Bubble + IQN

Ttobot

{ sejunpark, dujung, victor.cappa } @kaist.ac.kr

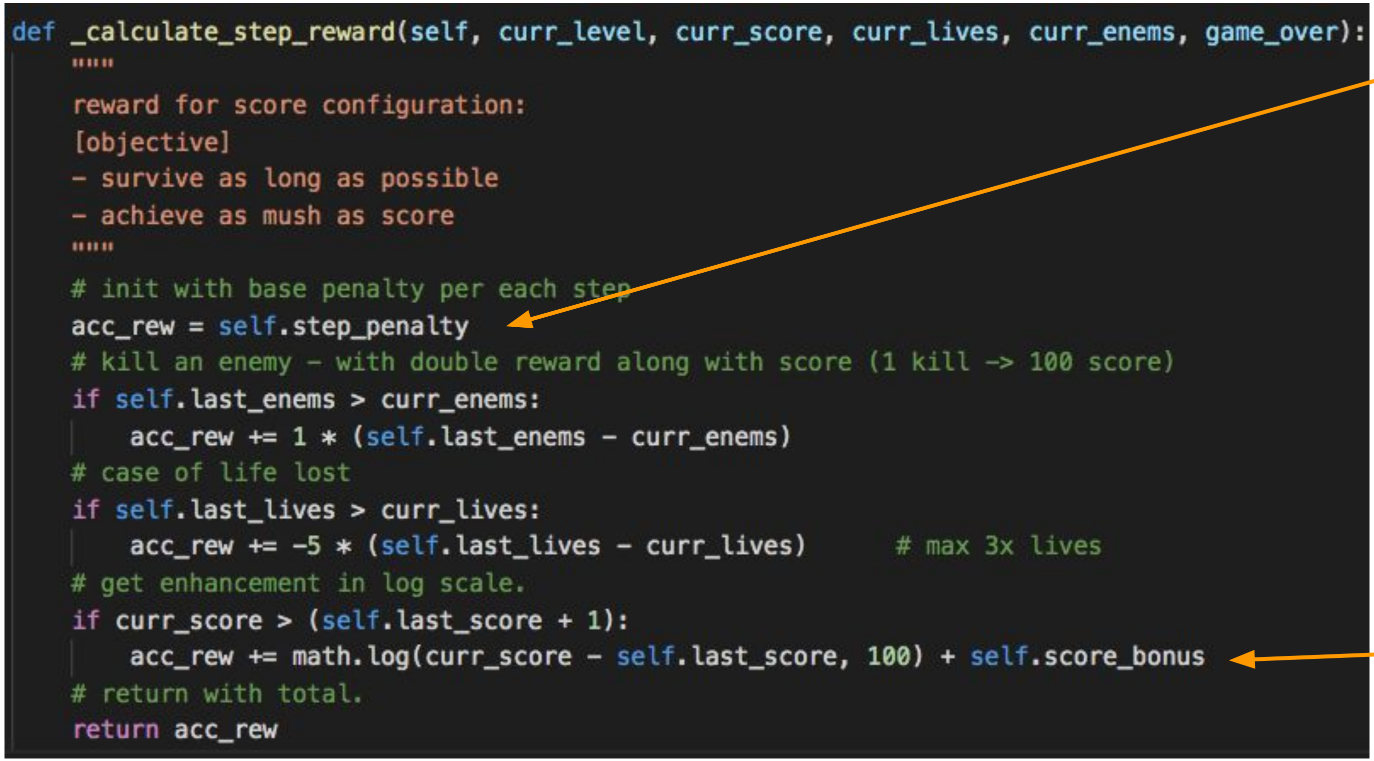
2020.JUL.01

1. Base code

- dopamine by google
  https://github.com/google/dopamine
- use basic code + config (DQN, Rainbow, IQN)
- customized Retro Environment and IQN Agent

