NAIST-IS-DD0361013

**博士論文**

# コミュニケーション実現のための階層型モジュール強化学習

杉本 徳和

2006 年 3 月 24 日

奈良先端科学技術大学院大学
情報科学研究科 情報生命科学専攻

本論文は奈良先端科学技術大学院大学情報科学研究科に
博士 (工学) 授与の要件として提出した博士論文である。

杉本 徳和

審査委員：

| | |
|---|---|
| 石井 信 教授 | 主指導教員 |
| 小笠原 司 教授 | 副指導教員 |
| 川人 光男 客員教授 | 副指導教員 |
| 柴田 智広 助教授 | 副指導教員 |
| 銅谷 賢治 客員助教授 | 副指導教員 |

# コミュニケーション実現のための階層型モジュール強化学習*

杉本 徳和

## 内容梗概

ヒトは複雑な環境をシンボル表現する事で高度な推論や認知的活動を可能にしており，そして見まね学習や協調作業など他者との相互作用がシンボル生成の源であるとされている．連続で高次元な環境にシンボルを割り当てる事や他者が用いているシンボルの意図を推定する事は一般に困難な問題であるが，本研究ではエージェントが過去の経験により獲得している行動即がそれらを解決すると仮定する．その時，エージェントの制御アルゴリズムが生成されるシンボルの性能を大きく左右する．そこで本研究では他の手法よりも柔軟な構造を持つ階層型強化学習手法を提案し，その枠組みを用いて他者のシンボルを理解したり新たなシンボルを生み出す枠組みを提案する．

本稿ではまず，MOSAIC モデルに基づいたモジュール構造を持つ 2 つの強化学習アルゴリズムについて説明する．1 つ目は複数個の状態予測モデルと報酬予測モデル，強化学習コントローラを持ち，環境の変化に応じて 3 者を適応的に切り替えながら制御を行うアルゴリズムである．2 つ目は MOSAIC モデルに基づいて環境を抽象化する階層型モジュール強化学習である．このアルゴリズムは状態予測誤差に基づいて環境を抽象化する下位層と，抽象化された状態空間で強化学習を行う上位層から構成される．上位層は各抽象化状態に対して選択するべきサブゴールを下位層に指示し，下位層がサブゴールを目標とし環境への出力を行う構造になっている．

*奈良先端科学技術大学院大学 情報科学研究科 情報生命科学専攻 博士論文，NAIST-IS-DD0361013, 2006 年 3 月 24 日.

後半では階層型モジュール強化学習を用いたコミュニケーションのモデル化について説明する．我々がコミュニケーションを行う際，観測された他者の軌道からその裏に隠れた高次の情報を推定している事は想像に難くない．その推定は一般に不良設定問題となるが，提案するモデルでは他者エージェントも MOSAIC モデルを基礎とし，似たような階層構造を持っていると仮定することで他者がどのモジュールを用いているかが推定できる．他者が使用しているモジュール系列をまねる見まね学習と他者のモジュール系列に応じて自身の行動選択を行う協調作業の枠組みを定式化する．

最後に本研究のまとめと議論を行い今後の展開について考察する．

**キーワード**

強化学習，モジュール階層構造，コミュニケーション，見まね学習，協調作業

# Hierarchical Reinforcement Learning: Commputational model of communication*

Norikazu Sugimoto

**Abstract**

Many cognitive skills, such as inference and planning, require a symbolic representation of the environment. These symbolic representations are generated through social interactions, e.g. imitation and cooperation. In this thesis, we assume that the motor control system is subserving action understanding and symbol formation for communication and cognitive skills. We propose a general framework, in which prediction models are utilized for both motor control and action understanding.

This thesis has two parts. In the first part, we propose two new algorithms based on the MOSAIC architecture (quote needed, Haruno/Kawato/Samejima). First, we present a modular reinforcement learning (RL) algorithm, where multiple forward models, reward models and RL controllers are adaptively combined, suitable for non-stationary environments. Second, we present a hierarchial modular RL algorithm for abstract representation of the environment. It has two layers: a top layer for selection of subgoals, and a bottom layer which selects its own goals.

In the second part, we discuss computational models of communication in the context of the hierarchial modular RL model. When we communicate with each

*Doctoral Dissertation, Department of Bioinformatics and Genomics, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DD0361013, March 24, 2006.

other, we may conceive higher order information such as intention just by observing the movement trajectories of one another. Then we can aquire skills by imitation or we can cooperate by selecting actions in accordance with each others' intentions. We introduce a modified hierarchial algorithm for estimation of higher order information from the trajectory of others, and we show that such symbolic information improve the learning of cognitive skills.

The thesis is concluded with a summary of the results, and suggestions of future developments.

**Keywords:**

reinforcement Learning, modularhierarchical architecture, communication, imitation, cooperation

# 目 次

# 図 目 次

# 表 目 次

# 第1章　序章

## 1. MOSAICモデル

非線形・非定常な環境へ柔軟に対応するため，環境を単純な状況や局所領域に分割し，それぞれにモジュール化された学習器を割り当てると言うアプローチがある．その様な学習システムの研究として “mixtures of expert” が良く知られている [23]．複数個のモジュール学習器が並列に存在し，それらの出力をゲート回路によって切り替えるシステムである．しかしモジュールの個数に比例してゲート回路の規模が大きくなってしまい学習が困難となる．

一方，Wolpert らによって提案された MOSAIC モデルでは 1 対の状態予測モデルと制御器が 1 つのモジュール学習器を成しており，状態予測の精度に応じて責任信号と呼ばれる重み付けの信号が計算される [43, 44, 18, 19]．それによって学習や出力の重み付けが行われ，責任信号が高かったモジュールのみ学習が行われ，低かったモジュールの学習結果は保持される．責任信号の計算とモジュールの学習を繰り返す事で個々のモジュール学習器の分化が実現され，霊長類の高い運動能力と柔軟な脱適応・再適応能力を実現するモデルとされている．

### 1.1 階層型MOSAICモデル

我々の日常において会話や複雑な運動課題などは毎時刻ごとのモジュール選択だけではなく，過去に用いられていたモジュールの情報を積極的に使用することで成り立っている．その様な文脈情報を処理するためには階層構造の導入が有効であり，Haruno らは MOSAIC を拡張して階層構造を持つ Hierarchical MOSAIC(HMOSAIC) を提案している [20]．各層は MOSAIC モデルにより構成され

ているため基本的に同じ構造を持っているが，上位層は下位層の責任信号を入力とし下位層が用いるべき理想的な責任信号を出力としている点で異なる．そして下位層は上位層の出力を事前分布として責任信号の計算を行う．最下位層は通常の MOSAIC と同じ入出力関係を持っている．上位層の各モジュールがそれぞれ責任信号の異なる時系列を学習することで複数の文脈情報を表現でき，また各層の構造が同じであるため容易に 3 層以上へと拡張可能である点が特徴である．

## 2. 強化学習

強化学習とは環境との相互作用の結果に対する評価，すなわち報酬の累積値が最大化されるように行動やその時系列を試行錯誤的に獲得する学習法である [38]．強化学習エージェントの目的は報酬の累積和が最大となるような行動決定則を学習することである．一般的な強化学習理論は離散な時間・状態・行動のもとで定式化されており，離散時間を $t_{dis}$，離散状態を $x_{dis}$，離散行動を $u_{dis}$ とすると即時的な報酬は

$$R(t_{dis}) = R(x_{dis}(t_{dis}), u_{dis}(t_{dis})) \tag{1.1}$$

と言うように状態と行動の関数として与えられる．また行動出力は状態の関数として

$$u_{dis} = \pi(\mathbf{x}_{dis}) \tag{1.2}$$

のように与えられ，$\pi$ は “方策 (policy)” と呼ばれる．ある方策 $\pi$ のもとでの累積報酬は

$$V^{\pi}(x_{dis}(t_{dis})) = R(t_{dis}+1) + \gamma R(t_{dis}+2) + \gamma^2 R(t_{dis}+3) + \cdots \tag{1.3}$$

$$= R(t_{dis}+1) + \gamma V^{\pi}(x_{dis}(t_{dis}+1)) \tag{1.4}$$

と定義され，$V^{\pi}$ は方策 $\pi$ のもとでの状態価値関数と呼ばれる．ここで $0 \leq \gamma < 1$ は報酬の割引率を決めるパラメータであり，1 に近いほど将来に受ける報酬を重視する事を意味する．強化学習の目的は，環境の各状態に対して状態価値関数 $V^{\pi}$

が最大となるような方策 $\pi$ を求めると言い換えることが出来る．また Watkins ら [41] は状態だけでなく，状態と行動の組に対する価値関数を定義している．

$$Q^{\pi}(x_{dis}(t_{dis}), u_{dis}(t_{dis})) = R(t_{dis}+1) + \gamma R(t_{dis}+2) + \gamma^2 R(t_{dis}+3) + \cdots \quad (1.5)$$

$$= R(t_{dis}+1) + \gamma \sum_{x'_{dis}} P(x'_{dis} \mid x_{dis}(t_{dis}), u_{dis}(t_{dis})) V^{\pi}(x'_{dis}) \quad (1.6)$$

ここで $Q^{\pi}$ は方策 $\pi$ のもとでの状態行動価値関数と呼ばれる．

## 2.1 連続版強化学習

Doya は環境の時間・状態・行動が連続な変数として表現される場合の強化学習理論を定式化している [13]．時刻 $t \in \Re$ における制御対象の状態を $\mathbf{x}(t) \in \Re^{N_s}$，制御出力を $\mathbf{u}(t) \in \Re^{N_c}$ とした時，環境の状態変化 $\dot{\mathbf{x}}(t)$ 及び報酬 $r(t) \in \Re$ は状態と出力の関数として

$$\dot{\mathbf{x}}(t) = F(\mathbf{x}(t), \mathbf{u}(t)) \quad (1.7)$$

$$r(t) = R(\mathbf{x}(t), \mathbf{u}(t)) \quad (1.8)$$

の様に与えられる環境を考える[1]．ここで，制御出力 $\mathbf{u}(t)$ は状態 $\mathbf{x}(t)$ をもとにする制御則 $\mathbf{u}(t) = \mu(\mathbf{x}(t))$ によって決定されるとする．連続時間での各状態における将来の報酬の重み付き期待値は，

$$V^{\mu}(\mathbf{x}(t)) = E\left[\int_t^{\infty} e^{-\frac{s-t}{\tau}} R(\mathbf{x}(s), \mathbf{u}(s)) ds\right] \quad (1.9)$$

と定義され，これが連続システムにおける状態価値関数となる．ここで $\tau$ は報酬の時間的重み付けを決める時定数である．強化学習の目標は式 (1.9) で表される価値関数が各状態についてなるべく大きくなるように制御則 $\mu$ を更新する事と定義される．価値関数は TD 誤差

$$\delta(t) \equiv R(t) + \frac{1}{\tau} V^{\mu}(\mathbf{x}) - \dot{V}(\mathbf{x}) \quad (1.10)$$

を誤差信号として逐次更新される．さらに TD 誤差は行動則 $\mu(t)$ を更新するためにも用いられる [13, 52].

[1] $N_s$，$N_c$ はそれぞれ状態空間，行動出力の次元数を表し，$\Re^{\bullet}$ は $\bullet$ 次元の縦ベクトルを表す．

# 3. モジュール強化学習

強化学習は学習対象の環境が複雑な場合，学習に必要な試行数が非現実的なほど膨大となる場合がある．また環境が非定常に変化する度に学習をやり直す必要があり効率が悪い．モジュール強化学習とは環境を複数個の局所領域に分割してそれぞれに独立した強化学習コントローラを配置する方式であり，各モジュールの担当範囲が局所領域に限定されるため学習に必要な施行数を抑えられ，また環境が切り替わるような場合でもモジュールを切り替えることで即座に対応できる利点がある．

## 3.1 Multiple Model-based Reinforcement Learning

Doya らは状態予測モデルと強化学習コントローラの対を 1 つのモジュールとしたモジュール強化学習方式 Multiple Model-based Reinforcement Learning(MMRL) を提案している [15, 46]．MMRL は MOSAIC モデルの強化学習版であり，逆モデルの学習法に強化学習を用いたものとなっている．良い予測を行った状態予測モデルほど高い値を持つ “責任信号” が計算され，その大きさに応じて状態予測モデルと強化学習コントローラの学習および制御出力の重み付けが行われる方式である．状態予測誤差に基づいた責任信号の計算とモジュールの学習を繰り返し行う事で個々のモジュールの分化が実現され，ダイナミクスが非定常に切り替わる環境下においても適切なモジュール選択が行える利点がある．

## 3.2 Modular reward

モジュール方式を用いる目的は局所的な学習器をそれぞれ個別に学習することで学習速度の向上や変動する環境への脱適応・再適応を行う点にあるが，その目的はしばしば大域的最適性と相反する要求となる．各モジュールの最適性が局所的なものに制限されているような方式は，例えばゴール付近でのみ報酬が与えられるような場合においてうまく機能しない．そこで Samejima らは擬似的な報酬 “modular reward” をモジュールに与える事で最適性を保障する方式を提案してい

る．この方式はモジュールが切り替わった際，切り替わり後のモジュールが獲得している価値を modular reward として切り替わり前のモジュールに与えると定式化されている．modular reward はモジュール切り替わり後にエージェントが受ける累積報酬を表しており，環境からの報酬と modular reward の 2 種類を用いる事で高速な学習と大域的最適性の 2 つが両立されるとしている．

# 4. 階層型強化学習

Singh は効率よく学習を行うためにはサブゴールの利用，探索の高効率化，状態表現の汎化などが解決されるべき課題であるとしている [35]．そして階層構造の導入はそれらの課題を解決するための有力な手段である．階層構造の導入により学習速度の改善が期待できるが，どの様な上位層を用いるかによって性能や特性が左右される．ここでは幾つかの階層型手法の紹介を行いそれぞれどの様な上位層を持っているのか，どの様な特徴があるのかを簡単に紹介する．

## 4.1 Morimoto 式階層型強化学習

Morimoto らは上位層が縮約された状態空間にてタスクの分割を行い，下位層が分割されたタスクの達成を行う階層型強化学習方式を提案している [29, 14]．上位層は状態空間の低次元化と離散化を行い，各離散状態に対する目標姿勢角を $Q(\lambda)$ 学習 [33] にて評価する．制御の各時刻において上位層が選択した目標姿勢角がサブタスクとなり下位層に伝えられ，下位層はそれを実現するための制御則を学習する．上位層には環境からの報酬が，そして下位層にはサブタスクの達成度合いに応じた報酬が与えられる．

## 4.2 Option

Sutton らは “option” と呼ばれるモジュールの一種を用いて行動の抽象化を行い，既存の強化学習理論を大きく変更することなく適用可能な枠組みを提案して

いる [39, 40]．各 option は状態の集合，方策そして終端確率の 3 者から構成されている．状態の集合は自分自身の担当範囲を決定しており，いったん選択された option は終端確率によって選択が解除されるまで獲得している方策に従った行動出力を行う．そして上位層は状態行動価値関数を学習し，各状態に対して累積報酬が最大となるような option を選択する．設定された終端確率に従って一定期間同じ方策を使い続ける事で行動の抽象化を行うのがこの手法の特徴である．この場合 option が切り替わる間隔は非定常となり，上位層は Semi-Markov Dicision Processe(SMDP) の環境で学習を行う事になる．通常の SMDP 環境では状態が切り替わる時にのみ学習が行われるが，終端確率による報酬予測値の期待値を用いて一時刻ごとに更新を行える点も特徴のひとつである．

## 4.3 Feudal Reinforcement Learning

Dayan らが提案した階層型強化学習方式は “Feudal Reinforcement Learning” と呼ばれ，上位ほど粗く分節化された状態空間を持ち，ある層の状態がひとつ下位における複数の状態を支配する層構造になっている [8]．ある層の状態は自分の支配者から命令を受け，それに応じて自分が支配している各状態に対して個別の命令を与える．その後，支配者から与えられる報酬によって自分が与えた命令の評価を行う．最上位層のみが環境からの報酬を受け，それ以外の層は上位層からの命令の達成度に応じた報酬を受ける．また最下位層が選択する命令が環境への行動出力となる．Dayan らは迷路課題に対して計 4 つの層を実装し，通常の強化学習手法よりも良い性能が得られたとしている．

## 4.4 Compositional Q-learning

Singh が提案した “Compositional Q-learning” は複数の状態行動価値関数を持ち，gating module が出力する重み付けの信号を用いてそれらを切り替える手法であり，サブタスクの時系列性に基づく階層構造を持った手法である [36]．“elemental task” と呼ばれるタスクの最少単位が存在し，それらの時系列により解くべきタスク (composite task) が構成されていると仮定した手法である．下位層には各

elemental task を担当する状態行動価値関数が配置され，上位層にはそれらを切り替える “gating module” が存在する．各状態行動価値関数は報酬予測誤差に基づいた尤度を持っており，事前分布による尤度の期待値を正規化したものがモジュールの選択確率として用いられる．学習は期待対数尤度の最大化を目的とし，gating module および各状態行動価値関数のパラメータは期待対数尤度をパラメータで偏微分した方向に更新される．予め各 elemental task に対する状態行動価値関数を学習しておけば composite task には gating module の学習のみ必要であり，新規の課題にも即座に対応できる利点がある．

### 4.5 MAXQ Value Function Decomposition

Dietterich は MAXQ と呼ばれる階層型強化学習を提案した [11]．MAXQ におけるタスク分割は対象となるタスクを頂点とし，上位ほど時間的に長く下位に行くほど短い時間区間で抽象化がなされたサブタスクが連結された木構造となっている．この木構造は “task graph” と呼ばれており，各サブタスクの上下関係を明示的に表現するためのものである．各サブタスクは局所的な方策，担当する局所領域，終端条件，擬似報酬の 4 者から構成されている．Sutton の option とよく似ているが，Sutton の階層型強化学習手法は上位の状態行動価値関数と下位の option 群という 2 階層構造であり，option 同士に上下関係はない．一方，MAXQ ではサブタスクが抽象化の粒度によって複数の層を構成しておりそれらの関係は task graph によって表されている．各層でどのサブタスクが有効になるかに応じて複数の方策を表現でき，それぞれに対する価値関数を個別に学習できる利点がある．反面，設計者の先見的知識によってタスクごとに task graph を設定しなければならない弱点がある．

## 5. マルチエージェントとコミュニケーション

近年ロボットの社会への進出が目覚しく，なかでも人とのコミュニケーション能力を売りにしたロボットの研究が盛んであり，工場内での作業のみならず我々

の日常生活を支援する場面が増えていく事が予想される．現時点においても音声認識により対話を行ったり画像認識により個人を識別したりする受付ロボットや人との協調作業を行うロボット，人のジェスチャを認識して行動するロボットなど様々なものが存在する．

シンボルを用いたコミュニケーションではシンボル生成の規則をエージェント間で統一しておく事により，環境の情報を交換したりお互いの状態を知ることができる．コミュニケーションシンボルに関する研究は大きく分けて，原始的なシンボルの発生メカニズムに関するものとシンボル同士の構造の進化メカニズムに関するものの二つ存在する．本節では前者に関する研究例を幾つか紹介する．

## 5.1 エージェント間で共通な体験

コミュニケーションが成立するためには共有信念の存在が必要不可欠となる．共有信念が存在するとは「A が命題 p を信じている」,「B が p を信じている」,「A と B が p を信じている」,「その事を A が信じている」と言う信念の入れ子構造が成立していることである．会話を例に挙げると，ある単語 p に対する意味づけが A と B で同じ必要があり，またその事を両者が信じる事が必要になる．共有信念が持つ無限の入れ子構造をいきなり扱うのは難しいため，まず環境の事象に対してエージェント間で共通な意味を持つシンボルをどう割り当てるかを考えると，エージェント間に共通体験が存在する必要がある [45]．言語体系が異なる 2 人のエージェントを考えると初めのうちは会話は成立しないが，環境の事象と未知の単語が同時に出現する頻度を測ることによりその単語の意味が徐々に理解できるものとされている．

