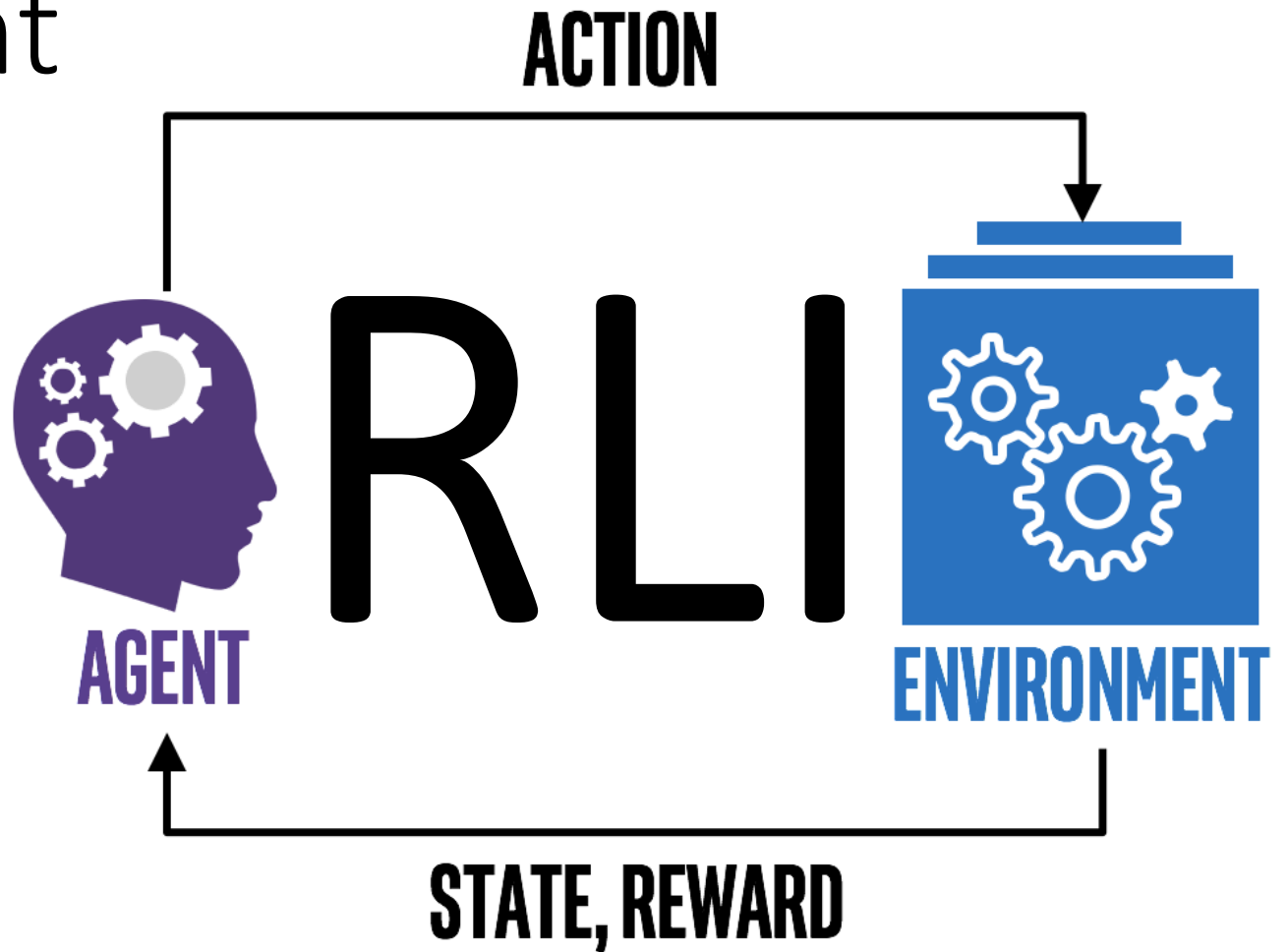
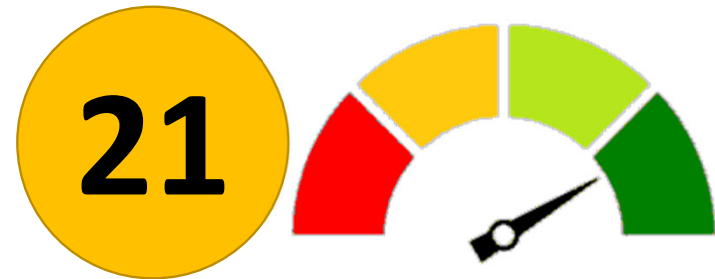


Reinforcement Learning Introduction



Rocket Landing

On its fifth attempt, SpaceX on Friday April 8th 2016 finally succeeded in landing its Falcon 9 rocket on a barge floating in the sea.