2. Reward

- bonus on score and killing enermy. but penalty on death



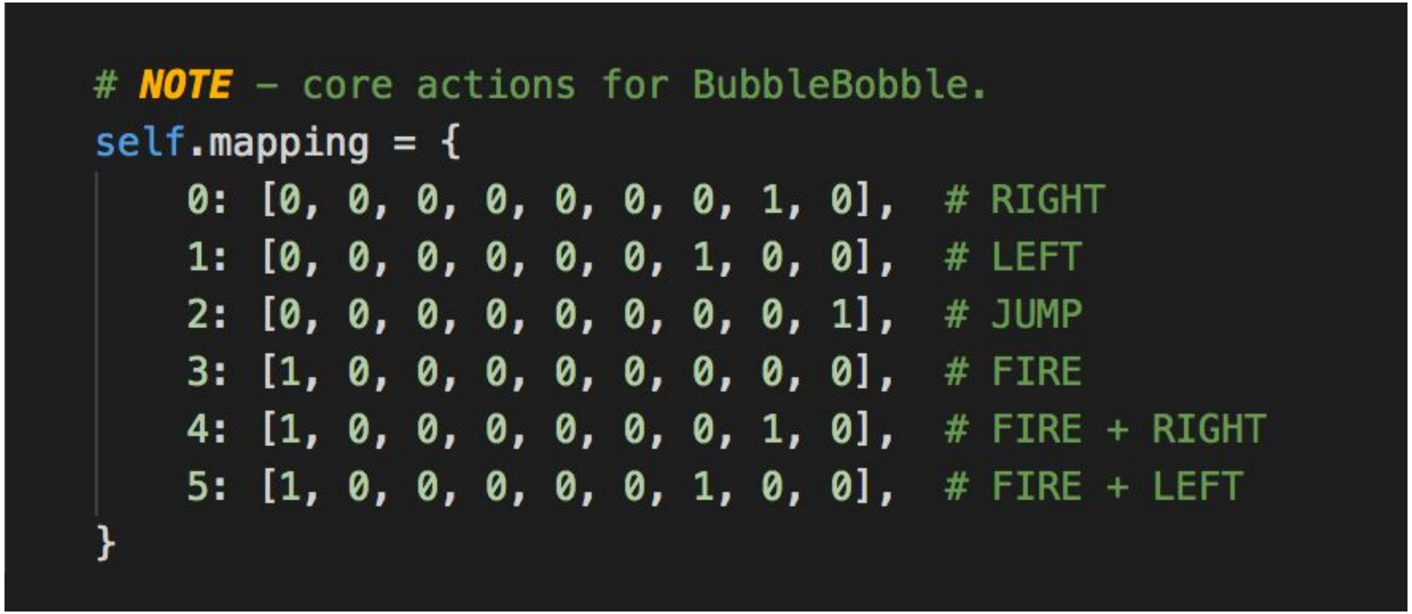
```python
def _calculate_step_reward(self, curr_level, curr_score, curr_lives, curr_enems, game_over):
    """
    reward for score configuration:
    [objective]
    - survive as long as possible
    - achieve as mush as score
    """
    # init with base penalty per each step
    acc_rew = self.step_penalty
    # kill an enemy - with double reward along with score (1 kill -> 100 score)
    if self.last_enems > curr_enems:
        acc_rew += 1 * (self.last_enems - curr_enems)
    # case of life lost
    if self.last_lives > curr_lives:
        acc_rew += -5 * (self.last_lives - curr_lives)      # max 3x lives
    # get enhancement in log scale.
    if curr_score > (self.last_score + 1):
        acc_rew += math.log(curr_score - self.last_score, 100) + self.score_bonus
    # return with total.
    return acc_rew
```

