电子科技大学
格拉斯哥学院
Glasgow College, UESTC

Logbook

From: 20/09/2019 To: 25/10/2019

Month	**List the main activities** (only few words per activity)	**Interaction with the supervisor**			**Any other form of supervisory interaction** (second supervisor, industry, fellows etc.)
		Number of meetings	Mode of meeting (face- to-face, online e.g., Skype, WeChat etc.)	Number of emails exchanged	
Sept.	1. kick-off meeting 2. paper reading 3. learning and reviewing	1	WeChat	0	Discussion with one of the students of the supervisor
Oct.	1. meeting and discussion 2. paper reading 3. learning and reviewing 4. project specifications & preliminary report	1	WeChat	0	None

Supervisory Meeting

Meeting Methods/Venue: Wechat

Date: 2019.9.23

Time: 18:15 ~ 19:15

Items for discussion:

1. An overview of my project (What I am supposed to do for the project)
2. Policies about Final Year Project in VoU and ShanghaiTech University

Record of discussion:

1. Prof. Rosendo would put me in contact with his students so that they could tell me the things that they're working on, and then I could work side by side with them and even have a publication before I start the master course.
2. Prof. Rosendo suggested that I could look through the LIMA Lab homepage and find it there are something that I feel more interested in.
3. In ShanghaiTech Prof. Rosendo has budget for visiting students. "When I could go there to spend a few months" may be a most important question because all of the projects require experiments with robots, and being there would make it possible to do those experiments and have results.
4. I sent my CV to Prof. Rosendo and he would talk to Yizheng and Jiahui (his two students) to see which one of them he could put me to work with.
5. At this point it might be early to describe the project on details. But it would be something in the lines of: Reinforcement Learning had been widely used in the last few years for problems, ranging from board games to simulations of robots. Although the learning process may require thousands of interations, new algorithms combine Bayesian learning to find a successful policy to solve the problem. For robots, specifically, such algorithms can be used to enable them to adapt to damages/malfunctions just a few seconds after the problem occurred.

Action list:

1. Have a talk with Prof. Rosendo's students
2. Brainstorm about the detail of the project

No. 2

Date 2019 · 9 · 24

Meeting with Miss. Jiahui

Meeting Method/Venue:	Date:	Time:
Wechat	2019.9.24	15:58 - 16:30

Items for discussion:

Miss Jiahui is one of Prof. Rosendo's students, having a meeting with whom could help me to have a overview of the working topics and topics of the lab. The items for discussion are mainly:

1. Her past and current work
2. Some recommended reading materials in the fields.

Record of discussion:

1. She was working on Bayesian Optimization of a quadruped robot, and she is trying to update the algorithm for better performance and set up a communication module on the robot to communicate with the PC for some more computing-expensive tasks.
2. She gave me some recommended papers, including

a. << A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning >> (Eric Brochu, et al., 2010)
— It gave an introduction to Bayesian Optimization and its classic application

b. << Gaussian Processes >> (Chuong B. Do et al., 2008)
— It gave an introduction to gaussian processes based on Bayesian Optimization

c. << Robots that can adapt like animals >> (Antoine Cully et al., 2015)
— It introduces an intelligent trial-and-error algorithm that allows robots to adapt to damage.

d. << Self-Organization of Locomotion in Modular Robots >> (Yvan Bourquin)
— It tuned the oscillatory parameters of robots using different numerical search methods.

e. << Bayesian Optimization of a Quadruped Robot During 3-Dimensional Locomotion >> (Jiahui et al., 2019)
— It is the work just has done by Miss Jiahui.

Action list

1. Read the materials above and take notes
2. Search for some relative materials on the Internet
3. Think about details of the project with the help of the reading

Summary of "A Visual Exploration of Gaussian Processes"

This is a note for a wonderful tutorial on Gaussian Processes online.

Multivariate Gaussian distributions

The multivariate Gaussian distribution is defined by a mean vector μ and a covariance matrix Σ.

$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \sim N(\mu, \Sigma)$

C The diagonal of Σ consists of the variance σ_i^2 of the i-th random variable. And the off-diagonal elements σ_{ij} describe the correlation between the i-th and j-th random variable.

Marginalization and Conditioning

When $P_{X,Y} = \begin{bmatrix} X \\ Y \end{bmatrix} \sim N(\mu, \Sigma) = N\left(\begin{bmatrix} \mu_X \\ \mu_Y \end{bmatrix}, \begin{bmatrix} \Sigma_{XX} & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_{YY} \end{bmatrix}\right)$

Then to marginalize out a random variable from a Gaussian distribution we can simply drop the variable from μ and Σ: $P_X(x) = \int_y P_{X,Y}(x,y)\,dy = \int_y P_{X|Y}(x|y) P_Y(y)\,dy$

And conditioning is defined by: $X|Y \sim N(\mu_X + \Sigma_{XY}\Sigma_{YY}^{-1}(Y - \mu_Y), \Sigma_{XX} - \Sigma_{XY}\Sigma_{YY}^{-1}\Sigma_{YX})$

Gaussian Processes

The key idea of Gaussian Processes is to model the underlying distribution of X together with Y as a multivariate normal distribution, while we denote the test data as X and training data as Y. Then we are interested in finding the μ and Σ of $P_{X|Y}$.

