LocoJump: Controlled Versatile Jumping from Locomotion of Quadruped Robot

Yulun Zhuang
Robotics
University of Michigan
Ann Arbor, United States
yulunz@umich.edu

***Index Terms*—Legged Locomotion, Trajectory Optimization, Reinforcement Learning**

I. Introduction

A. Background

The agile locomotion capabilities (e.g. parkour in Figure 1) of legged robots are of vital important yet highly challenged for applications like search and rescue. With the recent advancement in Reinforcement Learning (RL), its application in legged robot control has shown promising result to generate highly dynamic and robust locomotion policies. This project focuses on utilizing the state-of-the-art toolboxes to train RL aided model-based control policies for legged robots in simulation.

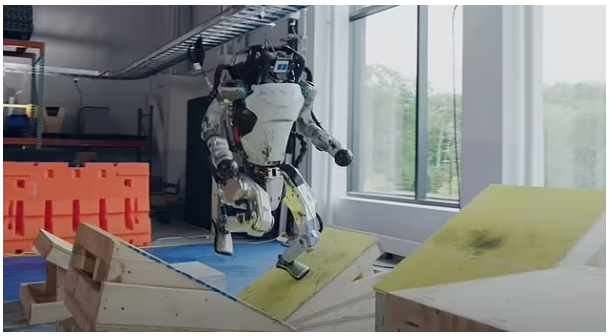

Fig. 1. The parkour motion performed by Atlas Robot

B. Related Works

- Optimized jumping of quadruped [1]
- Running jump via online optimization [2]
- Continuous jumping via MPC [3]
- Robust quadruped jumping via deep RL [4]
- Continuous jumping via relaxed centroidal QP [5]

This is the report for course ROB 590 Direct Study advised by Professer Yanran Ding.

- Jumping via learned acceleration residual [6]
- Learning aided centroidal QP locomotion [7]

C. Objectives

Enable the parkour capability of versatile jumping from running on a Unitree Go2 quadruped robot via RL assisted model-based control.

- Assumptions: Given desired landing states and contact schedule for jumping
- Planning: Heuristic acceleration planner by authoring vertical component of acceleration
- Tracking: Centroidal QP with learned residuals of reference

II. Reference Trajectory Generation

The transitions primarily investigated and optimized in this project are between jumping and running where the jumping and running gaits are chosen to be pronking and trotting respectively.

The reference trajectory for pronking and trotting are designed separately and parameterized by center-of-mass (COM) initial position (p_0), velocity (v_0) and target position (p_d) for each sequence of motion, while the transition between them are optimized individually. The main reason behind this design is that a complete reference trajectory can be automatically generated by random sampling given the number of sequence, since each reference motion can be connected with each other in arbitrary orders while continuity is ensured by optimizing transitions between them.

Note that the reference motion are designed in sagittal plane (x-z) of the robot for simplicity, but the robot are fully unconstrained in $\mathbb{R}^3$ during simulation.

A. Robot Modeling

Due to the nature of chosen gaits, the body of the quadruped robot are designed to remain upright during

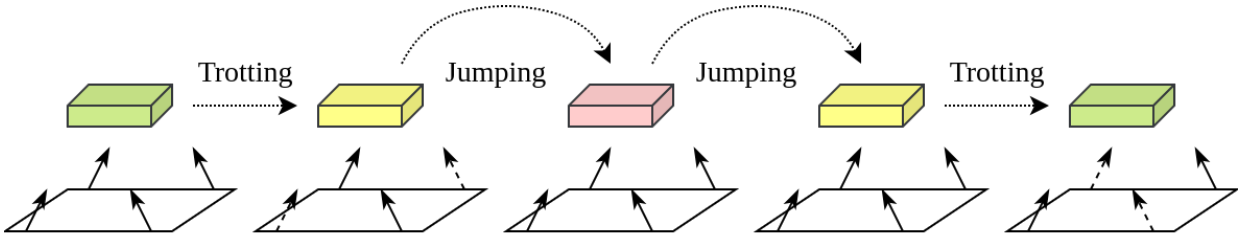

Fig. 2. A nominal parkour case in sagittal plane with controlled versatile jumping from trotting of a quadruped robot modeled as SRB

each gait, and the mass of four legs are small enough to be ignored compared to body. Therefore, the robot model is simplified as a single rigid-body (SRB, Figure 3). Consequently, the joint level dynamics and kinematics of the robot are not included in the optimization.

