

Energy-Efficient Motion Planner for Legged Robots

Alexander Schperberg^{†*}, Marcel Menner[‡], and Stefano Di Cairano[†]

Abstract—We propose an online motion planner for legged robot locomotion with the primary objective of achieving energy efficiency. The conceptual idea is to leverage a placement set of footstep positions based on the robot’s body position to determine when and how to execute steps. In particular, the proposed planner uses virtual placement sets beneath the hip joints of the legs and executes a step when the foot is outside of such placement set. Furthermore, we propose a parameter design framework that considers both energy-efficiency and robustness measures to optimize the gait by changing the shape of the placement set along with other parameters, such as step height and swing time, as a function of walking speed. We show that the planner produces trajectories that have a low Cost of Transport (CoT) and high robustness measure, and evaluate our approach against model-free Reinforcement Learning (RL) and motion imitation using biological dog motion priors as the reference. Overall, within low to medium velocity range, we show a 50.4% improvement in CoT and improved robustness over model-free RL, our best performing baseline. Finally, we show ability to handle slippery surfaces, gait transitions, and disturbances in simulation and hardware with the Unitree A1 robot.

I. INTRODUCTION

Legged robots excel on uneven terrain and in human-centric environments, but their efficiency is hindered by energy losses from foot-ground impacts, which accelerate battery depletion and increase motor heat, reducing operation time. Thus, formulating new energy conservation strategies are worthwhile endeavors. Hardware solutions such as Ranger [1] and ANYmal [2], reduce energy consumption using passive dynamics and mechanical springs. Software-based solutions, the focus of this work, can include various motion planning strategies, including optimization [3], Reinforcement Learning (RL) [4], and vision-based methods [5]. Although energy-efficient rewards or cost terms can be incorporated into the optimization or learning process, it is often challenging to assign significant weight to these terms without encountering local optima issues, where the robot fails to follow the desired velocity commands or avoids movements entirely.

Our approach achieves energy-efficient locomotion using a simple, geometry-based footstep planner that constrains motion based on fundamental physical principles. Elliptical placement sets beneath each hip dictate when and where steps occur. The planner’s parameters—ellipse shape, swing

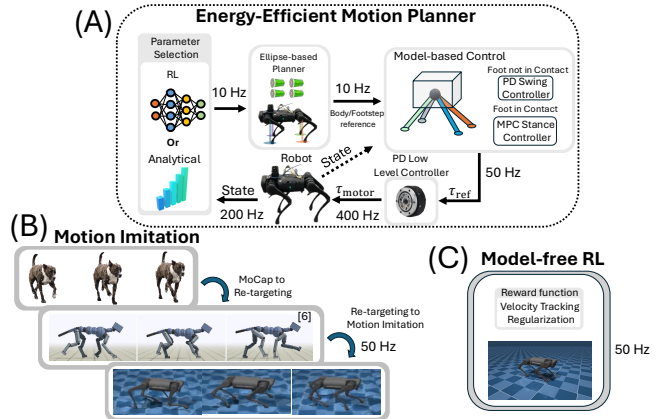


Fig. 1. We compare and evaluate our energy-efficient motion planner (A) (Sec. II), with motion imitation (B) (motion dataset from [6]) and model-free reinforcement learning (C) baselines.

time, and step height—are optimized to minimize the Cost of Transport (CoT) and maximize robustness to avoid singularity and favor non-slipping configurations. These parameters can be derived offline or adjusted through RL for improved disturbance handling. The proposed planner produces a more natural gait, where the robot moves its feet only when necessary, as defined by the placement set.

Although natural gaits can also be achieved by imitating animal motion [6], to the best of our knowledge, there has not been a clear analysis to evaluate whether imitating motion derived from biological animals is actually more energy-efficient for lower degree of freedom robots. Thus, in this work, we use CoT and robustness measures to compare our method against not only typical model-free RL approaches [7], but also against motion imitation, where the reference motions are derived from real animal movements (see Fig. 1).

Overall, we present the following **contributions**:

- 1) An online motion planning framework that naturally determines the gait by minimizing energy consumption using ellipses beneath the hip joints. Simulation and hardware experiments are presented.
- 2) The ellipses, along with step height and swing time of each leg are designed based on a parameter study that considers CoT and robustness measures. This study is achieved analytically and also through exploration of parameters using RL. Both are compared against each other for efficacy.
- 3) Our approach is compared against baselines, including model-free RL, and motion imitation using real biological dog motion priors as the reference.

*Corresponding author

[†]Alexander Schperberg, and Stefano Di Cairano are with Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, 02139, USA. {schperberg, dicairano}@merl.com

[‡]Marcel Menner was with Mitsubishi Electric Research Laboratories (MERL) during part of the work of this paper, Cambridge, MA, 02139, USA. {menner}@ieee.org

A. Related Works

Motion planning is a fundamental requirement for legged robot locomotion. Several approaches for stable motion have been proposed such as model-based (Sec. I-A.1) and learning-based methods including RL and motion imitation (Sec. I-A.2). Efficacy of these methods are typically evaluated primarily for velocity tracking, and only a few for energy-efficient locomotion (Sec. I-A.3).

1) *Model-based*: The Raibert controller [8] provides stable trajectories for legged robots that is computationally efficient, but not suitable for rough terrain nor considers energy-efficiency. For rough terrain, optimization-based techniques can be used, optimizing for the Zero-Moment Point (ZMP) through the Linear Inverted Pendulum approach [9]. Trajectory Optimization (TO) is employed if footsteps are not specified *a priori*, which helps expand the robot’s feasible workspace, and optimizes over contact forces using more complex centroidal dynamics [10]. However, TO methods are computationally expensive and difficult to apply for online-replanning.

2) *Reinforcement Learning and Motion Imitation*: To optimize over a high-precision model while being computationally efficient, model-free RL methods have become popular [7], [11], such as using a student/teacher policy in [7]. However, these methods require extensive hyperparameter tuning, heavy reward engineering, and safety consideration is less relevant as the underlying network output is difficult to understand. Another approach is through motion imitation, such as Imitation Learning (IL) [6], [12]. For example, replicating motion patterns observed in biological locomotion data collected from a real dog was proposed in [6]. By following such motion priors, the need to enforce walking behaviors such as achieving a certain step height, forward progress, or motion smoothness and so on, is greatly reduced, simplifying the number of reward terms necessary for legged locomotion. Although prior works using RL or IL can, and often do, include energy-efficient terms, the resulting policies tend to perform poorly if weights on these terms are too large. Also, these methods evaluate success based primarily on overall velocity tracking performance, not on energy-efficiency, as we do in this work.

