta) = \max(\sin \beta, \kappa \sin \beta), \quad \kappa \in [0, 1), \quad (4)$$

accounts for climbing work on ascents and for partial, rather than full, recovery of potential energy on descents. The first term in (3) penalizes walking slowly, the last penalizes walking fast, and the middle term carries the terrain dependence. The model is called an *interface* model because $c_{r,c}$ and $c_{d,c}$ summarize foot-terrain interaction (sinkage, slip, and compliance) rather than any property of the controller in isolation; this mirrors the granular-media analysis of [14] at a much coarser level.

Loose terrain also limits the speed that the controller actually achieves. Each class has a calibrated achievable speed $\bar{v}_c$, and the predicted realized speed is

$$\hat{v}(v; z) = \min\left(v, \sum_{c=1}^C z_c \bar{v}_c\right). \quad (5)$$

Equation (3) is always evaluated at $\hat{v}$, so the model captures the fact that sand both costs more per meter and is slower to cross.

Calibration. The parameter vector θ collects P_0 , the per-class coefficients $c_{r,c}$ and $c_{d,c}$, the recovery factor κ , and the per-class speeds $\bar{v}_c$; it is fitted from logged traverses. Battery voltage V and current I give the measured power $P^m(t) = V(t)I(t)$. Logs are cut into windows w of 2 s during which the class posterior under the feet is dominated by one class. For each window, the predicted energy is $\hat{E}_w(\theta) = \int_w \varepsilon(\hat{v}; z, \beta) \hat{v} dt$ using the estimated speed and terrain along the executed path. The coefficients are obtained by bounded least squares,

$$\theta^* = \arg \min_{\theta \in \Theta} \sum_w \left(\int_w P^m dt - \hat{E}_w(\theta) \right)^2 + \eta \|\theta - \theta_{\text{prior}}\|^2, \quad (6)$$

where Θ enforces $P_0, c_{r,c}, c_{d,c} \geq 0$ and $\kappa \in [0, 1)$, and the small ridge term η pulls poorly excited coefficients toward a prior. Because (3) is linear in $(P_0, c_{r,c}, c_{d,c})$ for fixed κ , the problem is a nonnegative least-squares fit nested in a one-dimensional search over κ . The achievable speeds $\bar{v}_c$ are fitted

separately as high quantiles of the realized speed under saturating commands. In simulation, the same procedure is applied with P^m replaced by the simulated electrical power of the joint actuators plus a constant auxiliary load. The calibrated model is evaluated only on terrain classes and slope ranges covered by the calibration logs; outside them, the planner falls back to the prior coefficients.

4.2 Cost-weighted range field

For ray k , the usable arc length is

$$\ell_k = \min(r_k - d_s, s_k^{\max}), \quad (7)$$

where d_s is a stopping margin. Rays with $\ell_k < \ell_{\min}$ are marked inadmissible. The effort cost of the ray is the predicted energy to traverse its usable part,

$$E_k = \sum_{i=0}^{N_k-1} \varepsilon(\hat{v}_{k,i}; z(\gamma_k(s_i)), \beta_{k,i}) \Delta s, \quad (8)$$

with $s_i = i \Delta s$, $N_k = \lfloor \ell_k / \Delta s \rfloor$, $\Delta s = 0.1$ m, $\hat{v}_{k,i} = \hat{v}(v_k; z(\gamma_k(s_i)))$, and $\beta_{k,i}$ the slope at $\gamma_k(s_i)$ along the path tangent. The predicted traversal time is $T_k = \sum_i \Delta s / \hat{v}_{k,i}$. The mean specific energy along the ray is $\bar{\varepsilon}_k = E_k / \ell_k$.

The cost-weighted range field is the set

$$\mathcal{F}_\lambda = \{(u_k, r_k, E_k, T_k, \rho_k(\lambda))\}_{k=1}^K, \quad (9)$$

where the *cost-weighted range* is

$$\rho_k(\lambda) = r_k \frac{1 + \lambda}{1 + \lambda \bar{\varepsilon}_k / \varepsilon_0}. \quad (10)$$

The quantity ρ_k equals the geometric range r_k on nominal flat ground or when $\lambda = 0$, shrinks on rays whose terrain is more expensive than nominal, and grows slightly on rays that are cheaper than nominal, for example on gentle descents. It expresses free range in units of “flat-ground-equivalent progress per unit of weighted cost,” and it reduces to FutureRay’s free range when $\lambda = 0$. The field is useful for visualization and as a drop-in replacement for r_k in any planner that consumes ranges; the planner below uses the full components of (9).

