States And Actions As Well As Rewards

Harry Li [‡], Ph.D.
Computer Engineering Department, San Jose State University
San Jose, CA 95192, USA

Email[†]: harry.li@ctione.com

***Abstract*—This note describes the states, actions, and rewards based on Unity AI model as reference guide for deep reinforcement learning experiment design.**

I. Introduction

This note is prepared based on the following material:

1. nnn-n-6DoF-Action-State-Reward-SS-2021-03-17.odt;

2. readme for Unity AI robotics training.

Robotic operations can be characterized as a robot making sequential movement, e.g., control actions in a stochastic environment. By the mathematical nature, these sequential decision process is modeled as a Markov Decision Process (MDP) which consists of

1. a set S of states, denoted as S = $\{s_1, s_2, ..., s_N\}$, and

2. a set A of actions, denoted as A = $\{a_1, a_2, ..., a_M\}$ as well as

3. a transition function T and

4. a reward function R,

denoted as a tuple $< S, A, T, R >$. When in any state $s \in S$, an action $a \in A$ will lead to a new state with a transition probability $P_T(s, a, s\prime)$, and a reward R(s, a) function.

The stochastic policy $\pi : S \to D$ maps from a space state to a probability over the set of actions, and $\pi(a|s)$ represents the probability of choosing action a at state s.

The goal is to find the optimal policy π^* to produce the highest rewards [Rein, 2020]:

$$\arg\max_{\pi \in \Omega_\pi} \{E[\sum_{k=0}^{H-1} \gamma^k R(s_k, a_k)]\} \tag{1}$$

For the stochastic nature of these rewards, we use statistical expectation as

$$E[\sum_{k=0}^{H-1} \gamma^k R(s_k, a_k)] = \int_{\inf} \gamma^k R(s_k, a_k) P(s_k, a_k) ds \tag{2}$$

Hence, we have the average discounted rewards under policy π.

II. States and Actions

Definition 1. *Trajectory τ. A trajectory τ of a robot motion is defined as a sequence of state-action pairs in time sequence $t_1, t_2, ..., t_N$, denoted as $\tau = (s_1, a_1, s_2, a_2, ..., s_N, a_N)$.*

Consider any trajectory τ of a robot motion, e.g., $\tau = (s_1, a_1, s_2, a_2, ..., s_N, a_N)$, the trajectory of the end effector in 3D space. See an end effector (a set of three in this case) from CTI's FD100 robot below,

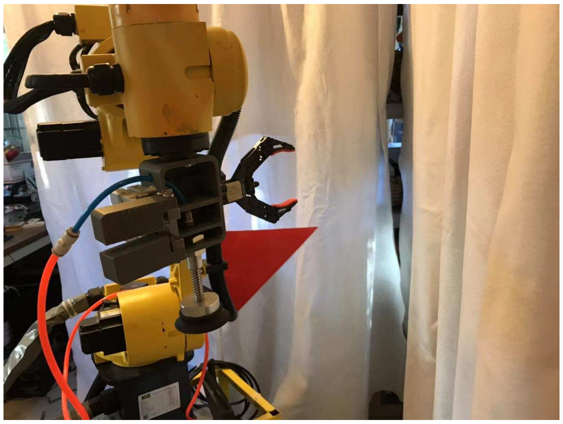

Fig. 1. An end effector (a set of three in this case) from CTI's FD100 robot below, whose movement forms a trajectory in 3D space".

The bigger view of the FD100 robot with an end effector (a set of three in this case) in the view.

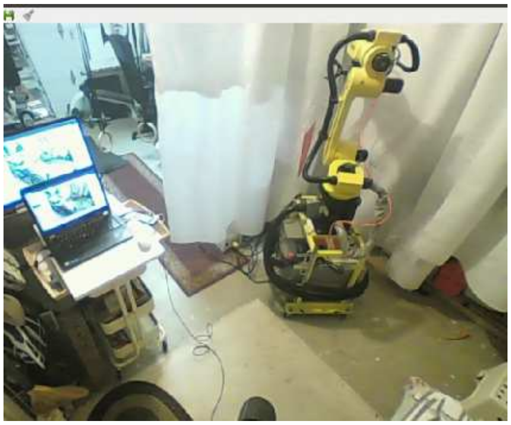

Fig. 2. The bigger view of the FD100 robot with an end effector (a set of three in this case) in the view.