Arita ら [1] は「敵」,「食料」と言った生命の維持に直結する事象へのシンボルの割り当てはある程度生得的に獲得されるとしている．ある集団において仲間が発したアラームの意味 (敵が来た，食料がある等) を理解できた固体のみが高い報酬を得られ，また次世代に遺伝子を残す事ができるという設定において，何世代か交代を繰り返すとその集団内に共通なシンボルが生まれる事をシミュレーションにより検証している．

Cangelosi はエージェントの学習アルゴリズムとして入力層・隠れ層・出力層

からなるネットワークを採用した実験を行っている [7]．入力層は環境の情報を入力する部分と他エージェントからの信号を入力する部分に分かれており，また出力層は行動出力を行う部分と他エージェントへ信号を送信する部分に分かれている．環境中には数種類のエサが配置されておりエージェントはそれらを獲得する事で報酬を得られるが，毒の入ったエサを食べると大きな負の報酬を得てしまう．エージェントはお互いに環境の情報を送信し合っており，他エージェントが送信してきた信号を理解できたエージェントは負の報酬を回避できる．高い報酬を得ていたエージェントが持つパラメータを次世代に伝える事で，数世代の世代交代により共通なシンボルが生成されたという結果が得られている．

## 5.2 連続な環境における共通なシンボルの生成

共通体験がシンボルの生成に欠かせない事は先行研究により示されているが，その多くは離散な環境を前提としている点に限界がある．実際の環境は常に連続な変数で表現され，それを設計者の都合により予め離散化を行ってしまうのでは本当の意味で記号接地問題 [17] を解決したとは言えない．エージェントがおかれている環境に応じて動的に離散化が行われるべきである．しかし連続な環境では各エージェントが持つ体験の共通部分を推定する事や他者が送信してきたシンボルの意味を推定する事は一般に不良設定問題となる．また状態の次元数が高い場合は観測された状態をそのまま扱う事は現実的ではなく，何らかの特徴量を用いる事が必要となる．これらの問題を解決する手段は大きく 3 種類に分けられる．

1 つ目は文脈情報を用いる手法である．他者が送信してきた信号にノイズや曖昧さが存在する場合，その意図を一意に解釈する事は難しい．そこで一定時間内における信号の時系列を用いる事が考えられる．Steels ら [37] はカメラを持った 2 人のエージェントに同じ状況を見せ，映像中の物体に名前を割り当てていくシミュレーションを行っている．送信側のエージェントは獲得している意味と語彙の対応表を用いて任意の対象の状況を送信し，受信側のエージェントは自分の対応表を用いて信号の解釈を試みる．そしてその解釈が誤っていたら対応表を更新するという流れを繰り返す．入力された映像は画像処理により離散的なシンボルへと変換されるが，その変換は多くの曖昧さを含むため扱いが難しい．そこで瞬

間的な情報ではなく一連の動作 (例えば「リンゴが動いた」等) に対する信号を送信しあう事により曖昧さが解決され，両者は物体の名前を共有できたという結果になっている．2 つ目は Hidden Markov Model(HMM) の利用である．野田は観測に隠れ状態が存在する場合の分節化を HMM によって解決している [54]．さらに分節化を行動則の変化にも応じて行う Merly 型 HMM を用いる事により，エージェントの状態のみならず意図をも分節化できるとしており，協調作業や見まね学習への応用が期待できる．Inamura らは連続版 HMM によって観測軌道の分節化とそのダイナミクスの認識を行う枠組みを提案している [22]．上位層は隠れ状態のダイナミクスの違いにより複数の動作を記憶し，任意の記憶した動作を再現できる．運動の認識と生成が同じ数学的モデルに基づいている点が特徴であり，ミメシス理論を実現する枠組みであるとしている．実ロボットを制御対象とすると状態空間が高次元となり分節化や特徴量の抽出が困難となるが，この手法はその様な問題を解決する手段として興味深い．3 つめは環境の状態予測モデルを用いた分節化である．状態予測モデルによる分節化は制御の各時刻においてどの状態予測モデルが最もダイナミクスの近似に貢献しているかに応じて分節化が行われ，一般的には Switching State-Space Models(SSM) として知られている [16]．Wolpert らはエージェントの制御アルゴリズムに MOSAIC モデルを採用する事で，自分の運動軌道の分節化だけではなく他者の運動の分節化も推定できるとしている [42]．MOSAIC モデルでは状態予測モデルと制御器の対が 1 つのモジュールを形成しており，どのモジュールが他者の運動軌道を最も良く説明するかを評価する．まず各モジュールの制御器に他者の運動軌道を入力する事で制御出力の推定値を得る．そしてそれぞれの制御出力の推定値を対となっている状態予測モデルに入力して，状態変化の推定値を得る．この状態変化の推定値は各モジュールが相手と同じ状況におかれた場合に自分の状態がどう変化するかを表している．両者の物理パラメータとモジュール構造に大きな差がないと仮定すれば，相手と同じ様な状態変化の出力を行ったモジュールを観測軌道を最も良く説明するものとして採用できる．この枠組みにより他者のモジュール系列を追従する見まね学習や階層化によるコミュニケーションシンボルの発現など，社会的・認知的な活動が実現されるとしている．

## 6. 研究背景

ロボットが実世界で働くにあたって全ての行動を予めプログラムしておく事は現実的に不可能であり，自律性の高い強化学習を基盤としたアルゴリズムがひとつの解決策として挙げられる．強化学習は学習速度の遅さや環境の非定常性への対応が弱点であり，それらを克服するためモジュール構造や階層構造を導入したアルゴリズムがいくつか提案されている．それらは環境を分節化する基準や必要となる事前知識の種類によってその特性が大きく左右される．我々の日常を見てみると多種多様なダイナミクスが存在しており，また行動の目標もその場その場で変化する．このようにダイナミクスと行動目標の両方が変化する状況を制御の対象としたモジュール強化学習理論は例が少なく，研究対象として十分に価値があるものと思われる．またダイナミクスが変化する環境下で働く階層型教科学習理論についても先行研究が少なく，有用な枠組みを提案する事が求められている．本研究ではそのようなモジュール，および階層型強化学習理論の開発を最初の目標とする．

強化学習の学習速度を改善するためにはモジュール構造や階層構造の導入の他に，学習済みのエージェントを教示者とする見まね学習も有効な手段である．見まね学習は教示者からどのような情報を得るかによってその性質が特徴付けられ，観測された情報をそのまま再現する事からその裏に隠れた高次の情報を推定するものまで様々な見まね学習が考えられる．観測された運動軌道と自分の状態の誤差をフィードバックする方式は最も単純なものとして考えられるが，フィードバックに時間遅れや観測ノイズがあると実現が困難になる．また一般に身体の物理パラメータには個体差があるため他者の最適軌道が自分にとっての最適軌道になるとは限らない．そこで高次の情報を用いて観測の曖昧さや個体差に起因する問題を軽減する事が考えられるが，観測軌道のみから高次の情報を推定する問題は一般に不良設定性を持つ．そこで本研究ではエージェントが過去の経験により獲得したモジュール構造を用いてその不良設定性を解決する枠組みを提案する．具体的にはエージェントの制御アルゴリズムとして提案する階層型強化学習理論が実装されていると仮定し，自分の上位層の状態と行動のうち他者の運動軌道を最も良く説明する組み合わせを推定するアルゴリズムを定式化する．

社会に進出するロボットを目指すためには学習速度の問題だけではなく，他のロボットやヒトとの相互作用をどう扱うかも問題となる．エージェント間に相互作用が存在する場合，他者から送信されてきたシンボルに応じた行動選択により協調的な振る舞いが可能となるが，シンボル生成の規則をエージェント間でどう共有するかという問題がある．複数のエージェントが存在する環境を考えると，各エージェントは各時刻において自分の内部状態に応じた運動軌道とシンボルを生成し，お互いにそれらを観測しあうという一般的な枠組みが考えられる．この時シンボル生成の規則がエージェント間で共通ならばある運動軌道に対して各エージェントは同じシンボルを生成するはずである．相手のシンボルと運動軌道を観測し，その誤差を減らすように内部状態からシンボルへの写像を更新していく事が求められる．そのためには運動軌道からそれを生成している内部状態を推定する必要があるが，シンボルがコミュニケーションにどう貢献するかはその推定手法の性能に左右される．そこで本研究で提案する階層型強化学習理論を用いてシンボル生成の規則を共有する枠組みを定式化しその性能を検証する．Wolpert らの推定手法 [42] と本研究で提案する内部状態の推定手法は本質的には同じであるが，Wolpert らの手法はエージェントの内部状態を環境の状態にのみ基づいて分節化している．一方，提案手法では行動の目標に関しても分節化を行うためより高い性能が期待できる．

# 7. 提案手法の概要

本研究ではエージェントの制御アルゴリズムは MOSAIC モデルに基づいたモジュール構造を持つと仮定する．そして観測された他者の運動軌道からモジュール構造を推定する事により，共通なシンボル生成の枠組みの一案を定式化する．一般に観測軌道のみからその生成に用いられたモジュール構造の推定は不良設定問題となるが，エージェントが過去の経験から獲得したモジュール構造を用いて他者の運動軌道を模擬する事によりその不良設定性が解決される．本稿ではまず MOSAIC モデルに基づいて連続な環境を分節化するモジュール強化学習方式 “combinatorial module-based reinforcement learning (CMRL)” の定式化を 2 章に

て行う．その手法は複数の状態予測モデルと報酬モデル，強化学習コントローラを持ち，予測誤差に基づいて 3 者を動的に組み合わせるアルゴリズムである．ダイナミクスと報酬関数が非定常に変化する環境であっても動的に適切な強化学習コントローラの選択が行える事を示す．そして 3 章にて CMRL に階層構造を導入した “Semi-Markov Switching State-Space Model-Based Reinforcement Learning (SSRL)” の定式化を行う．下位層が状態予測誤差に基づいて環境の分節化を行い，上位層が各分節における局所的な目標を選択すると言うアルゴリズムである．そして最後に 4 章にて共通なシンボル獲得のための枠組みを定式化する．エージェントは SSRL を制御アルゴリズムとし，制御の各時刻において選択されているモジュールに応じた運動軌道の生成とシンボルの発信を行える．各エージェントはお互いにシンボルの観測とモジュール構造の推定を繰り返す事により，各シンボルに対する意味付けを共有できる事を示す．

# 第2章　モジュール強化学習

## 1. モジュール強化学習

この章では “combinatorial module-based reinforcement learning (CMRL)” と名付けられたモジュール強化学習手法の定式化と実験を行う．強化学習は優れた特性を持つ学習理論であるが環境が複雑であると学習に必要な試行数は膨大なものとなり，また環境が非定常に変化する場合はその度に学習をやり直さなければならない．CMRL は環境をダイナミクスと報酬関数の特性によって複数の局所領域に分割し，複数の強化学習コントローラを動的に切り替える事でその欠点の克服を目標とする．

CMRL は複数の状態予測モデルと報酬予測モデル，強化学習コントローラの 3 種類のモジュール群から構成される (図 2.1)．複数の状態予測モデルと報酬予測モデルはそれぞれ局所的なダイナミクスと報酬信号を予測し，複数の強化学習コントローラは局所的な価値関数に基づいた局所最適制御を行う．CMRL による制御はまず環境の状態 $\mathbf{x}(t)$ と報酬 $r(t)$ を観測し MOSAIC モデルに従った環境の分節化を行う．その分節化は状態予測誤差と報酬予測誤差の 2 つの基準により行われる．そしてそれぞれの予測出力と分節化の結果をもとに，各強化学習コントローラの出力を重み付けする．重み付けされた制御出力は環境に与えられるとともに次の予測に用いられる．MOSAIC モデルは非線形・非定常な環境を適応的に分節化し，また各分節の活性度を滑らかに推定するため予測モデルや制御器の汎化性能を高められるのが特徴である [43].

これ以降では状態予測モデル，報酬予測モデル，強化学習コントローラを区別しやすくするためそれぞれのパラメータや変数を $f$，$r$，$c$ で修飾し，各分節は $i$, $j$，$k$ で番号付けを行う．まず 2 節にて 2 つの予測モデルによる環境の分節化を説

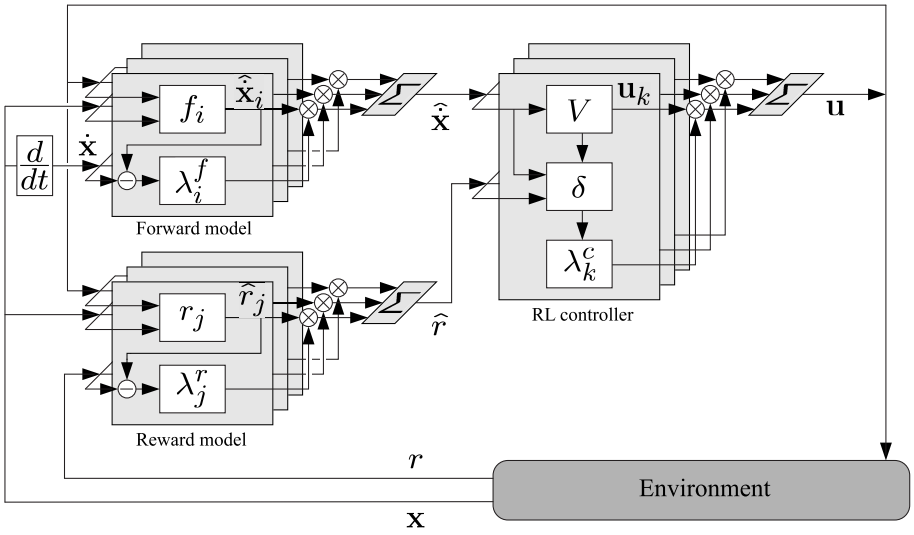
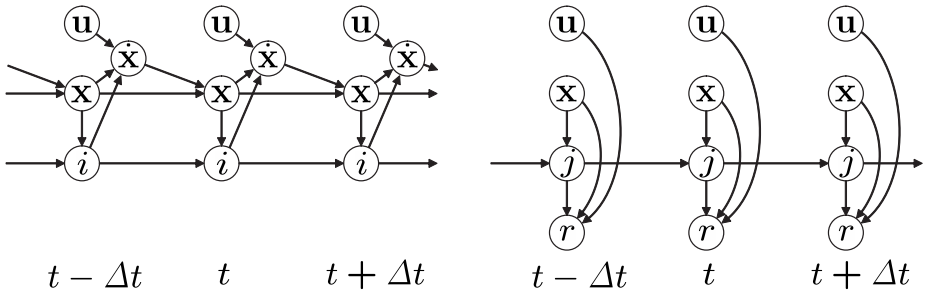



図 2.1 CMRL の概略図．左の二つが状態予測モデルと報酬予測モデル，右が強化学習コントローラを表す．2 つの予測モデルは環境の状態 $\mathbf{x}(t)$ と報酬 $r(t)$ を観測し，予測誤差に基づいて分節化を行う．強化学習コントローラはその分節化の結果に応じて出力の重み付けを決定する．

明し，3 節にて強化学習コントローラによる制御出力の方法について説明する．そしてシミュレーション実験により性能の検証を行う．




(a) ダイナミクスのグラフィカルモデル　　(b) 報酬のグラフィカルモデル

図 2.2 ダイナミクス・報酬のグラフィカルモデル．本稿で扱うシステムの時間 $t$ は連続であるが，分かりやすく離散的に表示した．状態変化 $\dot{\mathbf{x}}(t)$ は過去の時系列に依存するが，報酬 $r(t)$ は各時刻 $t$ の状態と出力にのみ依存する．

# 2. 複数の状態予測モデルと報酬予測モデルによる環境の分節化

この章では CMRL が仮定する環境の生成モデルについて述べた後で 2 つの予測モデルの説明を行い，その学習方法を説明する．制御の対象とする環境の状態変化 $\dot{\mathbf{x}}(t) \in \Re^{N_s}$ と報酬 $r(t) \in \Re$ は状態 $\mathbf{x}(t) \in \Re^{N_s}$ と制御入力 $\mathbf{u}(t) \in \Re^{N_c}$ によって決定されるとする．

$$\dot{\mathbf{x}}(t) = F(\mathbf{x}(t), \mathbf{u}(t)) + \mathbf{v}^f(t) \tag{2.1}$$

$$r(t) = R(\mathbf{x}(t), \mathbf{u}(t)) + v^r(t) \tag{2.2}$$

ここで $\mathbf{v}^f$ と $v^r$ はノイズであり，また $N_s$，$N_c$ はそれぞれ状態空間と制御信号の次元数である．このダイナミクスは $i \in \{1, ..., N^f\}$ で番号付けされる複数個の局所ダイナミクスに，また報酬は $j \in \{1, ..., N^r\}$ で番号付けされる複数個の局所報酬関数に分解できると仮定する．そして各時刻における状態 $\dot{\mathbf{x}}(t)$ と報酬 $r(t)$ はその時に活性化している局所ダイナミクスと局所報酬関数に従うものとする．複数の局所ダイナミクスが切り替わる環境のモデルは switching state-space models として知られている [16, 31]．CMRL ではダイナミクスだけではなく報酬関数も切り替わる様な環境のモデルを仮定している点に新規性がある．

## 2.1 生成モデル

まずダイナミクスの生成モデルを説明する．説明を簡単にするため，まず離散時間系 $s$ で定式化を行い最後に連続時間系 $t$ で再度定式化する．時刻 $s+1$ における状態 $\mathbf{x}(s+1)$ と隠れ変数 $i(s+1)$ の同時確率分布は一時刻前までに観測された状態と隠れ変数，制御入力の時系列を条件とした条件付確率分布となる．その条件付確率分布はベイズの法則を用いて以下の様に分解できる[1]．

$$\begin{aligned} &p(\mathbf{x}(s+1), i(s+1) \mid \mathbf{x}_{0:s}, \mathbf{u}_{0:s}, i_{0:s}) \\ &= p(\mathbf{x}(s+1) \mid i(s+1), \mathbf{x}_{0:s}, \mathbf{u}_{0:s}, i_{0:s}) P(i(s+1) \mid \mathbf{x}_{0:s}, \mathbf{u}_{0:s}, i_{0:s}) \end{aligned} \tag{2.3}$$

ここで右辺の一つ目の確率分布は $i$ 番目の局所領域におけるダイナミクスを表しているが，時刻 $s+1$ での状態は時刻 $s$ での状態と行動にのみ依存すると言う仮定から $p(\mathbf{x}(s+1) \mid i(s+1), \mathbf{x}(s), \mathbf{u}(s))$ とおく事ができる．また二つ目の確率分布は時刻 $s+1$ においてどの局所ダイナミクスが有効になるかを表しているが，その分布は一時刻前の状態と隠れ状態にのみ依存すると仮定すると $P(i(s+1) \mid \mathbf{x}(s), i(s))$ とおく事ができる．これらの変形により，ダイナミクスの生成モデルは最終的に以下の様になる

$$\begin{aligned} &p(\mathbf{x}(s+1), i(s+1) \mid \mathbf{x}_{0:s}, \mathbf{u}_{0:s}, i_{0:s}) \\ &= p(\mathbf{x}(s+1) \mid i(s+1), \mathbf{x}(s), \mathbf{u}(s)) P(i(s+1) \mid \mathbf{x}(s), i(s)) \end{aligned} \tag{2.4}$$

この生成モデルに従ったダイナミクスのグラフィカルモデルを図 2.2(a) に示す．CMRL は各局所ダイナミクスをそれぞれ担当する複数の状態予測モデル $f_i (i = 1, ..., N^f)$ を持ち，責任信号と呼ばれる隠れ変数の推定値によって各予測出力値を重み付けする．予測誤差に基づいて計算される責任信号によって非線形・非定常なダイナミクスを適応的に予測できる．

$$\hat{\mathbf{x}}(s+1) = \sum_{i=1}^{N^f} \lambda_i^f(s+1) \hat{\mathbf{x}}_i(s+1) \tag{2.5}$$

$$\hat{\mathbf{x}}_i(s+1) = f_i(\mathbf{x}(s), \mathbf{u}(s) \mid \phi_i^f(s)) \tag{2.6}$$