4.3 Global cost-to-go

A purely local score would favor short-horizon energy savings that lead into dead ends. A coarse graph $\mathcal{G}$ with 0.5 m spacing is therefore built over the known map, with edge cost

$$c_\lambda(e) = \ell_e + \lambda \frac{E_e}{\varepsilon_0}, \quad (11)$$

where ℓ_e is the edge length and E_e the energy predicted by (3) at the cruise speed along the edge. A backward Dijkstra search from the goal [34] gives the cost-to-go $H_\lambda(q)$ for every node, interpolated bilinearly between nodes. It is recomputed at 1 Hz and whenever λ changes. Using the same weighted cost in (11) and in the local score below keeps the local and global layers consistent, so the local planner does not trade energy at one rate while the global layer trades it at another.

4.4 Local planner

At each planning step the local planner selects

$$k^* = \arg \min_{k \in \mathcal{A}_t} J_k(\lambda), \quad (12)$$

over the admissible set $\mathcal{A}_t$, with

$$J_k(\lambda) = \ell_k + \lambda \frac{E_k}{\varepsilon_0} + H_\lambda(\gamma_k(\ell_k)) + w_c \phi_c(r_k) + w_s |\omega_k - \omega_{t-1}|. \quad (13)$$

The first two terms are the weighted cost of using ray k , the third is the cost-to-go from the ray’s end point, $\phi_c(r) = \max(0, r_{\text{safe}} - r)^2 / r_{\text{safe}}$ penalizes rays that end close to obstacles, and the last term discourages oscillation between neighboring commands. The selected command u_{k^*} is sent to the controller with its forward speed limited to the predicted achievable speed at the robot’s current position, $v = \hat{v}(v_{k^*}; z(p_t))$, so that the controller is not asked to track speeds the terrain does not allow. If $\mathcal{A}_t$ is empty, the robot stops and rotates in place toward the ray with the largest r_k .

For $\lambda \rightarrow \infty$, (13) is replaced by its lexicographic limit: rays are ranked by $E_k / \varepsilon_0 + H_\infty^E$, with H_∞^E the energy-only cost-to-go, and distance terms are used only to break ties. This is the energy-only baseline.

4.5 Parameter values

The command set uses $K = 21$ rays from 7 yaw rates and 3 forward speeds within the limits given in Section 3. The prediction horizon is $T = 2.5$ s, the range cap is $r_{\max} = 5$ m, the stopping margin is $d_s = 0.3$ m, the minimum usable length is $\ell_{\min} = 0.5$ m, and $r_{\text{safe}} = 1.0$ m. The weights w_c and w_s were set once on flat ground and held fixed across all terrain maps, λ values, and the field loop. The nominal specific energy ε_0 is taken from the calibrated model on firm flat ground at the cruise speed, so that it is defined consistently for each platform.

4.6 Properties of the scalar trade-off

The scalar λ is intended to act as an interpretable dial. The following property, a standard consequence of weighted-sum scalarization [32], makes this precise for exact route optimization.

Proposition (monotone trade-off). Let Π be a finite set of feasible routes and let $\pi_\lambda \in \arg \min_{\pi \in \Pi} J_\lambda(\pi)$ for J_λ in (2). If $0 \leq \lambda_1 < \lambda_2$, then $E(\pi_{\lambda_1}) \geq E(\pi_{\lambda_2})$ and $L(\pi_{\lambda_1}) \leq L(\pi_{\lambda_2})$. Moreover, for any $\lambda \in (0, \infty)$, π_λ is Pareto-optimal in (L, E) over Π .