The state of the SRB model of the robot is given by

$$\mathbf{x} = [\mathbf{p}\ \boldsymbol{\Theta}\ \mathbf{v}\ {}^{\mathcal{B}}\boldsymbol{\omega}]^T \tag{1}$$

where $\mathbf{p} \in \mathbb{R}^3$ is the position of COM, $\boldsymbol{\Theta} \in \mathbb{R}^3$ is the orientation of the SRB represented by Euler angles, $\mathbf{v} \in \mathbb{R}^3$ is the velocity of the COM, and $\boldsymbol{\omega} \in \mathbb{R}^3$ is the angular velocity of the SRB represented in the body frame $\mathcal{B}$.

The state $\mathbf{x}$ is controlled via the net external wrench applied on the robot's COM via the reaction forces at the feet. The SRB dynamics is given by

$$\begin{bmatrix} m\ddot{\mathbf{p}} \\ {}^{\mathcal{B}}\mathbf{I}\dot{\boldsymbol{\omega}} \end{bmatrix} = \sum_{i=0}^{n} \begin{bmatrix} \mathbf{f}_i + \mathbf{g} \\ \mathbf{r}_i \times \mathbf{f}_i \end{bmatrix} \tag{2}$$

where $\mathbf{f}_i \in \mathbb{R}^3$ the force at foot i, $\mathbf{r}_i \in \mathbb{R}^3$ the vector point from the position of foot i to COM, and n the number of feet in contact.

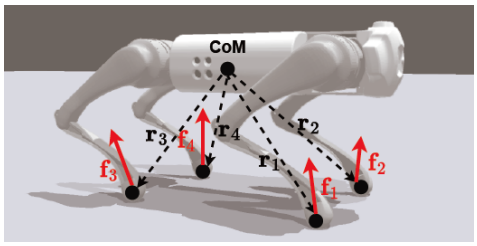

Fig. 3. SRB model representations on Go1 robot

B. Pronking Trajectory

The projectile motion equations are used to determine the state at liftoff that produces the desired CoM trajectory throughout flight, parameterized by the desired final position of the robot, e.g. jumping onto a stair or over a gap.

The desired position of the robot when it touches down $\mathbf{p}^{ref}(t_{TD}) \in \mathbb{R}^3$ is related to the reference trajectory of the robot throughout liftoff via

$$\mathbf{p}^{ref}(t_{TD}) = \mathbf{p}^{ref}(t_{LO}) + \mathbf{v}^{ref}(t_{LO})\Delta t_{FL} + \frac{1}{2}\mathbf{g}\Delta t_{FL}^2 \tag{3}$$

where $\mathbf{p}^{ref}(t_{LO}) \in \mathbb{R}^3$ and $\mathbf{v}^{ref}(t_{LO}) \in \mathbb{R}^3$ are the position and velocity of the robot when it lifts off the ground, Δt_{FL} is the duration of the flight phase, and $\mathbf{g} \in \mathbb{R}^3$ is the COM acceleration due to gravity.

Since the initial state of the robot is known, the reference position and velocity can be computed by integrating the reference acceleration $\mathbf{a}^{ref}(t) \in \mathbb{R}^3$ of the robot from zero to the time of liftoff t_{LO}.

The reference trajectories is generated by first manually authoring simple trajectories for the vertical component of the reference acceleration, which is similar to [2]. The heuristic used for vertical acceleration is the following.

$$a_z^{ref}(t) = (\beta + \frac{t}{t_{LO}}\gamma)\frac{1}{m}\sum_{i=1}^{n}\phi_i(t) f_{max,i} \tag{4}$$

where $\beta \in \mathbb{R}$ and $\gamma \in \mathbb{R}$ are empirical scaling parameters and $\phi_i(t) \in [0, 1]$ indicates whether the ith foot is in contact at time t.

Using the trajectory of $a_z^{ref}(t)$, the reference trajectories for $v_z^{ref}(t)$ and $p_z^{ref}(t)$ can be obtained via integration from the initial state. Consequently, the vertical component of (3) can be solved to obtain the duration of flight Δt_{FL}, given the landing height of $p_z^{ref}(t_{TD})$. Assuming constant acceleration in the forward and lateral directions, the forward and lateral components of (3) can be solved to obtain the complete $\mathbf{a}^{ref}(t)$ that result in the robot landing at $\mathbf{p}^{ref}(t_{TD})$. The generated trajectory is shown in Figure 5.

C. Trotting Trajectory

The major component of a trotting motion is in the forward direction, so a simple trapezoidal velocity profile

is designed in x axis while keeping zeros in other directions (Figure 7).

D. Transition Optimization

As shown in Figure 5, the heuristic jumping reference can not stabilize the robot after landing and prepare it for the next motion. To achieve this, the dynamic system during landing phase can be formulated as a point-mass model subjected to propulsion forces in the sagittal plane (Figure 4).

