Dynamic Jumps of an Agile Bicycle through Reinforcement Learning

Zhuochen Yuan
Tsinghua Shenzhen International Graduate School
Tsinghua University
Shenzhen, China
yuanzc22@mails.tsinghua.edu.cn

Linqi Ye
School of Future Technology
Shanghai University
Shanghai, China
yelinqi@shu.edu.cn

Houde Liu
Tsinghua Shenzhen International Graduate School
Tsinghua University
Shenzhen, China
liu.hd@sz.tsinghua.edu.cn

Zhang Chen
Department of Automation
Tsinghua University
Beijing, China
cz_da@tsinghua.edu.cn

Bin Liang
Department of Automation
Tsinghua University
Beijing, China
bliang@tsinghua.edu.cn

***Abstract*—Controlling bicycles during dynamic maneuvers, such as jumps, poses significant challenges due to the intricate demands of balance and stability. This study utilizes the Proximal Policy Optimization (PPO) algorithm to control both ground navigation and dynamic jumps. Stability on flat terrain is maintained through steering adjustments, while jumps are facilitated by vertical thrust. The proposed approach is validated through simulation, showing high success rates in jump execution, along with improved balance and adaptability. These results underscore the potential of reinforcement learning-based control systems in managing complex dynamic movements and contribute to the development of more robust motion control strategies for bicycles on both structured and unstructured terrains. The attached video is given in https://linqi-ye.github.io/video/flybike.mp4.**

Keywords—Bicycle jumping, reinforcement learning, motion control, Proximal Policy Optimization

I. INTRODUCTION

Bicycles, as a common form of transportation, are characterized by agile motion, precise control, and short response times [1]. However, the demands of complex operating environments and the need for dynamic stability [3][4] suggest further exploration is required in controlling bicycle motion, particularly jumping maneuvers, which are key to agile bicycle handling. Traditional approaches to bicycle control rely on dynamic models and control theories [2][5][6][7][9][10], but these methods often perform well only in simulation or controlled settings and are generally limited to tasks like steering and path tracking. Consequently, they struggle with uncertainties and variability introduced by environmental factors in dynamic jumping scenarios and lack the adaptability required for unstructured environments [8][11].

Reinforcement learning (RL) methods have increasingly gained attention as alternatives to conventional control systems due to their demonstrated stability in both structured and unstructured environments [13][15]. RL is based on the principle of training an agent to learn optimal actions by observing a dynamic environment and receiving rewards from interactions with the system, equipping the agent with generalizability across complex tasks and disturbances [12][14][17]. Randløv and Alstrøm [14] divided a cycling task into two components: maintaining balance and reaching a target, using RL to improve the learning efficiency of complex control tasks. Jie Tan [8] modeled bicycles and riders as articulated rigid-body systems and applied the policy gradient method [15] offline to teach optimal riding policies, achieving highly maneuverable control. Tuyen [16] utilized the deep deterministic policy gradient (DDPG) [17] algorithm to control bicycle balance and steering, demonstrating DDPG's effectiveness in continuous control tasks. However, most of these studies only focused on balance control and trajectory tracking, with limited attention to dynamic jumping tasks. Zheng [18] applied a continuous-action RL method combined with the DDPG algorithm to train bicycles for ramp jumping, but this approach relied heavily on inertia and did not explore the role of the rider in the jumping process.

Recently, Boston Dynamics presents its design of a jumping bicycle as shown in Fig. 1. The bicycle has attached a 2-link structure on its frame to provide thrust and can jump to a significant height (tethered).

Fig. 1. The jumping bicycle from Boston Dynamics.