Proof. Write $a = \pi_{\lambda_1}$, $b = \pi_{\lambda_2}$, and $e(\cdot) = E(\cdot) / \varepsilon_0$. Optimality gives $L(a) + \lambda_1 e(a) \leq L(b) + \lambda_1 e(b)$ and $L(b) + \lambda_2 e(b) \leq L(a) + \lambda_2 e(a)$. Adding the two inequalities yields $(\lambda_2 - \lambda_1)(e(a) - e(b)) \geq 0$, hence $e(a) \geq e(b)$. Substituting into the first inequality gives $L(a) - L(b) \leq \lambda_1(e(b) - e(a)) \leq 0$. For Pareto optimality, if some π' had $L(\pi') \leq L(\pi_\lambda)$ and $e(\pi') \leq e(\pi_\lambda)$ with at least one strict inequality, then $J_\lambda(\pi') < J_\lambda(\pi_\lambda)$ for $\lambda > 0$, a contradiction. $\square$

Two caveats follow directly. First, weighted sums reach only the supported points of the frontier; non-convex portions cannot be selected by any λ [33]. Second, the closed-loop system is a receding-horizon approximation with imperfect energy predictions, so the proposition is a design guide, not a guarantee for executed routes. The experiments therefore measure the frontier that the closed-loop system actually traces. The proposition also clarifies why $\lambda \rightarrow \infty$ is risky: the distance term, which guarantees that every executed meter is charged a positive cost independent of the terrain model, vanishes, and the

planner can circulate through regions whose predicted energy-to-go is nearly flat.

05 Experiments

5.1 Setup and platform

Simulation. The simulation study uses Isaac Lab [36] with 10 terrain maps with graded effort cost: flat maps, maps with gravel bands, maps with sand channels, and mixed maps. Each map is evaluated with 5 random seeds and 25 start-goal pairs per seed, giving 1250 navigation episodes per method. Terrain classes are rendered with class-specific contact parameters so that the effort cost differs between classes, and the energy model is calibrated on the simulated robot as described in Section 4.

Hardware. The physical platform is a Unitree Go2 quadruped running the same perception, planning, and locomotion stack. The field test uses a 2.4 km mixed-terrain loop with battery telemetry, and 10 runs are performed per λ setting. Battery voltage and current are logged and integrated to obtain electrical energy. Run-to-run scatter of the telemetry-derived energy on the field loop is $\pm 6\%$, attributed to battery temperature; all energy-per-kilometer comparisons on the field loop use the mean of the 10 runs.

Methods compared. (i) *Distance-only planner* ($\lambda = 0$): the same local planner and FutureRay ranges, with the energy terms removed. (ii) *Heuristic terrain avoidance*: the distance-only planner with a fixed, hand-set per-class multiplicative weight on path length, a common practice in semantic costmaps; it has no energy model and no trade-off scalar. (iii) *Energy-only routing* ($\lambda \rightarrow \infty$): the lexicographic limit described in Section 4. (iv) *Ours*: the predictive terrain-range map with $\lambda = 1.0$.

Metrics. Energy per kilometer (J/km) is integrated electrical energy divided by traveled distance. Arrival time (s) is the mean time from start to goal per navigation episode. Route length is the executed path length divided by the length of the shortest collision-free reference path computed on the global planning graph; because the reference path is constrained to the graph while executed trajectories are not, this ratio can fall slightly below 1. Goal success (%) is the fraction of episodes that reach the goal without collision or timeout.

5.2 Main comparison

Table 1 reports the main comparison.

The proposed planner has the lowest energy per kilometer, 1184 J/km. This is 181 J/km (13.3%) below the distance-only planner, 100 J/km (7.8%) below heuristic terrain avoidance, and 185 J/km (13.5%) below energy-only routing. It also has the shortest mean arrival time, 218 s, which is 30 s (12.1%) shorter than the distance-only planner, and the highest goal success, 96.8%, which is 6.1 percentage points above the distance-only planner.

The proposed planner accepts the longest routes (1.12 times the shortest reference path, compared with 1.06 for the distance-only planner). That it nevertheless arrives earlier is consistent with the speed model in (5): rays that cross sand and gravel are predicted to be both more expensive and slower, so a slightly longer route on firm ground is faster as well as cheaper. A plausible reading is that the distance-only planner commits to

the geometrically short route and then walks it at the reduced speed that loose terrain permits; per-episode speed logs would be needed to confirm this.

Energy-only routing is the least efficient method in energy per kilometer (1369 J/km), slightly worse than the distance-only planner (1365 J/km), and has the lowest goal success (81.5%). It has the shortest route-length ratio (0.98), so its deficit is not explained by longer routes; the recorded failure cases in Section 5.5 show that it can be trapped in detour loops, which spend energy and time without progress. Heuristic terrain avoidance improves on the distance-only planner in energy (1284 vs. 1365 J/km) and arrival time (231 vs. 248 s) but has lower goal success (87.1% vs. 90.7%), which is consistent with fixed weights that are sometimes too aggressive in avoiding a class.