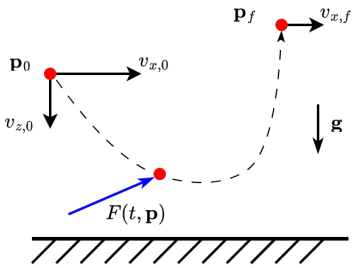

Fig. 4. Transition from landing state to a stabilized target state

- State: $\mathbf{s} = [\mathbf{p}\ \mathbf{v}]^T \in \mathbb{R}^4$
- Control: $\mathbf{a} = \sum_{i=0}^{4} F_i t^i := F(t,p)$
  where $\mathbf{p} = [F_0\ F_1\ F_2\ F_3]^T \in \mathbb{R}^{4\times 2}$
- Dynamics: $\dot{\mathbf{s}} = [\mathbf{v}\ F(t,\mathbf{p}) - \mathbf{g}]^T := f(t,\mathbf{s},\mathbf{p})$

The optimization can be formulated as a nonlinear programming problem (NLP) to find a set of coefficients $\mathbf{p}$ and time to stabilize t_f by minimizing the square some of parameterized ground reaction forces $F(t,\mathbf{p})$ while subjecting to boundary state constraints (5).

$$\begin{aligned} \min_{\mathbf{p},t_f} \int_0^{tf} & \|F(t,\mathbf{p})\|^2 dt \\ s.t. \quad & \mathbf{s}(0) = \mathbf{s}_0 \\ & \mathbf{s}(t_f) = \mathbf{s}_d \\ & \mathbf{s} \in \mathcal{S} \end{aligned} \tag{5}$$

where $\mathcal{S}$ is the boundary of robot's kinematic limits.

Since the integration upper bound t_f is also a optimization variable, time-transformation could be applied so that the system is integrated over a normalized time interval between $[0,1]$. Let $\tau = t/t_f$,

$$\begin{aligned} \min_{\mathbf{p},t_f} \int_0^{1} & \|F(t_f\tau,\mathbf{p})\|^2 t_f d\tau \\ s.t. \quad & \mathbf{s}(0) = \mathbf{s}_0 \\ & \mathbf{s}(1) = \mathbf{s}_d \\ & \mathbf{s} \in \mathcal{S} \end{aligned} \tag{6}$$

Therefore, this NLP can be solve via the single shooting method in CasADi [8].

E. Training Environment with Reference

The design of the centroidal policy is to mimic a MPC controller with receding horizon manner, which takes the input of a sequence of future reference states and output the residual of current reference state as a refinement, while minimizing the state tracking errors.

- Observation: $[\mathbf{x},\ \mathbf{r},\ \mathbf{x}_1^{ref}, \ldots,\ \mathbf{x}_{n_h}^{ref}]$
- Action: $[\Delta\mathbf{x}_1^{ref}, \Delta\mathbf{r}]$
- Reward: $\|\mathbf{x} - (\mathbf{x}_1^{ref} + \Delta\mathbf{x}_1^{ref})\|_{\mathbf{R}}$, e.t.c

where the subscript 1 denotes the index of current time step, n_h is the length of predictive horizon and $\mathbf{R}$ is a diagonal matrix represent the weight of each state dimension.

III. Experiments and Results

In this section, the implementation of simulation environment is discussed, and reference tracking curves are presented in Figures 5,6,7,8,9 for both single motion and multiple motions in sequence with or without optimized transitions.

A. Environment Setup

The experiment environment is setup in Isaac Gym [9], a high performance gpu-based physics simulation for robot learning. The codebase is implemented based on a fully paralleled (i.e. operate asynchronously in parallel) model-based quadruped controller framework from CAJun [5]. A novel reference trajectory generator is implemented to rollout a long reference by randomly sampling from a set of primitive motion sequences, and current reference states with a forward horizon are feed into the training environment according to the tracking progress w.r.t each robot in parallel.

B. Single Gait Tracking

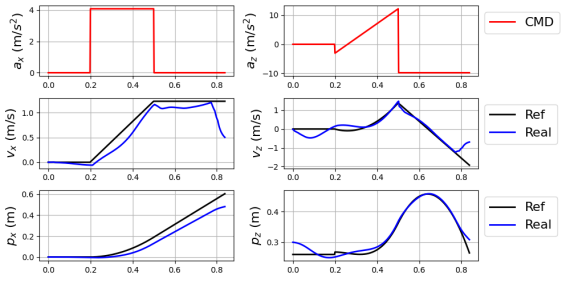

Fig. 5. Single pronking circle without optimized transition. The position tracking MSE error is 0.0026.

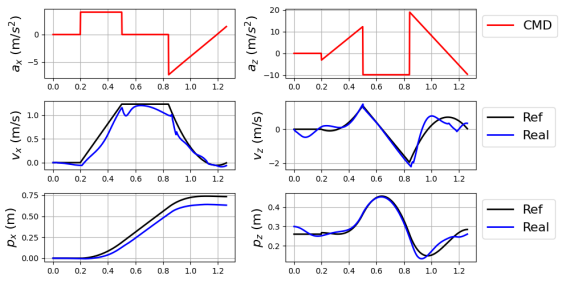

Fig. 6. Single pronking circle with optimized transition. The position tracking MSE error is 0.0017.