[1] $\bullet_{0:s}$ は $\bullet(0)$ から $\bullet(s)$ までの時系列を表す．

ここで $\phi_i^f$，$\lambda_i^f$ はそれぞれ状態予測モデルのパラメータ，責任信号を表す．また下付きの変数 $i$ がない場合には全てをまとめて表しているものとする（ $\phi^f = \{\phi_1^f, ..., \phi_{Nf}^f\}$，$\lambda^f = \{\lambda_1^f, ..., \lambda_{Nf}^f\}$）．

次に報酬の生成モデルについて説明する．時刻 $s+1$ における報酬 $r(s+1)$ と隠れ状態 $j(s+1)$ の同時分布は，一時刻前までに観測された状態と制御入力，隠れ状態の時系列を条件とした確率分布となる．報酬 $r(s+1)$ は一時刻前の状態 $\mathbf{x}(s)$ と行動 $\mathbf{u}(s)$，また隠れ状態 $j(s+1)$ は一時刻前の状態 $\mathbf{x}(s)$ と隠れ状態 $i(s)$ にのみ依存すると仮定すると報酬の生成モデルは以下の様に変形できる．

$$
\begin{aligned}
&p(r(s+1), j(s+1) \mid \mathbf{x}_{0:s}, \mathbf{u}_{0:s}, j_{0:s}) \\
&= p(r(s+1) \mid j(s+1), \mathbf{x}_{0:s}, \mathbf{u}_{0:s}, j_{0:s}) P(j(s+1) \mid \mathbf{x}_{0:s}, \mathbf{u}_{0:s}, j_{0:s}) \\
&= p(r(s+1) \mid j(s+1), \mathbf{x}(s), \mathbf{u}(s)) P(j(s+1) \mid \mathbf{x}(s), j(s))
\end{aligned} \tag{2.7}
$$

ここで右辺の一つ目の確率分布は時刻 $s+1$ における報酬，また二つ目の確率分布は隠れ状態 $j(s+1)$ の確率分布を表している．この生成モデルに基づいた報酬のグラフィカルモデルを図 2.2(b) に示す．CMRL は複数の報酬予測モデル $r_j$ によって各局所報酬関数を近似し，隠れ状態の推定値である責任信号を用いて重み付けを行う．

$$
\hat{r}(s+1) = \sum_{j=1}^{N^r} \lambda_j^r(s+1) \hat{r}_j(s+1) \tag{2.8}
$$

$$
\hat{r}_j(s+1) = r_j(\mathbf{x}(s), \mathbf{u}(s) \mid \phi_j^r(s)) \tag{2.9}
$$

ここで $\phi_j^r$，$\lambda_j^r$ はそれぞれ報酬予測モデルのパラメータと責任信号を表す．また下付きの変数 $j$ がない場合には全てをまとめて表しているものとする（ $\phi^r = \{\phi_1^r, ..., \phi_{Nr}^r\}$，$\lambda^r = \{\lambda_1^r, ..., \lambda_{Nr}^r\}$）．

CMRL では状態予測モデルと報酬予測モデルの予測性に基づいて分節化を行う．責任信号 $\lambda^f$，$\lambda^r$ の計算とその大きさに応じて予測モデルのパラメータ $\phi^f$，$\phi^r$ の学習を行う事を繰り返す事でダイナミクスと報酬関数が非定常に切り替わる環境を適応的に予測するモデルが得られる．次節では状態予測モデルと報酬予測モデルの責任信号の推定とパラメータの学習方法について説明する．

## 2.2 責任信号の推定

状態予測モデル，報酬予測モデルの責任信号は各時刻における隠れ状態 $i$，$j$ の推定値であり，それぞれ尤度と事前分布の積に比例したものとして求められる．状態予測モデルの尤度はパラメータ $\phi_i^f$ のもとで観測値 $\mathbf{x}(s+1)$ を，また報酬予測モデルの尤度はパラメータ $\phi_j^r$ のもとで観測値 $r(s+1)$ を予測する誤差に基づいて計算される．それぞれの予測誤差がガウス分布に従うと仮定した場合，各尤度は以下の様に決定される．

$$p(\mathbf{x}(s+1)|i(s),\mathbf{x}(s),\mathbf{u}(s),\phi_i^f(s)) = \frac{1}{(2\pi)^{D_s/2}{\sigma_i^f}^{D_s}} \exp\left[-\frac{1}{2{\sigma_i^f}^2}\|\hat{\mathbf{x}}_i(s+1)-\mathbf{x}(s+1)\|^2\right] \tag{2.10}$$

$$p(r(s+1)|j(s),\mathbf{x}(s),\mathbf{u}(s),\phi_j^r(s)) = \frac{1}{(2\pi)^{1/2}\sigma_j^r} \exp\left[-\frac{1}{2\sigma_j^{r2}}(\hat{r}_j(s+1)-r(s+1))^2\right] \tag{2.11}$$

ここで $p(\mathbf{x}(s+1)|i(s),\mathbf{x}(s),\mathbf{u}(s),\phi_i^f(s))$ が状態予測モデルの尤度，$p(r(s+1)|j(s),\mathbf{x}(s),\mathbf{u}(s),\phi_j^r(s))$ が報酬予測モデルの尤度を表しており，${\sigma_i^f}^2$，$\sigma_j^{r2}$ はそれぞれの予測誤差の分散である．事前分布は時刻 $s+1$ における状態と報酬 $\mathbf{x}(s+1)$，$r(s+1)$ を観測する前の情報から求められる分布である．生成モデルの定義より状態予測モデルの隠れ状態 $i(s+1)$ は 1 時刻前の状態 $\mathbf{x}(s)$ と隠れ状態 $i(s)$ に，また報酬予測モデルの隠れ状態 $j(s+1)$ は 1 時刻前の状態 $\mathbf{x}(s)$ と隠れ状態 $j(s)$ に依存すると仮定される．

$$\hat{\lambda}_i^f(s+1) = P(i(s+1) \mid \mathbf{x}(s), i(s), \phi_i^f(s)) \tag{2.12}$$

$$\hat{\lambda}_j^r(s+1) = P(j(s+1) \mid r(s), j(s), \phi_j^r(s)) \tag{2.13}$$

ここで $\hat{\lambda}_i^f(s+1)$，$\hat{\lambda}_j^r(s+1)$ が状態予測モデルと報酬予測モデルの責任信号の事前分布である．以上をまとめると責任信号はそれぞれ以下の様に定義される．

$$\lambda_i^f(s+1) \equiv \frac{p(\mathbf{x}(s+1) \mid i(s+1),\mathbf{x}(s),\mathbf{u}(s),\phi_i^f(s))\hat{\lambda}_i^f(s+1)}{\sum_{i'=1}^{Nf} p(\mathbf{x}(s+1) \mid i'(s+1),\mathbf{x}(s),\mathbf{u}(s),\phi_{i'}^f(s))\hat{\lambda}_{i'}^f(s+1)} \tag{2.14}$$

$$\lambda_j^r(s+1) \equiv \frac{p(r(s+1) \mid j(s+1),\mathbf{x}(s),\mathbf{u}(s),\phi_j^r(s))\hat{\lambda}_j^r(s+1)}{\sum_{j'=1}^{Nr} p(r(s+1) \mid j'(s+1),\mathbf{x}(s),\mathbf{u}(s),\phi_{j'}^r(s))\hat{\lambda}_{j'}^r(s+1)} \tag{2.15}$$

次に責任信号の事前分布 $\hat{\lambda}_i^f$，$\hat{\lambda}_j^r$ について説明する．事前分布は空間的局所性，時間的連続性と言った 2 つの仮定から推定される [15, 46]．各モジュールは状態空間上に担当範囲を持ち，その範囲内では選択確率が高くなると言う仮定に基づいたものが空間的局所性である．MOSAIC モデルに基づいたモジュール化は予測モデルの予測誤差に基づいて行われるが，状態空間が高次元になった場合にさらにモジュール化を促進するためにこの仮定が用いられる．そして責任信号は急激に変化しないという仮定に基づくものが時間的連続性である．モジュール学習においてモジュールの切り替わりが頻繁に起こってしまうと，個々のモジュールの学習が進まない上にモジュール間の干渉も起きてしまい好ましくない．その様な事態を回避する事が時間的連続性を用いる理由であり，1 時刻前のモジュールの選択状況に応じたり，尤度を時間的に平滑化したものにより実現される．ここで各状態予測モデルと予測モデルが持つ担当範囲がガウス分布に従い，また責任信号の時間的な滑らかさは平滑化された予測誤差に基づくものであるとすると事前分布はそれぞれ

$$\hat{\lambda}_i^f(s+1) = \frac{1}{(2\pi)^{(D_s+1)/2}|\Sigma_i^f|^{1/2}{\sigma_i^f}^{D_s}} \exp\left[-\frac{1}{2}(\mathbf{x}(s)-\mathbf{x}_i^f)^T{\Sigma_i^f}^{-1}(\mathbf{x}(s)-\mathbf{x}_i^f) - \frac{1}{2{\sigma_i^f}^2}\|\dot{\mathbf{x}}_i(s)-\dot{\mathbf{x}}(s)\|^2\right] \quad (2.16)$$

$$\hat{\lambda}_j^r(s+1) = \frac{1}{(2\pi)^{(D_s+1)/2}|\Sigma_j^r|^{1/2}\sigma_j^r} \exp\left[-\frac{1}{2}(\mathbf{x}(s)-\mathbf{x}_j^r)^T{\Sigma_j^r}^{-1}(\mathbf{x}(s)-\mathbf{x}_j^r) - \frac{1}{2{\sigma_j^r}^2}(\hat{r}(s)-r(s))^2\right] \quad (2.17)$$

と記述できる．ここで $\mathbf{x}_i^f$，$\mathbf{x}_j^r$ は担当範囲の中心を表し $\Sigma_i^f$，$\Sigma_j^r$ はその分散を表し，制御の各時刻において環境の状態と担当範囲の中心が近いものほど事前分布の値が高くなる．また 1 時刻前の予測誤差にも基づく事で時間的連続性を実現している．

## 2.3 連続時間系における生成モデルと責任信号の定式化

この節では離散時間系 $s$ で定式化してきた環境の生成モデルと責任信号の推定を連続時間系 $t$ へ拡張する．その場合，各状態予測モデルと報酬予測モデルの出力は各時刻における状態変化と報酬の予測値になる．

$$\hat{\dot{\mathbf{x}}}_i(t) = f_i(\mathbf{x}(t), \mathbf{u}(t) \mid \phi_i^f(t)) \tag{2.18}$$

$$\hat{r}_j(t) = r_j(\mathbf{x}(t), \mathbf{u}(t) \mid \phi_j^r(t)) \tag{2.19}$$

また責任信号は以下の様に定式化される．

$$\lambda_i^f(t) \equiv \frac{p(\dot{\mathbf{x}}(t) \mid i(t), \mathbf{x}(t), \mathbf{u}(t), \phi_i^f(t))\hat{\lambda}_i^f(t)}{\sum_{i'=1}^{Nf} p(\dot{\mathbf{x}}(t) \mid i'(t), \mathbf{x}(t), \mathbf{u}(t), \phi_{i'}^f(t))\hat{\lambda}_{i'}^f(t)} \tag{2.20}$$

$$\lambda_j^r(t) \equiv \frac{p(r(t) \mid j(t), \mathbf{x}(t), \mathbf{u}(t), \phi_j^r(t))\hat{\lambda}_j^r(t)}{\sum_{j'=1}^{Nr} p(r(t) \mid j'(t), \mathbf{x}(t), \mathbf{u}(t), \phi_{j'}^r(t))\hat{\lambda}_{j'}^r(t)} \tag{2.21}$$

尤度は予測誤差を用いて以下の様に定義される．

$$p(\dot{\mathbf{x}}(t)|i(t), \mathbf{x}(t), \mathbf{u}(t), \phi_i^f(t)) = \frac{1}{(2\pi)^{D_s/2}{\sigma_i^f}^{D_s}} \exp\left[-\frac{1}{2{\sigma_i^f}^2}\|\hat{\dot{\mathbf{x}}}_i(t) - \dot{\mathbf{x}}(t)\|^2\right] \tag{2.22}$$

$$p(r(t)|j(t), \mathbf{x}(t), \mathbf{u}(t), \phi_j^r(t)) = \frac{1}{(2\pi)^{1/2}\sigma_j^r} \exp\left[-\frac{1}{2{\sigma_j^r}^2}(\hat{r}_j(t) - r(t))^2\right] \tag{2.23}$$

また責任信号の事前分布は

$$\hat{\lambda}_i^f(t) = \frac{1}{(2\pi)^{(D_s+1)/2}|\Sigma_i^f|^{1/2}{\bar{\sigma}_i^f}^{D_s}} \exp\left[-\frac{1}{2}(\mathbf{x}(t) - \mathbf{x}_i^f)^T {\Sigma_i^f}^{-1}(\mathbf{x}(t) - \mathbf{x}_i^f) - \frac{1}{2{\bar{\sigma}_i^f}^2}E_i^f(t)\right] \tag{2.24}$$

$$\hat{\lambda}_j^r(t) = \frac{1}{(2\pi)^{(D_s+1)/2}|\Sigma_j^r|^{1/2}\bar{\sigma}_j^r} \exp\left[-\frac{1}{2}(\mathbf{x}(s) - \mathbf{x}_j^r)^T {\Sigma_j^r}^{-1}(\mathbf{x}(s) - \mathbf{x}_j^r) - \frac{1}{2{\bar{\sigma}_j^r}^2}E_j^r(t)\right] \tag{2.25}$$

となる．ここで $E_i^f(t)$，$E_j^r(t)$ は状態予測誤差と報酬予測誤差を時間的に平滑化したものであり，${\bar{\sigma}_i^f}^2$，${\bar{\sigma}_j^r}^2$ はそれらの分散を表している．

## 2.4 パラメータの学習

予測モデルの効果的な学習則はEMアルゴリズム [10] より与えられる．EMアルゴリズムは隠れ状態の確率分布を推定するEステップとパラメータの尤度を最大化するMステップを繰り返す手法である．責任信号は各値が非ゼロでかつ和が1となる信号であるため隠れ状態の確率分布ととらえる事ができる．つまり責任信号の計算をEステップとみなす事でMステップをパラメータの更新に用いられる．一時刻ごとに責任信号の計算とパラメータの更新を行う事で逐次的に更新が行える．以下は時刻 $t$ において環境の状態 $\mathbf{x}(t)$ が観測された時の更新手順である．

**Eステップ**　時刻 $t$ における更新前のパラメータ $\phi^{f\prime}(t)$，$\phi^{r\prime}(t)$ に対する責任信号 $\lambda^f(t \mid \phi^{f\prime}(t))$，$\lambda^r(t \mid \phi^{r\prime}(t))$ を求める．

**Mステップ**　時刻 $t$ での観測データに対する期待対数尤度は以下の様に定義される．

$$L(\phi^f \mid \phi^{f\prime}(t)) = \sum_{i=1}^{N^f} \lambda_i^f(t \mid \phi^{f\prime}(t)) p(\mathbf{x}(t), i(t) \mid \phi^f) \tag{2.26}$$

$$L(\phi^r \mid \phi^{r\prime}(t)) = \sum_{j=1}^{N^r} \lambda_j^r(t \mid \phi^{r\prime}(t)) p(r(t), j(t) \mid \phi^r) \tag{2.27}$$

この期待対数尤度が上昇する方向にパラメータを更新する．

# 3. 強化学習コントローラ

この節では強化学習コントローラの定式化を行い，制御出力をどのように重み付けして環境に出力するかを説明する．$k \in \{1, ..., N^c\}$ 番目の強化学習コントローラは $\phi_k^c(t)$ をパラメータとする局所的な価値関数 $V_k(\mathbf{x}(t) \mid \phi_k^c(t))$ を持ちその勾配に基づいて制御信号 $\mathbf{u}_k(t)$ を出力する．グリーディな方策での制御出力は状態予測モデルの勾配と出力のコスト関数を用いて決定さる．そうして出力された各制

御信号は責任信号によって重み付けされ環境に出力される．

$$\mathbf{u}(t) = \sum_{k=1}^{N^c} \lambda_k^c(t)\mathbf{u}_k(t) \tag{2.28}$$

$$\mathbf{u}_k(t) = S'^{-1}\left(\left(\frac{\partial\hat{\dot{\mathbf{x}}}(t)}{\partial\mathbf{u}}\right)^T\left(\frac{\partial V_k(\mathbf{x}(t)\mid\phi_k^c(t))}{\partial\mathbf{x}}\right)^T\right) \tag{2.29}$$

ここで $\lambda_k^c(t)$ が強化学習コントローラの責任信号，$S'^{-1}()$ はコスト関数を出力 $\mathbf{u}$ で偏微分したものの逆関数である．次節ではこの責任信号 $\lambda_k^c(t)$ の求め方とパラメータ $\phi_k^c(t)$ の更新方法を述べる．

## 3.1 責任信号の推定

強化学習コントローラの責任信号 $\lambda_k^c(t)$ は局所価値関数 $V_k(\mathbf{x}(t)\mid\phi_k^c(t))$ の尤度と事前分布 $\hat{\lambda}_k^c(t)$ の積に比例して推定される．

$$\lambda_k^c(t) = \frac{p(V(t)\mid k(t), r(t), \mathbf{x}(t), \dot{\mathbf{x}}(t), \phi_k^c(t))\hat{\lambda}_k^c(t)}{\sum_{k'=1}^{N^c} p(V(t)\mid k'(t), r(t), \mathbf{x}(t), \dot{\mathbf{x}}(t), \phi_{k'}^c(t))\hat{\lambda}_{k'}^c(t)} \tag{2.30}$$

ここで $p(V(t)\mid k(t), r(t), \mathbf{x}(t), \dot{\mathbf{x}}(t), \phi^c(t))$ が尤度，$\hat{\lambda}_k^c(t)$ が責任信号の事前分布を表す．この尤度は局所価値関数の近似の良さに応じて決定され，その近似の良さは TD 誤差 [38, 13] を用いて評価できるとする．各局所価値関数 $V_k(\mathbf{x}(t)\mid\phi_k^c(t))$ の TD 誤差 $\delta_k(t)$ が分散 $\sigma_k^{c2}$ の正規分布に従うとすると尤度は

$$p(V(t)\mid k(t), r(t), \mathbf{x}(t), \dot{\mathbf{x}}(t), \phi_k^c(t)) = \frac{1}{(2\pi)^{1/2}\sigma_k^c}\exp\left[-\frac{1}{2\sigma_k^{c2}}\delta_k(t)^2\right] \tag{2.31}$$

$$\delta_k(t) = \hat{r}(t) - \frac{1}{\tau}V_k(t) + \frac{\partial V_k(\mathbf{x}(t)\mid\phi_k^c(t))}{\partial\mathbf{x}}\hat{\dot{\mathbf{x}}}(t) \tag{2.32}$$

となる．ここで $0 < \tau < \infty$ は時定数である．TD 誤差の計算 (式 (2.32)) において，観測された状態変化 $\dot{\mathbf{x}}(t)$ と報酬 $r(t)$ ではなく予測モデルの出力値 $\hat{\dot{\mathbf{x}}}(t)$，$\hat{r}(t)$ を用いる事により，観測にノイズや時間遅れ，隠れ状態がある課題にも対処可能となる．

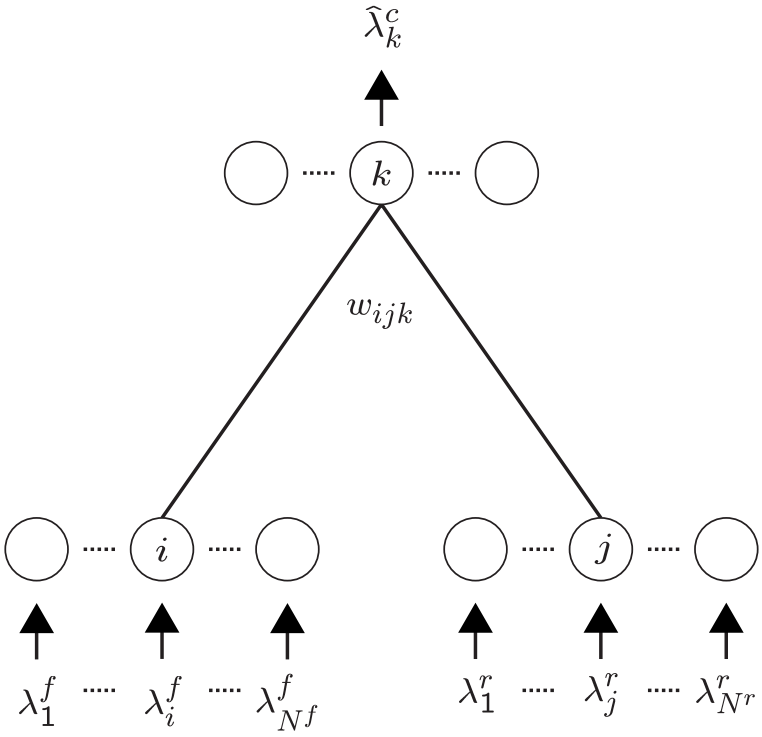



図 2.3 Prior network of RL-controller.