Kernels

In Gaussian processes it is often assumed that $\mu = 0$ and kernels are widely used to model the desired shape of the function. $\Sigma = Cov(X, X) = k(t, t')$

- stationary kernels: only dependent on their relative position — RBF kernel, periodic kernel
- non-stationary kernels: dependent on an absolute position — linear kernel

Prior and Posterior Distribution

In the context of Bayesian inference, the case without observing any training data called prior distribution P_X.

For measurement errors or uncertainty, we need to add an error term $\epsilon \sim N(0, \psi^2)$ to each of our training points: $Y = f(X) + \varepsilon$,

Then $P_{X,Y} = \begin{bmatrix} X \\ Y \end{bmatrix} \sim N(0, \Sigma) = N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \Sigma_{XX} & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_{YY} + \psi^2 I \end{bmatrix}\right)$

Conclusion

It could be seen that Gaussian processes offer a flexible framework for regression and several extensions exist that make them even more versatile.

Reference

Jochen Görtler et al., 2019 A Visual Exploration of Gaussian Processes
https://distill.pub/2019/visual-exploration-gaussian-process/

No. 4

Date 2019. 9. 30

First Exploration of Bayesian Optimization

Part of this note is from the paper "A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning" (Eric Brochu et al., 2010) and a tutorial video "Optimization using ML methods" on YouTube (https://youtu.be/K_qNcLY3XU1) helps for my understanding a lot.

1. BO is applicable in situations where one does not have a closed-form expression for the objective function, but where one can obtain observations of this function at sampled values. It is particularly useful when these evaluations are costly, when one does not have access to derivatives, or when the problem at hand is non-convex

2. Surrogate model. A popular surrogate model for Bayesian optimization are Gaussian processes (GPs). GPs define a prior over functions and we can use them to incorporate prior beliefs about the objective function. The GP posterior is cheap to evaluate and is used to propose points in the search space where sampling is likely to yield an improvement.

3. Acquisition functions. Proposing sampling points in the search space is done by acquisition functions. They trade off exploitation and exploration and the goal is to maximize the acquisition function to determine the next sampling point. Popular acquisition functions are "maximum probability of improvement" (MPI), expected improvement (EI) and upper confidence bound (UCB).

4. Optimization algorithm. The Bayesian Optimization procedure is as followed. For $t = 1, 2, \cdots$ repeat:

   ① Find the next sampling point x_t by optimizing the acquistion function over the GP:

   $$x_t = \operatorname{argmax}_x u(x | D_{1:t-1})$$

   ② Obtain a possibly noisy sample $y_t = f(x_t) + \epsilon_t$ from the objective function f.

   ③ Add the sample to previous samples $D_{1:t} = \{D_{1:t-1}, (x_t, y_t)\}$ and update the GP

Summary of "Bayesian Optimization of a Quadruped Robot During 3-Dimensional Locomotion"

This is a paper done by one of the students of Prof Rosendo, Miss Jiahui Zhu. It uses Bayesian Optimization (BO) for parametric search for gait controllers of a quadruped robot. The objective function $f(\cdot)$ is the vertical displacement between initial and end position.

Gaussian Processes It introduces the Matérn kernel and adopts it in this work, represented by:

$$k_{Matérn}(d) = \sigma_p^2 \frac{2^{1-\upsilon}}{\Gamma(\upsilon)} \frac{\sqrt{2\upsilon}^{\upsilon}}{l} K_{\upsilon} \frac{\sqrt{2\upsilon}}{l}$$

Bayesian Optimization It uses the probability of improvement (PI) as acquisition functions, defined as:

$$PI(x) = \phi\left(\frac{\mu(x) - y_{max} - \xi}{\sigma(x)}\right)$$

Robotic Platform The robot use Arduino nano to control eight servo-motors and each of its four legs contains two servomotors to control its hips and knees. Therefore, the robot can move freely in 3-D space.