5.3 Ablation

Table 2 removes one component at a time. *w/o effort cost per ray* keeps the energy model and the λ -weighted global cost-to-go but sets $E_k = 0$ in (13), so energy enters only through the coarse graph. *w/o calibrated energy model* replaces θ^* in (6) with the uncalibrated prior coefficients θ_{prior} . *w/o planner scalar (fixed λ)* removes the explicit scalar from the planner and instead folds a fixed trade-off into the cost terms, so that the trade-off is no longer set as one consistent λ across the local and global layers.

Every component contributes to both energy and success. Removing the per-ray effort cost has the largest effect: energy rises by 113 J/km (9.5%) to 1297 J/km and goal success falls by 10.2 percentage points to 86.6%. Energy information on the coarse graph alone therefore does not substitute for energy attached to the control-aligned rays, where the planner actually chooses commands. Replacing the calibrated model with prior coefficients raises energy by 66 J/km (5.6%) and lowers goal success to 89.7%, which indicates that the ranking of gravel and sand relative to firm ground is sensitive to the calibrated interface coefficients. Removing the explicit scalar raises energy by 42 J/km (3.5%) and lowers success to 89.0%. The energy difference for this last variant is smaller than the $\pm 6\%$ per-run telemetry scatter on the field loop, so the energy effect of this variant should be interpreted with caution even though it is computed from run means; the success-rate drop of 7.8 percentage points is the clearer signal.

5.4 Energy-time frontier

Table 3 reports the sweep over the trade-off scalar.

Across the sweep, energy per kilometer decreases and arrival time increases monotonically with λ , which matches the ordering predicted by the proposition in Section 4 (with arrival time in place of route length), and no setting is dominated by another in both quantities. Increasing λ from 0 to 10 lowers energy by 326 J/km (23.1%) and raises arrival time by 102 s (54.5%). The returns diminish quickly. The marginal saving per second of additional arrival time is 10.3 J/km/s between $\lambda = 0$ and 0.5, 5.5 between 0.5 and 1.0, 2.5 between 1.0 and 2.0, 1.0 between 2.0 and 5.0, and 0.39 between 5.0 and 10. Between $\lambda = 5.0$ and 10, a 7 J/km reduction costs 18 s. The setting $\lambda = 1.0$ used in Table 1 captures 228 J/km (70%) of the total 326 J/km saving available in the sweep while incurring 31 s (30%) of the 102 s total increase in arrival time. The energy differences at the high- λ end of the sweep are small relative to the $\pm 6\%$

Table 1: Main comparison. Lower is better for energy, arrival time, and route length; higher is better for goal success. Ours uses $\lambda = 1.0$.

Method	Energy per km (J/km)	Arrival time (s)	Route length ($\times$ shortest)	Goal success (%)
Distance-only planner ($\lambda = 0$)	1365	248	1.06	90.7
Heuristic terrain avoidance (fixed weights)	1284	231	1.01	87.1
Energy-only routing ($\lambda = \infty$)	1369	242	0.98	81.5
Ours	1184	218	1.12	96.8

Table 2: Ablation of the predictive terrain-range map.

Variant	Energy per km (J/km)	Goal success (%)
Ours (cost-weighted range field)	1184	96.8
w/o effort cost per ray	1297	86.6
w/o calibrated energy model	1250	89.7
w/o planner scalar (fixed λ)	1226	89.0

Table 3: Sweep of the trade-off scalar λ .

Trade-off λ	Energy per km (J/km)	Arrival time (s)
0	1412	187
0.5	1289	199
1.0	1184	218
2.0	1121	243
5.0	1093	271
10	1086	289

telemetry scatter, so the precise position of the frontier beyond $\lambda = 2.0$ is less certain than its shape.

The $\lambda = 1.0$ row of the sweep coincides with the Ours row of Table 1. The $\lambda = 0$ row of the sweep (1412 J/km, 187 s) does not coincide with the distance-only baseline row of Table 1 (1365 J/km, 248 s), although both are labeled $\lambda = 0$; the two tables were compiled as separate records, and the source of the difference is not resolved in the available logs. For this reason, comparisons in this paper are made within a table and never across rows of different tables.

5.5 Failure cases

Three recorded failure cases qualify the results.