Inspired by Boston Dynamics' work, we present a novel agile bicycle system as shown in Fig. 2, which is a rigid-body system consisting of a bicycle and a rider, and use reinforcement learning to train dynamic jumping motion based on this system model. The rider, represented as a movable mass block, contributes to the platform jump through traction motion. Compared to single bicycle motion, the addition of the mass-block structure and the rapid vertical

changes during jumping introduce new challenges for system balance. The main contributions of this paper are as follows:

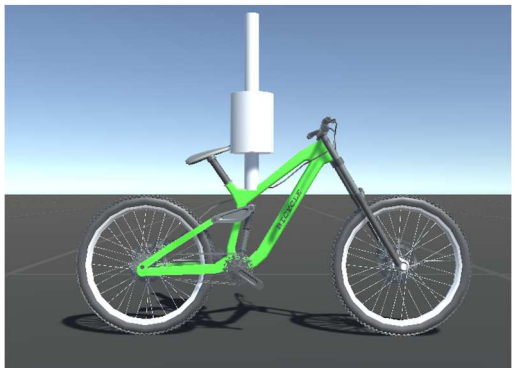

Fig. 2. The proposed agile bicycle as the research subject which consists of the bicycle body and the rider (mass-block structure).

- We propose a bicycle model for the dynamic jumping that incorporates a mass-block structure, enabling significant changes in vertical velocity through traction motion.
- We place additional emphasis on the bicycle's vertical velocity during training, alongside a reward term for maintaining balance, and apply Proximal Policy Optimization (PPO) [19] for reinforcement learning.
- The effectiveness and stability of the method are validated through step-jumping tests in a simulated environment, demonstrating the applicability of the reinforcement learning approach in dynamic settings requiring fast and accurate responses. This research lays the foundation for future studies on applying deep reinforcement learning to more complex motion control systems.

II. Methods

In this work, we apply the PPO algorithm to train the control policy for the system, see Fig. 3. The reinforcement learning framework is modeled as a Markov Decision Process (MDP), defined by the tuple (S, A, P_a, R_a) [18], where:

- S represents the set of all possible states,
- A represents the action space,
- $P(s_{t+1}|s_t, a_t)$ is the state transition function,
- $R(s_{t+1}|s_t, a_t)$ is the immediate reward function.

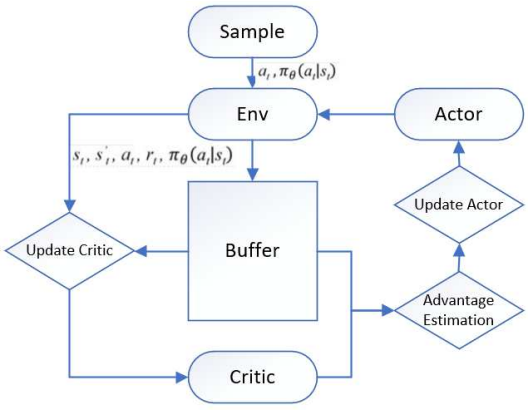

Fig. 3. Diagram of Reinforcement Learning Method.

At each time step t, the agent selects an action a_t from the policy based on the current state s_t. The MDP then computes the next state s_{t+1} and the corresponding reward r_t, providing feedback to the agent. The objective is to learn a policy $\pi(a_t|s_t)$ that maximizes the cumulative discounted rewards over time, which is calculated as:

$$J(\pi) = E_\pi(\sum_{t=0}^{\infty} \gamma^t r_t) \quad (1)$$

where γ is the discount factor, determining how much future rewards are valued.

Policy Network: To train the policy $\pi(a_t|s_t)$, we use an actor-critic architecture with the Proximal Policy Optimization algorithm (PPO) optimizing the policy. PPO is well-suited for continuous control tasks like hopping, as it maintains a balance between exploration and exploitation while ensuring stable and efficient learning. The actor network outputs the control action, while the critic network estimates the value function of each state, helping the agent evaluate the quality of its actions.

State Space: The state space consists exclusively of proprioceptive information related to the bicycle's state. This includes joint angles θ, joint velocities $\dot{\theta}$, the bicycle's body orientation (pitch θ_x, yaw θ_y, and roll θ_z), and the linear and angular velocities v, ω of the body. This rich set of proprioceptive data enables the agent to effectively perceive the bicycle's posture and dynamics, making informed decisions to maintain stability and control. However, it does not include any external environmental data such as terrain features or obstacles, making the model entirely reliant on internal feedback for decision making.

Action Space: The action space is continuous, controlling various parameters such as the position of mass-block structure, which determines the jumping velocity, and the steering angle, which adjust the orientation and stability of the bicycle, and the velocity of the back wheel, which determines the forward speed of the bicycle. These actions provide smooth control over the bicycle's motion, enabling it to adapt its internal configuration to maintain balance and achieve agile and robust jumping.

