Intelligent Systems and Robotics 2024: Homework 2

In this assignment, you will learn and practice using the powerful open-source codebase, Humanoid-Gym, to develop a reinforcement learning policy for a humanoid robot.

Your main task is to implement RMA: Rapid Motor Adaptation for Legged Robots within the Humanoid-Gym framework. RMA is a method that empowers robots to rapidly adapt their locomotion to various terrains, obstacles, or conditions, thereby improving stability, efficiency, and overall performance across diverse environments. Completing the baseline will earn you 40% of the homework marks.

To begin, follow the step-by-step instructions for installing and setting up the development environments, as well as for configuring Humanoid-Gym. Next, train a baseline policy. Once your setup is complete, implement the RMA algorithm using the provided code framework. This section will account for the remaining 60% of your assignment grade.

Please note:

- Before beginning this task, please be aware that you may need to prepare **GPU resources** to complete this assignment. A minimum of 6GB GPU memory is required. If your GPU has only 4GB memory, we provide a setting for 4GB GPU (see README.md). Your grade will be based on the quality of your implementation, so you are eligible for the full score regardless of the task you choose.
- You can complete this assignment as long as you follow the hints we give, however, we still encourage you to read the RMA paper.
- During implementation, it is beneficial to refer to the corresponding functions in the base class, as most implementations involve making only minor modifications to these functions.
- **Collaboration policy.** Discussions with others are encouraged. However, you should implement the idea on your own.
- You will run at least 3 learning programs, which may take in total **5 hours on 3060, or 2.5 hours on 3090**. Please take care of the ddl.
- If you have any questions about this assignment, please fell free to contact TA (Chengming Shi) via WeChat or email scm23@mails.tsinghua.edu.cn.

Exercise 0: Install the codebase (20%)

Please refer to README.md for installation. We provide installation instructions on both Ubuntu and WSL.

ATTENTION! For any issues, please refer to the Troubleshooting section in README.md at first.

After Installation, you can run `play_demo.py`. It will load a trained model in `logs/exported/` and generate a video of humanoid walking.

```
cd humanoid
python scripts/play_demo.py
```

The video is placed in `videos/example/`. For the video you can get 20 points.

Exercise 1: Run baseline algorithm (20%)

1. Train PPO on the plane-walking task

```
cd humanoid
python scripts/train.py --headless --use_wandb --task=humanoid_ppo --run_name=baseline
```

Refer to Wandb if you want to use wandb. You can remove the flag `--use_wandb` to avoid wandb sync and use tensorboard instead. Please note your **run name shall not contain the underline symbol** '_' since we use it to split run names.

The default setting will use about 4GB GPU memory. If your GPU has only 4GB memory, you can refer to the setting for 4GB GPU memory in README.md, which uses about 3.5GB. The checkpoints are placed in `logs/XBot_ppo/`. For more details about the arguments, please refer to READEME.md or train.py.

2. Test your baseline policy

```
python scripts/play.py --headless --task=humanoid_ppo --run_name=baseline
```

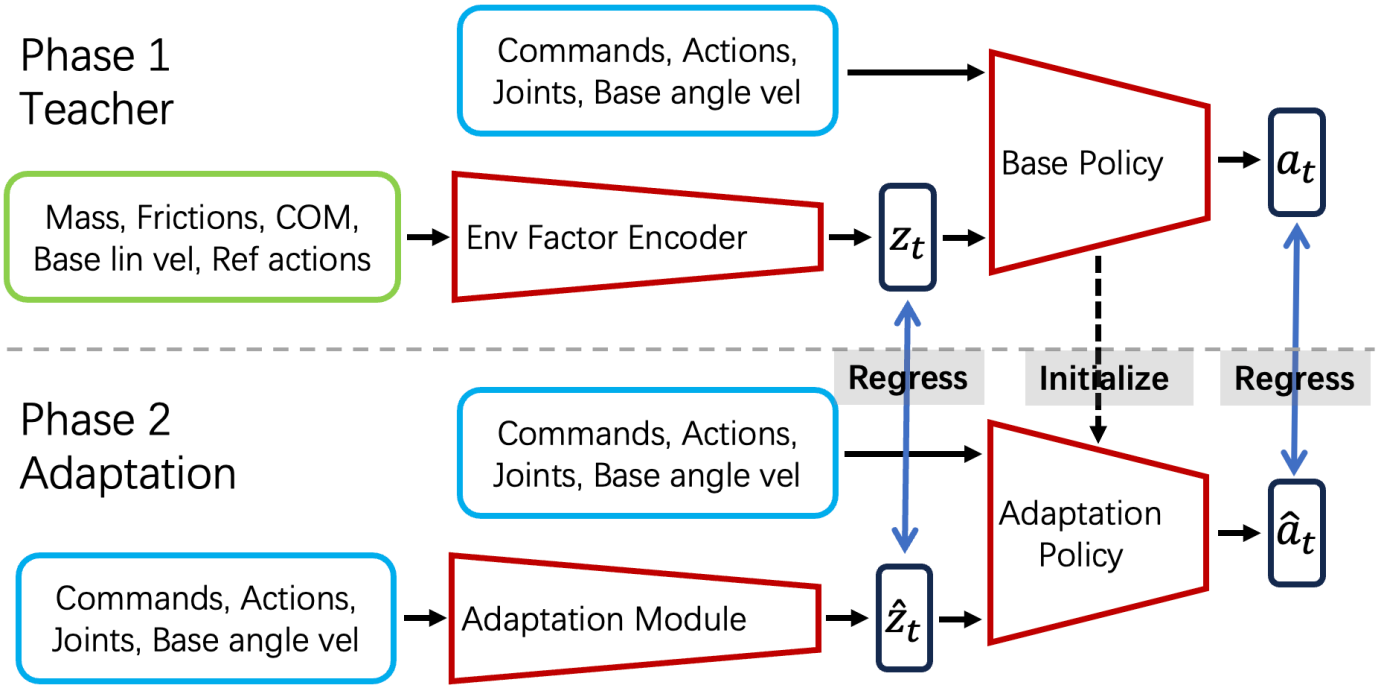
Please note the run name should match your training run name. The script will generate a video of your trained model. The video is placed in `videos/XBot_ppo/`. For more details about the arguments, please refer to play.py.