III. Reward

Definition 2. *Reward r. A reward r is defined as numerical value assigned to each state-action pair $s_i a_i$, e.g., formulate as $r : S \times A \to R$, where $S \times A = (s_1a_1, s_2a_1, ..., s_1a_N, s_2a_N, ..., s_Na_1, ..., s_Na_N)$.*

We can build tables for S, A and R repectively, as follows

TABLE I
TABLE I. STATE TABLE FROM UNITY AI ROBOT

Category	Description	Note % Improvement
$s_{j1-angle}$	[,]	Continuous
$s_{j1-speed}$	[,]	Continuous
$s_{j1-accel}$	[,]	Continuous
$s_{j2-angle}$	[,]	Continuous
$s_{j2-speed}$	[,]	Continuous
$s_{j2-accel}$	[,]	Continuous
...	...	...
$s_{j6-angle}$	[,]	Continuous
$s_{j6-speed}$	[,]	Continuous
$s_{j6-accel}$	[,]	Continuous

TABLE II
TABLE II. ACITION TABLE FROM UNITY AI ROBOT

Category	Description	Note % Improvement
$a_{j1-angle}$	[,]	Continuous
$a_{j1-speed}$	[,]	Continuous
$a_{j1-accel}$	[,]	??? Check this
$a_{j2-angle}$	[,]	Continuous
$a_{j2-speed}$	[,]	Continuous
$a_{j2-accel}$	[,]	Continuous
...	...	...
$a_{j6-angle}$	[,]	Continuous
$a_{j6-speed}$	[,]	Continuous
$a_{j6-accel}$	[,]	??? Check this

The design of reward function $R(s_t, a_t)$ is defined based on the general guidelines from Unity AI Robot github [???]

1. When the arm hits the ground, a Hefty Penalty (-1) is given, e.g.,

$$R(s_t, a_t) = -1 \tag{3}$$

where s_t = (ground state), then the training episode is terminated. Note each pisode is defined as one full cycle of training.

2. When the arm reaches the target, a Hefty Reward (1) is given,

$$R(s_t, a_t) = 1 \tag{4}$$

where s_t = (target state). The target state can be detected when the end effector $P_{end}(x, y, z)$ reaches the object(target) $P_{tgt}(x, y, z)$, e.g.,

$$||P_{end}(x, y, z) - P_{tgt}(x, y, z)|| \leq \epsilon \tag{5}$$

Then end the episode;

3. When the arm reaches closer to the target, a marginal reward is assigned based on the normalized difference in distance to the targe. That is to define the previous arm's (end effector) position as $P_{end}(x(t-1), y(t-1), z(t-1))$, if the current position is $P_{end}(x(t), y(t), z(t))$, then the distances to the target position can be defined as

$$d(P_{end}(t-1), P_{tgt}) = ||P_{end}(t-1) - P_{tgt}||_2 \tag{6}$$

and

$$d(P_{end}(t), P_{tgt}) = ||P_{end}(t) - P_{tgt}||_2 \tag{7}$$

if we have

$$d(P_{end}(t), P_{tgt}) \leq d(P_{end}(t-1), P_{tgt}) \tag{8}$$

then, the reward is defined as the distance gained to the target

$$R(s_t, a_t) = \frac{d(P_{end}(t), P_{tgt}) - d(P_{end}(t-1), P_{tgt})}{d(P_{end}(t), P_{tgt})} \tag{9}$$