step-penalty:
0.0005

score_bonus:
0.02

3. Action

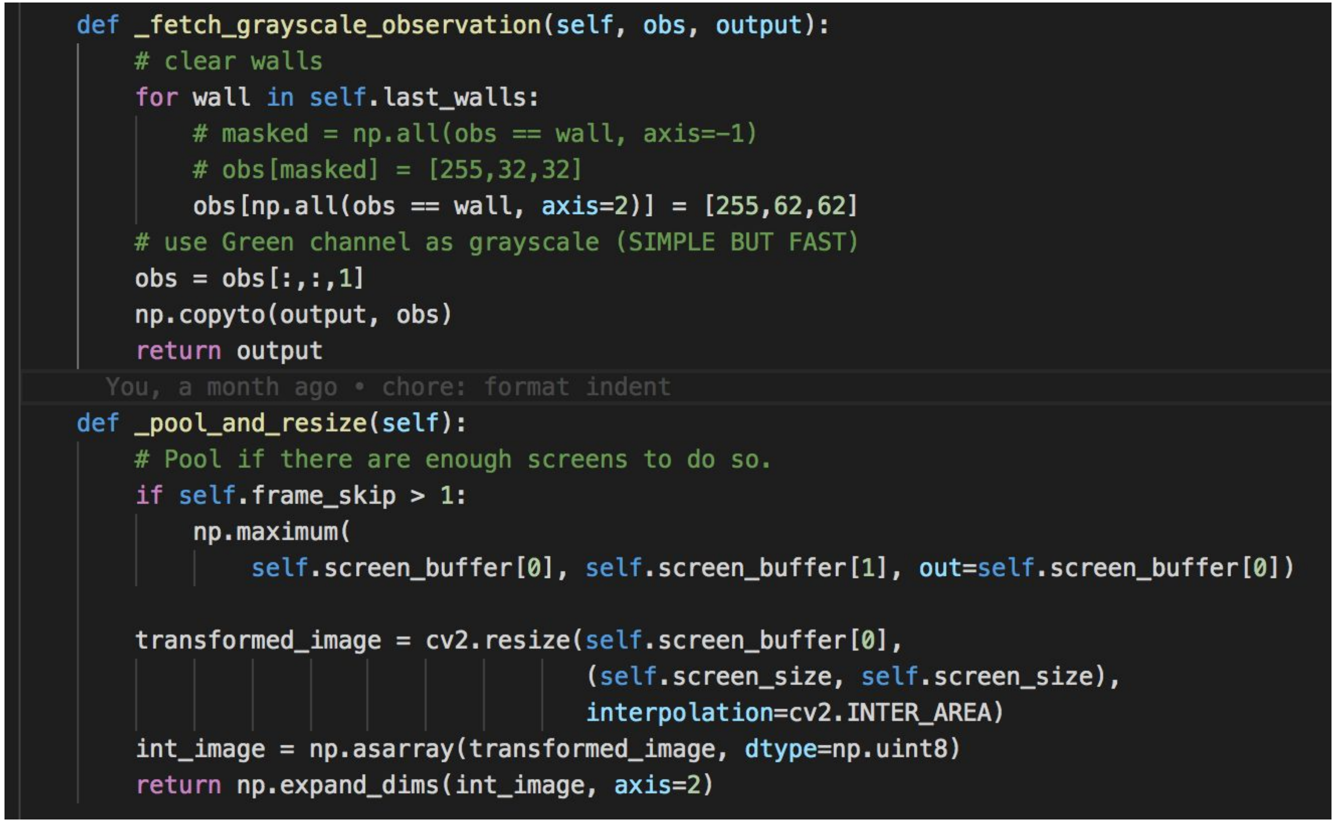
- Discrete action-space with key combinations



```
# NOTE - core actions for BubbleBobble.
self.mapping = {
    0: [0, 0, 0, 0, 0, 0, 0, 1, 0],  # RIGHT
    1: [0, 0, 0, 0, 0, 0, 1, 0, 0],  # LEFT
    2: [0, 0, 0, 0, 0, 0, 0, 0, 1],  # JUMP
    3: [1, 0, 0, 0, 0, 0, 0, 0, 0],  # FIRE
    4: [1, 0, 0, 0, 0, 0, 0, 1, 0],  # FIRE + RIGHT
    5: [1, 0, 0, 0, 0, 0, 1, 0, 0],  # FIRE + LEFT
}
```