Reward Function: The reward function r_{total} is designed to incentivize stability, minimize energy consumption, and encourage forward progression. It consists of several components:

- Live Reward: A constant reward $r_{live} = 1$ is provided at each time step, which is needed for keeping the agent learning continuously.
- Orientation Reward: The orientation reward penalizes the bicycle for deviating from a stable posture, which is based on the body's pitch θ_x and roll θ_z:

$$r_{orix} = w_{ori1} \times min(|\theta_x|, |360° - \theta_x|) \quad (2.1)$$

$$r_{oriz} = w_{ori2} \times min(|\theta_z|, |360° - \theta_z|) \quad (2.2)$$

  where w_{ori1} and w_{ori2} correspond to the weights of the two part rewards. In this case, both are set to -0.05 in our work.
- Velocity Reward: The velocity reward measures the change in the vertical direction of the smart body and increases if the speed is greater, which ensures that the bicycle can take off with sufficient speed. It is calculated as follows:

$$r_{vel} = w_h \times |v_y| \quad (3)$$

where v_y is the vertical velocity of the bicycle's body and w_h is the weight of the velocity reward which is set to 1 here.

Overall, the comprehensive reward function ensures that the bicycle learns to maintain balance, move efficiently, and adapt its internal state based on the feedback, which is the sum of the components mentioned above:

$$r_{total} = r_{orix} + r_{oriz} + r_{vel} \quad (4)$$

Policy Optimization Process: PPO optimizes the policy by repeatedly interacting with the environment, updating the policy network to maximize the expected cumulative reward. Furthermore, we set termination conditions to end the learning process early in case of extreme situations: If the body's rotation around the Z-axis exceeds 45°, it means that the intelligence may have tipped over on its side, and the round ends with a large negative bonus:

$$min(|\theta_z|, |360° - \theta_z|) > 45° \quad (5)$$

When the termination criteria are met, the episode resets to its initial state, and a new training episode begins.

III. Experiments

A. Simulation Setup

The PPO-based reinforcement learning approach is implemented in Unity 3D using the ML-Agents toolkit. Unity 3D offers a highly customizable simulation environment that enables precise modeling of bicycle dynamics and terrain interactions, while the ML-Agents toolkit provides an effective framework for training and deploying reinforcement learning algorithms in complex 3D environments.

In our simulation, the bicycle is modeled as a rigid-body system, and the rider is represented by a mass-block structure. The environment consists of flat terrain for ground navigation and obstacles for dynamic jumps. The bicycle is tasked with navigating the terrain and performing jumps, while key variables such as steering angle, forward angle, vertical velocity, and forward velocity are recorded for analysis.

During the simulation, we first test the bicycle's dynamic jumping ability by leaping onto the step-like terrain. This is followed by descending two steps in rapid succession immediately after landing, to evaluate the bicycle's ability to regain balance after the aerial phase. Detailed terrain is shown in Fig. 4.

The training process involves continuously updating the control strategy as the agent interacts with the environment, learning from the feedback provided by the reward function. The simulation runs in real time, with each episode ending either after a successful dynamic jump or when the bicycle becomes unstable and the environment is reset.

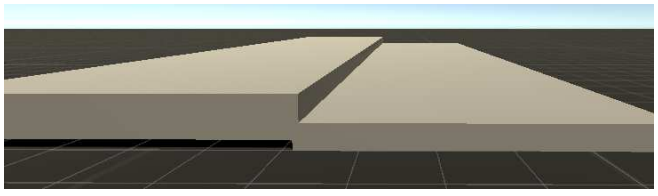

Fig. 4. The detailed step-like terrain which is a platform consist of two stages.

B. Simulation Results

The bicycle is trained for $800k$ steps with the PPO algorithm. The reward curve is shown in Fig. 5. We then test the obtained control policy on the step-like terrain. The results can be found in the attached video: https://linqi-ye.github.io/video/flybike.mp4. The snapshot of the bicycle jumping process is shown in Fig. 6.