Exercise 2: Implement RMA Teacher (30%)

RMA is a teacher-student learning approach, illustrated in the following figure. In the first phase, the teacher takes in some unobtainable and implicit observations, such as friction, terrain height, and motor strength, to learn an Env Factor Encoder. The latent output of the encoder is then used as an additional input for the policy.

In the second phase, the student uses the obtainable observations to train an Adaption Module. The aim of RMA is to facilitate the output of the Adaptation Module and the latent output of the Env Factor Encoder to be as similar as possible by applying teacher-student learning.

You can refer to the original paper RMA: Rapid Motor Adaptation for Legged Robots. In this project, we modify the framework to suit the humanoid robot. Specifically, we make the policy module trainable and add action differences into the loss.



*Unobtainable data in green, *Obtainable data in blue, *Trainable Modules in red

1. Implement the phase 1: teacher

Please implement all the #TODO blocks in

- envs/custom/humanoid_rma.py
- envs/custom/humanoid_rma_config.py
- algo/rma/rma.py
- algo/rma/actor_encoder.py

2. Run RMA Teacher on the stair-walking task.

```
python scripts/train.py --headless --use_wandb --task=humanoid_rma --run_name=rma-teacher
```

Please note your run name shall not contain the underline symbol '_' since we use it to split run names. The default setting will use about 5GB GPU memory. If your GPU has only 4GB memory, you can turn to plane task (refer to setting for 4GB GPU memory in README.md).

3. Test your RMA Teacher

```
python scripts/play.py --headless --task=humanoid_rma --run_name=rma-teacher
```

You should test on 2 tasks: 1cm upstair and 3cm downstair. If your training task is plane-walking you only need to test on plane.

Exercise 3: Implement RMA Adaptation (30%)

1. Implement the phase 2: adaptation

Please implement all the #TODO blocks in

- algo/rma/rma_adaptation.py

2. Run RMA Adaptation on the stair-walking task

```
python scripts/train.py --headless --use_wandb --task=humanoid_adaptation --
teacher_run_name=rma-teacher --run_name=rma-adaptation
```

The `teacher_run_name` should match your RMA Teacher `run_name`. Also note your run name shall not contain the underline symbol '_'. The GPU memory usage is basically the same as RMA Teacher training, and you can turn to plane task if your GPU has only 4GB memory.

3. Test your RMA Adaptation

```
python scripts/play.py --headless --task=humanoid_adaptation --run_name=rma-adaptation
```

You should test on 2 tasks: 1cm upstair and 3cm downstair. If your training task is plane-walking you only need to test on plane.

Optional Exercises

Please note theses exercises have **no extra points**.

1. Run baseline on stair tasks

You can run PPO on the stair task can compare the result with RMA. To do so you should change settings in `envs/custom/humanoid_config.py`.

- `XBotLCfg.terrain.mesh_type='trimesh'`
- `XBotLCfg.commands.ranges`, same with `XBotLRMACfg.commands.ranges`

If you do it, you can submit your result videos.

2. Run RMA to converge

In previous trainings the learning iteration number is 500, which is insufficient to converge, so your model may be not able to walk on high stairs. If you have enough GPU resource you can train it longer by loading the half-trained model and setting a higher iteration number. If you do it, you can submit your result videos. Please write down the total training iteration number in the file name.

Submit Your Homework

You need to pack the following files/dirs into a zip file:

- `humanoid/`: The directory should include `envs/`, `algo/`, and all files in these subdirs.
- `demo.mp4`: The video from `play_demo.py`
- `ppo.mp4`: The video of the baseline.
- The video of RMA Teacher. You should submit either of these:
  - `rma-teacher-stairup.mp4` and `rma-teacher-stairdown.mp4`

    - `rma-teacher-plane.mp4`
- The video of RMA Adaptation. You should submit either of these:
    - `rma-adaptation-stairup.mp4` and `rma-adaptation-stairdown.mp4`
    - `rma-adaptation-plane.mp4`
- `optional/`: A subdir containing videos of the optional exercises. No fixed file names, just make it clear.

Please name your zip file with your student ID and Chinese name, e.g. `2023311763-施铖铭.zip`, and submit it through Web Learning.