3) *Optimizing for Energy-Efficiency*: Only few works primarily focus on energy-efficient locomotion to help reduce power consumption of the robot’s actuators. One such example can be found in [13]. Unlike our work, in [13], the step-height needs to be pre-determined, and the gait timings are a function of a pre-defined distance instead of being modifiable during operation. Although our work does not employ vision data as shown in [14], by considering energy-efficiency for various step heights implicitly provides information on which postures are optimal (energy wise) on uneven terrain. Similar to [14], we also consider slipping using force ellipsoids.

Other works aim at energy-efficiency by either optimizing locomotion parameters [15], employing policy gradient RL [4], or with adaptive control methods [16]. However, these works do not consider the effects of minimizing energy-consumption while accounting for gait transitions which is

computationally costly if using a programming solver [17], [18]. In contrast, here we address the decision at which forward velocity it becomes more energy-efficient to transition from a gait to another, e.g., walking to trot. The proposed method is also general enough not to need the exact order of legs that transition from stance to swing as required in [19].

B. Preliminaries

Here we describe the low-level controllers to track the reference trajectories from the proposed energy-efficient planner.

1) *Swing Controller*: The swing controller is similar to [20], which computes the torque for foot i for all three joints of the robot as¹:

$$\begin{aligned} \boldsymbol{\tau}_i = & \mathbf{J}_i^\top [\mathbf{K}_p (\mathbf{p}_{i,\text{ref}} - \mathbf{p}_{i,\text{cur}}) + \mathbf{K}_d (\mathbf{v}_{i,\text{ref}} - \mathbf{v}_{i,\text{cur}})] \quad (1a) \\ & + \mathbf{J}_i^\top \boldsymbol{\Lambda}_i \mathbf{a}_{i,\text{ref}} + \mathbf{V}_i (\dot{\mathbf{q}}_{i,\text{cur}}) + \mathbf{G}_i (\mathbf{q}_{i,\text{cur}}) \quad (1b) \end{aligned}$$

where $\boldsymbol{\tau}_i \in \mathbb{R}^3$ is the joint torque, $\mathbf{q}_{i,\text{cur}} \in \mathbb{R}^3$ and $\dot{\mathbf{q}}_{i,\text{cur}} \in \mathbb{R}^3$ are the current joint position and velocity of foot i , $\mathbf{J}_i \in \mathbb{R}^{3 \times 3}$ is the foot Jacobian, \mathbf{K}_p and \mathbf{K}_d are the proportional and derivative (PD) gain matrices, $\mathbf{p}_{i,\text{ref}} \in \mathbb{R}^3$ and $\mathbf{p}_{i,\text{cur}} \in \mathbb{R}^3$ are the reference and current footstep positions (calculated using forward kinematics on current joint encoders) in the body frame, $\mathbf{v}_{i,\text{ref}} \in \mathbb{R}^3$ and $\mathbf{v}_{i,\text{cur}} \in \mathbb{R}^3$ are the reference and current footstep velocities in the body frame, $\mathbf{a}_{i,\text{ref}} \in \mathbb{R}^3$ is the reference footstep acceleration in the body frame, $\mathbf{V}_i \in \mathbb{R}^3$ is the torque due to the Coriolis and centrifugal forces, $\mathbf{G}_i \in \mathbb{R}^3$ is the torque due to gravity, and $\boldsymbol{\Lambda}_i \in \mathbb{R}^{3 \times 3}$ is the operational mass matrix.

2) *Stance Controller*: For the legs in stance, i.e., making contact with the ground, we use a Model Predictive Controller (MPC) to calculate the ground reaction forces to track the reference trajectory of the body, or $\mathbf{X}_{\text{ref}} \in \mathbb{R}^{12}$, where $\mathbf{X}_{\text{ref}} = [\boldsymbol{\Theta}^\top, \mathbf{r}^\top, \boldsymbol{\omega}^\top, \mathbf{v}^\top]^\top$, $\boldsymbol{\Theta}$ is the robot’s orientation, \mathbf{r} is the CoM base position, $\boldsymbol{\omega}$ is the angular velocity, and \mathbf{v} is the linear velocity, each vector with x , y , and z components. While details of such controller are in [20], here, the ground reaction forces are used to calculate the joint torques that serve as input to the motor’s torque controller:

$$\boldsymbol{\tau}_{i,t} = \mathbf{J}_{i,t}^\top \mathbf{R}_{b,i,t}^{w,\top} \mathbf{f}_{i,t} \quad (2)$$

where $\mathbf{f}_{i,t} \in \mathbb{R}^3$ is the force vector associated with leg i as subset of \mathbf{f}_t , and $\mathbf{R}_{b,i,t}^{w,\top}$ is the rotation matrix from world to body frame of leg i at time step t .

3) *Reinforcement Learning through PPO*: We employ Proximal Policy Optimization (PPO) [21] for both our optimal parameter study (Sec. II), and for our model-free and motion imitation baselines. Only the reward formulation differs for each use case. In short, PPO enhances the policy $\pi_\theta(\mathbf{a} | \mathbf{O})$ by maximizing the clipped objective function:

$$L(\theta) = \mathbb{E}_t \left[\min \left(r_t(\theta) \hat{A}_t, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_t \right) \right], \quad (3)$$

Here, $r_t(\theta)$ denotes the ratio of the new policy to the old one, \hat{A}_t represents the estimated advantage, and ϵ is a threshold

¹We omit time step t for simplicity.

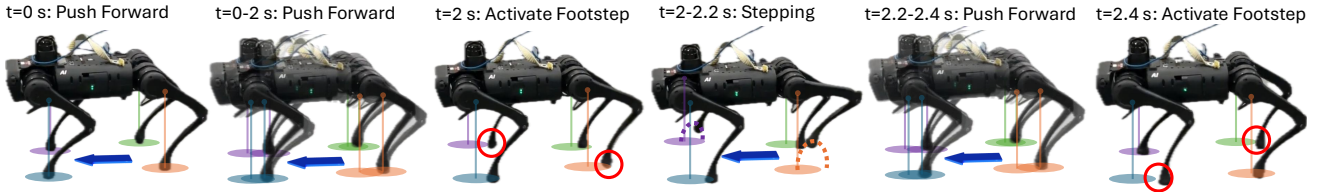


Fig. 2. We show the process of our energy-efficient motion planner. With a desired forward velocity, from $t = 0 - 2$ s, the robot’s body pushes forward using the MPC stance controller from Sec. I-B.2, until one or more legs are outside of the ellipse (red circle), which activates the footstep as shown at $t = 2$ s. The robot then takes a step, $t = 2 - 2.2$ s, and the process repeats. Note, the center position of the ellipse is beneath the hip joint at all times.

that constrains policy updates to maintain stability. At each time step t , the PPO-derived policy outputs an action \mathbf{a}_t , which is then executed on the system: $\mathbf{a}_t = \pi_\theta(\mathbf{O}_t)$, where \mathbf{O}_t refers to the current observation.