Detour loops at the energy-only end. On sand channel map 7, energy-only routing trapped the robot in a 40 s detour loop twice, and goal success on this map at $\lambda = 10$ was 88%. The loop arises because, without a meaningful distance term, the local score barely penalizes motion that does not reduce the cost-to-go, so the planner circulates around the channel instead of committing to a crossing or to a long bypass. This is the mechanism anticipated after the proposition in Section 4, and it shows that the energy-optimal endpoint of the frontier is not free: the lowest energy per kilometer in Table 3 comes with reduced reliability on maps where cheap terrain forms closed corridors. This is consistent with the low goal success of energy-only routing in Table 1. That the recorded success drop on this map occurs already at $\lambda = 10$ indicates that the high- λ end of the sweep inherits part of the same vulnerability, and it is a further reason, beyond the diminishing energy returns, to operate at moderate λ .

Over-avoidance by fixed heuristic weights. Heuristic terrain avoidance over-avoids gravel. In the recorded case, its route was 1.41 times the shortest reference path and its energy was 13% above that of the proposed planner at $\lambda = 1.0$. Fixed per-class weights do not distinguish a narrow gravel band,

which is cheap to cross, from a wide gravel field, which is expensive; the per-ray energy integral in (8) does, because it accumulates cost only over the portion of the ray that actually lies on gravel. Averaged over all episodes, the heuristic route-length ratio is 1.01 (Table 1), so over-avoidance is episodic rather than systematic, but where it occurs it is costly.

Telemetry scatter. Battery telemetry scatter on the field loop is $\pm 6\%$ and is attributed to battery temperature. This scatter is of the same order as several of the reported differences, most notably between the fixed- λ ablation and the full method (3.5%) and between adjacent points at the high- λ end of the sweep. All field energy-per-kilometer comparisons therefore use the mean of 10 runs, but no per-condition confidence intervals are reported in the compiled tables.

REPRODUCIBILITY NOTE

Reported values are the compiled per-method aggregates from the simulation study (10 maps, 5 seeds, 25 goal pairs each) and the Go2 field loop (2.4 km, 10 runs per λ setting). Per-map breakdowns and confidence intervals are not part of the released tables.

06 Limitations

Several limitations remain. First, the energy model in (3) is deliberately low-dimensional: it uses a small set of terrain classes and a single speed-dependent term per class. Terrain whose energetics depend on moisture, compaction, or vegetation density beyond the class label is represented only through the class posterior, and calibration covers only the classes and slope ranges present in the logs. Second, the per-ray energy depends on the terrain classifier; a misclassified patch translates directly into a mispredicted effort cost, and the method currently propagates no classification uncertainty into the cost. Risk-aware formulations such as [30, 31] would be a natural extension. Third, the weighted-sum scalarization reaches only supported points of the distance-energy frontier [33], and the monotonicity property holds only for exact route optimization, not for the closed-loop receding-horizon system. Fourth, the hardware study uses one platform and one 2.4 km loop, and the $\pm 6\%$ battery-temperature scatter is comparable in size to some reported differences; temperature-compensated energy estimation would sharpen the comparison. Fifth, the compiled $\lambda = 0$ records in the main comparison and the sweep do not agree, as discussed in Section 5, which limits cross-table comparison. Finally, λ is fixed for an entire run. A natural next step is to adapt λ to the remaining battery charge and the remaining distance to the goal, which the explicit scalar makes straightforward to express.

07 Conclusion

This paper extended control-aligned future-range prediction into a predictive terrain-range map in which every control-aligned ray carries both its free range and the electrical energy needed to use it. The effort cost is computed from a calibrated interface energy model that accounts for terrain class, slope, and terrain-limited speed, and a local planner with a consistent global cost-to-go trades distance against energy through a single dimensionless scalar λ . In simulation on graded-cost terrain maps and on a Unitree Go2 field loop with battery telemetry, the planner with $\lambda = 1.0$ reached 1184 J/km, compared with 1365 J/km for a distance-only planner, with shorter arrival time (218 vs. 248 s) and higher goal success (96.8% vs. 90.7%). Sweeping λ traced a monotone energy-time frontier with strongly diminishing returns beyond $\lambda = 2.0$, and the recorded failures showed that the energy-only endpoint of that frontier can trap the robot in detour loops. The results suggest that attaching physically calibrated energy to control-aligned range estimates is a simple and interpretable way to make legged navigation energy-aware.

