IL Code Base

IN CASE YOU FIND ANY BUGS OR ABNORMAL BEHAVIOR THEN PLEASE LET ME KNOW.

This repository is based on the original Gym-Pybullet -Drones repository and will act as the starter code for our project. We will make use of the `learning` package to implement our project.

Collaboration

Before we begin, we can setup a protocol for unifying the code we will be implementing. The repository can be divided into 3 seperate branches as per their roles which are as follows-

Karush(kar)	Mohammad(mhd)	Reza(rz)
IL-IL	RL-RL	IL-RL

Each contributor would only commit to his own branch, for instance Karush will commit to `kar` branch. This will ensure that our code stays clean and does not present abnormal behavior. Towards the end, we will merge these branches into a single code base.

Structure

The main contents of our project reside in the `learning` folder. Following is the directory structure-

- `algos`- The folder will consist of all our algorithms implemented as part of the IL and RL framework. As a starting point, I have added `TD3` and `DDPG`. These implementations make use of a predefined class structure and functions which should be followed while writing our program. This would allow smoother integration. Any additional functions should be added to the `utils.py` file.
- `configs`- The folder consists of all arguments as configurations. These are hyperparameters and tuning values which may need to be changed over time. You can use them in a similar fashion as the command line arguments. The default configs are in `defaults.yaml` file. Corresponding to each program, we wil have a config file. For instance, Karush will make a config file `il_il.yaml` consisting of arguments for his experiments.
- `results`- Results will automatically be stored here once the program completes execution.

Another thing, the base models currently make use of kinematic state inputs (vectors) and do not learn from images. The models output raw actions (torques) rather than depth segmentation/viewpoints. I am looking into incorporating this in our code. You coudl start as well.

Requirements and Installation

The repo was written using *Python 3.7* and tested on *Ubuntu 18.04*

On *Ubuntu*

Major dependencies are `gym`, `pybullet`, `stable-baselines3`, and `rllib`

```
pip3 install --upgrade numpy Pillow matplotlib cycler
pip3 install --upgrade gym pybullet stable_baselines3 'ray[rllib]'
```

Video recording requires to have `ffmpeg` installed, on *macOS*

```
$ brew install ffmpeg
```

On *Ubuntu*

```
$ sudo apt install ffmpeg
```

The repo is structured as a Gym Environment and can be installed with `pip install --editable`

```
$ git clone https://github.com/utiasDSL/gym-pybullet-drones.git
$ cd gym-pybullet-drones/
$ pip3 install -e .
```

Usage

Our project is implemented using the `singleagent.py` file. This is a common file which will run all our implementations upon integration.

To run an `SAC` agent on the `takeoff` task use the following-

```
python singleagent.py --configs configs/defaults.yaml takeoff
```

This will train the agent for `1e5` timesteps and save results in the `results` folder.

The default settings will train a `SAC` agent on the `hover` task as per the following-

```
python singleagent.py --configs configs/defaults.yaml hover
```

Custom implementations can be trained using their respective `config` files.

NOTE FOR MASTER BRANCH- Do not change the files until a major change is required. Make changes to your branch first. IF they work then we will incorporate them in the master branch.

Changes

Following are the changes which I have made in comparison to the original gym-pybullet-drones implementation-

- `singleagent.py`- The original implementation makes use of baselines3 to train agents. This will not allow us to complete our project as baselines3 trains the agent under the hood. Thus, I switched the

baselines3 algos with our own custom methods.
- `algos`- Similar to above, we now have our own algorithms in the `algos` directory which are used by the `singleagent.py` file.
- `configs`- All configurations are organized better since we are a total of 3 collaborators.

Development

So what is a good place to start your work? Look at the following-

- Code study- I would suggest you to study and understand the `learning` package before making any changes.
- `algos`- Follow a similar line of coding as in the `algos` folder as this will lead to easier integration and faster progress.
- `configs`- Make sure that your arguments are clean and tuned. A `configs.yaml` is a great way to tune your parameters.
- New files- Incase you wish to make a new file for your code, then please do so in the `algos` folder. This will keep the directory consistent.
- `utils`- Any additional utilities such as saving/logging files or plotting results should be implemented as functions in the `utils.py`. This is to help each other while using the code base. I am currently in the process of building this file.

Future Changes

If all goes well, we would be able to make the folowing changes-

- `multiagent.py`- Hopefully we would be able to scale the project to multiple agents.
- novel scenarios- We could try to transfer knowledge from one task to another based on privilidged information.