The observation space at time step t for our model-free baseline is denoted by \mathbf{O}_t^M while for our motion imitation baseline as well as for optimizing our planner parameter selection (see Sec. II) is denoted by \mathbf{O}_t^I (joints from all legs i are considered in the observation space):

$$\begin{aligned} \mathbf{O}_t^M &= [\mathbf{q}_{\text{cur}}, \dot{\mathbf{q}}_{\text{cur}}, \boldsymbol{\tau}, \mathbf{X}_{\text{cur}}, \mathbf{a}_{t-1}]_t, \\ \mathbf{O}_t^I &= [\mathbf{q}_{\text{cur}}, \dot{\mathbf{q}}_{\text{cur}}, \boldsymbol{\tau}, \mathbf{X}_{\text{cur}}, \mathbf{a}_{t-1}, \mathbf{q}_{\text{ref}}, \mathbf{X}_{\text{ref}}]_t, \end{aligned} \quad (4)$$

where the new variables include current body state described by \mathbf{X}_{cur} , the previous actions by \mathbf{a}_{t-1} , and reference joint trajectory by \mathbf{q}_{ref} . Note, although not shown explicitly in equation (4), we do include the observational history up to N_{prev} as part of our overall observation space, or $\mathbf{O}_t = [\mathbf{O}_{t:t-N_{\text{prev}}}]$. During model-free and motion imitation training, \mathbf{a}_t represents desired joint positions, whereas in energy-efficient motion planning it serves as parameters of our planner such as swing time or ellipse shape (Sec. II).

Different combination of rewards were used depending on the RL training, although hyper-parameters were kept constant and w , σ_{track} represent weights on reward terms (see Table I). For the motion imitation baseline, we used all the terms listed in (1), and (2) in Table I. We used data from [6] as motion priors, which involve re-targeting mocap data from a real dog. For the model-free RL baseline we used all the terms listed in (2), and (3) in Table I. Finally, for the parameter selection study described in detail in Sec. II, we used only the terms listed in (4) in Table I. Note, we do not include all details of our hyper-parameter selection and domain randomization, as they follow similar structure as found in many other RL locomotion works such as [7].

II. ENERGY-EFFICIENT MOTION PLANNER

Our planner is based on the relative position of the center of mass of the robot and the robot’s feet. In particular, it uses the geometric shape of an ellipse as the placement set for the feet. For each leg of the robot, the center point of the ellipse is always located directly beneath the hip joint, which imposes a stable position. Other shapes for the placement sets can be used, but ellipses are an intuitive geometric shape for this purpose. We define this ellipse through their major/minor axis, r_{ell}^x and r_{ell}^y , with center point position, i.e., the robot’s hip position, p_{hip}^x and p_{hip}^y . As the robot’s center of mass

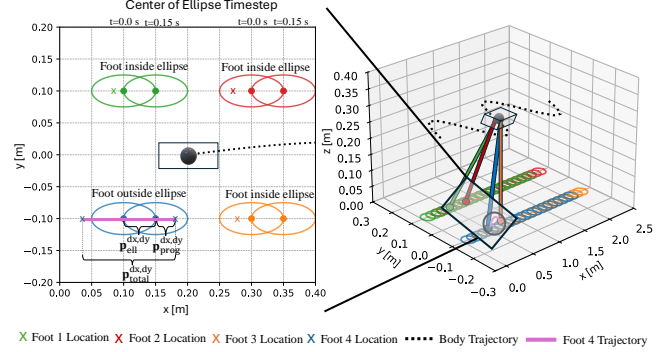


Fig. 3. We show a 2D representation of the footstep ellipses and their variable representation as described in Sec. II on the left-hand side. A full 3D sequence is given in Fig. 2.

moves using a desired longitudinal and lateral commanded velocity, \mathbf{V}_{des} , through the ground reaction forces calculated by the MPC [20] in Section I-B.2, at each time step, we calculate the difference between the current footstep position, p_{cur}^x and p_{cur}^y , and the hip position:

$$p_{\text{ell}}^{dx} = p_{\text{cur}}^x - p_{\text{hip}}^x \quad (5a)$$

$$p_{\text{ell}}^{dy} = p_{\text{cur}}^y - p_{\text{hip}}^y. \quad (5b)$$

Given by a contact matrix \mathbf{C}_{ref} to indicate which feet are in stance or swing phase, using p_{ell}^{dx} and p_{ell}^{dy} , we can check whether the current foot is outside this ellipse for each leg. For each foot in contact, the check can be represented by:

$$\frac{(p_{\text{ell}}^{dx})^2}{(r_{\text{ell}}^x)^2} + \frac{(p_{\text{ell}}^{dy})^2}{(r_{\text{ell}}^y)^2} > 1. \quad (6)$$

If the check in (6) is true for any of the legs, we plan for the next footstep swing trajectory and corresponding body trajectory. The planner outputs the reference footstep position, velocity, and acceleration given by \mathbf{p}_{ref} , $\dot{\mathbf{p}}_{\text{ref}}$, $\ddot{\mathbf{p}}_{\text{ref}}$ respectively, and the body trajectory, calculated using Euler discretization on the commanded velocity, \mathbf{V}_{des} , and current state, \mathbf{X}_{cur} , to yield the reference \mathbf{X}_{ref} .

When all feet are inside their corresponding ellipse, the reference trajectory stays in place. In other words, if the user commands a forward velocity, the MPC controller pushes the robot’s body in that direction with all feet in stance. Once one of the foot locations is outside the ellipse, the planner produce a swing trajectory. The reference trajectory for the footstep in swing is calculated by moving the current footstep