次に事前分布について説明する．各強化学習コントローラの尤度は TD 誤差により決定されるが，TD 誤差は一般にノイズの影響を受けやすく分散の大きな信号である．また価値関数の学習が十分でない段階においては TD 誤差自体が信頼できないため，尤度のみに頼って責任信号を決定する事は危険である．そこで図 2.3 のようなネットワークを考え，その出力を強化学習コントローラの事前分布として採用する．このネットワークはある状態予測モデルと報酬予測モデルの組に対して頻繁に選択されていた強化学習コントローラほど高い事前確率を出力するネットワークである．状態予測モデルの責任信号 $\lambda^f$ と報酬予測モデルの責任信号 $\lambda^r$ が入力，事前分布 $\hat{\lambda}_k^c$ が出力であり，$i$，$j$ 番目の入力ノードと $k$ 番目の出力ノードが荷重 $w_{ijk}$ で結合される．頻繁に選択されている状態予測モデル，報酬予測モデル，強化学習コントローラの組ほど結合荷重 $w_{ijk}$ が大きくなるように更新することで，経験に基づいた事前分布が出力されるようになる．時刻 $t$ におけ

る責任信号が $\lambda^f(t)$，$\lambda^r(t)$，$\lambda^c(t)$ である時，その様な更新は例えば以下の様に行う事で可能となる．

$$\dot{w}_{ijk}(t) = \begin{cases} 1 - \alpha w_{ijk}(t) & (if \;\; i = \mathrm{argmax}_i\, \lambda^f(t), j = \mathrm{argmax}_j\, \lambda^r(t)) \\ -\alpha w_{ijk}(t) & (otherwise) \end{cases} \tag{2.33}$$

ここで $\alpha$ は $w_{ijk}$ を発散させないためのパラメータである．その様にして更新される結合荷重 $w_{ijk}$ を用いて，$k$ 番目の出力 $\lambda^c_k(t)$ を以下の様に決定する．

$$\hat{\lambda}^c_k(t) = \frac{\sum_{i=1}^{N^f}\sum_{j=1}^{N^r}\lambda^f_i(t)\lambda^r_j(t)\exp(w_{ijk})}{\sum_{i=1}^{N^f}\sum_{j=1}^{N^r}\sum_{k'=1}^{N^c}\lambda^f_i(t)\lambda^r_j(t)\exp(w_{ijk'})} \tag{2.34}$$

## 3.2 局所価値関数の学習

各強化学習コントローラが持つ局所価値関数は TD 誤差を最小にするように更新される．各局所価値関数を局所的な報酬最大化に限定してよい場合は以下の様に責任信号によって重み付けされた TD 誤差の二乗値を評価関数とする．

$$e_k(t) = \frac{1}{2}\lambda^c_k(t)|\delta_k(t)|^2 \tag{2.35}$$

$$\dot{\phi}^c_k(t) = -\eta^c\frac{\partial e_k(t)}{\partial \phi^c_k(t)} \tag{2.36}$$

ここで $\eta^c$ は学習係数である．

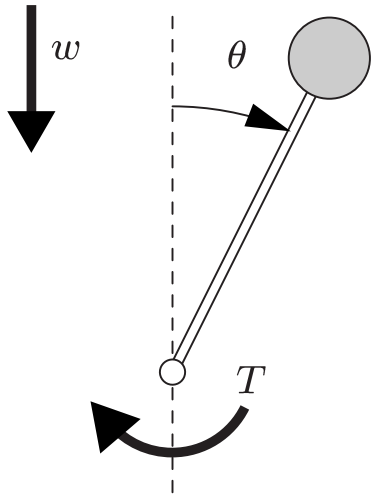



図 2.4　実験の制御対象とした単振り子 ($m = 0.5$[kg], $l = 1$[m], $\mu = 0.01$[kg $\cdot$ m$^2$/sec]). 振り子の角度 $\theta$ は 0[rad] を倒立状態，時計回りを正とした. 制御入力 $T$[Nm] の上限値は 10[N] とした.

## 4. 実験：非定常に変化するダイナミクスと報酬

モジュール構造をもつアルゴリズムの利点は非定常に変化する環境や新規の環境への柔軟性である. 特に CMRL はダイナミクスと報酬のモデルをそれぞれ個別に切り替えられるため，その両方が変化する環境下で最も効果的に働く. この章ではダイナミクスと報酬がそれぞれ異なる環境をランダムに切り替えた時，CMRL のモジュールがどのように適応するのかを検証する. そして新規の環境へどう対応するのかも検証する.

制御対象は図 2.4 のような単振り子を用いる. 状態変数及び制御変数は

$$\mathbf{x}(t) = [\theta(t) \;\; \dot{\theta(t)}]^T \tag{2.37}$$

$$\mathbf{u}(t) = T(t) \tag{2.38}$$

とした. 振り子をぶら下がった状態 ($\theta(0) = \pi$[rad]) から倒立の状態 ($\theta(0) = 0$[rad]) へ到達させる非線形な制御課題である.

環境には上下に風が吹き，その変化により 2 種類のダイナミクス $(F_d, F_u)$ が存在する．ダイナミクス $F_d$ では下方向に 5[N] の力が振り子の先端にかかり，ダイナミクス $F_u$ では上方向に 5[N] の力が振り子の先端にかかる設定とした．また報酬関数も 2 種類存在し $(R_a, R_b)$，報酬関数 $R_r$ では右斜めの角度 $\theta = \frac{45}{180}\pi$[rad]，報酬関数 $R_l$ では左斜めの角度 $\theta = -\frac{45}{180}\pi$[rad] で最大の報酬となる．

状態予測モデル，報酬予測モデルにはそれぞれ線形モデルと 2 次モデルを採用した．また強化学習コントローラの価値関数は関数近似器を用いて学習し，その勾配に基づいて出力を行うとした．それぞれの詳細を以下に示す．

**状態予測モデル**　状態予測モデルの関数系には課題の目的に応じて様々なものを用いられるが，機械学習の分野では定式化と解析の簡便さから線形モデルが頻繁に用いられる．そこでこの実験でも以下の様な線形モデルを用いた

$$f_i(\mathbf{x}(t), \mathbf{u}(t)) = A_i(\mathbf{x}(t) - \mathbf{x}_i^f) + B_i\mathbf{u}(t) + c_i \tag{2.39}$$

ここで $A_i \in \Re^{N_s \times N_s}$，$B_i \in \Re^{N_s \times N_c}$ はパラメータ行列，また $c_i \in \Re^{N_s}$ はバイアス項を表す．各状態予測モデルの尤度は，予測誤差が分散 ${\sigma_i^f}^2$ のガウス分布に従うものとして式 (2.22) を採用した．責任信号の事前分布は用いず，$\hat{\lambda}_i^f(t) = \frac{1}{N^f}$ とした．以上をまとめると学習するべき状態予測モデルのパラメータは $\phi_i^f = \{A_i, B_i, c_i, \mathbf{x}_i^f$ となる．

**報酬予測モデル**　制御の分野では線形ダイナミクスのもとで 2 次のコストを用いた理論や解析の研究が数多くなされてきた．報酬予測モデルも任意の関数系を用いられるが，以下の様な 2 次形式を用いる事で制御の世界で築かれてきた様々な理論を応用できる．

$$r_j(\mathbf{x}(t), \mathbf{u}(t)) = -(\mathbf{x}(t) - \mathbf{x}_j^r)^T Q_j(\mathbf{x}(t) - \mathbf{x}_j^r) - \mathbf{u}(t)^T R_j \mathbf{u}(t) + q_j \tag{2.40}$$

責任信号の事前分布には時間的連続性，空間的局所性に基づくものを用いた．ここで $Q_j \in \Re^{N_s \times N_s}$，$R_j \in \Re^{N_c \times N_c}$ はパラメータ行列，また $q_j$ はバイアス項である．責任信号の事前分布は空間的局所性と時間的連続性の両方を用いた．各報酬予測モデルの担当範囲は $\mathbf{x}_j^r$ を中心とし分散 $\Sigma_j^r$ のガウス分布とし，また平滑化された

予測誤差の分散が $\bar{\sigma}_j^{r\,2}$ であるとして式 (2.17) を採用した．以上より学習するべき報酬予測モデルのパラメータは $\phi_j^r = \{Q_j, R_j, q_j, \mathbf{x}_j^r, \sigma_j^{r\,2}, \Sigma_j^r, \bar{\sigma}_j^{r\,2}\}$ となる．

**強化学習コントローラ**　各強化学習コントローラは価値関数 $V_k$ を学習し，その勾配をもとに制御出力を行う．それぞれの価値関数 $V_k$ は normalized gaussian network (NGNet) [28] を用いて近似した．NGNet の基底は $15 \times 15$ 個を $[-\pi \sim \pi \ \ -3\pi \sim 3\pi]^T$ の範囲に格子状に配置した．

$$V_k(\mathbf{x}(t)) = \sum_{s=1}^{N^w} w_{ks}(\mathbf{x}(t)) b_{ks} \tag{2.41}$$

$$w_{ks}(\mathbf{x}(t)) = \frac{\exp\left[-\frac{1}{2}(\mathbf{x}(t) - \mu_{ks})^T \Sigma_{ks}^w (\mathbf{x}(t) - \mu_{ks})\right]}{\sum_{s'=1}^{N^w} \exp\left[-\frac{1}{2}(\mathbf{x}(t) - \mu_{ks'})^T \Sigma_{ks'}^w (\mathbf{x}(t) - \mu_{ks'})\right]} \tag{2.42}$$

ここで $N^w$ は k 番目の強化学習コントローラの近似に用いた基底の個数で，$w_{ks}(\mathbf{x}(t))$, $b_{ks}$ はそれぞれ s 番目の基底関数とその重み係数である．

各基底の重み係数 $b_{ks}$ は exponential eligibility trace[13] を用いて学習し，その際の学習係数は 10，次定数を 0.5，適格度係数を 0.1 とした．

## 4.1 状態予測モデルと報酬予測モデルの学習

まず状態予測モデルと報酬予測モデルの学習を行った．ランダムな初期状態と制御入力により生成したサンプルデータをもとにオンラインで学習を行い，強化学習コントローラは用いなかった．学習の手順は以下の通りである．

- 出力として Ornstein-Uhlenbeck 型 Brown 運動を用いた．

$$\dot{T}(t) = -T(t) + \mathcal{N}(0, 10)$$

- 1 試行の長さは 0.5[sec] とし，計 1000 試行行った．
- 最初の 500 試行はダイナミクス $F_d$ と報酬関数 $R_r$ を提示し，それぞれ状態予測モデル $f_1$ と報酬予測モデル $r_1$ を用いて近似を行った．
- 501 試行目において状態予測モデル $f_2$ と報酬予測モデル $r_2$ を投入した．

- 後半の 500 試行は組み合わせ $(F_d,\ R_r)$ と $(F_u,\ R_l)$ を 1 試行毎にランダムに切り替えながら提示し学習を行った．

状態予測モデルの学習結果を図 2.5(a)，報酬予測モデルの学習結果を図 2.5(b) に示す．

## 4.2 非定常な環境での強化学習

複数個の強化学習コントローラが変化する環境と新規の環境に対してどのように適用するのかを調べる実験を行った．環境は 2 種類 $(E_{dr}$，$E_{ul})$ を用意し，環境 $E_{dr}$ はダイナミクス $F_d$ と報酬関数 $R_r$，環境 $E_{ul}$ はダイナミクス $F_u$ と報酬関数 $R_l$ の組み合わせとした．最初の 2500 試行は環境 $E_{dr}$ と環境 $E_{ul}$ をランダムに提示し，強化学習コントローラの学習を行い，2501 試行目移行は新たな環境環境 $E_{ur}$ を生成し，3 種類をランダムに切り替えた．環境 $E_{ur}$ はダイナミクス $F_u$ と報酬関数 $R_r$ の組み合わせである．状態予測モデルと報酬予測モデルは 4.1 節で学習済みのものを用い，それらのパラメータは実験中において固定とした．また 1 試行の長さは 20[sec]，シミュレーションの時間幅は 0.02[sec] とした．

**結果 1: コントローラの数が十分に多い場合**　まずコントローラの数が変化する環境の個数に比べて十分に多い場合 $(N^c = 5)$ の結果を示す．図 2.6(a) は各環境における累積報酬の変化を表しており，10 回の実験の結果を平均して表示してある．図 2.6(b) は各コントローラの使用率の例を表しており，各環境にそれぞれ 1 つずつのコントローラが担当している事が見て取れる．

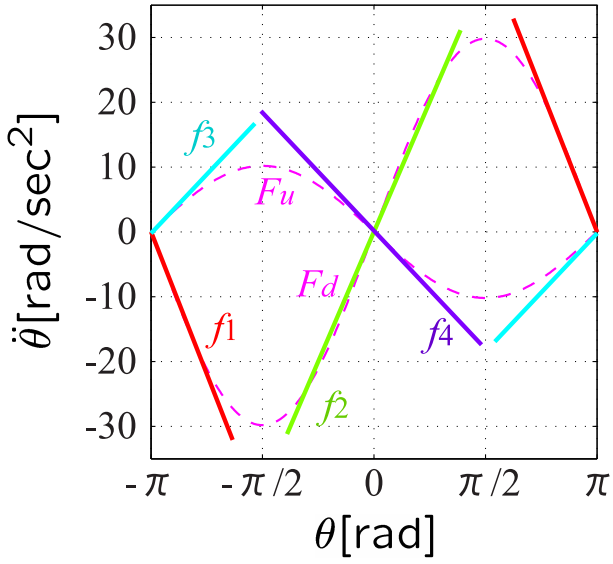
**結果 2: コントローラの数が少ない場合**　まずコントローラの数が少ない場合 $(N^c = 2)$ の結果を示す．実験 1 と同じく，図 2.7(a) は各環境における累積報酬の変化を表しており，10 回の実験の結果を平均して表示してある．図 2.7(b) は各コントローラの使用率の例を表しており，最初に提示される 2 つの環境には 1 つずつのコントローラが担当しているが，途中で現れる環境 $E_{ba}$ には 2 つのコントローラが半分ずつ対応している事が見て取れる．

## 4.3 考察

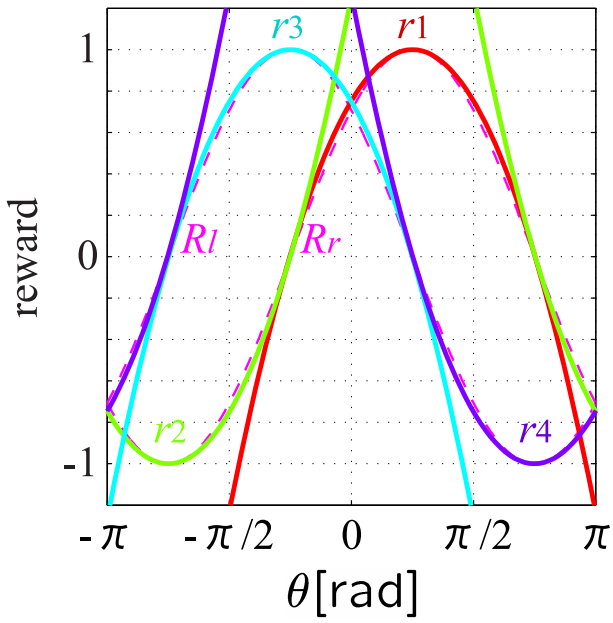
モジュール学習の利点は環境の変化に応じて制御則を切り替えられる点にある．コントローラの数が十分に多い場合は各環境にそれぞれ一つずつのコントローラが担当し，制御が行える事を示した．また新規の環境が提示された場合，それまで選択されていなかったコントローラが新たに担当する結果となり，非定常な環境に対する提案手法の柔軟性を示す事が出来た．

またコントローラの数が少ない場合は新規の環境に対し，既存のコントローラの出力を足し合わせる事で制御を行う結果となった．このことは TD 誤差でコントローラの環境に対する貢献度を決めると言う，提案手法の利点を最大限に活用した結果であると言える．

2 つの実験を比較すると，環境 $E_{dr}$ ではコントローラ数が少ない方が高い性能を示している．図 4.2(b) を見るとこの環境を担当するコントローラが固定されていない事が見てとれる．コントローラ数が多い場合はコントローラの担当範囲が固定されないため，結果として学習が進まなくなっていると予想される．

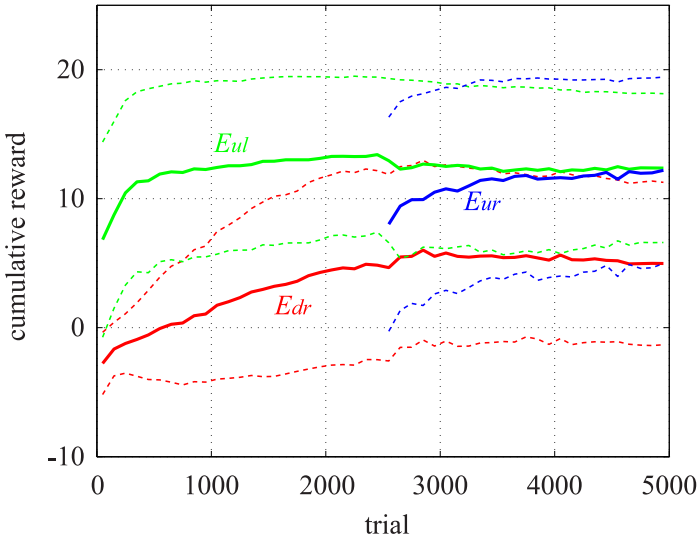



(a)

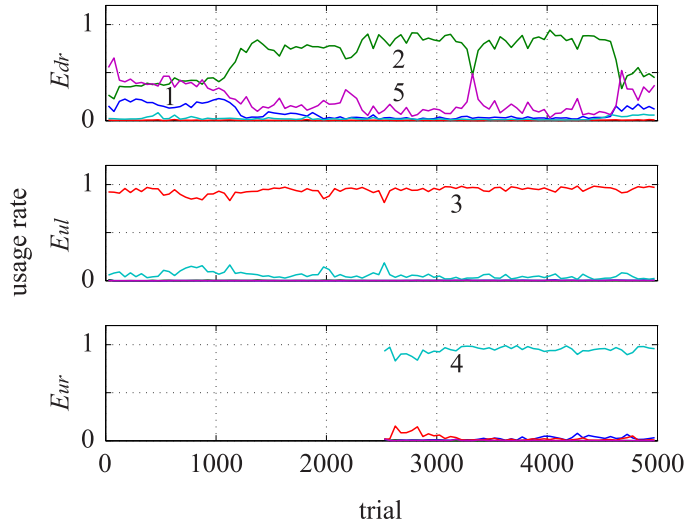



(b)

図 2.5　(a) が状態予測モデルによるダイナミクスの近似，(b) が報酬予測モデルによる報酬関数の近似を表す.