4. Pre-processing

- Green channel as grey scaled image by 84x84

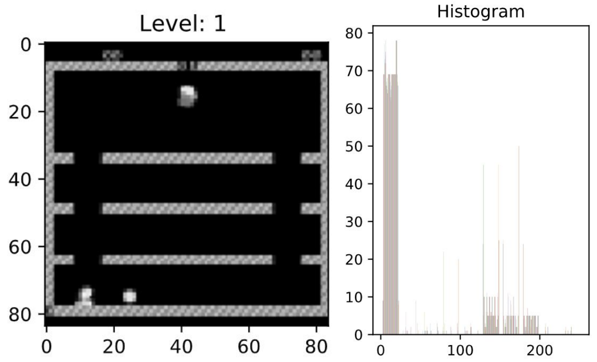


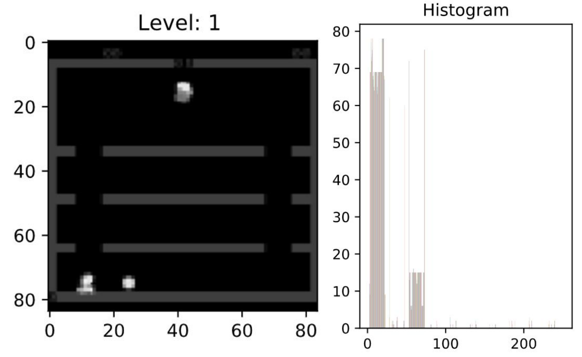
```
    def _fetch_grayscale_observation(self, obs, output):
        # clear walls
        for wall in self.last_walls:
            # masked = np.all(obs == wall, axis=-1)
            # obs[masked] = [255,32,32]
            obs[np.all(obs == wall, axis=2)] = [255,62,62]
        # use Green channel as grayscale (SIMPLE BUT FAST)
        obs = obs[:,:,1]
        np.copyto(output, obs)
        return output
      You, a month ago • chore: format indent
    def _pool_and_resize(self):
        # Pool if there are enough screens to do so.
        if self.frame_skip > 1:
            np.maximum(
                self.screen_buffer[0], self.screen_buffer[1], out=self.screen_buffer[0])

        transformed_image = cv2.resize(self.screen_buffer[0],
                                       (self.screen_size, self.screen_size),
                                       interpolation=cv2.INTER_AREA)
        int_image = np.asarray(transformed_image, dtype=np.uint8)
        return np.expand_dims(int_image, axis=2)
```

5. Performance

- GTX 1080 Ti (12GB) + 24xCore CPU w/ 64GB RAM
- 250k steps / train. 10k steps / eval.

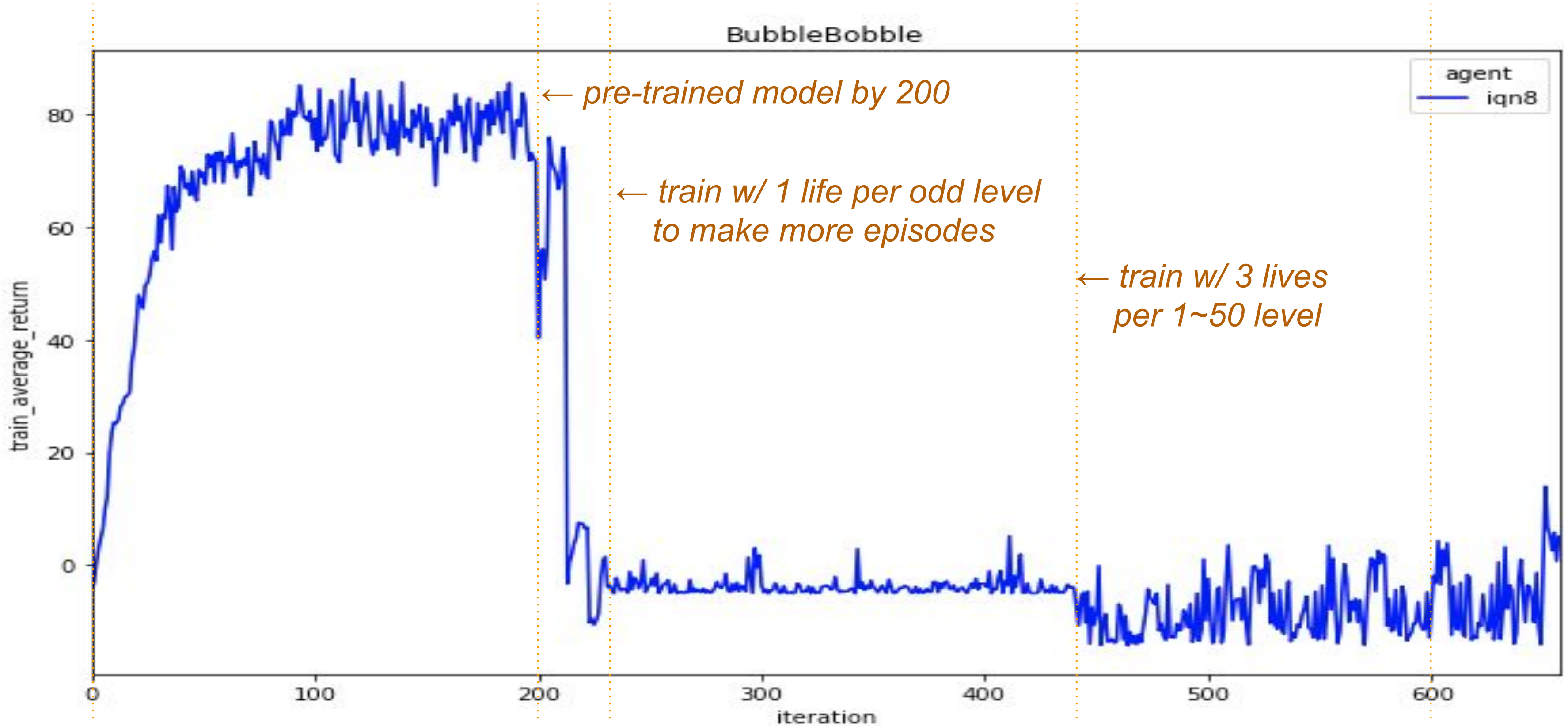
	steps/sec	iteration/day	rewards/train
No Wall Filter	90	~25	-
Wall Filter	60 (-30%)	~17	10% improved