Algorithm 1: Energy-Efficient Motion Planner

```

1 while Robot in Operation do
2    $\mathbf{X}_{\text{cur}}, \mathbf{p}_{\text{cur}} \leftarrow \text{StateEstimation}(\text{See [22]})$ 
   // Get user commands (10 Hz)
3    $\mathbf{V}_{\text{des}} \leftarrow \text{Joystick}$ 
   // Get planner parameters
4    $[\Delta T, h_{\text{step}}, X^z, r_{\text{ell}}^x, r_{\text{ell}}^y] \leftarrow$ 
   Analytical or RL (Sec. II)

   // For each leg
5    $\mathbf{p}_{\text{ell}}^{dx} = \mathbf{p}_{\text{cur}}^x - \mathbf{p}_{\text{hip}}^x$ 
6    $\mathbf{p}_{\text{ell}}^{dy} = \mathbf{p}_{\text{cur}}^y - \mathbf{p}_{\text{hip}}^y$ 

   // Check if foot in contact is outside
   ellipse
7   Outside Ellipse?  $\leftarrow$ 
   checkEllipse( $\mathbf{p}_{\text{ell}}^{dx}, \mathbf{p}_{\text{ell}}^{dy}, r_{\text{ell}}^x, r_{\text{ell}}^y$ )

   // If outside the ellipse, calculate a new
   reference trajectory
8   if Outside Ellipse = True :
9      $\mathbf{p}_{\text{prog}}^{dx} = V_{x,\text{des}}^B \Delta T + \sqrt{X^z/g}(V_{x,\text{cur}}^B - V_{x,\text{des}}^B)$ 
10     $\mathbf{p}_{\text{prog}}^{dy} = V_{y,\text{des}}^B \Delta T + \sqrt{X^z/g}(V_{y,\text{cur}}^B - V_{y,\text{des}}^B)$ 
11     $\mathbf{p}_{\text{ref}}, \dot{\mathbf{p}}_{\text{ref}}, \ddot{\mathbf{p}}_{\text{ref}}, \mathbf{C}_{\text{ref}} \leftarrow$ 
    Sinusoidal( $dt, \Delta T, \text{gait}, \mathbf{p}_{\text{prog}}^{dx,dy}, \mathbf{p}_{\text{ell}}^{dx,dy}, h_{\text{step}}$ )
12     $\mathbf{X}_{\text{ref}} \leftarrow \text{interp}(\mathbf{X}_{\text{cur}}, \mathbf{V}_{\text{des}})$ 

    // If no legs are outside the ellipse nor
    currently in swing phase, we put all
    legs in stance, and include the
    reference commanded body velocity to
    push body forwards
13   if Swing Phase Complete = True :
14      $\mathbf{X}_{\text{next}}, \mathbf{p}_{\text{next}}, \dot{\mathbf{p}}_{\text{next}}, \ddot{\mathbf{p}}_{\text{next}}, \mathbf{C}_{\text{next}} \leftarrow$ 
      $\mathbf{X}_{\text{cur}}, \mathbf{p}_{\text{cur}}, \mathbf{0}^{12}, \mathbf{0}^{12}, \mathbf{1}^4$ 
     // Otherwise, iterate through the
     reference trajectory
15   else:
16      $\mathbf{X}_{\text{next}}, \mathbf{p}_{\text{next}}, \dot{\mathbf{p}}_{\text{next}}, \ddot{\mathbf{p}}_{\text{next}}, \mathbf{C}_{\text{next}} \leftarrow$ 
      $\mathbf{X}_{\text{ref}}[it], \mathbf{p}_{\text{ref}}[it], \dot{\mathbf{p}}_{\text{ref}}[it], \ddot{\mathbf{p}}_{\text{ref}}[it], \mathbf{C}_{\text{ref}}[it]$ 
17   Send references to swing (1) and stance (2)
   controllers.
18 end

```

position first to the center of the ellipse, and then adding an additional term, p_{prog}^{dx} and p_{prog}^{dy} (see Fig. 3), to account for the desired commanded velocity:

$$p_{\text{prog}}^{dx} = V_{x,\text{des}}^B \Delta T / 2 + \sqrt{X^z/g}(V_{x,\text{cur}}^B - V_{x,\text{des}}^B) \quad (7a)$$

$$p_{\text{prog}}^{dy} = V_{y,\text{des}}^B \Delta T / 2 + \sqrt{X^z/g}(V_{y,\text{cur}}^B - V_{y,\text{des}}^B) \quad (7b)$$

where X^z is the height of the robot, g is the gravity term, ΔT is the footstep timing for the swing phase, and B is to symbolize that we expect the desired velocity, $V_{x,\text{des}}^B$ and $V_{y,\text{des}}^B$, to be in the body frame. Note, this is derived from the Raibert Heuristic [8], where the the feedback gain k is chosen to be equal to $\sqrt{X^z/g}$.

TABLE I

REWARD FUNCTIONS. SEE SEC. I-B.3 FOR DETAILS ON USAGE.

(1) Term (Imitation)	Equation
Joint Position Reward	$\left(e^{-\frac{\sum \ \mathbf{q}_{\text{cur}} - \mathbf{q}_{\text{ref}}\ ^2}{\sigma_{\text{track}}}} \right) w_1$
Foot Position Reward	$\left(e^{-\frac{\sum \ \mathbf{p}_{\text{cur}} - \mathbf{p}_{\text{ref}}\ ^2}{\sigma_{\text{track}}}} \right) w_2$
(2) Term (Tracking)	Equation
Linear Velocity in XY Plane	$\left(e^{-\frac{\ \mathbf{v}_{\text{cur},xy} - \mathbf{v}_{\text{ref},xy}\ ^2}{\sigma_{\text{track}}}} \right) w_3$
Angular Velocity in Z Axis	$\left(e^{-\frac{\ \omega_{\text{cur},z} - \omega_{\text{ref},z}\ ^2}{\sigma_{\text{track}}}} \right) w_4$
Linear Velocity in Z Axis	$-(\ \mathbf{v}_{\text{cur},z} - \mathbf{v}_{\text{ref},z}\ ^2) w_5$
(3) Term (Regularization)	Equation
Joint Accelerations	$-(\sum \ \dot{\mathbf{q}}\ ^2) w_6$
Action Rate	$-(\sum \ \mathbf{a}_t - \mathbf{a}_{t-1}\ ^2) w_7$
Foot Airtime	$(\sum t_{\text{air}} 1_{\text{new.contact}}) w_8$ [7]
Self-collision	$(1_{\text{collision}}) w_9$ [7]
(4) Term (Parameter Selection)	Equation
Cost of Transport	$-(\text{CoT}) w_{10}$ eq.(9)
Robustness	(Manipulability) w_{11} eq.(10)
Number of Time Steps	$(t/t_{\text{max}}) w_{12}$

The reference trajectory from the current footstep \mathbf{p}_{cur} to the next footstep or \mathbf{p}_{next} is created using a sinusoidal implementation:

$$\bar{a}_z = h_{\text{step}} \frac{1}{2} \left(\frac{2\pi}{\Delta T} \right)^2 \quad (8a)$$

$$a_{z,t} = \bar{a}_z \cos \left(\frac{2\pi}{\Delta T} t \right) \quad (8b)$$

$$v_{z,t} = \bar{a}_z \frac{\Delta T}{2\pi} \sin \left(\frac{2\pi}{\Delta T} t \right) \quad (8c)$$

$$p_{z,t} = \bar{a}_z \left(\frac{\Delta T}{2\pi} \right)^2 \left(1 - \cos \left(\frac{2\pi}{\Delta T} t \right) \right) \quad (8d)$$

with step height h_{step} , maximum vertical acceleration \bar{a}_z , and:

$$\bar{a}_x = (p_{\text{ell}}^{dx} + p_{\text{prog}}^{dx}) \frac{1}{2} \left(\frac{\pi}{\Delta T} \right)^2 \quad (8e)$$

$$\bar{a}_y = (p_{\text{ell}}^{dy} + p_{\text{prog}}^{dy}) \frac{1}{2} \left(\frac{\pi}{\Delta T} \right)^2 \quad (8f)$$

$$a_{x/y,t} = \bar{a}_{x/y} \cos \left(\frac{\pi}{\Delta T} t \right) \quad (8g)$$

$$v_{x/y,t} = \bar{a}_{x/y} \frac{\Delta T}{\pi} \sin \left(\frac{\pi}{\Delta T} t \right) \quad (8h)$$

$$p_{x/y,t} = \bar{a}_{x/y} \left(\frac{\Delta T}{\pi} \right)^2 \left(1 - \cos \left(\frac{\pi}{\Delta T} t \right) \right) \quad (8i)$$

with maximum lateral acceleration $\bar{a}_{x/y}$. Note that the proposed footstep motion planner does not rely on the choice of trajectory. Different implementations can be used as well. In particular, for climbing stairs or traversing rubble and obstacles, (8) may be modified to alter the touchdown point.

