First part report

1. Code Explanation

The project utilizes the Isaac Gym environment to simulate the movement and behavior of the Go2 robot. Based on the framework provided by the Legged Gym library, secondary development is carried out. The core of the entire framework consists of two classes:

Core Class: The go2robot class is the core of the entire code, located in the legged_gym/envs/go2/go2_robot.py folder. It inherits from the pre-defined parent class LeggedRobot in the base folder. This class covers five aspects of reinforcement learning, including environment creation, agent addition, reward design, exploration mechanism, and training promotion (already divided in the LeggedRobot class). In the first part, we did not adjust the content of the parent class LeggedRobot, but we will override the methods of the LeggedRobot class in subsequent parts two and three.

Configuration Class: The Go2RoughCfg class is also a core part of the framework, located in the legged_gym/envs/go2/go2_config.py folder, inheriting from the parent class LeggedRobotCfg in the base folder. It is a class included in the go2robot class. Go2RoughCfg contains the parameters required for reinforcement learning training and operation. In the first part, we adjusted some properties of the Go2RoughCfg class, mainly including the following two parts:

- **Robot Parameters**: Includes init_state, URDF asset, etc.
- **Control Parameters**: Adjust PD control parameters, use position control, and calculate torque output for the simulator's motors using the PD controller.

Test: We wrote a test script to test our class, located in the legged_gym/tests/test_env.py file. Normally, some parameters need to be passed, but we have configured these parameters with default values, so no parameters are needed. By entering python test_env.py, it should run directly. Using the dummy_policy() function, we obtain an action of type pytorch.Tensor with dimensions of 1x12, corresponding to the target positions of the twelve joints of the quadruped robot. The range is set between (-1, 1) to avoid excessively large inputs. When running the script, you will notice the robot shakes in the environment because we used a random strategy to generate random actions. If all elements of the action are set to 0, you will notice that the robot remains stationary in the environment. go2robot contains the attribute max_episode_length, and when this value exceeds the set threshold, the reset() function will be called to reset. Therefore, you will see that the robot in the simulator resets after a while.

Figure 1 and 2 shows the initial Issac Gym environment where the quadruped robot is going to be tested.

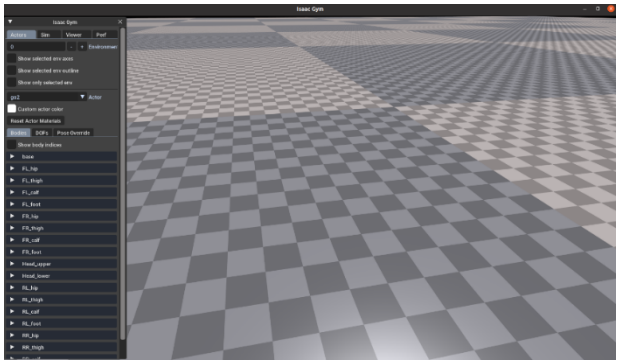

Figure 1: Flat terrain

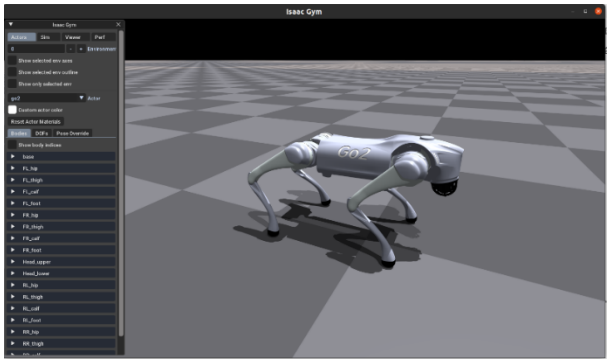

Figure 2: Quadruped robot

2. Existing Codes/Libraries

- **Isaac Gym**: NVIDIA's Isaac Gym provides a powerful environment to simulate robotic systems. The library offers an integrated physics simulation tool that can handle multiple robots in real-time. This was used extensively for the environment setup, including terrain generation and real-time physics computation.
    - **Physics Engine**: Isaac Gym's GPU-based physics engine allows for the high-performance simulation of multiple robots or complex movements. The project leverages the engine for quick iterations and experiments with different control strategies.
    - **Assets and Pre-configured Models**: Pre-configured robotic assets were customized to match the Go2 model for smooth integration with the simulation environment.
- **Legged Gym:** Legged Gym is a physics-based reinforcement learning environment designed specifically for training legged robots such as quadrupeds. It integrates with NVIDIA Isaac Gym and GPU-accelerated physics engines, facilitating efficient learning of robotic control strategies. Legged Gym provides various standard quadrupedal robot models and terrains supporting the development of movement strategies in complex environments. By leveraging large-scale parallel training in simulated environments, developers can accelerate the learning and optimization of control strategies. We need to use its built-in classes and functions (e.g., Legged_Robot and Legged_Robot_Cfg classes) to improve our development efficiency.
- **RSL_RL:** This release contains an optimized version of the Proximal Policy Optimization (PPO) algorithm tailored for GPU-accelerated simulators. It is a dependency of Legged Gym. Although this library has not been used in the first phase of the project, it will be utilized in later stages.
- **Pytorch:** PyTorch is an open-source deep learning framework known for its excellent flexibility and ease of use. It is a dependency of other libraries. In the first part, we used PyTorch's built-in data structure Tensor. In subsequent phases, we will need it to build the network used for training.

3. Reflections and Lessons Learned

- During the project proposal phase, we were unaware of the existence of such a comprehensive open-source framework as Legged Gym. It includes almost all components needed for the reinforcement learning simulation training of quadruped robots. To ensure workload, we will try using newer reinforcement learning algorithms for training and replace Legged Gym's existing reinforcement learning algorithms in subsequent stages. We will also experiment with various terrains to ensure that our training results have better generalizability. These were not mentioned in our previous proposal.