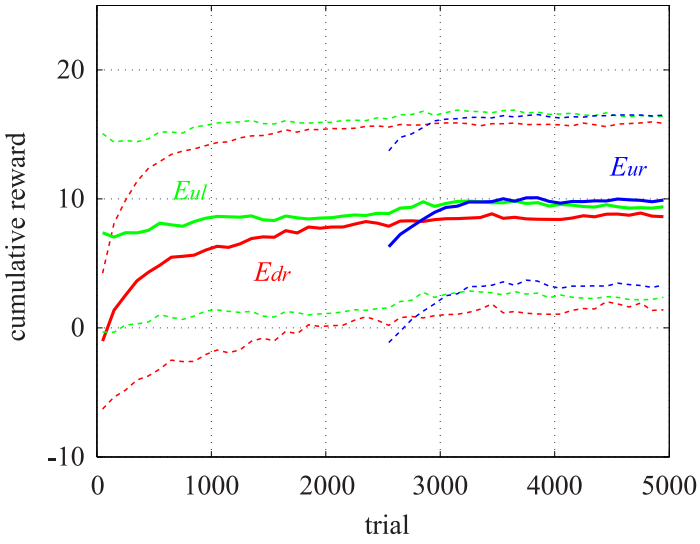



(a) それぞれの環境における累積報酬の変化．赤が環境 $E_{dr}$，緑が環境 $E_{ul}$，青が環境 $E_{ur}$ での結果を表している．

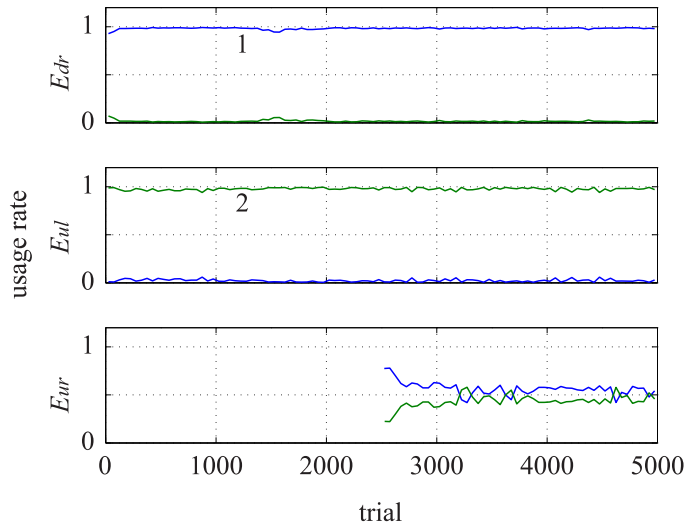



(b) 上から順に環境 $E_{dr}$，$E_{ul}$，$E_{ur}$ において各コントローラが使用されていた割合を示している．

図 2.6　(a) は各環境における累積報酬の変化，(b) はそれぞれの環境においてどのコントローラが用いられていたかを示している．




(a) それぞれの環境における累積報酬の変化．赤が環境 $E_{dr}$，緑が環境 $E_{ul}$，青が環境 $E_{ur}$ での結果を表している．比較のためコントローラの数が多い場合の結果を細い破線にて表示してある．




(b) 上から順に環境 $E_{dr}$，$E_{ul}$，$E_{ur}$ において各コントローラが使用されていた割合を示している．

図 2.7　(a) は各環境における累積報酬の変化，(b) はそれぞれの環境においてどのコントローラが用いられていたかを示している．

# 5. 二足歩行制御

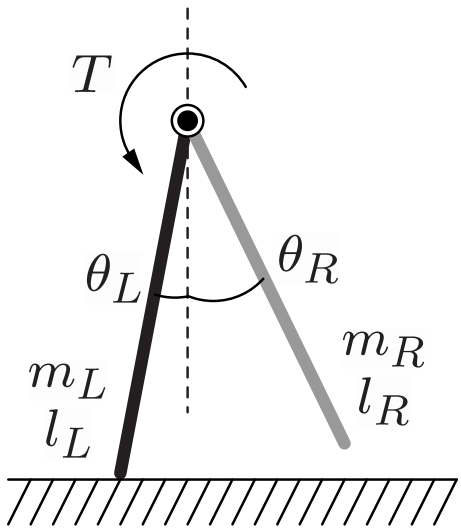



図 2.8 二足歩行ロボット．二つのリンクから構成される ($D_s = 4, D_c = 1$)．遊脚を前に振り出すときは地面に衝突しないようにした．回転角は時計回りを正とした．足の質量は $m_L = m_R = 0.5$[kg]，長さは $l_L = l_R = 0.1016$[m] とした．また出力の上限は $|T| \leq 2$[N] とした．

最後に二足歩行ロボットの制御に挑戦する．制御対象は図 2.8 の様に二つのリンクから構成されるものを用いた．遊脚を前に出す時は地面と接触しないように設定した．左右の足の角度はそれぞれ $\theta_L, \theta_R$ とし，股関節に加わるトルクを $T$ とした ($\mathbf{x} = [\theta_L \ \theta_R \ \dot{\theta}_L \ \dot{\theta}_R]^T$，$\mathbf{u} = T$)．

2 種類の報酬関数 $R_a$，$R_b$ を用意し，左足が接地していれば $R_a$ を，右足が設置していれば $R_b$ を提示した．

$$R_a(\mathbf{x}(t), \mathbf{u}(t)) = -\left(\theta_L(t) + \frac{30}{180}\pi\right)^2 - \left(\theta_R(t) - \frac{15}{180}\pi\right)^2 - 0.05T(t)^2 \quad (2.43)$$

$$R_b(\mathbf{x}(t), \mathbf{u}(t)) = -\left(\theta_L(t) - \frac{15}{180}\pi\right)^2 - \left(\theta_R(t) + \frac{30}{180}\pi\right)^2 - 0.05T(t)^2 \quad (2.44)$$

また両足が浮いている場合の報酬は 0 とした．

### 状態予測モデルと報酬予測モデル

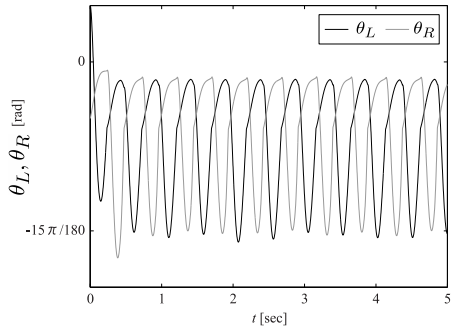
状態予測モデルは式 (2.39) の様な線形モデルを 20 個，報酬予測モデルは式 (2.40) の様な 2 次形式を 2 個用意した．ランダムな初期状態と制御入力による試行を繰り返す事で状態変化と報酬の系列を生成し，それをもとに各パラメータ $\phi_i^f$，$\phi_j^r$ の学習を行った．生成した全てのデータに対して責任信号 $\lambda_i^f(t)$，$\lambda_j^r(t)$ を求め，期待対数尤度を最大化する作業をパラメータが十分に収束するまで行った．その詳細は付録 A にまとめた．状態予測モデル，報酬予測モデルともに責任信号の事前分布は用いなかった ($\hat{\lambda}_i^f(t) = 1/N^f$，$\hat{\lambda}_j^r(t) = 1/N^r$)．

### 強化学習コントローラ

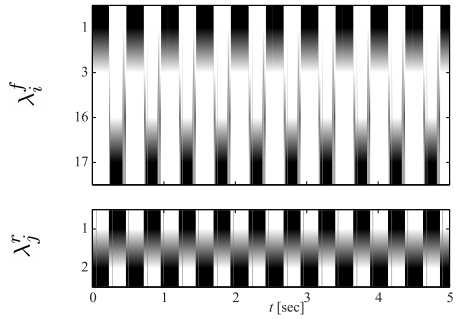
各強化学習コントローラの価値関数は実験 1，2 と同様に全ての状態予測モデルと報酬予測モデルの組み合わせから付録 A の手法を用いて導出した．導出は実験の前に行い，実験中は固定とした．価値関数の次定数 $\tau$ は 0.5，TD 誤差の分散 $\delta_k^{c\,2}$ は $0.1^2$ で固定とした．

### 結果

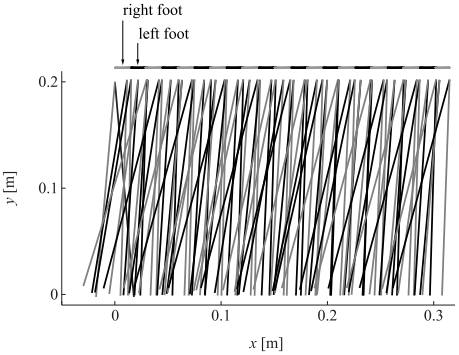
図 2.9(a)-(d) に結果を示す．図 2.9(a) は脚の角度 $\theta_L(t)$，$\theta_R(t)$ の変化を示す．開始直後は少し乱れているものの，0.5[sec] 以降は安定しており左右の足が周期的に動いている事が分かる．図 2.9(b) は状態予測モデルと報酬予測モデルの責任信号 $\lambda_i^f(t)$，$\lambda_j^r(t)$ の変化を示す．1，17 番目の状態予測モデルが主に用いられており，その 2 つが切り替わる際に 3，16 番目が用いられている．図 2.9(c) は歩行の様子を 0.1[sec] 毎に表しており，灰色が右足，黒色が左足を示している．また上部にどちらの足が接地しているかを表示してある．図 2.9(d) は周期運動の軌道と状態予測モデル，報酬予測モデルが切り替わる様子を示している．左足が接地している状態では 3, 16, 17 番目，また左足が接地している状態では 1 番目の状態予測モデルが用いられており，足が接地すると同時に報酬関数も切り替わっている．



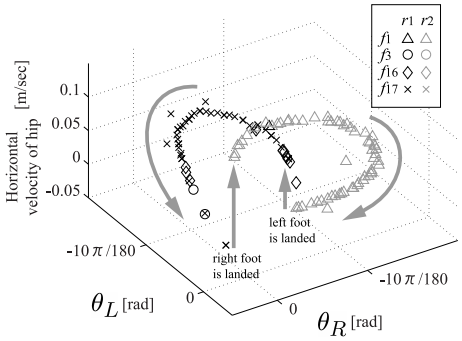

(a) 足の角度の変化．黒色が左足，灰色が右足であることを示している．




(b) 責任信号の変化．状態予測モデルは実際に用いられていた 4 個のみを表示した．




(c) 歩行の様子を 0.1[sec] 毎に表示した．上部の線は黒の時は左足が，灰色の時は右足が接地している事を表している．




(d) 軌道と予測モデルの切り替わりを同時に表示した．縦軸は関節部の水平方向の速度である．各点は責任信号の最大値をもつ予測モデルが表示されている．軌道が不連続に飛んでいる箇所にて足の接地が行われている．

図 2.9 実験 3 の結果.

# 6. 比較

この節では CMRL と Doya らによって提案されている MMRL[15] の違いについて考察してみる．一番の大きな違いは環境を分節化する基準であると言える．CMRL ではダイナミクスと報酬の予測性に基づいて分節化を行っているが，MMRL ではダイナミクスの予測性にのみ基づいている．CMRL は 2 つの基準の分節化によりダイナミクスと報酬関数の両方が切り替わる環境にも対応可能である事をシミュレーションによって示した．しかし MMRL を用いて同様の実験を行った場合，エージェントは報酬関数の切り替わりを認識できずモジュールを切り替えられない事が予測される．2 つ目の違いは強化学習コントローラの選択方法である．MMRL では各予測モデルと強化学習コントローラが対になっているが，CMRL では分離されているためそれぞれ個別に選択できる．一般に強化学習コントローラは多くのパラメータを必要とするため，その個数を減らす事は学習速度や記憶容量の観点から見て重要である．CMRL では強化学習コントローラの個数が少ない場合であっても，TD 誤差によって尤もらしいものを選択する事ができる．

# 7. 本章の議論

本章では環境のダイナミクスを推定する状態予測モデルと報酬を予測する報酬予測モデルを用い，両方の予測性に基づいて環境を分節化する手法を提案した．従来手法の MMRL ではダイナミクスの非定常性にのみ適応可能であったが，提案する分節化の基準により報酬関数の非定常性にも対応できた．また強化学習コントローラの制御に対する貢献度を TD 誤差を用いて決定した．それにより，強化学習コントローラの個数が変動する環境の個数よりも少ない場合であっても，尤もらしいものを選択したり幾つかを組み合わせて制御が行える．

# 第3章　階層型モジュール強化学習

## 1. 階層型モジュール強化学習

優れた可能性を持つ強化学習 [38] は数々の理論的発展と様々な問題への適用が行われてきた [24]．しかし通常の強化学習を実問題の様に膨大な数のパラメーターを必要とする問題へ適用した場合，学習に必要な時間は実現不可能なほど長くなる．Singh [35] は効率よく学習を行うためにはサブゴールの利用，探索の高効率化，状態表現の汎化などが解決されるべき課題であるとしている．サブゴールを設定する事で大きなタスクを分解する事ができ，必要なパラメータの個数や学習に必要な時間を減らせられる．また探索するべき空間が広いほど局所的な解に陥りやすいため，大域的な探索を効率よく事が求められる．そして状態空間を抽象化しその中では必要なパラメータや方策がある程度似通っていると仮定することで，パラメータ数が減ったり頻繁な行動選択を避けられたりする効果が得られる．

モジュール構造・階層構造の導入は Singh が挙げる課題を解決する手段のひとつである．Dayan ら [8] は解像度の粗い上位層を設ける事で効率よく探索を行う “Feudal Reinforcement Learning” を提案している．粗く分割した状態空間で行われる大域的な探索と，下位の細かい状態空間で行われる局所的な探索を同時に行える手法である．Dietterich[11] はマルコフ決定過程（Markov decision processes, MDP）を分解し，それぞれに価値関数を割り当てる “MAXQ Value Function Decomposition” を提案している．長い時間幅で分割を行う上位層と短い時間幅で分割を行う下位層により効率の良い探索を実現するとともに，各層においてどのモジュールが用いられているかに応じて複数の文脈に対する価値関数を表現できる．また Sutton ら [40, 39] は一般的な強化学習の枠組みを大きく変え

ずに行動の抽象化を行う “Option” の提案を行っている．これまでの研究により階層構造の導入は学習時間を大幅に短縮できる事が示されているが，その多くは離散時間・離散状態を対象とした研究である．離散時間・離散状態を対象とした強化学習は定式化や収束証明のしやすさ・アルゴリズムの実装のしやすさなどの利点がある．しかし実世界においてヒトやロボットが環境と相互作用を行う場合，観測される環境の状態や報酬信号，また出力する制御信号は一般に連続な値を持つ．環境を実験者が予め離散化したり階層構造を作りこまなければならない手法は，最も困難な課題を実験者が問題に対する事前知識に頼って解決してしまっているので一般性がないと言わざるを得ない．連続なシステムを扱う時にこそ階層型強化学習の本質的な困難が顕在化すると言うのが我々の考えであり，本稿はその様な問題に取り組むための枠組みとして連続時間・連続状態を対象とした階層構造の構築を行う．

抽象化された状態空間で行動選択を行う上位層を設け学習時間を短縮する事が階層型強化学習の目的のひとつであり，Morimoto らの研究においても上位層の有用性が示されている [29]．そのような階層構造を用いようとした場合 “どのように状態や行動を抽象化するか” が最も難しく，また性能に大きくかかわってくる．非線形な連続システムでは環境の特性を実験者が把握しにくいため抽象化が特に難しく，自動的に抽象化が行える手法が必要となる．Wolpert ら [43, 49] による MOSAIC モデルは制御出力と感覚フィードバックの入出力特性に基づいて環境の分節化が自動で行え，その文節ごとに制御器を切り替える事で非線形・非定常な環境への適応が可能であるとしている．そこでこの章では MOSAIC モデルに基づいて切り分けられる分節を抽象化状態とする階層型モジュール強化学習の定式化を行う．下位層は状態の抽象化を行い上位層は各抽象化状態でどのサブゴールを目標とするべきかを強化学習により獲得する．そして下位層は上位層が選択したサブゴールの達成度に応じて与えられる “下位報酬” を最大とするような制御則の学習を行う．

MOSAIC モデルに基づいた強化学習手法としてこれまでに Doya らによる Multiple Model-based Reinforcement Learning (MMRL) [15, 46] や杉本らによる Combinatorial Model-based Reinforcement Learning (CMRL) [48] が提案されている．

しかしそれらは階層構造を持っておらず予測モデルの出力誤差に基づいて制御器の出力を重み付けしていたため，モジュールの切り替えの選択において累積報酬の最大化が考えられていないという欠点があった．

提案手法において行動選択を行う上位層は標準的な離散系強化学習の枠組みで記述される．そのためこれまでに離散系を対象として提案されている手法を容易に適用できる．その様な上位層と連続な環境の間を取り持つ下位層の存在が本研究における最大の新奇性である．そこでまず，MOSAIC モデルによる環境の抽象化について 1.1 節で説明する．そして 2 節で提案手法の定式化を行い 3 節でシミュレーションによる実験を行う．最後に 6 節でまとめを行う．

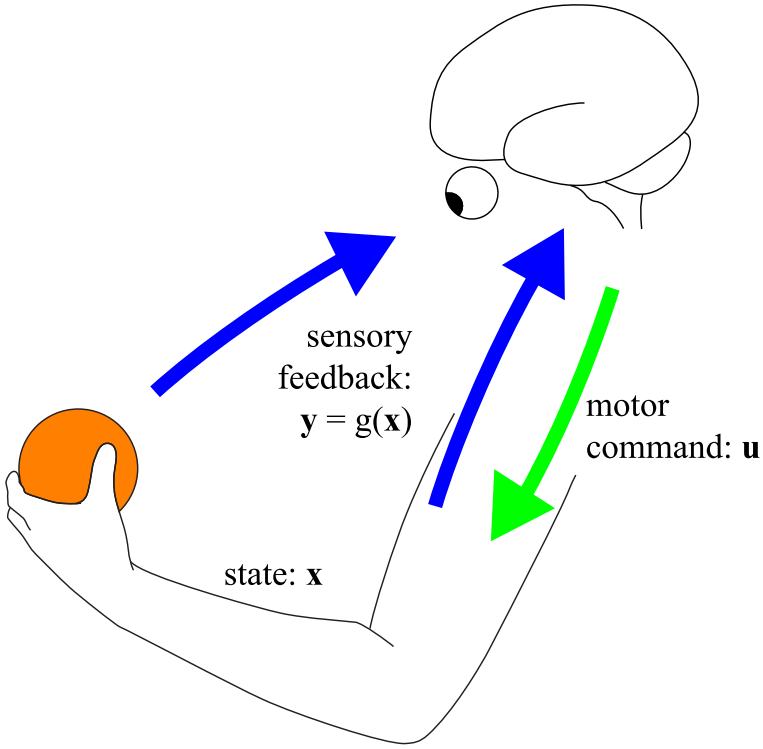



図 3.1 状態 $\mathbf{x}$ を観測関数 $g$ を通した感覚フィードバック $\mathbf{y}$ として観測する．制御出力 $\mathbf{u}$ と感覚フィードバック $\mathbf{y}$ の入出力の時系列から環境の状態を推定し，かつ抽象化を行う．

## 1.1 MOSAIC モデルに基づく状態の抽象化:連続システムと SMDP

この章では MOSAIC モデルに基づいた状態の抽象化について説明する．本論文で対象とする環境は，ある時刻 $t \in \Re$ における状態 $\mathbf{x}(t) \in \Re^{D_s}$ と制御出力 $\mathbf{u}(t) \in \Re^{D_c}$ のもとでの状態変化 $\dot{\mathbf{x}}(t) = \frac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t}$ と観測変数 $\mathbf{y}(t) = g(\mathbf{x}(t)) + \nu(t)$ の同時分布が以下の様に分解できると仮定する（図 3.1，$\nu(t)$ は観測ノイズ）[1]．

$$p(\dot{\mathbf{x}}(t), \mathbf{y}(t) \mid \mathbf{x}(t), \mathbf{u}(t)) = \sum_{i=1}^{N^f} \lambda_i^f(t) p(\mathbf{y}(t) \mid \mathbf{x}(t), i) p(\dot{\mathbf{x}}(t) \mid \mathbf{x}(t), \mathbf{u}(t), i)$$