The contact matrix $\mathbf{C}_{\text{ref}} \in \mathbb{R}^4$ specifying which leg is in stance or swing phase can be pre-determined if a certain gait is desired, e.g., trot or walk, or can be determined by the algorithm. E.g., in the pre-determined case for a trot gait, if one of the footsteps is outside the defined ellipse, the foot along with the diagonally opposing footstep are constrained to move at the same time from stance into swing. In this case, we only move the leg and its diagonally opposing leg in alternating order, e.g., if legs 1 and 3 were in swing previously, we must move legs 2 and 4 next. In the non-pre-determined case, we remove such constraints (except that we cannot move the same leg twice). Instead, we only move the leg that passes (6), allowing \mathbf{C}_{ref} to freely vary. In both cases, since foot movement follows (6), stance phase duration varies with current body position relative to the feet. Thus, unique gaits and footstep timings for each step are generated.

We now detail how to enforce energy-efficiency for our motion planner. A common metric for energy-efficient planning is the Cost of Transport (CoT), which can be computed using the joint velocity $\dot{\mathbf{q}}_t$, torque $\boldsymbol{\tau}_t$, and distance ΔD traveled:

$$\text{CoT} = \frac{\sum_{t=k-N}^k \max\{\dot{\mathbf{q}}_t^\top \boldsymbol{\tau}_t, 0\} dt}{\Delta D} \quad (9)$$

where k is the current time step discretized with sampling time dt , and N is the number of time steps within total time of locomotion considered.

For each desired velocity commanded by the user (\mathbf{V}_{des}), we determine ideal parameters for the proposed planner that achieve not only the lowest CoT but also favors robot configurations for which the legs in stance can most easily produce ground reaction forces. To quantify the favorable configurations, we leverage force manipulability ellipsoids [23]. These ellipsoids provide a concise and intuitive representation of the force and moment capabilities of a robot at a particular point in its workspace. Mathematically, the motor torque vector is mapped into a unit sphere $\boldsymbol{\tau}_i^\top \boldsymbol{\tau}_i \leq \mathbf{1}$ in joint space and into the ellipsoids $\mathbf{f}_i^\top \mathbf{J}_i \mathbf{J}_i^\top \mathbf{f}_i \leq \mathbf{1}$ in task space, where \mathbf{J}_i and \mathbf{f}_i are the Jacobian and end-effector ground reaction forces, respectively. We can estimate a representation of the volume of these force ellipsoids through the force manipulability measure [23]. A high force manipulability measure indicates a better ability to apply forces in various directions, while a lower manipulability measure indicates a more limited ability to do so. To calculate this force manipulability measure, we first take the eigenvalues, $\boldsymbol{\lambda}$, of the matrix $U = (\mathbf{J}_i \mathbf{J}_i^\top)^{-1}$, and:

$$\text{manipulability measure} = \sqrt{\max(\boldsymbol{\lambda}) / \min(\boldsymbol{\lambda})}. \quad (10)$$

Thus, our goal is to find the parameters for each commanded body velocity that minimize the CoT and maximize the manipulability measure of the force ellipsoids for increased robustness. To do so, we compute both CoT and the force manipulability measure as a function of the planner parameters using Mujoco [24] simulation as our physics model for propagating from the current to the next time step. In particular, we study the impact of the swing time ΔT ,

TABLE II
PARAMETERS USED FOR PARAMETER STUDY

Name	Description	min	max	units
$V_{x,\text{des}}^B$	Desired Body Velocity	0.05	1.00	$\frac{\text{m}}{\text{s}}$
ΔT	Swing Time	0.10	0.25	s
h_{step}	Step height	0.05	0.15	m
X^z	Robot Height	0.28	0.311	m
$r_{\text{ell}}^x, r_{\text{ell}}^y$	Ellipse axis in x/y direction	0.01	0.15	m

body velocity in the x -direction $V_{x,\text{des}}^B$, step height h_{step} , robot height X^z , and the ellipse size of the planner r_{ell}^x and r_{ell}^y . For each combination of parameters, we calculate the average CoT and force manipulability measure during about 20 seconds of locomotion. The range of parameters tested is described in Table II. To calculate the CoT and the force manipulability measure, the joint torques for the feet in swing and stance are needed. These torques can be received directly from Mujoco. We call this method of parameter selection as Energy-Efficient Motion Planner using the *analytical approach* (or EEMP-Analytical), because we can find these parameters without any learning methods—we simply iterate through a range of velocities to find which parameter combination produced the ideal CoT and manipulability measure for each commanded velocity and employ a look-up table during evaluation.

In addition to this *analytical approach*, we also employ a simple RL formulation using PPO (Sec. I-B.3) to find these parameters through exploration and compare against the *analytical approach*. We term this as EEMP-RL. In this approach, the actions are the parameters of our planner from Table II, or $\mathbf{a}_t = [\Delta T, h_{\text{step}}, X^z, r_{\text{ell}}^x, r_{\text{ell}}^y]_t$, and the observations are described by \mathbf{O}_t^i in equation (4), using IK on \mathbf{p}_{ref} to get \mathbf{q}_{ref} . Our reward function consist of only the three terms listed in (4) of Table I, which is to penalize the agent for high CoT, reward high manipulability, and to reward the agent for not falling to the ground—where t is the current time step during an episode, and t_{max} is the maximum allotted time step per episode or 20 seconds of locomotion.

The overall footstep planner is described in Algorithm 1, and visually shown in 3D and 2D in Figs. 2 and 3 respectively. Summarizing, we first get the state of the robot [22], and then receive the desired velocity command from the user. Planner parameters are then selected through either EEMP-Analytical or EEMP-RL. We then check (6) to see if any of the footsteps in contact with the ground are currently outside the defined ellipse. If one of the footsteps is outside the ellipse, we calculate the footstep trajectory (7), (8) and the body trajectory through Euler discretization for the commanded velocity. Additionally, we check if we fully iterated through the swing trajectory based on the swing time ΔT . If so, we produce four stance phase reference trajectories for all legs, i.e., the robot feet are commanded to stay in position, while the body reference, $\mathbf{X}_{\text{next}} \in \mathbb{R}^{12}$, is such that it moves in the direction of the desired velocity. If the swing phase is not completed, we proceed to the next

TABLE III

BEST PARAMETERS FOR WALK / TROT GAIT ACROSS VELOCITY RANGE

Name	Description	Value	units
ΔT	Swing Time	0.25	s
h_{step}	Step height	0.10	m
X^z	Robot Height	0.31	m
$r_{\text{ell}}^x, r_{\text{ell}}^y$	Ellipse axis in x/y directions	[0.07, 0.05]	m

time step of the swing reference trajectory. References are then sent to our downstream stance and swing controllers.