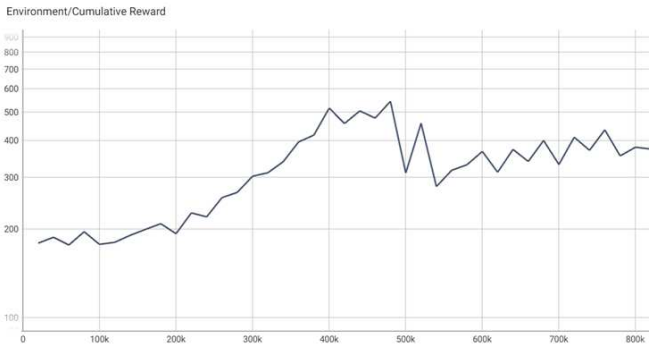

Fig. 5. The reward curve during training episodes.

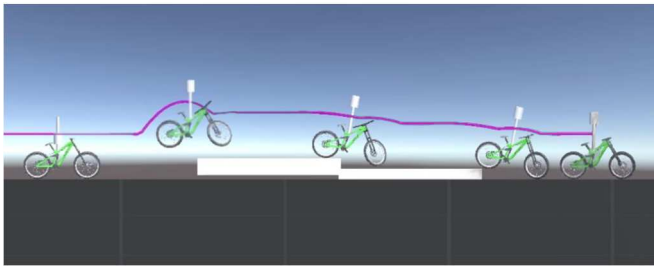

Fig. 6. The bicycle jumping snapshots.

Jumping Height. Fig. 7 illustrates the variation in the bicycle's body height as it traverses the designed terrain. The bicycle begins in a stable state, ascends to a two-step-high platform at the third second, and then continuously descends two steps before reaching the ground and stabilizing around the eighth second. The figure indicates that the bicycle successfully executes stabilization maneuvers, including both ascent and descent, demonstrating its ability to adapt to altitude changes while maintaining stability. As shown in Fig. 7, the maximum jumping height of the bicycle during this process is $1.13\ m$.

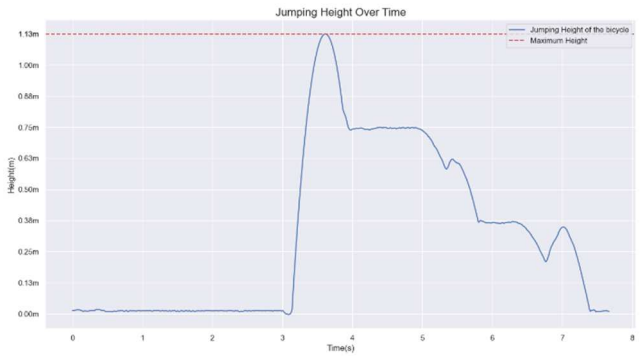

Fig. 7. The height trajectory of the bicycle.

Steering and Forward Angles. The second set of data focuses on the steering and forward angles over time, as illustrated in Fig. 8. During the jump phase, the steering angle oscillated between approximately $-40°$ and $40°$, with several fluctuations occurring particularly after 4 seconds. This indicates that significant adjustments are made to the control

strategy to maintain balance during the landing and recovery phases. Meanwhile, the forward angle varies between $-20°$ and $10°$, with a notable drop around 4 seconds into the jump. These rapid changes indicate that the control strategy adjust for the bicycle's forward tilt during the air phase, ensuring a smooth landing without excessive forward or backward lean.

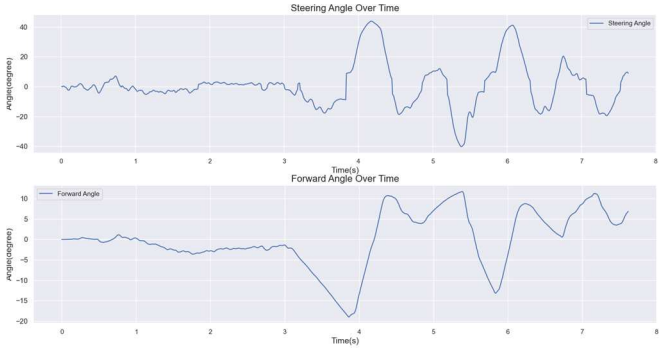

Fig. 8. Steering and forward angles over time.