[1] $\Re$ はスカラー，$\Re^m$ は $m$ 次元の縦ベクトルを表す．ベクトルに関する微分については付録 B 章に補足してある．

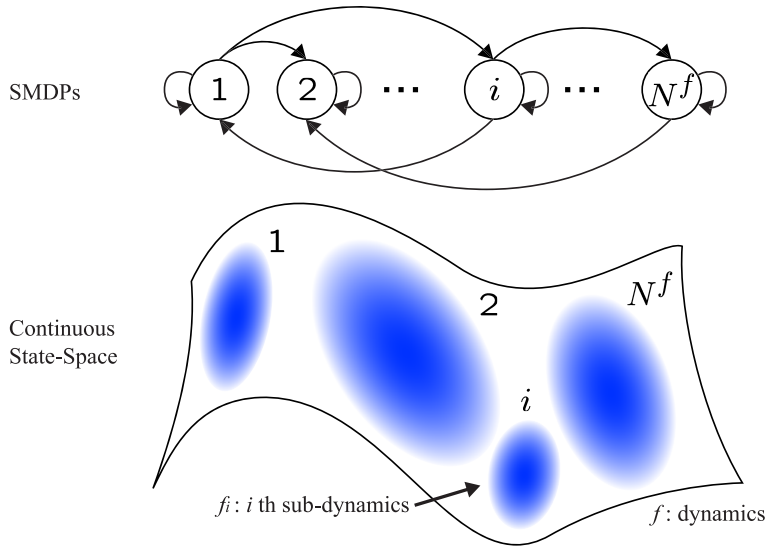



図 3.2　下図は環境の状態 $\mathbf{x}$ とその変化 $\dot{\mathbf{x}}$ が成すダイナミクスを表している．それを複数の局所領域に分割しインデックス $i = 1, ..., N^f$ を割り当てると上図の様にそのインデックス $i$ を離散状態とした SMDP モデルが考えられる．

ここで $p(\dot{\mathbf{x}}(t) \mid \mathbf{x}(t), \mathbf{u}(t), i)$ は局所的なダイナミクスを表し $N^f$ がその個数，$\lambda_i^f(t)$ が責任信号と呼ばれる重み付けのための信号である．一般に複数個の局所ダイナミクスが切り替わる環境のモデルは “Switching State-Space Models (SSM)” と呼ばれる [16, 31]．提案手法は MOSAIC モデルに基づいて局所ダイナミクスが切り替わる SSM として環境をモデル化し，各時刻において有効になっている局所ダイナミクスのインデックスを上位層の状態として採用する（図 3.2）．

MOSAIC モデルに基づいた環境の分節化は観測 $\mathbf{y}(t)$ の時系列から現在の状態 $\mathbf{x}(t)$ を推定し，状態予測モデルの予測誤差に基づいて責任信号 $\lambda_i^f(t)$ を求める事で実現される．状態予測モデルは時刻 $t$ における状態 $\mathbf{x}(t)$ と入力 $\mathbf{u}(t)$ から状態変化 $\dot{\mathbf{x}}(t)$ の予測値 $\hat{\dot{\mathbf{x}}}(t)$ を出力するモデルである（$\hat{\dot{\mathbf{x}}}(t) = f_i(\mathbf{x}(t), \mathbf{u}(t))$）．

$$p(\hat{\dot{\mathbf{x}}}(t) \mid \mathbf{x}(t), \mathbf{u}(t), i) = f_i(\mathbf{x}(t), \mathbf{u}(t)) + \omega(t) \tag{3.1}$$

ここで $\omega(t)$ はシステムノイズである．状態 $\mathbf{x}(t)$ の推定にはカルマンフィルタ [25] やモンテカルロ法 [12] などの手法が利用できる．これ以降ではそれらの手法によ

り状態 $\mathbf{x}(t)$ の推定値 $\tilde{\mathbf{x}}(t)$ が既に得られたものとして責任信号の説明を行う．

責任信号 $\lambda^f(t)$ は時刻 $t$ までの状態 $\tilde{\mathbf{x}}(t)$ とその変化 $\tilde{\dot{\mathbf{x}}}(t)$ および制御出力 $\mathbf{u}(t)$ の時系列の事後分布として推定される[2]．

$$\lambda^f(t) = \{\lambda_1^f(t), \cdots, \lambda_i^f(t), \cdots, \lambda_{N^f}^f(t)\} \tag{3.2}$$

$$\lambda_i^f(t) \equiv P(i|\tilde{\dot{\mathbf{x}}}_{0:t}, \tilde{\mathbf{x}}_{0:t}, \mathbf{u}_{0:t}) \tag{3.3}$$

責任信号は各状態予測モデル $f_i$ の尤度を正規化する事で求められるが，その尤度を状態予測誤差 $\hat{\dot{\mathbf{x}}}(t) - \tilde{\dot{\mathbf{x}}}(t)$ に基づいて計算するのが責任信号の特徴である．そのような分節化によって非線形・非定常な環境を適応的に制御できる事がこれまでの研究で示されている [43, 15, 19, 20]．よって各時刻 $t$ において最も高い責任信号をもつ状態予測モデルのインデックスは環境を良く抽象化していると言え，そのインデックスを状態変数とする上位層は MOSAIC モデルの利点を有する状態空間で行動選択を行える．抽象化状態のインデックスを $i$，選択可能なサブゴールのインデックスを $j$ とすると，上位層が扱う環境はセミマルコフ決定過程 (semi-Markov decision processes, SMDP) [6, 27, 32] としてモデル化される．SMDP は連続時間・離散状態のシステムをモデル化したもので Markov decision processes(MDP) の一種である．SMDP は $\langle \mathbf{S}, \mathbf{A}, P, R, L \rangle$ の 5 つの組によって定義され，$\mathbf{S}$ は状態の集合，$\mathbf{A}$ は行動の集合，$P$ は各時刻の状態と行動に依存した状態遷移確率，$R$ は報酬関数，$L$ は次の状態に遷移する時間の分布を表している [26]．この定義に従うと上位層の環境は以下の様な SMDP と言える．

- 状態の集合は状態予測モデルのインデックス $\mathbf{S} = \{1, \cdots, i, \cdots, N^f\}$ となる．
- 行動の集合はサブゴールのインデックス $\mathbf{A} = \{1, \cdots, j, \cdots, N^r\}$ となる．
- 状態繊維確率 $P$ は離散状態 $i$ と行動 $j$ に依存した分布 $P(i'|i, j)$ を持つ．
- 状態が $i'$ に切り替わった時，状態 $i$(と行動 $j$) に対する報酬 $R(t)$ は連続な報酬を状態 $i$ に滞在していた時の経路に沿って積分したものとして与えられる．

[2] $\bullet_{0:t}$ は時刻 0 から $t$ の間に観測された変数 $\bullet$ の時系列を表す．

- 次状態に遷移するまでの時間 $l$ はある分布 $P(l|i',i,j)$ を持つ

上位層の状態繊維確率は $P(i'|i,j)$ とおいたが実際はその裏に下位の連続状態 $\mathbf{x}(t)$ と連続行動 $\mathbf{u}(t)$ が隠れており，上位層の状態が $i$ に遷移した時にどのような連続状態 $\mathbf{x}(t)$ と連続行動 $\mathbf{u}(t)$ を伴っていたかによって状態繊維確率 $P(i'|i,j)$ の分布は変化する．また報酬 $R(t)$ も同じく下位の連続状態 $\mathbf{x}(t)$ と連続行動 $\mathbf{u}(t)$ に左右される．そのため厳密には SMDP の枠組みから少し外れるが，定式化の簡単のためこのシステムを SMDP とみなして定式化を進めていく．

SMDP 環境における最適状態価値関数 $V^*$ は以下のような Bellman 方程式を満さなければならない [6].

$$V^*(i) = \max_j \left[ \sum_{i'=1}^{N^f} P(i'|i,j) \int_0^\infty P(l|i',i,j) \left\{ \int_t^{t+l} e^{-\frac{s-t}{\tau}} R(s) ds + e^{-\frac{l}{\tau}} V^*(i') \right\} dl \right] \tag{3.4}$$

右辺中括弧内の第一項 $\int_t^{t+l} e^{-\frac{s-t}{\tau}} R(s) ds$ は時刻 $t$ から $t+l$ まで状態 $i$ に滞在した時に得る報酬の重み付累積和で，$\tau$ はその時定数である．また第二項中の $e^{-\frac{l}{\tau}}$ は次状態 $i'$ に遷移するまでの時間 $l$ に応じた割引率である．それらを状態遷移確率 $P(i'|i,j)$ と確率分布 $P(l|i',i,j)$ で平均化している．時間が連続で，また遷移する時間が分布を持つことを除けば MDP における定義と同様である．同じく最適行動価値関数 $Q^*(i,j)$ は以下を満たす.

$$Q^*(i,j) = \sum_{i'=1}^{N^f} P(i'|i,j) \int_0^\infty P(l|i',i,j) \left\{ \int_t^{t+l} e^{-\frac{s-t}{\tau}} R(s) ds + e^{-\frac{l}{\tau}} \max_{j'} Q^*(i',j') \right\} dl \tag{3.5}$$

以上の様に MOSAIC モデルに基づいた SSM はセミマルコフ的に局所ダイナミクスが切り替わる事より，“Semi-Markov Switching State-space Models” と呼ぶことにする．次章ではこのモデルに基づいた階層型強化学習の定式化を行う．

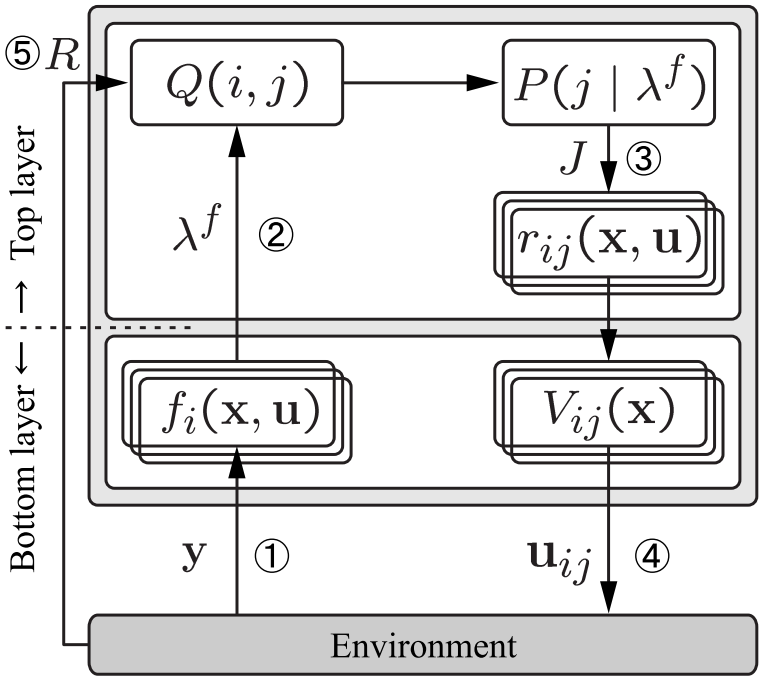



図 3.3 SSRL の構造を示す．変数 $J$ は上位層が選択した下位報酬関数のインデックスである．$\lambda^f$ は責任信号の集合 $\{\lambda_1^f, \cdots, \lambda_i^f, \cdots \lambda_{Nf}^f\}$ を表す．

## 2. アルゴリズム

提案する階層型強化学習アルゴリズムを “Semi-Markov Switching State-Space Model-Based Reinforcement Learning (SSRL)” と名付ける（図 3.3）．SSRL は上下二層から構成され上位層は抽象化状態のインデックス $i$ を状態変数，サブゴールのインデックス $j$ を行動変数として強化学習を行う．そして選択したサブゴール $j$ の達成度に応じた報酬を下位層に与える．抽象化状態 $i$ でのサブゴール $j$ に対応する報酬関数を $r_{ij}(\mathbf{x}(t), \mathbf{u}(t))$ で表し，環境からの報酬と区別するために下位報酬関数と呼ぶ．下位層の働きは大きく分けて二つあり，一つ目は連続状態 $\mathbf{x}(t)$ の抽象化である．下位層は複数の状態予測モデル $f_i(\mathbf{x}(t), \mathbf{u}(t))$ を持っており，制御の各時刻において責任信号 $\lambda^f(t)$ を上位層に伝える．二つ目は上位層から与えられる下位報酬の最大化である．抽象化状態 $i$ と選択された下位報酬関数 $r_{ij}((\mathbf{x}(t), \mathbf{u}(t)))$ に対応する下位価値関数 $V_{ij}(\mathbf{x}(t))$ を学習し，それに基づいて制御出力 $\mathbf{u}_{ij}(\mathbf{x}(t))$ を

| | 状態 | 行動 |
|---|---|---|
| 上位層 | 抽象化された状態<br>のインデックス $I$ | 下位報酬関数<br>のインデックス $J$ |
| 下位層 | 連続状態 $\mathbf{x}$ | 行動出力 $\mathbf{u}$ |

表 3.1 各層における状態変数と行動変数.

環境に与える．以上より上位層・下位層の状態と行動は以下の様にまとめられる．

学習は図 3.3 の丸字番号で示した順に従い，環境を観測する毎に繰り返し行われる．各丸字番号の大まかな内容は以下の通りである．

① 下位層：時刻 $t$ までに得た観測 $\mathbf{y}_{0:t}$ と制御出力 $\mathbf{u}_{0:t}$ から環境の状態 $\mathbf{x}(t)$ とその変化 $\dot{\mathbf{x}}(t)$ を推定

② 下位層：責任信号 $\lambda^f(t)$ を推定し上位層に伝える

③ 上位層：行動価値関数 $Q$ を参照して行動を選択

④ 下位層：上位層が選択した行動を実現するための行動出力を生成

⑤ 上位層：報酬 $R(t)$ を観測し行動価値関数 $Q$ を更新

次節以降では各層の詳細を説明する．

## 2.1 上位層

### 行動選択と下位価値関数の学習

観測にノイズや隠れ状態が存在する時，責任信号 $\lambda^f(i)$ はある分布を持ち信念状態の一種とみなされる．そこで行動選択は行動価値関数 $Q$ を責任信号 $\lambda^f(t)$ で重み付けしたものを用いて行う．行動選択確率は，例えば Boltzmann 分布に従う

とすると逆温度パラメーター $\beta$ を用いて以下の様に記述される．

$$P(j|\lambda^f(t)) = \frac{\exp\left[\beta \sum_{i=1}^{Nf} \lambda_i^f(t) Q(i,j)\right]}{\sum_{j'} \exp\left[\beta \sum_{i=1}^{Nf} \lambda_i^f(t) Q(i,j')\right]} \tag{3.6}$$

上位層における行動選択と行動価値関数の更新は上位層の状態が変化した時，つまり最大責任信号を持つ状態予測モデルのインデックスが変化した時に行われる．時刻 $t$ から $t+l$ の区間において上位層の状態が $I$，行動が $J$ であり，時刻 $t+l$ において上位層の状態が $I'$ へ変化したとすると状態行動価値関数 $Q(I,J)$ の更新は以下の様に行われる[3]．

$$Q(I,J) := Q(I,J) + \alpha\Big[\int_t^{t+l} e^{-\frac{s-t}{\tau}} R(\mathbf{x}(s),\mathbf{u}(s))ds + e^{-\frac{l}{\tau}} \max_j Q(I',j) - Q(I,J)\Big] \tag{3.7}$$

$\alpha$ は学習係数，$0 < \tau < \infty$ は割引の時定数である．

#### 下位報酬関数の設定

上位層はある状態遷移確率 $P(i' \mid i,j)$ にしたがって遷移する抽象化状態を環境として学習を行う．その状態遷移確率は各抽象化状態 $i$ に対する下位報酬関数 $r_{ij}(\mathbf{x}(t),\mathbf{u}(t))$ によって決定されるが，その状態遷移が確定的であるほど上位層の学習は行いやすい．そこで下位報酬関数 $r_{ij}(\mathbf{x}(t),\mathbf{u}(t))$ として隣り合った抽象化状態の領域内に頂点を持つ凸関数を採用する．その様な報酬関数を下位層に与えれば，任意の抽象化状態への遷移確率が確定的になる事が期待できる．

### 2.2 下位層

#### 連続状態の抽象化

下位層はまず，時刻 $t$ までの観測 $\mathbf{y}_{0:t}$ と制御出力 $\mathbf{u}_{0:t}$ から現時刻のダイナミクスの推定値 $\tilde{\mathbf{x}}(t), \tilde{\dot{\mathbf{x}}}(t)$ を求める．そして責任信号 $\lambda_i^f(t)$ の推定を行う．時刻 $t$ の責

[3]最大責任信号を持つ状態予測モデルのインデックスを大文字の $I$，式 (3.6) の確率分布に従って選択された行動のインデックスを大文字の $J$ で表す．

任信号 $\lambda_i^f(t)$ は現在の状態 $\mathbf{x}(t)$ と平滑化された予測誤差から求められる $\hat{\lambda}_i^f(t)$ を事前分布とし，状態変化 $\tilde{\mathbf{x}}(t)$ が推定された後の事後分布として推定される．

$$\lambda_i^f(t) \equiv \frac{p(\tilde{\mathbf{x}}(t) \mid i(t), \tilde{\mathbf{x}}(t), \mathbf{u}(t), \phi_i^f(t))\hat{\lambda}_i^f(t)}{\sum_{i'=1}^{Nf} p(\tilde{\mathbf{x}}(t) \mid i'(t), \tilde{\mathbf{x}}(t), \mathbf{u}(t), \phi_{i'}^f(t))\hat{\lambda}_{i'}^f(t)} \tag{3.8}$$

$$\tag{3.9}$$

各状態予測モデルの尤度は予測誤差が分散 $\Phi_i^f \in \Re^{D_s \times D_s}$ に従う正規分布だとした場合，以下の様に定義される[4]．

$$p(\tilde{\mathbf{x}}(t)|\tilde{\mathbf{x}}(t), \mathbf{u}(t), i) = \frac{1}{(2\pi)^{D_s/2}|\Phi_i^f|^{D_s/2}} \exp\left[-\frac{1}{2}(\hat{\mathbf{x}}_i(t) - \tilde{\mathbf{x}}(t))^T \Phi_i^f (\hat{\mathbf{x}}_i(t) - \tilde{\mathbf{x}}(t))\right] \tag{3.10}$$

ここで $\hat{\mathbf{x}}_i(t)$ は $i$ 番目の状態予測モデル $f_i(\mathbf{x}(t), \mathbf{u}(t))$ の予測出力値である．責任信号の事前分布 $\hat{\lambda}_i^f(t)$ は空間的局所性と時間的連続性の 2 つから求められる．空間的連続性は各状態予測モデルがダイナミクスの入力空間 $\mathbf{x}$ に対して局所的な担当範囲を持つと言う仮定，また時間的連続性は責任信号は頻繁に変化しないと言う事前知識に基づいて出力される．各状態予測モデルの担当範囲が $\mathbf{x}_i^f$ を中心としたガウス分布を持ち，また時間的連続性が状態予測誤差を平滑化したものによって実現されるとすると責任信号の事前分布 $\hat{\lambda}_i^f(t)$ は以下の様になる．

$$\hat{\lambda}_i^f(t) = \frac{1}{(2\pi)^{(D_s+1)/2}|\Sigma_i^f|^{1/2}\bar{\sigma}_i^{f\,D_s}} \exp\left[-\frac{1}{2}(\tilde{\mathbf{x}}(t) - \mathbf{x}_i^f)^T \Sigma_i^{f\,-1}(\tilde{\mathbf{x}}(t) - \mathbf{x}_i^f) - \frac{1}{2\bar{\sigma}_i^{f\,2}} E_i^f(t)\right] \tag{3.11}$$