III. PARAMETER STUDY RESULTS

We first applied the EEMP-Analytical approach (Sec. II) to examine how optimal parameter selection affects CoT and manipulability across various lateral velocities (x -direction) during 20 seconds of locomotion and setting the gait to be either in walk or trot to compare both. Our analysis showed that the parameters in Table III consistently yielded the lowest CoT and highest manipulability *on average*, i.e., across the velocity range in walk/trot. However, optimizing parameters for each velocity improved CoT by 35.5% in the trot gait and 13.3% in the walk gait over using the constant set of parameters from Table III.

Fig. 4 shows contour plots of the parameter selection, where for each commanded velocity we show the ellipse's major x axis, r_{ell}^x (toward the locomotion direction), swing time, and color coding represented by CoT. The results of the trot gait are shown on the left and of the walk gait on the right half. For the trot gait, we show that high swing times are critical at low velocities, whereas the selection for r_{ell}^x appears less important. As velocity increases to 0.50 m/s, the combination of r_{ell}^x and swing time becomes critical, where a swing time of about 0.18 seconds and r_{ell}^x of about 0.06 to 0.08 meters, or high swing time and high r_{ell}^x leads to optimal CoT. However, at velocities going towards 1.0 m/s, lower swing times of 0.14-0.16 seconds become more favorable, while r_{ell}^x has minimal impact in this range. This is intuitive since, at higher velocities, the footstep often falls outside the ellipse, triggering constant step initiation regardless of r_{ell}^x .

For the walk gait, we see the same trend as for the trot gait in low velocities. However, different from the trot gait, a high swing time is more favorable at nearly all choice of velocities. Although the r_{ell}^x parameter seems more important if the user selects lower swing times, at higher velocities r_{ell}^x loses effect, particularly at 1.0 m/s due to a similar reason as for the trot gait. We make a comparison for the CoT and manipulability measure between the trot and walk gait in Fig. 5. In the velocity range up to 0.4 m/s, the walk gait has both a more favorable CoT and manipulability measure relative to trot. This is expected as energy consumption is estimated by summing joint accelerations and torques across all legs, with the swing phase being most significant. Since walk gait has one leg in swing phase at a time, it results in lower energy consumption across all velocities, as confirmed by our results. As the manipulability measure is based on how well the robot can produce ground reaction forces at

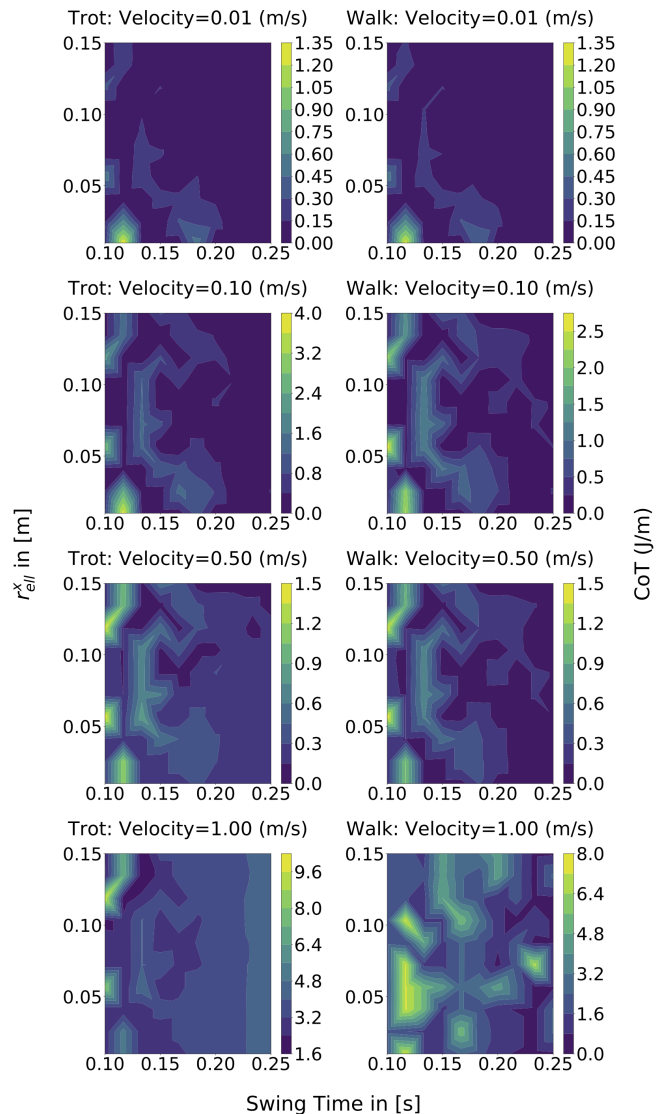


Fig. 4. Contour plots for different commanded velocities (namely, 0.01, 0.10, 0.50, and 1.00 m/s), and for varying major axis of the planning ellipse r_{ell}^x (y-axis) and swing time (x-axis). The color varies with the values of CoT, from yellow to blue. The graphs on the left half show the trot gait results, while the graphs on the right half show the walk gait results.

any given time, the walk gait is expected to show higher performance as well since it has three legs in stance phase, instead of two legs for the trot gait. Overall, the walk gait used 59% of energy of the trot gait in this velocity range. At velocities exceeding 0.4 m/s (marked by the red line in Fig. 5), the walk gait becomes unstable, evidenced by a sharp increase in CoT and a decline in the manipulability measure. This instability is consistently observed in both simulation and hardware experiments at velocities above 0.4 m/s.

Finally, we also compare EEMP-Analytical with parameter selection using RL (EEMP-RL, Sec. II). As shown in Fig. 6, EEMP-RL (red) slightly outperforms EEMP-Analytical (blue) in both CoT and manipulability. This indicates two key points: first, our method achieves efficient locomotion without requiring RL, as both methods show

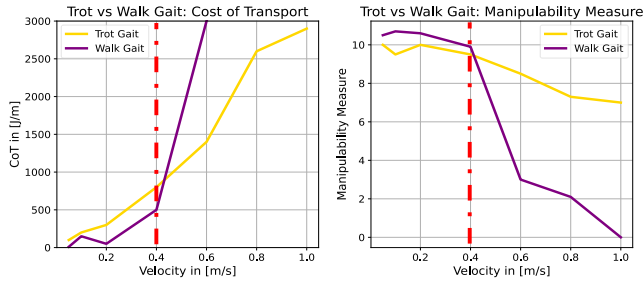


Fig. 5. CoT (left) and the manipulability measure (right) for the trot (yellow) and walk (purple) gaits at various velocities. The red line indicates the velocity at which walk gait becomes unstable.

good performance; second, RL improves performance by better handling disturbances from estimation errors, contacts, or velocity transitions. For instance, RL can optimize parameters in continuous space for smoother gait changes, instead of relying on a discretized look-up table (i.e., EEMP-Analytical), thereby enhancing overall energy efficiency.