Vertical and Forward Velocities. The third set of data focuses on vertical and forward velocity. As illustrated in Fig. 9, vertical velocity rises sharply during the jump and then drops rapidly. Forward velocity remains around $3\,m/s$ throughout, with a slight decrease during the jump to compensate for the change in roll angle. Upon leaving the step, each peak in vertical velocity corresponds to the landing of either the front or rear wheel. Forward velocity begins to increase as the front wheel leaves the platform and returns to $3m/s$ once both wheels have landed. These patterns indicate the effectiveness of the control strategy in enabling the bicycle to quickly regain balance after landing.

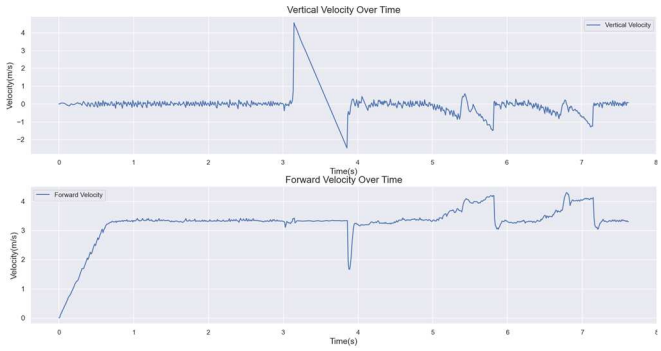

Fig. 9. Vertical and forward velocity over time.

Analysis Between Angles and Velocities. The final set of data displays both steering and forward angles along with the corresponding vertical and forward velocities. The vertical velocity exhibits a sharp drop, indicating the point where the bicycle reaches its peak height and begins descending. This behavior occurs concurrently with significant changes in both steering and forward angles, as the control policy adjusts to prepare for landing. The forward velocity remains relatively stable, with small fluctuations throughout the episode. Notably, the forward velocity decreases slightly during the airborne phase, before regaining momentum as the bicycle touches down.

The relation between the angles and velocities highlights the intricate balance control achieved by the PPO-based control strategy. The bicycle maintains forward velocity between 2 to 4 m/s, with vertical velocity ranging from -2 to $2\,m/s$, indicating that the control strategy effectively manages momentum and velocities to achieve an agile jump.

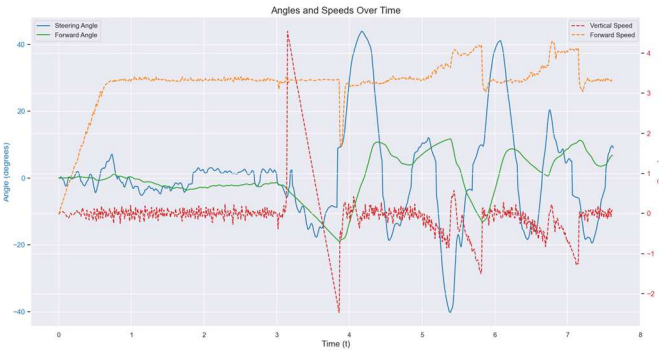

Fig. 10. Angles and velocities over time.

These results confirm that the control strategy effectively manages both vertical jumps and forward velocities, enabling the bicycle to maintain forward velocity while minimizing the risk of instability during the aerial phase. The descent and the recovery after the jumping demonstrate the stability of the PPO-based control strategy in handling highly dynamic jumping motions.

IV. Conclusion

We make use of a PPO-based reinforcement learning approach to control dynamic jumps of the bicycle-rider system modeled with a mass-block structure. Simulation results demonstrate the method's ability to manage smooth ground motion and complex jumping maneuvers, while also highlighting its capacity to adjust key variables such as steering angle, forward angle and velocity to maintain stability and achieve landings during dynamic jumps. The interaction between vertical and forward velocity underscores the effectiveness of reinforcement learning strategies in managing vertical ascent and descent while ensuring continuous forward progress.

The experiment shows that the PPO-based control strategy successfully balances the bicycle during jumping, makes real-time adjustments to maintain stability, and effectively recovers after landing. These findings confirm the potential of reinforcement learning in handling complex dynamic tasks like bicycle jumping, which require precise control and adaptation.