ここで $E_i^f(t)$ は状態予測誤差を時間的に平滑化したものであり，$\bar{\sigma}_i^{f\,2}$ はそれらの分散を表している．

この節で説明した責任信号 $\lambda^f(t)$ の計算によって SSRL は連続な環境を分節化する．責任信号は状態予測誤差に基づく事でダイナミクスの出力空間を，空間的局所性に基づく事でダイナミクスの入力空間を分節化している．この様に入出力の両方で環境を分節化している事が SSRL の特徴のひとつである．

[4] $\Re^{m \times n}$ は $m$ 行 $n$ 列の行列を表す．

#### 下位の行動則の学習

下位層は状態予測モデル $f_i(\mathbf{x}(t), \mathbf{u}(t))$ と下位報酬関数 $r_{ij}(\mathbf{x}(t), \mathbf{u}(t))$ の組み合わせに対する下位状態価値関数 $V_{ij}(\mathbf{x}(t))$ を持っており，上位層から与えられる下位報酬をもとに学習する．最適下位状態価値関数 $V^*_{ij}(\mathbf{x}(t))$ は以下の Bellman 方程式を満たす [13].

$$\frac{1}{\xi}V^*_{ij}(\mathbf{x}(t)) = \max_{\mathbf{u}}\left[r_{ij}(\mathbf{x}(t), \mathbf{u}(t)) + \frac{\partial V^*_{ij}(\mathbf{x}(t))}{\partial \mathbf{x}} f_i(\mathbf{x}(t), \mathbf{u}(t))\right] \tag{3.12}$$

ここで $0 < \xi \leq \tau$ は時定数である．最適下位状態価値関数 $V^*_{ij}(\mathbf{x}(t))$ に従った最適制御出力 $\mathbf{u}^*_{ij}(\mathbf{x}(t))$ は以下の様になる．

$$\mathbf{u}^*_{ij}(\mathbf{x}(t)) = S'^{-1}\left(\left(\frac{\partial f_i(\mathbf{x}(t), \mathbf{u}(t))}{\partial \mathbf{u}}\right)^T \left(\frac{\partial V^*_{ij}(\mathbf{x}(t))}{\partial \mathbf{x}}\right)^T\right) \tag{3.13}$$

ここで $S'()$ は出力のコスト関数を出力 $\mathbf{u}$ で偏微分したものであり，$S'^{-1}()$ はその逆関数を表す．下位層は上位層の状態と行動に応じた下位価値関数を学習しながら制御出力を行う．

## 2.3 下位価値関数の最適化

階層型強化学習において，Feudal Reinforcement Learning[8] や Morimoto らの手法 [29] の様に，上位が指定したサブゴールの達成度合いに応じて下位への報酬が決定される方式では達成度合いの評価方法によって最終的な性能が大きく左右される．そして，SSRL の場合における達成度合いの評価方法は下位報酬関数の形状に相当する．この節では真に最適な制御則を獲得するために，下位報酬関数から導かれる下位状態価値関数を最適化する手法を定式化する．

まず最初に各下位状態価値関数 $v_{ij}(\mathbf{x}(t))$ を基底とみなし，それらの線形和によって新たな下位状態価値関数を考える．拡張された下位状態価値関数は $V^k_i(\mathbf{x}(t))$ で表され，各状態予測モデル $f_i$ に対して $N^c$ 個ずつ用意される $(k = 1, ..., N^c)$.

$$V^k_i(\mathbf{x}(t)) = \sum_{j=1}^{N^r} w^k_{ij} v_{ij}(\mathbf{x}(t)) \tag{3.14}$$

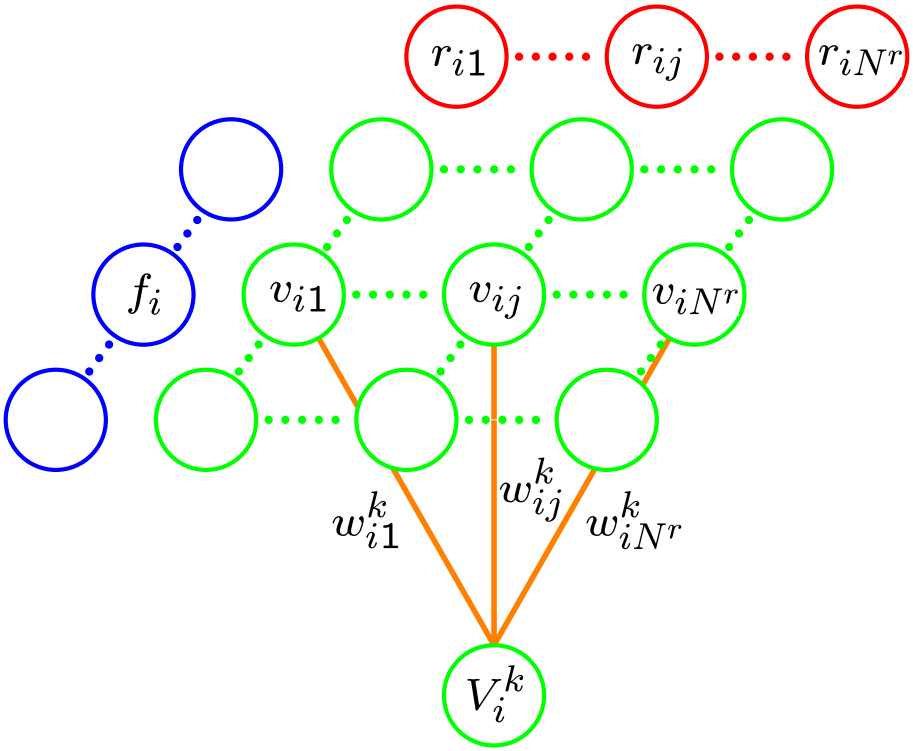



図 3.4 拡張された下位状態価値関数 $V_i^k$ は複数個の下位状態価値関数 $v_{ij}$ の重み付和で表される．

ここで $w_{ij}^k$ は重みのパラメータであり，これを更新する事で拡張された下位状態価値関数 $V_i^k(\mathbf{x}(t))(k = 1, ..., N^c)$ を最適化する．また拡張された下位状態価値関数は $V_i^k(\mathbf{x}(t))$ に対する最適制御出力則 $\mu_i^k(\mathbf{x}(t))$ は

$$\mathbf{u}_i^k(\mathbf{x}(t)) = S'^{-1}\left(\left(\frac{\partial f_i(\mathbf{x}(t), \mathbf{u}(t))}{\partial \mathbf{u}}\right)^T \left(\frac{\partial V_i^k(\mathbf{x}(t))}{\partial \mathbf{x}}\right)^T\right) \tag{3.15}$$

となる．この拡張に伴って，上位層の学習は下位報酬関数の選択から拡張された下位状態価値関数の選択へと変更される．よって式 (3.6) で表される上位層の行動選択確率は拡張された下位状態価値関数のインデックス $k$ を用いて以下の様に

再定式化される．

$$P(k|\lambda^f(t)) = \frac{\exp\left[\beta \sum_{i=1}^{N^f} \lambda_i^f(t) Q(i,k)\right]}{\sum_{k'} \exp\left[\beta \sum_{i=1}^{N^f} \lambda_i^f(t) Q(i,k')\right]} \tag{3.16}$$

またこの確率分布に従って選択された離散行動を $K$ とする事で，状態行動価値関数の更新即も以下の様に再定式化される．

$$Q(I,K) := Q(I,K) + \alpha\Big[\int_t^{t+l} e^{-\frac{s-t}{\tau}} R(\mathbf{x}(s),\mathbf{u}(s))ds + e^{-\frac{l}{\tau}} \max_k Q(I',k) - Q(I,K)\Big] \tag{3.17}$$

これは変更前の更新則 (式 (3.7)) の行動のインデックスを $J$ から $K$ に置き換えたものであり，更新の間隔は同じ様に上位層の離散状態が変化した時に行われる．次に重みのパラメータ $w_{ij}^k$ の更新方法について述べる．更新のための評価関数 $E(t)$ は拡張された下位状態価値関数 $V_i^k(\mathbf{x}(t))$ に対する TD 誤差 $\delta_i^k(t)$ を状態予測モデルの責任信号 $\lambda_i^f(t)$(式 (3.8)) と行動選択確率 $P(k \mid \lambda^f(t))$ で重み付けしたものを用いる．

$$E(t) = \frac{1}{2} \sum_{i=1}^{N^f} \sum_{k=1}^{N^c} \lambda_i^f(t) P(k \mid \lambda^f(t)) \delta_i^k(t)^2 \tag{3.18}$$

$$\delta_i^k(t) = R(t) - \frac{1}{\xi} V_i^k(\mathbf{x}(t)) + \frac{\partial V_i^k(\mathbf{x}(t))}{\partial \mathbf{x}} \dot{\mathbf{x}}(t) \tag{3.19}$$

重み $w_{ij}^k$ の更新はこの評価関数が減少する方向へ行われ，勾配法を用いた場合の更新即は以下の様になる．

$$\dot{w}_{ij}^k = -\eta \frac{\partial E(t)}{\partial w_{ij}^k} \tag{3.20}$$

$$\begin{aligned} \frac{\partial E(t)}{\partial w_{ij}^k} &= \sum_{i=1}^{N^f} \sum_{k=1}^{N^c} \lambda_i^f(t) P(k \mid \lambda^f(t)) \delta_i^k(t) \frac{\partial \delta_i^k(t)}{\partial w_{ij}^k} \\ &= \sum_{i=1}^{N^f} \sum_{k=1}^{N^c} \lambda_i^f(t) P(k \mid \lambda^f(t)) \delta_i^k(t) \left\{ -\frac{1}{\xi} v_{ij}(\mathbf{x}(t)) + \frac{\partial v_{ij}(\mathbf{x}(t))}{\partial \mathbf{x}} \dot{\mathbf{x}}(t) \right\} \end{aligned} \tag{3.21}$$

ここで $\eta$ は学習係数である．

基底関数 $v_{ij}(\mathbf{x}(t))$ の近似能力が不十分な場合，重み $w_{ij}^k$ の学習だけでは真の価値関数に近づける事はできず個々の基底 $v_{ij}^k(t)$ をも更新しなければならない．ゴールにたどり着いた時刻を $T$，ゴールでの状態価値を $V_{max}$ とすると各時刻 $t$ での価値は $\hat{V}(t) = e^{-\frac{T-t}{\tau}} V_{max}$ と求められる．この $\hat{V}(t)$ に下位価値関数の出力が近づくように個々の基底関数を更新する．各時刻 $t$ における下位価値関数の出力を $V(t)$ とした場合の評価関数は

$$G(t) = \frac{1}{2}(V(t) - \hat{V}(t))^2 + \xi\phi(\mathbf{x}(t)) \tag{3.22}$$

と書ける．ここで $\phi$ は滑らかさを補償する項，$0 \leq \xi$ はその度合いを決めるパラメータである．

# 3. シミュレーション: 倒立振子の制御

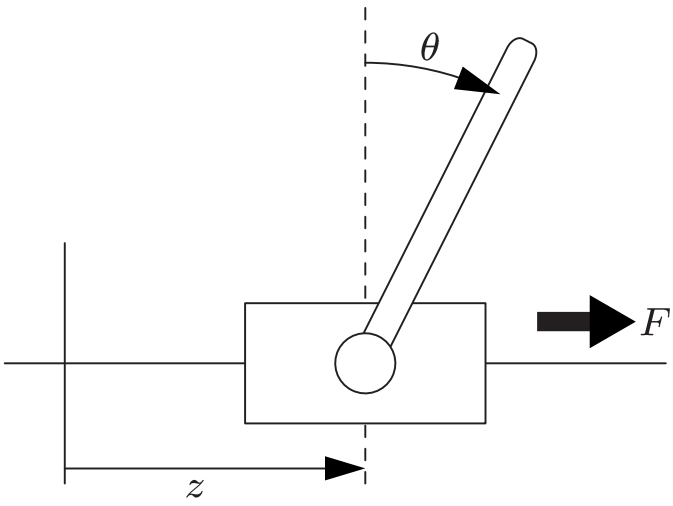



図 3.5 倒立振子の振り上げ課題．台車が移動可能な幅に制限はないものとした．台車の横方向にのみ力 $F$ を加える事ができ，その上限は $|F| < 20$[N] とした．状態変数の次元 $D_s$ は 4，制御入力の次元 $D_c$ は 1 である．

図 3.5 のような倒立振子 ($\mathbf{x} = [z\ \theta\ \dot{z}\ \dot{\theta}]^T, \mathbf{u} = [F]$) の振り上げ課題を行う ($D_s = 4, D_c = 1$)．簡単のため各観測関数は $g_i(\mathbf{x}(t)) = \mathbf{x}(t) + \mathcal{N}(0, 0.1I_{D_s})$ [5]とした．物理パラメータは Barto ら [3] が用いたものと同じ設定にし，制御の間隔は 0.01[sec] とした．またシステムノイズとして $\mathcal{N}(0, 0.1I_{D_s})$ を与えた．

1 試行の長さは 10[sec] とした．各試行の初期状態は振子がぶら下がった状態 ($\theta = \pi$[rad]) から $\pm\frac{15}{180}\pi$[rad] の範囲でランダムに選択した．報酬は振子が倒立した状態 ($\theta = 0$[rad]) の周りで大きく与えられる正の報酬から出力のコスト $S(\mathbf{u}(t)) = \frac{1}{2}0.0001F(t)^2$ を引いたもので与えた．

$$R(\mathbf{x}(t), \mathbf{u}(t)) = \exp\left[-\frac{1}{2}\theta(t)^2\right] - \frac{1}{2}0.0001F(t)^2 \tag{3.23}$$

[5] $I_\bullet$ は $\bullet$ 次元の単位行列

比較のため 2 つの手法（SSRL，標準的な強化学習）を用いて実験を行った．2 つの手法の設定を以下に述べる．

## 3.1 SSRL

下位層の状態予測モデル $f_i$ は以下のような線形モデルを用意した．

$$f_i(\mathbf{x}, \mathbf{u}) = A_i(\mathbf{x} - \mathbf{x}_i^f) + B_i\mathbf{u} + c_i \tag{3.24}$$

$$\tag{3.25}$$

また責任信号 $\lambda_i^f(t)$ の事前予測値 $\hat{\lambda}_i^f(t)$ は空間的局所性に基づくもののみ用いた．

$$\hat{\lambda}_i^f(t) = \frac{1}{(2\pi)^{D_s/2}|\Sigma_i^f|^{D_s/2}} \exp\left[-(\mathbf{x}(t) - \mathbf{x}_i^f)^T {\Sigma_i^f}^{-1}(\mathbf{x}(t) - \mathbf{x}_i^f)\right] \tag{3.26}$$

予測誤差の分散 $\Phi_i^f$ は等方であると仮定し，各対角成分を $0.1^2$ で固定とした．その他のパラメータ $A_i \in \Re^{D_s \times D_s}$，$B_i \in \Re^{D_s \times D_c}$，$c_i \in \Re^{D_s}$，$\mathbf{x}_i^f \in \Re^{D_s}$，$\Sigma_i^f \in \Re^{D_s \times D_s}$ は EM アルゴリズム（付録 E 章を参照）を用いて予め学習したものを実験に用いた．状態予測モデルの個数 $N^f$ が多すぎると予測に寄与しない状態予測モデルが発生してしまうため，責任信号の平均値 $(\langle 1 \rangle_i)$ が限りなく 0 に近くなり M ステップにおいてパラメータの推定値が無限大となってしまう．今回の実験では状態予測モデルの個数が 100 個以上の場合においてその様な現象が見られたため $N^f = 100$ とした．

責任信号の空間的局所性の役割はモジュールの頻繁な切り替わりを抑える事である．しかしながら今回の実験では状態予測モデルの学習をオフラインで行ったため，責任信号の空間的局所性を仮定しなくとも安定して学習が可能である．また，学習が十分に収束し各状態予測モデルの分化が実現されている場合，頻繁な切り替わりはおきにくいと思われる．以上の理由から事前責任信号の空間的局所性を用いなかった．

上位層の下位報酬関数は獲得された状態予測モデルをもとにし，次のように自動的に設定した．ある抽象化状態 $i$ の中心 $\mathbf{x}_i^f$ と他の抽象化状態の中心の間の距

離を調べ，その距離が短かった順に自分自身を含めた 5 つを選択する．そうして選ばれた各抽象化状態の中心 $\mathbf{x}_i^f$ に頂点を持ち，その領域の逆共分散 $\Sigma_i^{f\,-1}$ を傾きとした 2 次関数 $-(\mathbf{x}(t)-\mathbf{x}_i^f)^T\Sigma_i^{f\,-1}(\mathbf{x}(t)-\mathbf{x}_i^f)$ を下位報酬関数として設定した．また行動価値関数 $Q$ は Adaptive RTDP（付録 C 章）を用いて行い，行動選択は $\epsilon$-greedy[38] を用い，1/10 の確率でランダムな行動選択を行うとした．また行動価値関数の時定数 $\tau$ は 2 とした．

下位価値関数 $V_{ij}$ はすべての状態予測モデル $f_i$ と下位報酬関数 $r_{ij}$ の組み合わせに対して用意し，NGNet[28] を用いて学習した．下位価値関数 $V_{ij}$ は $1\times5\times5\times5$ 個の基底を状態予測モデル $f_i$ の中心 $\mathbf{x}_i^f$ の周りに配置し，exponential eligibility trace[13] によってパラメータを更新した．その際，学習係数は 0.1，下位価値関数の時定数 $\xi$ は 1.0，適格度の係数は 0.5 とした．下位層の出力 $\mathbf{u}$ は各時刻における上位層の状態と行動に対応した状態予測モデル $f_I$ と下位価値関数 $V_{IJ}$ を用いて

$$\mathbf{u}(t) = S^{-1}\left(\left(\frac{\partial f_I(\mathbf{x}(t),\mathbf{u}(t))}{\partial \mathbf{u}}\right)^T\left(\frac{\partial V_{IJ}(\mathbf{x}(t))}{\partial \mathbf{x}}\right)^T\right) \tag{3.27}$$

とした．

## 3.2 標準的な強化学習

階層構造を持たない標準的な強化学習手法で行動則の獲得を行った．価値関数の近似には NGNet を用いた．NGNet の基底は $1\times20\times25\times25$ 個を状態空間へ格子状に配置し，報酬 $R(t)$ をもとに学習を行った．行動出力 $\mathbf{u}(t)$ は SSRL の下位層と同じ様に状態予測モデルと価値関数の勾配をもとに生成した．状態予測モデルは SSRL の場合と同じ条件で事前に学習したものを用いた．学習に用いた価値関数の学習係数，時定数とその適格度の係数および基底の個数と分散は SSRL の下位層と同じものを用いて公平な条件になるように設定した．

## 3.3 結果

長さが 10000 試行の実験を 10 回行った時の結果を図 3.6 に示す．各点は 1000 試行毎，つまり計 10000 試行分の平均を表す．また上下の線分は標準偏差を表す．

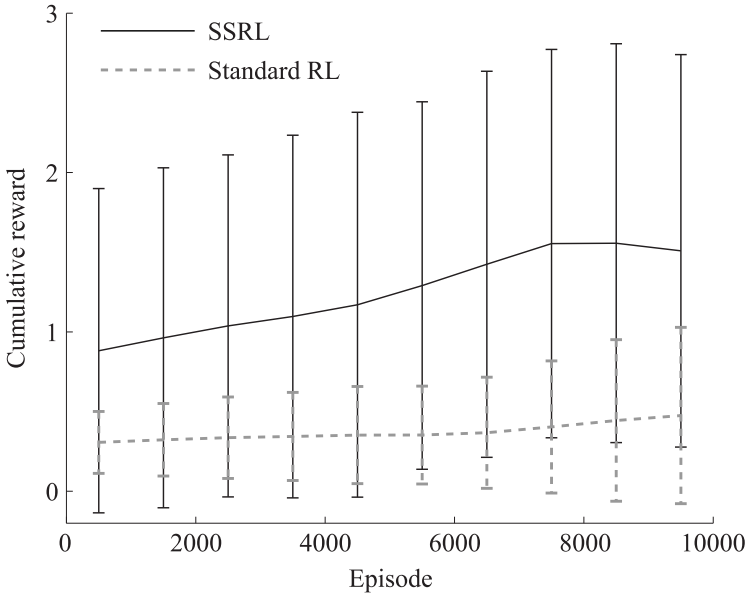
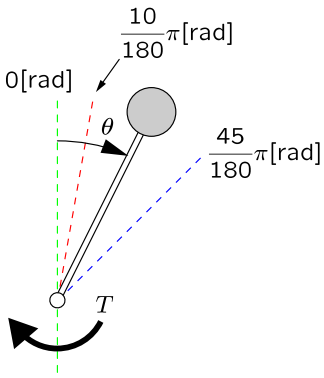