Note,

$$1 \geq R(s_t, a_t) \geq 0. \tag{10}$$

4. When the arm moves far from the target, based on the similar notion for in 3, we define a reward as a marginal penalty as how far is it moved away from the target as:

if

$$d(P_{end}(t), P_{tgt}) \geq d(P_{end}(t-1), P_{tgt}) \tag{11}$$

$$R(s_t, a_t) = \frac{d(P_{end}(t), P_{tgt}) - d(P_{end}(t-1), P_{tgt})}{d(P_{end}(t), P_{tgt})} \tag{12}$$

Note,

$$-1 \leq R(s_t, a_t) \leq 0. \tag{13}$$

TABLE III
TABLE III. REWARD TABLE

Category	Description	Note % Improvement
$s_{j1-angle} \times a_{j1-angle}$	[,]	Continuous
$s_{j1-speed} \times a_{j1-angle}$	[,]	Continuous
$s_{j1-accel} \times a_{j1-angle}$	[,]	Continuous
...	...	...

Note, check the datasheet of FD100 to fill in the numerical values in the description section of the above tables.

IV. POLICY ON ROBOT OPERATIONS (UNDER CONSTRUCTION)

Definition 3. *Policy π. A policy π in Robotics is a set of guidelines for a robot controller to follow to deliver its control action a_i upon its current state s_i, which is denoted as $\pi(a_i|\ s_i)$.*

A policy π leads to the robot control to deliver its control action as a mapping function $s_i \rightarrow a_i$.

A policy pi can be either stochastic which is characterized by a conditional probability as

$$\pi(a_i|s_i) : s_i \rightarrow Pr(a_i|s_i), \tag{14}$$

or deterministic

$$\pi(a_i|s_i) : s_i \rightarrow a_i = \mu(s_i). \tag{15}$$

V. Policy As DNN (Under Construction)

We now introduce a notation to policy π as π_θ where a set of variables which affects π. In deep reinforcement learning (DRL), a policy π_θ is formulated as a deep neural network (DNN), where θ is the general parameter storing all the networks weights and biases $W = (w_{i,j})$.

Definition 4. *Policy π_θ. A policy π_θ is a deep neural network (DNN), where θ is the collection of all parameters of the neural networks (NN) weights and biases, simply denoted as $W = (w_{i,j})$.*

A typical realization of π_θ is a gaussian Multi-Layer Perceptron (MLP) net, which samples the action to be taken from a gaussian distribution of actions over states as follows

$$\pi_\theta(a_i|s_i) = \frac{1}{\sqrt{(2\pi)^{n_a} det \sum_\theta(s)}} E[-\frac{(||a - \mu_\theta(s)||)^2}{2\sum_\theta(s)}] \quad (16)$$

Example 1. Supppose we have $s_1 = -1, s_2 = 2$ and $a_1 = 10$ and $a_2 = 20$, find

(1) $\sum_\theta(s)$ and $det \sum_\theta(s)$,

(2) $\mu_\theta(s)$,

(3) $\pi_\theta(a_i|s_i) =?$

Sol:

(1) $\sum_\theta(s) = \frac{1}{2}(s_1 + s_2)$,

Acknowledgment

I would like to express my thanks to CTI One engineering member and CTI One engineering intern team, as well as to SJSU graduate research assistants for letting me introducing DRL in Robotics to our next project and for the codings in Deep Learning, Robotis and embedded systems.

VI. Quiz

1. Suppose the target position is P_{tgt} = $(110, 110, 200)$, the robot end effector previous postion is $P_{end}(x(t-1), y(t-1), z(t-1))$ = $(11.35, 113.6, 201.5)$, the current position is $P_{end}(x(t), y(t), z(t))$ =$(23.1, 114, 203.1)$, find the reward function $R(s_t, a_t)$=?

References

[1] [Franceschetti, 2020] Andrea Franceschetti, Elisa Tosello, Nicola Castaman, and Stefano Ghidoni, ”Robotic Arm Control and Task Training through Deep Reinforcement Learning”, https://arxiv.org/pdf/2005.02632.pdf, May 2020.

[2] [Reinforcement, 2020] Reinforcement learning, https://en.wikipedia.org/wiki/Reinforcement _learning, 2020.