References

- [1] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter. Learning Agile and Dynamic Motor Skills for Legged Robots. *Science Robotics*, 4(26), 2019. arXiv:1901.08652.
- [2] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning Quadrupedal Locomotion over Challenging Terrain. *Science Robotics*, 5(47), 2020. arXiv:2010.11251.
- [3] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning Robust Perceptive Locomotion for Quadrupedal Robots in the Wild. *Science Robotics*, 7(62), 2022. arXiv:2201.08117.
- [4] N. Rudin, D. Hoeller, P. Reist, and M. Hutter. Learning to Walk in Minutes Using Massively Parallel Deep Reinforcement Learning. *Conference on Robot Learning (CoRL)*, 2021. arXiv:2109.11978.
- [5] A. Kumar, Z. Fu, D. Pathak, and J. Malik. RMA: Rapid Motor Adaptation for Legged Robots. *Robotics: Science and Systems (RSS)*, 2021. arXiv:2107.04034.
- [6] T. Zang, S. Wang, H. Huang, and S. Cheng. FutureRay: Control-Aligned Future Range for Agile Quadruped Navigation. arXiv:2608.01197, 2026.
- [7] A. Agarwal, A. Kumar, J. Malik, and D. Pathak. Legged Locomotion in Challenging Terrains Using Egocentric Vision. *Conference on Robot Learning (CoRL)*, 2022. arXiv:2211.07638.
- [8] Z. Zhuang, Z. Fu, J. Wang, C. Atkeson, S. Schwertfeger, C. Finn, and H. Zhao. Robot Parkour Learning. *Conference on Robot Learning (CoRL)*, 2023. arXiv:2309.05665.
- [9] X. Cheng, K. Shi, A. Agarwal, and D. Pathak. Extreme Parkour with Legged Robots. arXiv:2309.14341, 2023.
- [10] D. Hoeller, N. Rudin, D. Sako, and M. Hutter. ANYmal Parkour: Learning Agile Navigation for Quadrupedal Robots. *Science Robotics*, 9(88), 2024. arXiv:2306.14874.
- [11] G. B. Margolis and P. Agrawal. Walk These Ways: Tuning Robot Control for Generalization with Multiplicity of Behavior. *Conference on Robot Learning (CoRL)*, 2022. arXiv:2212.03238.
- [12] Z. Fu, A. Kumar, J. Malik, and D. Pathak. Minimizing Energy Consumption Leads to the Emergence of Gaits in Legged Robots. *Conference on Robot Learning (CoRL)*, 2021. arXiv:2111.01674.
- [13] Y. Yang, T. Zhang, E. Coumans, J. Tan, and B. Boots. Fast and Efficient Locomotion via Learned Gait Transitions. *Conference on Robot Learning (CoRL)*, 2021. arXiv:2104.04644.
- [14] C. Li, T. Zhang, and D. I. Goldman. A Terradynamics of Legged Locomotion on Granular Media. *Science*, 339(6126):1408–1412, 2013.
- [15] T. Miki, L. Wellhausen, R. Grandia, F. Jenelten, T. Homberger, and M. Hutter. Elevation Mapping for Locomotion and Navigation Using GPU. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022. arXiv:2204.12876.
- [16] D. Hoeller, L. Wellhausen, F. Farshidian, and M. Hutter. Learning a State Representation and Navigation in Cluttered and Dynamic Environments. *IEEE Robotics and Automation Letters*, 6(3), 2021. arXiv:2103.04351.
- [17] N. Rudin, D. Hoeller, M. Bjelonic, and M. Hutter. Advanced Skills by Learning Locomotion and Local Navigation End-to-End. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022. arXiv:2209.12827.
- [18] J. Lee, M. Bjelonic, A. Reske, L. Wellhausen, T. Miki, and M. Hutter. Learning Robust Autonomous Navigation and Locomotion for Wheeled-Legged Robots. *Science Robotics*, 9(89), 2024. arXiv:2405.01792.
- [19] Z. Fu, A. Kumar, A. Agarwal, H. Qi, J. Malik, and D. Pathak. Coupling Vision and Proprioception for Navigation of Legged Robots. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. arXiv:2112.02094.
- [20] F. Yang, C. Wang, C. Cadena, and M. Hutter. iPlanner: Imperative Path Planning. *Robotics: Science and Systems (RSS)*, 2023. arXiv:2302.11434.
- [21] P. Roth, J. Nubert, F. Yang, M. Mittal, and M. Hutter. ViPlanner: Visual Semantic Imperative Learning for Local Navigation. *IEEE International Conference on Robotics and Automation (ICRA)*, 2024. arXiv:2310.00982.
- [22] A. Loquercio, E. Kaufmann, R. Ranftl, M. Müller, V. Koltun, and D. Scaramuzza. Learning High-Speed Flight in the Wild. *Science Robotics*, 6(59), 2021. arXiv:2110.05113.
- [23] D. Fox, W. Burgard, and S. Thrun. The Dynamic Window Approach to Collision Avoidance. *IEEE Robotics and Automation Magazine*, 4(1):23–33, 1997.
- [24] G. Williams, P. Drews, B. Goldfain, J. M. Rehg, and E. A. Theodorou. Information-Theoretic Model Predictive Control: Theory and Applications to Autonomous Driving. *IEEE Transactions on Robotics*, 34(6), 2018. arXiv:1707.02342.
- [25] G. Kahn, P. Abbeel, and S. Levine. BADGR: An Autonomous Self-Supervised Learning-Based Navigation System. *IEEE Robotics and Automation Letters*, 6(2), 2021. arXiv:2002.05700.
- [26] M. G. Castro, S. Triest, W. Wang, J. M. Gregory, F. Sanchez, J. G. Rogers III, and S. Scherer. How Does It Feel? Self-Supervised Costmap Learning for Off-Road Vehicle Traversability. *IEEE International Conference on Robotics and Automation (ICRA)*, 2023. arXiv:2209.10788.
- [27] J. Frey, M. Mattamala, N. Chebrolu, C. Cadena, M. Fallon, and M. Hutter. Fast Traversability Estimation for Wild Visual Navigation. *Robotics: Science and Systems (RSS)*, 2023. arXiv:2305.08510.
- [28] A. J. Sathyamoorthy, K. Weerakoon, T. Guan, J. Liang, and D. Manocha. TerraPN: Unstructured Terrain Navigation Using Online Self-Supervised Learning. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022. arXiv:2202.12873.

