Title

Bowen Zhi

Human and Biological Robotics
Imperial College London

Word Count:

March 2025

Template for the Thesis of Imperial College London

ABSTRACT

This is abstract. The appendix of this article is used for example, like this ”See Appendix A for details”.

Key Words: Key1 Key2 Key3 Key4

Contents

ABSTRACT I
Contents II
1 Introduction 1
2 Dynamic Programming 2
  2.1 Parameterisation 2
  2.2 Graphical representation 2
Reference 3
A Matlab example 4
B Python example 4

Chapter 1 Introduction

The format of this template is for reference only, the content is only for the purpose of displaying the format and is meaningless. For examlpe, we can have a matlab example in A.

Chapter 2 Dynamic Programming

2.1 Parameterisation

Kaelbling et al. (1996) discusses central issues of reinforcement learning, including trading off exploration and exploitation, establishing the foundations of the field via Markov decision theory, learning from delayed reinforcement, constructing empirical models to accelerate learning, making use of generalization and hierarchy, and coping with hidden state. Python example in B。

They (Kaelbling et al.; 1996) discusses central issues of reinforcement learning, including trading off exploration and exploitation, establishing the foundations of the field via Markov decision theory, learning from delayed reinforcement, constructing empirical models to accelerate learning, making use of generalization and hierarchy, and coping with hidden state.

2.2 Graphical representation

The Imperial College's Coat of arms is shown in Figure 2-1.

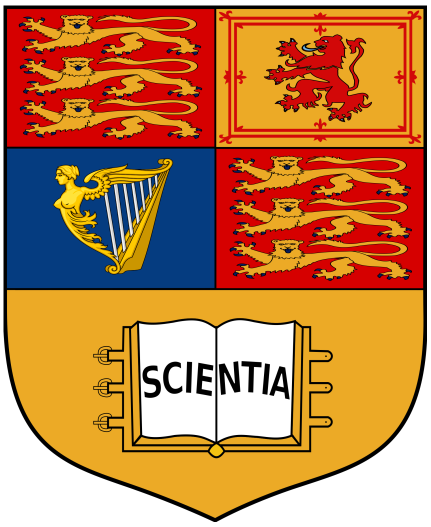

Figure 2-1 ICL arms

References

Kaelbling, L. P., Littman, M. L. and Moore, A. W. (1996). Reinforcement learning: A survey, Journal of artificial intelligence research **4**: 237–285. 2

Appendix A Matlab example

```matlab
function f=det(P)
n=size(P,4);
f=0;
if n==2
f=conv(P(:,:,1,1),P(:,:,2,2))-conv(P(:,:,1,2),P(:,:,2,1));
end
```

Appendix B Python example

```python
import math
def hix(x):
if x > 0:
return x
else:
return 0

def trans1(x1, x2, n, s1, s2, lock):
# 1.  (A1,B1) -> (A1,B1)
if lock == 1:
dt = 22 * (math.floor(x1 / 2) + math.floor(x2 / 2))
dw = math.floor(x1 / 2) * n + math.floor(x2 / 2) * (150 - n)
return dt, dw, lock
else:
return 0, 0, lock
```