Algorithms:

- DQN;

- REINFORCE-PG;

Key Features:

- Environment is compatible to gym.env;

- Support providing to algorithm state (coordinates, velocity, etc.) and sensor data (images from different sensors, from different cameras). Observation is a pair of <State, Data>, where one of the members can be None.

Architecture:

1. Environment (gym.Env interface).

2. DataProviders

3. Scheduler (algorithm processing)

4. Agent (PG-like or DQN-like, or other).

5. Network interface.

Diagram: Scheduler uses (Environment, DataProvider, Agent), Agent uses Network.

Issues (TODO):

**1. DataProvider Logic.**

*a. Entities: TransformationRule, StorageDict (just dict essentially)*

*b. Flow:*

b.1. Observation → StorageDict(numpy),

b.2. ( [StorageDict(numpy)], [TransformationRule] )→ [StorageDict(numpy)],

b.3. StorageDict(numpy) → StorageDict(Tensor),

b.4. StorageDict(numpy) {→ (pack) ← (unpack) } (numpy\_flat\_array, flat\_array\_description),

b.5. StorageDict(Tensor) {→ (pack) ← (unpack) } (Tensor, tensor\_split\_description)

*c. Transformations:* stack several images, compress image, duplicate image if not enough for stacking.

*Note:* Storage Dict: numpy/tensor + description {packed or unpacked}. Description is to be used to make network.

**2. Debugging Tools:**

a. Visualization inside environment + saving images as an option.

b. Visualization inside rewards.

c. Make flag in environment if to save images or not.

d. Imaging inside generate\_session.

**3. Scheduler: non-implemented logic.**

a. Network configuration based on StorageDict (its description part). {enironment → network description}

b. Agent calls and interaction.

c. Mean rewards/summary + Tensorboard integration.

**4. Environment**

- flat array logic: remove or fill

**5. Action Space**

- implement reset method

**6. Rewards**

a. Abstract Rewards: requriements

b. Abstract Rewards: submission

c. Supported Environments field

d. Methods that configure for concrete environment.

**7. General Config**

- Reward Config

- Action Config

- Environment Config: what to track and transformations

- Launch Config: method selection

**8. DQN**

- make compatible to the new architecture

**9. Checkpointing**

- Pytorch checkpoints (also np.random.state)

*Next first steps:*

1. DataProvider logics.

2. Rewards logic.

3. Actioni space logic