- [29] T. Guan, D. Kothandaraman, R. Chandra, A. J. Sathyamoorthy, K. Weerakoon, and D. Manocha. GA-Nav: Efficient Terrain Segmentation for Robot Navigation in Unstructured Outdoor Environments. *IEEE Robotics and Automation Letters*, 7(3), 2022. arXiv:2103.04233.
- [30] D. D. Fan, K. Otsu, Y. Kubo, A. Dixit, J. Burdick, and A.-A. Agha-Mohammadi. STEP: Stochastic Traversability Evaluation and Planning for Risk-Aware Off-Road Navigation. *Robotics: Science and Systems (RSS)*, 2021. arXiv:2103.02828.
- [31] X. Cai et al. EVORA: Deep Evidential Traversability Learning for Risk-Aware Off-Road Autonomy. arXiv:2311.06234, 2023.
- [32] R. T. Marler and J. S. Arora. Survey of Multi-Objective Optimization Methods for Engineering. *Structural and Multidisciplinary Optimization*, 26(6):369–395, 2004.
- [33] I. Das and J. E. Dennis. A Closer Look at Drawbacks of Minimizing Weighted Sums of Objectives for Pareto Set Generation in Multicriteria Optimization Problems. *Structural Optimization*, 14(1):63–69, 1997.
- [34] P. E. Hart, N. J. Nilsson, and B. Raphael. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.
- [35] V. Makoviychuk et al. Isaac Gym: High Performance GPU-Based Physics Simulation for Robot Learning. arXiv:2108.10470, 2021.
- [36] M. Mittal et al. Orbit: A Unified Simulation Framework for Interactive Robot Learning Environments. *IEEE Robotics and Automation Letters*, 8(6), 2023. arXiv:2301.04195.
- [37] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain Randomization for Transferring Deep Neural Networks from Simulation to the Real World. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017. arXiv:1703.06907.
- [38] A. Kadian et al. Sim2Real Predictivity: Does Evaluation in Simulation Predict Real-World Performance? *IEEE Robotics and Automation Letters*, 5(4), 2020. arXiv:1912.06321.