Parameter Configuration Each servomotor is controlled by an oscillator, with a sinusoidal equation

$$x(t) = x_0 + A\sin(t/\tau + \varphi),$$

while these sine waves must have the same frequency but they can differ in phases, amplitude and the midpoint. After compressing the search space with symmetrical parameter configuration, the problem is reduced to 10-dimensional parameters. The search space is as followed

Parameter	Control	Value
LF/LH/RH/RF Hip	midpoint x_i	[255 295 305 345]
LF/LH/RH/RF Knee	midpoint x_j	[215 255 295 305]
LF/RF Hip	amplitude A_i	[50 55 60 65]
LH/RH Hip	amplitude A_j	[60 65 70 75]
LF/RF Knee	amplitude A_m	[25 30 35 40]
LH/RH Knee	amplitude A_n	[20 25 30 35]
LF/RF Hip	phase φ_i	[-90 0 90 180]
LH/RH Hip	phase φ_j	[-90 0 90 180]
LF/RH Knee	phase φ_m	[-90 0 90 180]
LH/RF Knee	phase φ_n	[-90 0 90 180]

Result The results show that there are major advantages of BO, such as data-efficiency during the optimization process. The robot starts from zero knowledge about its morphology and after 23 interations it reaches optimal behaviour.

No. 6

Date : 2019 · 10 · 4

Supervisory Meeting

Meeting Methods / Venue: Wechat

Date: 2019.10.4

Time: 17:27 - 18:05

Items for discussion:

1. The similarity between my project and Jiahui's (What can I learn from the previous work)
2. What is the differences between Bayesian Optimization and Reinforcement Learning
3. Is it possible to combine these two methods?

Record of discussion:

1. I can choose to use reinforcement learning to do a similar task as Miss Jiahui's.
2. At the moment Jiahui is the only one working on this parametric search, while many others in the lab are working on RL.
3. I could work with either one or the other. If I work with BO, my chances of publishing are higher, as less people are studying it, so I have more chances to shine.
4. Since my project title is about reinforcement learning, I may find a way to combine the two methods. And there is a way: I could use bayesian learning to search for RL parameters.
5. I could search online for Deep PILCO to get some reading materials.

Action list:

1. Find and read some literature about PILCO.
2. Find and read some literature about combination of reinforcement learning and bayesian optimization.

Summary of the Difference Between Bayesian Optimization and Reinforcement Learning

Since Bayesian Optimization (BO) and Reinforcement Learning (RL) are two foundamental tools for my project, it is meaningful and important to compare the two methods and make it possible for combination.

Review of RL in the context of robotics. The "actions a" available to the robot might be the torque sent to motors or the desired accelerations sent to an inverse dynamics control system. A function π that generates the motor commands (i.e. the actions) based on the incoming ball and current internal arm observations (i.e. the state) would be called the "policy". A reinforcement learning problem is to find a policy that optimizes the long-term sum of "reward $R(s,a)$": a reinforcement learning algorithm is one designed to find such a (near-) optimal policy. (Jens Kober et al., 2014, "Reinforcement Learning in robotics: A survey")

Compare BO with RL. A tutorial video on YouTube ("Marc Deisenroth: Fast Robot Learning with Gaussian Processes", https://youtu.be/ft7y60sE2fXc) helps me on comprasion of these two approaches, as the table shown below:

	Cost	Dynamics model	Policy learning	# Parameters
RL	✓	✓	✓	$\leq 10{,}000$
BO	×	×	✓	≤ 20

- If a good dynamics model can be learned and a cost function can be defined, RL-based methods provide better flexibility.
- Bayesian optimization is a more general/easier framework for learning a few parameters, but it does not scale to many parameters.

Fast Reinforcement Learning. Probabilistic Inference for Learning Control (PILCO) is a way to combine the two methods. Its high-level steps are as followed:

1. Probabilistic model for transition function f (System identification)
2. Compute long-term predictions $p(x_1|\theta), \cdots, p(x_T|\theta)$
3. Policy improvement { Compute expected long-term cost $J(\theta) = \sum_t \mathbb{E}[c(x_t)|\theta]$
   Find parameters θ that minimize $J(\theta)$
4. Apply controller

Two examples: ① Standard Benchmark Problem: Cart-Pole Swing-up

Marc Peter Deisenroth et al., 2011, PILCO: A Model-Based and Data-Efficient Approach to Policy Search

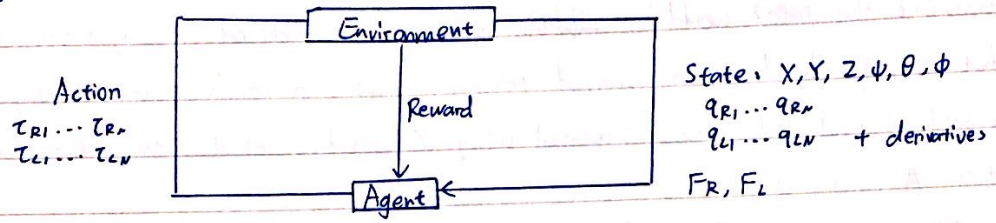
② Learning to Pick up Objects

B. Bischoff et al., 2014, Policy Search For Learning Robot Control Using Sparse Data

Deep Reinforcement Learning for Walking Robots in MATLAB

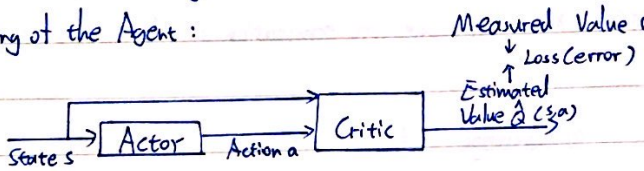
This note is based on a tutorial on YouTube ("Deep Reinforcement Learning for Walking Robots - MATLAB and Simulink Robotics Arena", https://youtu.be/6DL5M9b2j6I) which could be useful when doing comparative experiment later.