IV. EXPERIMENTAL VALIDATION

1) *Controller Architecture and Baseline Comparisons:* In simulation and hardware, the velocity commands are provided by the operator, via a joystick. For state estimation we use a similar approach as [22]. (A) in Fig. 1 summarizes the control architecture and control frequency used in this paper for both Mujoco simulation and hardware experiments. We also show two other baseline methods for achieving legged locomotion, namely through motion imitation and model-free RL (Sec. I-B.3). The results of training these baselines, along with our optimal parameter selection (EEMP-RL) is shown in Fig. 7. We note that we trained our RL baselines on flat terrain for 100 million time steps, and on rough terrain for 50 million time steps (thus, the drop in rewards occur due to the switch of terrains, which is expected), and we required only 10 million time steps directly on rough terrain for EEMP-RL.

2) *Mujoco Simulation Results:* For 20 seconds of locomotion per desired velocity, we compare our method, EEMP-Analytical (blue) and EEMP-RL (red), without predefining walk/trot for parity, with motion imitation (orange) and model-free RL (green) baselines using CoT, manipulability, and velocity tracking as evaluation metrics. Overall, our method performs best at ≤ 0.6 m/s across all metrics (i.e., CoT, manipulability, and velocity tracking). At higher velocities, model-free RL and motion imitation slightly outperform EEMP methods in CoT, but perform similarly in velocity tracking and manipulability using EEMP-RL. This is expected, as higher velocities shrink the ellipses to a point, requiring constant stepping, thus reducing the energy-efficiency benefits of our planner. Additionally, nonlinear dynamics at high speeds challenge the linear MPC stance controller assumption underlying the EEMP framework, a limitation of this work. Addressing this may require nonlinear models, at the cost of higher computation, or learning such models offline with neural nets. Notably, model-free RL benefits from tightly coupling the planner and controller

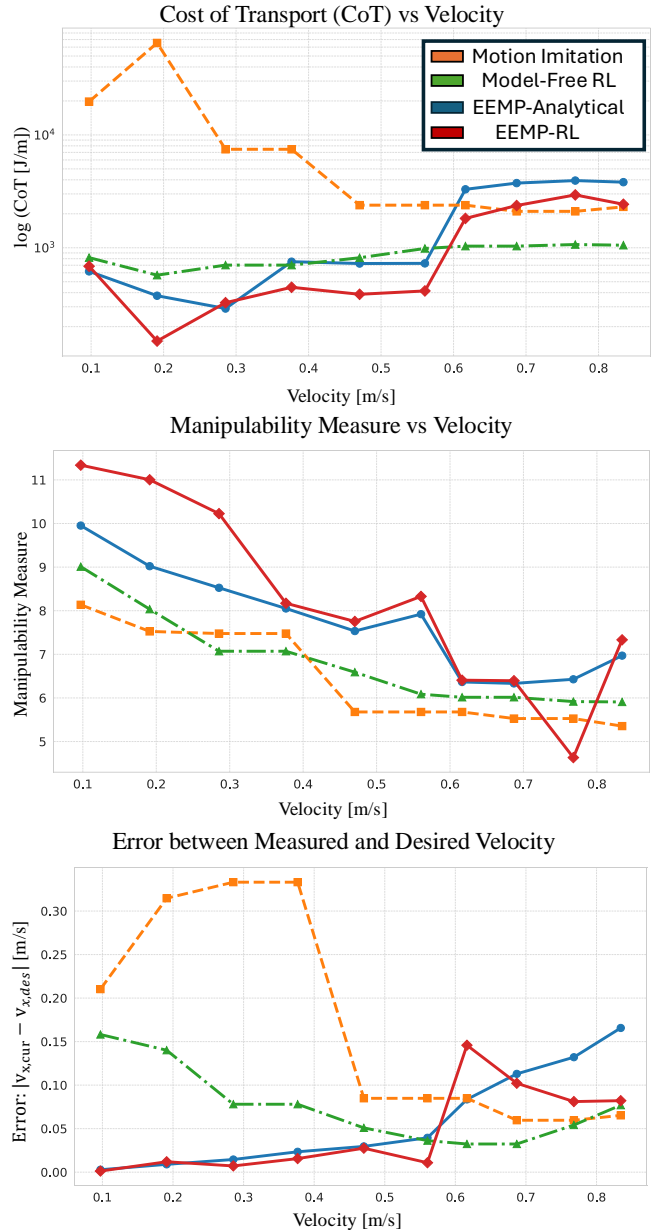


Fig. 6. CoT, manipulability measure, and velocity tracking (y-axis) for various desired velocities (x-axis) are shown, comparing our Energy-Efficient Motion Planner (EEMP) using the analytical approach and RL approach for parameter selection in blue and red respectively, motion imitation in orange, and model-free RL in green.

(unlike our EEMP methods), which proves advantageous at higher speeds. Interestingly, motion imitation underperforms compared to model-free RL at nearly all velocities, suggesting that biological imitation may be less effective for lower-DoF robots, such as a 3-DoF per leg quadruped. Based on these findings, the most optimal energy-efficient motion would be one that uses EEMP-RL for velocities ≤ 0.6 m/s, and then transition to model-free RL.

3) *Hardware Validation:* We teleoperated the robot shown in Fig. 8 using a desired commanded walking speed to test how the proposed footstep planner and control framework

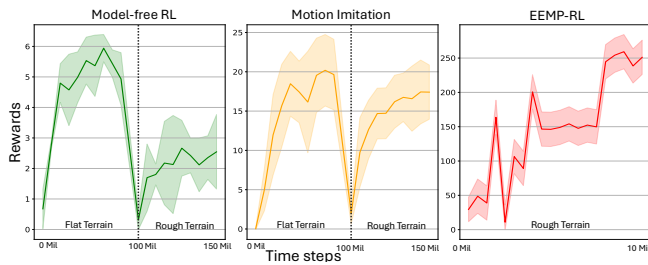


Fig. 7. Rewards for model-free RL and motion imitation baselines, and our Energy-Efficient Motion Planner using RL (EEMP-RL).