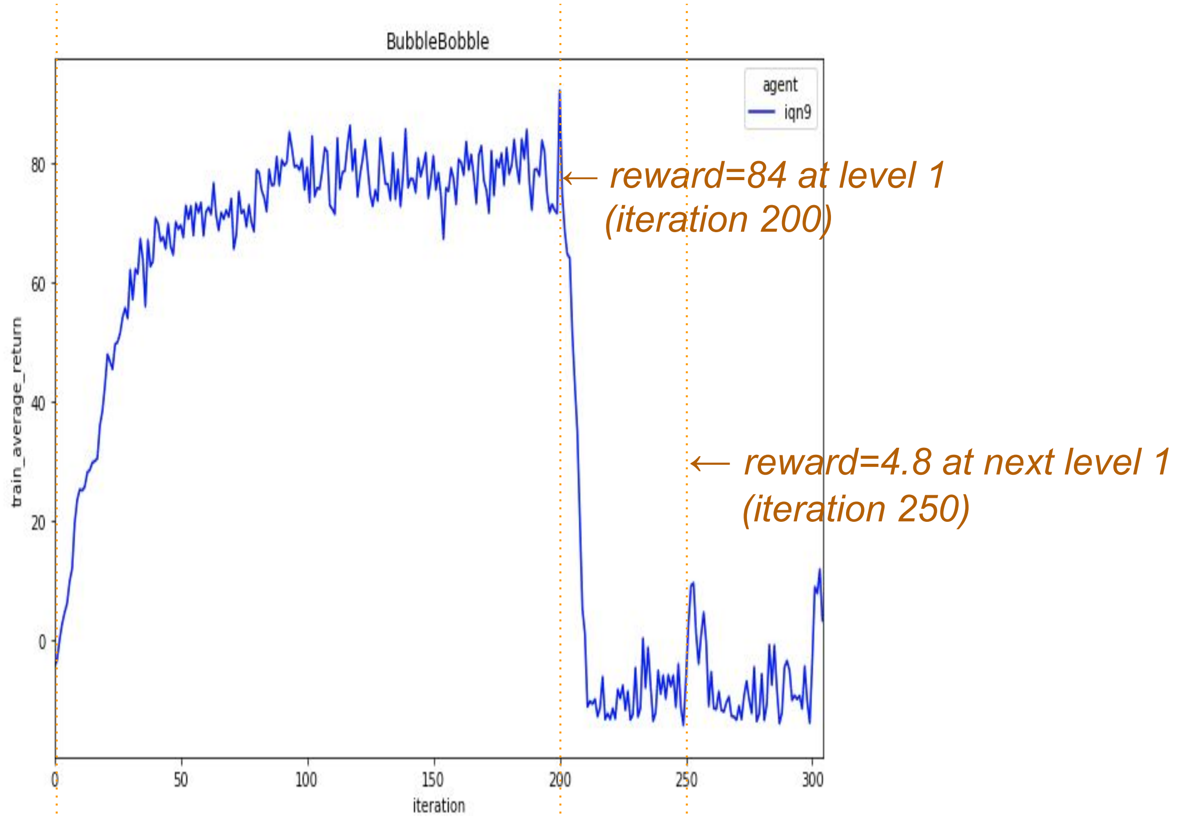
7. Experimental

- Prepared the trained IQN-Agent with 1 level by 200 iteration.



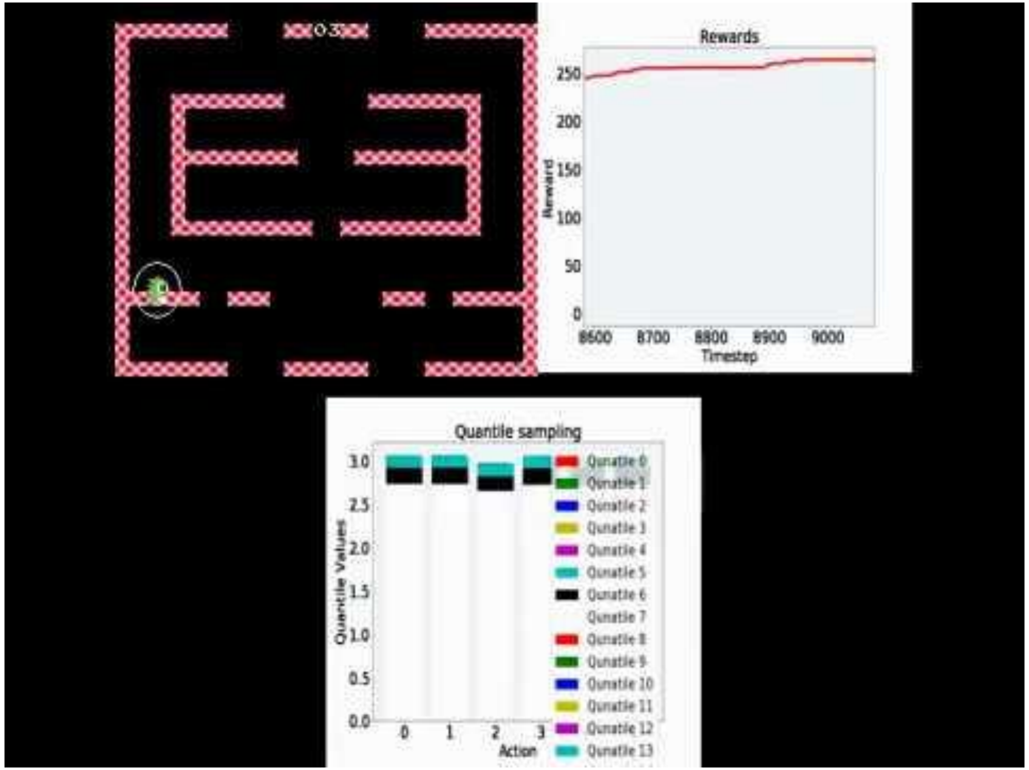
8. Train Forgot?

- Reward at Level 1. Down to 4.8 from 84 after training each 50 levels in sequence



APPENDIX

Implicit Quantile Network (IQN) Tensorflow Agent playing BubbleBobble game


Rewards
Reward
Timestep
8600
8700
8800
8900
9000
Quantile sampling
Quantile Values
Action
Qunatile 0
Qunatile 1
Qunatile 2
Qunatile 3
Qunatile 4
Qunatile 5
Qunatile 6
Qunatile 7
Qunatile 8
Qunatile 9
Qunatile 10
Qunatile 11
Qunatile 12
Qunatile 13