Deep Deterministic Policy Gradient (DDPG): A reinforcement learning algorithm can handle continuous action space (Timothy P. Lillicrap et al., 2015, Continuous Control with Deep Reinforcement Learning)

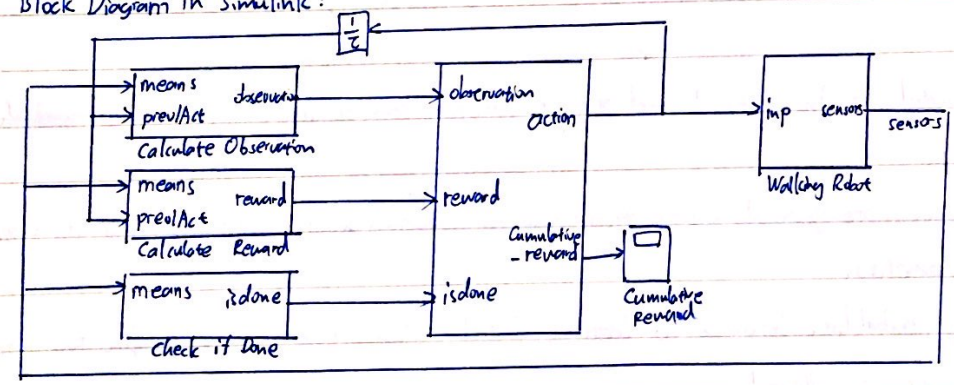


Reinforcement Learning: Use past experiences to maximize expected reward

Training of the Agent:



Block Diagram in Simulink:



The example files could be downloaded on http://bit.ly/2HBxe79 or https://www.mathworks.com/matlabcentral/fileexchange/64227-matlab-and-simulink-robotics-arena-walking-robot

I would do the experiment later.

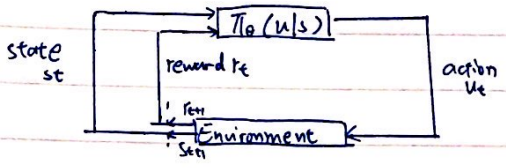
ps: It is shown in the video that the training process require a long time, which is impractical in physical robot and reflects the problem of RL.

Notes for Policy Search

Before exploring the method of PILCO, I find it necessary to learn more about policy search in the context of Reinforcement Learning. An informative and understandable lecture on YouTube in this field is found, whose title is "Policy Search for Reinforcement Learning" by Pieter Abbeel. (https://youtu.be/dEHyTBJ1pSc, and the slide is available on http://bit.ly/2xcW3Rp)

The first note of this lecture is as followed.

Policy Optimization



Consider control policy parameterized by parameter vector θ

$\max_\theta E[\sum_{t=0}^{H} R(s_t)|\pi_\theta]$

Often stochastic policy class (smooths out the problem)

$\pi_\theta(u|s)$: probability of action u in state s

Why Policy Optimization

Often π can be simpler than Q or V (e.g., robotic grasp)

V: doesn't prescribe action (would need dynamics model)

Q: need to be able to efficiently solve $\arg\max_u Q_\theta(s,u)$

(Challenge for continuous/high-dimensional action spaces)

Successful Examples

Kohl & Stone, 2004, Policy Gradient Reinforcement Learning for Fast Quadrupedal Locomotion

Ng et al, 2004, Autonomous Inverted Helicopter Flight via Reinforcement Learning

Tedrake et al, 2005, Learning to Walk in 20 Minutes

Kober & Peters, 2009, Policy Search for Motor Primitives in Robotics

Mnih et al, 2015, Asynchronous Methods for Deep Reinforcement Learning (A3C)

Silver et al, 2014, Deterministic Policy Gradient Algorithms (DPG)

Lillicrap et al, 2015, Continuous control With Deep Reinforcement Learning (DDPG)

Schulman et al, 2016, High-Dimensional Continuous Control Using Generalized Advantage Estimation (TRPO+GAE)

Levine, Finn, et al, 2016, Guided Policy Search

Silver, Huang, et al, 2016, Mastering the Game of Go with Deep Neural Network And Tree Search (AlphaGo)

(I've organize above materials for further reading)