While the current PPO-based reinforcement learning approach has demonstrated effects in controlling dynamic jumps in a simplified bicycle-rider system, there are several aspects remaining further improvement, particularly when addressing more complex terrains and optimizing velocity. The following areas are recommended for future investigation:

- **Complex Terrain Handling.** The current model operates in a controlled environment. Future work should focus on incorporating more diverse terrains, such as uneven surfaces, slopes, and obstacles, to enhance the bicycle's ability to travel in various environments more smoothly and efficiently.
- **Optimizing Speed and Direction.** To enable faster traversal, further refinement is needed in controlling both direction and velocity to ensure real-time adjustments in acceleration, deceleration, and steering, while maintaining stability during jumps.

- **Timing Optimization For Jumps.** Optimizing jump timing can improve obstacle avoidance and help maintain momentum. Enhancing the bicycle's ability to predict and plan jumps in advance will lead to smoother and more efficient maneuvers.

REFERENCES

[1] Åström, K. J., Klein, R. E., Lennartsson, A. Bicycle dynamics and control: adapted bicycles for education and research[J]. IEEE Control Systems Magazine, 2005, 25(4): 26-47.

[2] Meijaard J P, Papadopoulos J M, Ruina A, et al. Linearized dynamics equations for the balance and steer of a bicycle: a benchmark and review[J]. Proceedings of the Royal society A: mathematical, physical and engineering sciences, 2007, 463(2084): 1955-1982.

[3] Limebeer D J N, Sharp R S. Bicycles, motorcycles, and models[J]. IEEE Control Systems Magazine, 2006, 26(5): 34-61.

[4] Schwab A L, Meijaard J P, Kooijman J D G. Some recent developments in bicycle dynamics[C]//Proceedings of the 12th World Congress in Mechanism and Machine Science. Moscow, Russia: Russian Academy of Sciences, 2007: 1-6.

[5] Keo L, Yamakita M. Controlling balancer and steering for bicycle stabilization[C]//2009 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, 2009: 4541-4546.

[6] Consolini L, Maggiore M. Control of a bicycle using virtual holonomic constraints[J]. Automatica, 2013, 49(9): 2831-2839.

[7] Hess R, Moore J K, Hubbard M. Modeling the manually controlled bicycle[J]. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, 2012, 42(3): 545-557.

[8] Tan J, Gu Y, Liu C K, et al. Learning bicycle stunts[J]. ACM Transactions on Graphics (TOG), 2014, 33(4): 1-12.

[9] Seekhao P, Tungpimolrut K, Parnichkun M. Development and control of a bicycle robot based on steering and pendulum balancing[J]. Mechatronics, 2020, 69: 102386.

[10] Persson N, Ekström M C, Ekström M, et al. Trajectory tracking and stabilisation of a riderless bicycle[C]//2021 ieee international intelligent transportation systems conference (itsc). IEEE, 2021: 1859-1866.

[11] He K, Deng Y, Wang G, et al. Learning-based trajectory tracking and balance control for bicycle robots with a pendulum: A Gaussian process approach[J]. IEEE/ASME Transactions on Mechatronics, 2022, 27(2): 634-644.

[12] Sutton R S, Barto A G. Reinforcement learning: An introduction[M]. MIT press, 2018.

[13] Zai A, Brown B. Deep reinforcement learning in action[M]. Manning Publications, 2020.

[14] Randløv J, Alstrøm P. Learning to Drive a Bicycle Using Reinforcement Learning and Shaping[C]//ICML. 1998, 98: 463-471.

[15] Peters J, Schaal S. Reinforcement learning of motor skills with policy gradients[J]. Neural networks, 2008, 21(4): 682-697.

[16] Chung T C. Controlling bicycle using deep deterministic policy gradient algorithm[C]//2017 14th international conference on ubiquitous robots and ambient intelligence (URAI). IEEE, 2017: 413-417.

[17] Lillicrap T P. Continuous control with deep reinforcement learning[J]. arXiv preprint arXiv:1509.02971, 2015.

[18] Zheng Q, Wang D, Chen Z, et al. Continuous reinforcement learning based ramp jump control for single-track two-wheeled robots[J]. Transactions of the Institute of Measurement and Control, 2022, 44(4): 892-904.

[19] Schulman J, Wolski F, Dhariwal P, et al. Proximal policy optimization algorithms[J]. arXiv preprint arXiv:1707.06347, 2017.