Reinforcement Learning Introduction

Assignment 21.00

Rocket landing upgrade

Date: Mar/21 2025

Due Date: Apr/10 2025 [SESSION 27]

Ongoing grading (maximum)



50 Individual Assignment [mini-groups]

Assignment Description

Overview

Using as an starting point the code presented in the session for the “Rocket Landing” practice, and included in the files attached, this assignment consists of 2 parts/questions:

Assignment Part.01 → Explain and document your understanding of the Actor-Critic algorithm employed. (file “policy.py”) [33% points] and tweak it to account for the “entropy” regularization term [17% points]

- Comment specially the **update_ac()** method and your interpretation of the loss functions utilized and their relationship with the theoretical background of such methods
- In the code provide you will notice the ominous “ unstitched code  commented sections. Try to properly adapt them so that your algorithm includes this “entropy” variation.
- Do your own research (and document it) to characterize and understand this type of “modality” within Actor-Critic.
- Unfortunately, though it seems quite straightforward, the “re-stitching” of this code is not easy at all. You will have to handle the conformance of the tensor shapes for “probs” (remember they are calculated in a different method **get_action()**, etc...) So, this requires some skills with tensors and pytorch and a good understanding of the general “training” data flow. Do not despair if finally you get stuck here. Just document your theoretical findings and/or ideas explored and leave the code “commented out” so everything is still “executable”... [that will be good enough]

Assignment Description

Overview

Assignment Part.02 → Create a proper “rocket_env.py” that implements an OpenAI-gymnasium-compatible version of the “Rocket.py” environment [10% points]

- Analyze the possible approaches for such conversion (refactoring, wrapping, subclassing) and describe their pros/cons (in terms of simplicity, maintainability and/or clarity)
- NOTES:
 - the “rocket_env.py” is just a dummy/stub approach, not functional
 - this step is additionally a necessary requisite for Part.03 too

Assignment Description

Overview

Assignment Part.03 → Migrate your training process to another RL algorithm (for example: it could be the same A2C, SAC or PPO, etc) using the implementation provided in the “stable-baselines-3” framework [40% points]

- Do the training for the “starship” rocket [task = “landing”]
- NOTE: “rocket.py” → class Rocket(object): →

```
def __init__( self, max_steps, task='hover', rocket_type='falcon',  
              viewport_h=768, path_to_bg_img=None ):
```
- Ergo: instantiate class with rocket_type=‘starship’ [if you want to be very “professional” this could be implemented with a new parameter in “rocket_env.py” → class RocketEnv() → __init__(), but for me it’s good enough if you just hardcode it]
- Document and chart the results obtained in the training process (do about 20k episodes)
- Explain or comment briefly which ideas (rewards/loss) you could have for improving additionally the average “precision” with respect to the exact aiming of the landing target point

Assignment Description

Overview

For each part/question you should elaborate a self-contained directory-zip file with your “.py” and “.ipynb” files (derived from the documents delivered with the assignment), providing (when required) a clear concise explanation of the conceptual approach, the diagrams or models representing the problem, comments for the code used and the explanation of the solution

SUBMIT YOUR WORK BY ZIPPING TOGETHER THE FINAL CODE (COMPLETED WITH YOUR ANSWERS) IN A SINGLE SUBMISSION FILE NAMED:

“RLI_21_00 – mini-group number XY.zip” (see the attached .txt for the mini-groups)

Administrative and additional notes

- Find attached (in the zip file) the configuration of the mini-groups for the assignment:

→ RLI_2024-25 (A or B) - ASSIGNMENT 2 RLI_21_00 TEAMS.txt

Only submit **ONE zip file per group**. Choose ONE member of each group for that submission. In case you send more than one, ensure that all the submissions by the different members are identical, as the system will choose just ONLY one of them (randomly) for the review and evaluation of the whole group.

- As usual, **the code should run without warnings or errors**. All additional required files should be included in the ZIP file, and if you were using any “novel” additional external library for your work, clearly indicate a brief description of its purpose, the version used and the “!pip install <package>” required for its installation