Research Internship 1st Report
Reinforcement Learning Overview and Tactile Reward Setting

TIANMING QIU
Institute for Cognitive Systems
September 14, 2017

1 Reinforcement Learning Overview

Some basic knowledges following are from Sutton's *Reinforcement learning: An introduction*[1] and UCL course on RL given by David Silver.

1.1 Markov Decision Process

A Markov decision process is a discrete-time state transition system. It can be described formally with 5 components as a 5-tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma)$.

$\mathcal{S}$ represents state space, $\mathcal{A}$ is action space, $\mathcal{P}$ means transition probability. If the transition probability satisfies $P(s_{t+1}|s_t) = P(s_{t+1}|s_t, s_{t-1}, \ldots, s_1)$, that means it satisfies Markov property. Then this decision process is called Markov decision process. r is reward. γ is a discount factor.

1.2 Model-based Dynamic Programming

'Model-based' means the knowledge of environment is known, mathematically means the transition probability matrix $\mathcal{P}$ and reward r are already known. Two widely-used algorithms regarding dynamic programming are:

- Value iteration
- Policy iteration

1.3 Model-free Reinforcement learning: MC and TD

1.3.1 Monte-Carlo Reinforcement Learning

Monte-Carlo method has no knowledge of MDP transitions and rewards. It learns directly from episodes of experience and the episodes are all complete,

that is different from TD learning methods. The idea of Monte-Carlo method is $value = mean_return$, which is in the algorithm represented as:

Update value $V(S_t)$ toward *actual* return G_t:
$V(S_t) \leftarrow V(S_t) + \alpha(G_t - V(S_t))$

1.3.2 Temporal-Difference Learning

Same as MC, TD learning also has no knowledge of MDP transitions and rewards, it has to estimate value function from episodes of experience as well. The different is TD learns from *incomplete* episodes.

Update value $V(S_t)$ toward *estimated* return $R_{t+1} + \gamma V(S_{t+1})$:
$V(S_t) \leftarrow V(S_t) + \alpha(R_{t+1} + \gamma V(S_{t+1}) - V(S_t))$

Above shows the similarity and difference between MC and TD learning. Also there are two main TD learning methods:

- ON-policy: Sarsa
- OFF-policy: Q-learning

1.4 Q-learning

Q-learning is most widely-used reinforcement learning algorithm, which is a kind of TD-learning. Following is the algorithm:

Q-learning: An off-policy TD control algorithm

Initialize $Q(s,a)$, $\forall s \in \mathcal{S}$, $\forall a \in \mathcal{A}$, arbitrarily, and $Q(\text{terminal-state},\cdot) = 0$
Repeat (for each episode):
  Initialize S
  Repeat (for each step of episode):
    Choose A from S using policy derived from Q (e.g., ϵ-greedy)
    Take action A, observe R, S'
    $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)]$
    $S \leftarrow S'$
  Until S is terminal

1.5 Deep Q Network

If the state space is very big even against infinite, it will be trouble to use traditional q-learning method. Minh introduced a combination of deep learning network with q-learning called DQN algorithm.[2] And this method works pretty well in Atari and Alpha Go. It stores transition $(s_t, a_t, r_{t+1}, s_{t+1})$ in replay memory $\mathcal{D}$, and uses neuronal network to calculate Q values.

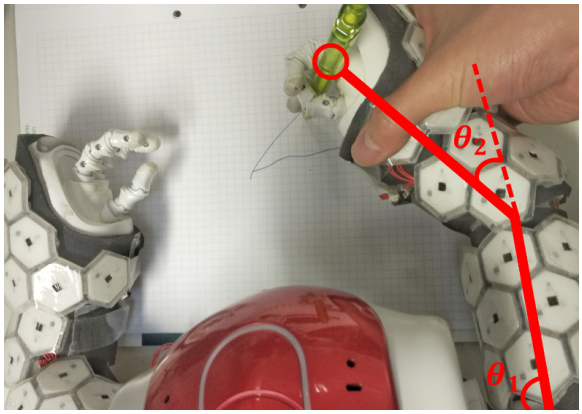

Figure 1: Draw a Triangle

2 Tactile Reward and Modeling

2.1 Task

A very first the task could be train the NAO to draw a triangle like in figure 1. Human will catch the robot hand and give the tactile feedback when robot is drawing. At the very beginning, the freedom of robot arm will be dropped to 2. Only shoulder and elbow joints will be used and two variables θ_1 and θ_2 can be changed and will be learned.

Further the task could be write letters following a personal font. And like images style transfer [3] we could do something 'fonts' transfer for robot manipulation. This part I am not familiar with, just a simple idea.

2.2 MDP Modeling

- **States**: The ranges of both θ_1 and θ_2 are nearly 90°and they will be discretized into 90 levels. The state space will be $90 \times 90 = 810$ dimensions. It is already very large, maybe some methods should be taken to reduce the dimension. Or directly use DQN.
- **Actions**: $\theta_i = \theta_i \pm 1°$
- **Transition probability**: Joint angles will be read from NAO itself and $\mathcal{P}$ will be unknown before. This is a model-free problem.
- **Reward**: Tactile reward can be written as a average force value feedback:

$$r = -\frac{1}{N}(\sum_{i=1}^{N} f_i - f_0)^2$$

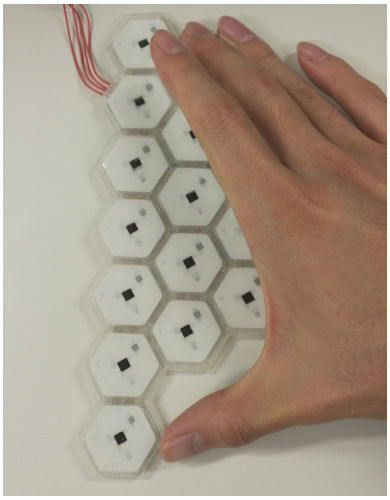

Figure 2: Artificial Skin Cell

wihch f_i means the force data form the ith skin cell and f_0 means the average force when human just catches the NAO's hand and no guidance or pressure would like to give. N is the total number of skin cells.

In figure 2 the skin around one hand include 14 skin cells and fit the size of human beings' hand. So it could record the each accurate change of human hand.

- **Discount factor**: γ will be temporarily chosen as 0.9.

3 Future Plan

3.1 Manipulate Q-learning

I am not familiar with DQN, so I would like to manipulate the traditional Q-learning on NAO at first.

3.2 Modify the MDP Model

Reward function will influence the result a lot. I am not sure weather the modeling is suitable or not. And I also have 3-4 different ideas about how to represent the state space. So I need to first run some codes and find the issues, then make some necessary changes.

3.3 DQN

Since robot arm has a continuous state space or the discretized state space are super large, DQN method need to be considered and it is proved by Alpha Go and Google's robot arm picking up manipulations[4].

3.4 Add a Guidance like PbD?

For reinforcement learning, if the robot arm fails to draw a pre-defined shape it will end an episode. I am figuring out can I give some guidance through tactile sensors and how, something like *Programming by Demostration*[5].

3.5 More Functions of Artificial Skin

Consider about more manipulations on artificial skin and some further task. For example, instead of average force value how to give the direction feedback through different skin cell?

References

1. Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

2. Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

3. Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*, pages 694–711. Springer, 2016.

4. Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 3389–3396. IEEE, 2017.

5. Aude Billard, Sylvain Calinon, Ruediger Dillmann, and Stefan Schaal. Robot programming by demonstration. In *Springer handbook of robotics*, pages 1371–1394. Springer, 2008.