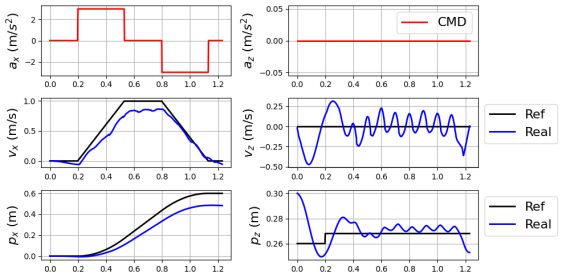

Fig. 7. Single trotting circle with optimized transition. The position tracking MSE error is 0.0032.

C. Continuous Jumping

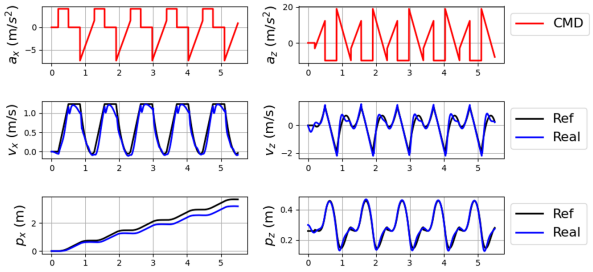

Fig. 8. Continuous jumping, video link

D. Transitions between Trotting and Jumping

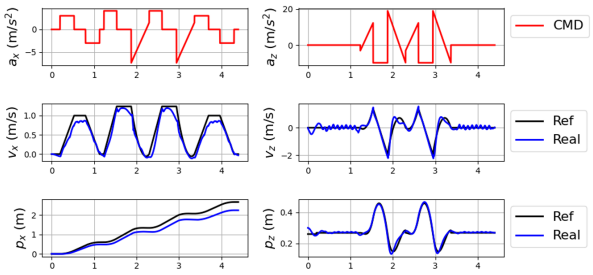

Fig. 9. Transitions between trotting and jumping, video link

E. Centroidal Policy Training

1) Training with reference: video link

- No good results have been obtained till the submission of this report
- Lessons learned: precise domain knowledge is needed to properly combine model-based method with RL, e.g. tuning hyperparameter from both sides

2) Training without reference: video link

- Trained surprisingly good without reference
- Just switched gait scheduler after each gait circle

IV. Conclusions

In conclusion, LocoJump offers a parallelizable and extendable framework for achieving controlled versatile jumping capabilities in legged robots. Through a combination of trajectory optimization, and RL aided model-based control methods, experimental results validate the effectiveness of the proposed approach, showcasing smooth and stable transitions between different motion modes and accurate tracking of reference trajectories.

References

[1] Q. Nguyen, M. J. Powell, B. Katz, J. Di Carlo, and S. Kim, "Optimized jumping on the mit cheetah 3 robot," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 7448–7454.

[2] M. Chignoli and S. Kim, "Online trajectory optimization for dynamic aerial motions of a quadruped robot," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 7693–7699.

[3] C. Nguyen, L. Bao, and Q. Nguyen, "Continuous jumping for legged robots on stepping stones via trajectory optimization and model predictive control," in *2022 IEEE 61st Conference on Decision and Control (CDC)*. IEEE, 2022, pp. 93–99.

[4] G. Bellegarda, C. Nguyen, and Q. Nguyen, "Robust quadruped jumping via deep reinforcement learning," *arXiv preprint arXiv:2011.07089*, 2020.

[5] Y. Yang, G. Shi, X. Meng, W. Yu, T. Zhang, J. Tan, and B. Boots, "Cajun: Continuous adaptive jumping using a learned centroidal controller," in *Conference on Robot Learning*. PMLR, 2023, pp. 2791–2806.

[6] Y. Yang, X. Meng, W. Yu, T. Zhang, J. Tan, and B. Boots, "Continuous versatile jumping using learned action residuals," in *Learning for Dynamics and Control Conference*. PMLR, 2023, pp. 770–782.

[7] Z. Xie, X. Da, B. Babich, A. Garg, and M. v. de Panne, "Glide: Generalizable quadrupedal locomotion in diverse environments with a centroidal model," in *International Workshop on the Algorithmic Foundations of Robotics*. Springer, 2022, pp. 523–539.

[8] J. A. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "Casadi: a software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, pp. 1–36, 2019.

[9] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa *et al.*, "Isaac gym: High performance gpu-based physics simulation for robot learning," *arXiv preprint arXiv:2108.10470*, 2021.