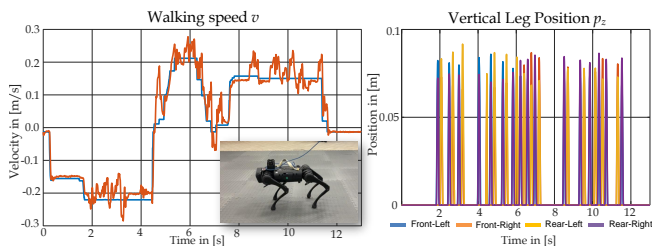


Fig. 8. Results of teleoperation with our architecture to demonstrate robustness against disturbances on hardware. Using a joystick, we commanded the robot with velocities shown in blue arbitrarily selected, while orange indicates the actual velocity received from state estimation (left graph). The corresponding vertical leg positions are shown in the right graph.

handles disturbance rejection. Fig. 8 shows an example of a trot gait, where the left plot shows the commanded forward velocity and the robot’s velocity from the state estimator. The vertical leg trajectories are shown on the right figure. From about 6.5 to 8.5 s, the vertical positions of the legs are zero indicating that all four legs are in stance. The left plot shows a forward walking speed in this period. This is a feature of the planner, the robot only executes a step when necessary, specifically, when the foot location is outside the specified ellipse. Thus, the body tracks the commanded velocity while conserving energy for some seconds by pushing the stance legs to move the body forward without taking a step.

V. CONCLUSIONS

A motion planner for legged robots was developed that is energy efficient. We show a 34.5% improvement in CoT over a baseline with fixed parameters for trot gait, and a 13.3% improvement in walk gait by employing a parameter design framework that uses a motion model based on Newtonian mechanics. Overall, we found that the walk gait used 59% of energy of the trot gait. For low to medium velocity, our method outperforms model-free RL, in CoT, manipulability, and velocity tracking. We show a 50.4% improvement in CoT over this baseline. We further validated our framework in hardware experiments, and could successfully demonstrate robustness against disturbances, such as slipping. Significantly, we presented a motion planner that is easy to implement, computationally efficient, and saves energy by having the robot only make a step when necessary. Coupling of leg dynamics, other geometric shape representations, and providing further energy-efficient comparisons in hardware are left as future work.

REFERENCES

- [1] P. A. Bhounsule, J. Cortell, and A. Ruina, “Design and control of ranger: An energy-efficient, dynamic walking robot,” *Adaptive Mobile Robotics*, pp. 441–448, 2012.
- [2] M. Hutter *et al.*, “Anymal - a highly mobile and dynamic quadrupedal robot,” in *2016 IEEE IROS*, pp. 38–44, 2016.
- [3] A. Schperberg, S. D. Cairano, and M. Menner, “Auto-tuning of controller and online trajectory planner for legged robots,” *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7802–7809, 2022.
- [4] Z. Sun and N. Roos, “An energy efficient dynamic gait for a Nao robot,” in *2014 IEEE ICARSC*, pp. 267–272, 2014.
- [5] L. Chen, S. Ye, C. Sun, A. Zhang, G. Deng, T. Liao, and J. Sun, “Cnns based foothold selection for energy-efficient quadruped locomotion over rough terrains,” in *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 1115–1120, 2019.
- [6] X. B. Peng, E. Coumans, T. Zhang, T.-W. E. Lee, J. Tan, and S. Levine, “Learning agile robotic locomotion skills by imitating animals,” in *Robotics: Science and Systems*, 07 2020.
- [7] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, “Learning quadrupedal locomotion over challenging terrain,” *Science Robotics*, vol. 5, no. 47, 2020.
- [8] M. H. Raibert, *Legged Robots That Balance*. USA: Massachusetts Institute of Technology, 1986.
- [9] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa, “Biped walking pattern generation by using preview control of zero-moment point,” in *2003 IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1620–1626, 2003.
- [10] A. W. Winkler, C. D. Bellicoso, M. Hutter, and J. Buchli, “Gait and trajectory optimization for legged systems through phase-based end-effector parameterization,” *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1560–1567, 2018.
- [11] K. Ito and F. Matsuno, “A study of reinforcement learning for the robot with many degrees of freedom - acquisition of locomotion patterns for multi-legged robot,” in *2002 IEEE International Conference on Robotics and Automation*, vol. 4, pp. 3392–3397, 2002.
- [12] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne, “Deepmimic: example-guided deep reinforcement learning of physics-based character skills,” *ACM Trans. Graph.*, vol. 37, July 2018.
- [13] H.-K. Shin and B. K. Kim, “Energy-efficient gait planning and control for biped robots utilizing the allowable ZMP region,” *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 986–993, 2014.
- [14] L. Chen, S. Ye, C. Sun, A. Zhang, G. Deng, T. Liao, and J. Sun, “CNNs based foothold selection for energy-efficient quadruped locomotion over rough terrains,” in *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 1115–1120, 2019.
- [15] W. Xi and C. D. Remy, “Optimal gaits and motions for legged robots,” in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3259–3265, 2014.
- [16] A. Mutka, E. Kočo, and Z. Kovačić, “Adaptive control of quadruped locomotion through variable compliance of revolute spiral feet,” *IJARS*, vol. 11, no. 10, p. 162, 2014.
- [17] N. Chakraborty, S. Berard, S. Akella, and J. Trinkle, “A geometrically implicit time-stepping method for multibody systems with intermittent contact,” *IJRR*, vol. 33, no. 3, pp. 426–445, 2014.
- [18] Z. Manchester, N. Doshi, R. J. Wood, and S. Kuindersma, “Contact-implicit trajectory optimization using variational integrators,” *IJRR*, vol. 38, no. 12-13, pp. 1463–1476, 2019.
- [19] Z. Fu, A. Kumar, J. Malik, and D. Pathak, “Minimizing energy consumption leads to the emergence of gaits in legged robots,” in *Conference on Robot Learning (CoRL)*, 2021.
- [20] J. Di Carlo, P. M. Wensing, B. Katz, G. Bleidt, and S. Kim, “Dynamic locomotion in the MIT Cheetah 3 through convex model-predictive control,” in *2018 IEEE/RSJ IROS*, pp. 1–9, 2018.
- [21] J. Schulman *et al.*, “Proximal policy optimization algorithms,” <https://arxiv.org/abs/1707.06347>, 2017.
- [22] M. Menner and K. Berntorp, “Simultaneous state estimation and contact detection for legged robots by multiple-model kalman filtering,” in *2024 (ECC)*, pp. 2768–2773, 2024.
- [23] C. Melchiorri, “Force manipulability ellipsoids for general manipulation systems,” *IFAC Proceedings Volumes*, vol. 27, no. 14, pp. 235–240, 1994. Fourth IFAC Symposium on Robot Control, Capri, Italy, September 19-21, 1994.
- [24] E. Todorov, T. Erez, and Y. Tassa, “Mujoco: A physics engine for model-based control,” in *2012 IEEE/RSJ IROS*, pp. 5026–5033, IEEE, 2012.