図 3.6 2 つの手法による累積報酬の変化．乱数の種を変えながら 10 回実験を行った．10 回の実験と 1000 試行における平均値と標準偏差を表す．実線が SSRL，破線が標準的な強化学習による結果を表す．

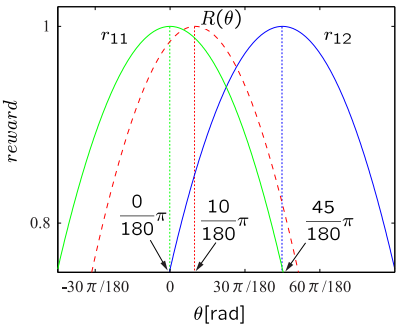
価値関数の学習に用いた関数近似器，状態予測モデルの性能は同じであるにもかかわらず提案手法が勝る結果となった．標準的な強化学習は連続な状態空間全体に報酬が伝播するのを待たなければ振り上げられないが，SSRL は抽象化された状態空間で報酬を伝播させる事が出来るためはるかに高速に学習を行えた．SSRL は早い段階で報酬が広く伝播していたが，標準的な強化学習の場合は初期状態の付近に報酬が伝播する前に実験が終わってしまいほとんど振り上げる事ができず，最後の 100 試行における振り上げの成功率は SSRL が 39.875[%]，標準的な強化学習が 2.125[%] であった．

SSRL の実験において公平な比較を行うため NGNet を用いて下位価値関数の学習を行ったが，下位価値関数 $V_{ij}$ は組になっている状態予測モデル $f_i$ と下位報酬関数 $r_{ij}$ を環境の代替とする事で直接制御則 $\mathbf{u}_{ij}(\mathbf{x})$ を導ける．特に今回の実験の様に状態予測モデル $f_i$ が線形，下位報酬関数 $r_{ij}$ が 2 次形式の場合は linear quadratic

controller (LQC)[4] として容易に求められる [15, 46, 48]．提案手法の実用性を高めるためには下位の行動側を LQC として与えた方が良く，今回の実験と同じ条件で下位の行動則を LQC とした場合の成功率は最初の 100 試行の段階で 91.5[%] となった．




(a) 単振り子




(b) 報酬関数と下位報酬関数

図 3.7 (a) は制御対象とした単振り子を表す．緑と青の破線は基底として用意した下位報酬関数の頂点の角度，赤の破線は環境から与えられる報酬の頂点の角度を表している．(b) は報酬関数と下位報酬関数をそれぞれ表示したものである．緑と青の実線はそれぞれ下位報酬関数 $r_{11}$，$r_{12}$ を表しており，赤の破線が環境からの報酬 $R(\theta)$ を表している．

# 4. シミュレーション: 下位状態価値関数の最適化

この節では本章 2.3 節で定式化を行った下位状態価値関数の最適化アルゴリズムを検証する．制御対象は図 3.7(a) の様な単振り子を用い，状態変数 $\mathbf{x}(t)$ と制御変数 $\mathbf{u}(t)$ はそれぞれ

$$\mathbf{x}(t) = [\theta(t) \ \ \dot{\theta}(t)]^T \tag{3.28}$$

$$\mathbf{u}(t) = [T(t)] \tag{3.29}$$

とした．状態予測モデルは線形モデルを採用し，$\mathbf{x} = [0 \ \ 0]^T$ 付近を近似するためのものを 1 つだけ用意した．

$$f_1(\mathbf{x}(t), \mathbf{u}(t)) = \begin{bmatrix} 0 & 1 \\ 9.8 & -0.01 \end{bmatrix} \mathbf{x}(t) - \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{3.30}$$

環境からの報酬関数 $R(\mathbf{x}(t), \mathbf{u}(t))$ は振り子の角度と出力のコストに応じて与え，

振り子が右に $\frac{10}{180}\pi$[rad] 傾いた状態で最大となるような設定とした．

$$R(\mathbf{x}(t),\mathbf{u}(t)) = \cos\left(\theta(t) - \frac{10}{180}\pi\right) - \frac{1}{2}0.001T(t)^2 \tag{3.31}$$

下位報酬関数は上に凸な2次形式とし，0[rad] で最大となる $r_{11}(\mathbf{x}(t),\mathbf{u}(t))$ と $\frac{45}{180}\pi$[rad] で最大となる $r_{12}(\mathbf{x}(t),\mathbf{u}(t))$ の 2 つを用意した．

$$r_{11}(\mathbf{x}(t),\mathbf{u}(t)) = \theta(t)^2 - \frac{1}{2}0.001T(t)^2 \tag{3.32}$$

$$r_{12}(\mathbf{x}(t),\mathbf{u}(t)) = (\theta(t) - \frac{45}{180}\pi)^2 - \frac{1}{2}0.001T(t)^2 \tag{3.33}$$

図 3.7(b) に報酬関数および下位報酬関数を表示した．赤い破線が式 (3.31) の報酬関数を表し，緑の実線が下位報酬関数 $r_{11}(\mathbf{x}(t),\mathbf{u}(t))$，青の実線が下位報酬関数 $r_{12}(\mathbf{x}(t),\mathbf{u}(t))$ をそれぞれ表す．この図から分かる様に 2 つの下位報酬関数の頂点と環境からの報酬の頂点は一致しておらず，2 つの下位報酬関数はどちらが選ばれたとしても環境からの報酬を最大とする行動則は得られない．

次に下位状態価値関数の説明を行う．基底となる下位状態価値関数は 2 つ用意し，下位状態価値関数 $v_{11}(\mathbf{x}(t),\mathbf{u}(t))$ は状態予測モデル $f_1(\mathbf{x}(t),\mathbf{u}(t))$ と下位報酬関数 $r_{11}(\mathbf{x}(t),\mathbf{u}(t))$，また下位状態価値関数 $v_{12}(\mathbf{x}(t),\mathbf{u}(t))$ は状態予測モデル $f_1(\mathbf{x}(t),\mathbf{u}(t))$ と下位報酬関数 $r_{12}(\mathbf{x}(t),\mathbf{u}(t))$ のパラメータを用い，時定数 $\xi = 1$ のもとで最適なものを導出した．その詳細は付録 D にまとめた．また拡張された下位状態価値関数は 2 つ用意し ($N^c = 2$)，重みパラメータ $w$ と下位状態価値関数 $v$ を用いて以下の様に表した．

$$V_1^1(\mathbf{x}(t),\mathbf{u}(t)) = w_{11}^1 v_1 1(\mathbf{x}(t),\mathbf{u}(t)) + w_{12}^1 v_{12}(\mathbf{x}(t),\mathbf{u}(t)) \tag{3.34}$$

$$V_1^2(\mathbf{x}(t),\mathbf{u}(t)) = w_{11}^2 v_1 1(\mathbf{x}(t),\mathbf{u}(t)) + w_{12}^2 v_{12}(\mathbf{x}(t),\mathbf{u}(t)) \tag{3.35}$$

重みパラメータの初期値は $w_{11}^1 = w_{12}^2 = 1$，$w_{12}^1 = w_{11}^2 = 0$ とし，式 (3.19) を評価関数として勾配法により更新した．その時の学習係数は 10 とした．下位状態価値関数の最適化は重み $w_{ij}^k$ の更新のみ行い，基底 $v_{ij}(\mathbf{x}(t),\mathbf{u}(t))$ の書き換えは行わなかった．

上位層の行動選択確率は Boltzmann 分布とし式 (3.16) に従って算出した．パラメータは学習係数 $\alpha = 0.1$，時定数 $\tau = 2$，行動選択の逆温度は $\beta = trial$ として

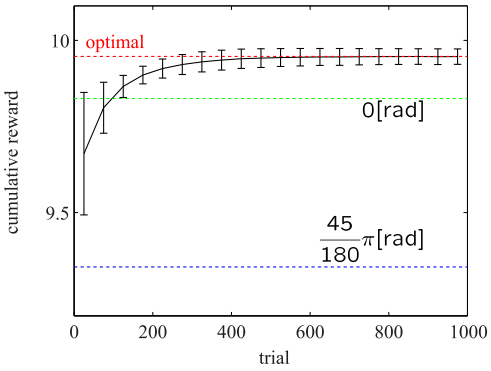


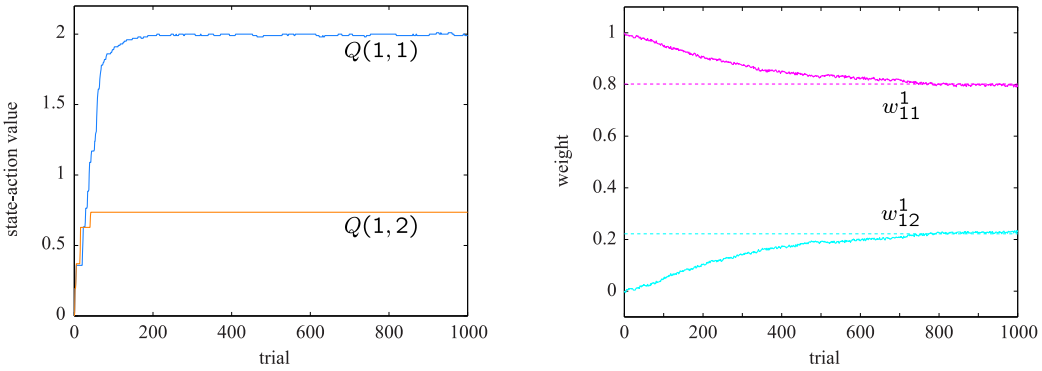

図 3.8　累積報酬の変化．それぞれ 50 試行と 100 回の実験の平均値と標準偏差を表示した．赤の破線は最適値を表している．また緑と青の破線はそれぞれ下位状態価値関数 $v_{11}(\mathbf{x}(t), \mathbf{u}(t))$，$v_{12}(\mathbf{x}(t), \mathbf{u}(t))$ のみを制御に用いた場合の値を示している．

焼きなましを行った．上位層の更新は離散状態が変化した時に行うとしたが，この実験では状態予測モデルを 1 つしか用意していない．そこで 10[sec] に 1 回の割合で式 (3.17) の更新を強制的に行う設定とした．

単振り子の初期状態は $\mathbf{x}(0) = [-\frac{30}{180}\pi \sim \frac{30}{180}\pi \;\; 0]^T$ の間で一様な乱数として与えた．シミュレーションの時間幅は 0.02[sec] とし，時間が 10[sec] を超えるかもしくは振り子の角度が $-\frac{90}{180}\pi < \theta < \frac{90}{180}\pi$ の範囲を超えた場合を試行の終了とした．1000 試行を 1 回の実験とし，乱数の種を変えながら合計で 100 回実験を行った．状態 $\mathbf{x}(t)$ と報酬 $R(t)$ にはそれぞれ分散が $0.01^2$ と $0.1^2$ の正規分布を観測ノイズとして与え，またシステムノイズとして $\dot{n}(t) = -n(t) + N(0, 2)$ を与えた．

## 4.1 結果

図 3.8，3.9 に実験の結果を表す．図 3.8 は 1 試行中の累積報酬の学習曲線を表している．緑の破線は下位報酬関数 $v_{11}(\mathbf{x}(t), \mathbf{u}(t))$ のみを制御に用いた場合，青の




(a) 状態行動価値関数 $Q(i,k)$ の変化　　(b) 重みパラメータ $w_{ij}^k$ の変化

図 3.9　100 回のうち 1 回の実験を例として表示したものである．(a) は上位の状態行動価値関数の変化を表しており，拡張された 2 つの下位状態価値関数に対する価値を表している．この例では $k=1$ に対する価値が大きくなっており，拡張された下位状態価値関数 $V_1^1(\mathbf{x}(t))$ が主に使用されている事が分かる．(b) は拡張された下位状態価値関数 $V_1^1(\mathbf{x}(t))$ を決定している重みパラメータ $w_{11}^1$，$w_{12}^1$ の変化を表示している．破線はそれぞれの重みの最適値を表している．

破線は下位報酬関数 $v_{12}(\mathbf{x}(t),\mathbf{u}(t))$ のみの場合，また赤の破線は最適値[6]を表している．なお，それぞれの破線は単振り子の初期状態を $\mathbf{x}(0)=[-\frac{30}{180}\pi \sim \frac{30}{180}\pi \;\; 0]^{\mathrm{T}}$ の間で一様な乱数として与えて制御をそれぞれ 1000 回行い，その平均値を表示してある．黒色の実線が下位状態価値関数の最適化を行った場合を表しており，50 試行毎に幅を区切って 100 回の実験における平均と標準偏差を表示した．この図より個々の下位状態価値関数 $v_{11}(\mathbf{x}(t),\mathbf{u}(t))$，$v_{12}(\mathbf{x}(t),\mathbf{u}(t))$ を用いたのでは最適な性能を得られないが，重みパラメータ $w_{ij}^k$ の最適化を行う事により最適な性能を得られている事が分かる[7]．

[6]報酬関数 (式 (3.31)) を $\mathbf{x}=[\frac{10}{180}\pi \;\; 0]$ の付近で 2 次関数を用いて近似したものを下位報酬関数とし，そこから付録 D 章の手法を用いて解析的に導いた最適下位状態価値関数を制御に用いた結果を最適値として採用した．その際の状態予測モデルは式 (3.30) を用い，時定数は $\xi=1$ とした．

[7]最適値は初期状態を乱数として与えた場合の平均値としているため，標準偏差が赤の破線で示した最適値を超えている場合がある．

最適化によるパラメータの変化の一例を図 3.9 に示す．図 3.9(a) は拡張された下位状態価値関数 $V_1^1(\mathbf{x}(t), \mathbf{u}(t))$(式 3.34)，$V_1^2(\mathbf{x}(t), \mathbf{u}(t))$(式 3.35) に対する価値を表している．拡張された下位状態価値関数 $V_1^2(\mathbf{x}(t), \mathbf{u}(t))$ に対する価値はおよそ 50 試行目で上昇が止まっている一方 $V_1^1(\mathbf{x}(t), \mathbf{u}(t))$ に対する価値は最大値[8]まで上昇している．このことから拡張された下位状態価値関数 $V_1^1(\mathbf{x}(t), \mathbf{u}(t))$ が制御に貢献していると分かる．そこで拡張された下位状態価値関数 $V_1^1(\mathbf{x}(t), \mathbf{u}(t))$ を決定している重みパラメータ $w_{11}^1$，$w_{12}^1$ の変化を図 3.9(b) に示した．紫と水色の実線が重み $w_{11}^1$，$w_{12}^1$ を表しており，破線はその最適値となっている．

[8] $\theta$ に関する環境からの報酬は $\theta = \frac{10}{180}\pi$ で最大となるが，出力のコストが影響するため累積報酬を最大とするための姿勢は $\theta \simeq \frac{6.589}{180}\pi$ となる．その姿勢で定常状態となった時の報酬値は $R(t) \simeq 0.993$ である．この実験において上位層の時定数は $\tau = 2$ としているため，$Q(1, 1)$ の最大値は 1.986 となる．

# 5. 比較

本研究で提案したSSRLを含め，これまでに複数の階層型強化学習が提案されている．それらは環境を抽象化する方法や報酬の取り扱い，必要な事前知識などの違いによって性能や適応可能な課題が特徴付けられる．この節ではそれぞれの手法の特徴をまとめてSSRLとの比較を行う．比較は以下に示した6つの手法を対象とする．

- SSRL
- Morimotoらにより提案されている手法 (以下Morimoto方式) [29, 14]
- Option [39, 40]
- Feudal Reinforcement Learning (以下Feudal RL) [8]
- Composite Q-learning (以下CQL) [36]
- MAXQ Value Function Decomposition (以下MAXQ) [11]

それぞれの手法は序章にて説明を行っている．比較の結果は表3.2にまとめてある．

まず分節化の方法において大きく2つに分けられる．1つ目は環境の状態空間を局所領域に分割する手法であり，SSRL，Morimoto方式，Option，Feudal RLがそれに相当する．SSRLの分割方法は状態予測モデルの空間的局所性と予測誤差に基づいており，ダイナミクスの入力・出力の両方で分割することができる．ダイナミクスの入出力データをもとに状態予測モデルを学習することで，似たような入出力関係を持つ領域ごとに環境を自動的に分節化できる点が特徴である．Morimoto方式は環境の状態のうち必要な次元を抜き出したうえで離散化を行っており，上位層は縮約された状態空間を扱う事で大域的な探索を効率よく行えるとしている．状態空間のうちどの次元を削除するのか，および離散化の粒度は設計者が決定しなければならない．Optionにおける分節化は状態空間上の局所化と上位層によるモジュールの選択の2つの基準で行われている．Optionの各モジュールは状態空間上に担当範囲と方策をもっており，分節化する際はまず各状態に対して有効になっているモジュールを調べる．候補が複数ある場合，上位層は価値関

数に基づいて候補の中から適切な方策を持っているモジュールを選択し，その出力が環境に与えられる．最後に環境からの報酬を観測することで上位の価値関数の更新を行う．いったんモジュールが選択されると設定された終端確率によって選択が解除されるまで同じモジュールの方策が制御に用いられる．頻繁な行動選択を避けることで大域的な探索が可能となる手法であるが，各モジュールの担当範囲と終端確率は設計者が事前に設定しなければならない．Feudal RL は状態空間を複数の局所領域に分けることで分節化を行っており，異なる解像度によって局所化された層が複数個設けられている点に特徴がある．モジュールは上位になるほど広い担当範囲を持っており，まず最上位層に所属するモジュール群の中から現在の状態を担当範囲に持つものが一つ選択される．選択されたモジュールには一つ下の層のモジュールのいくつかが支配下に置かれており，支配下にある各モジュールのなかから現在の状態を担当範囲に持つものがさらに選択される．そして支配下のモジュールに対してサブゴールを指定し，指定された下位のモジュールはさらに自分の支配下にあるモジュールに対してサブゴールを指定する．各層のモジュール群がもつ担当範囲に重複は無い．異なる複数の解像度においてそれぞれ探索を行う階層型強化学習の一般的な枠組みと言うのがこの手法の位置付けであり，いくつの層を設けるかや局所化の解像度は設計者が問題に応じて決めなければならない．これら 4 つの手法を比べると Morimoto 方式と Feudal RL はある状態に対するモジュールの候補は一つのみであるのに比べ[9]，Option と SSRL は複数の候補を持つことが出来る．Option は各モジュールの担当範囲が重複することが許されているためある状態に対して複数個のモジュールの候補が挙がる場合があり，上位層の状態行動価値関数に基づいてその中から適切なものが選択される．制御の目標が変化した場合，ある程度まとまった領域ごとにモジュールを切り替えられるため高速な再適応が行えると言う特徴がある．SSRL も同じく状態空間上での分割に加えてダイナミクスの出力空間上でも分割を行っており，複数のダイナミクスが非定常に切り替わるような環境への適応を可能としている点が特徴である．また分節化を行うモジュールと各文節における行動即を決定する

[9]Feudal RL は 3 層以上の層構造を持つことが出来るため複数の候補があるように思える．しかし環境への出力を行う最下層において，状態とモジュールは 1 対 1 の対応となっているため状態と方策も 1 対 1 となる．

モジュールは別々に選択することが出来るため，Option と同様に目標の変化に対する高速な再適応が期待できる．2 つ目はタスクが複数のサブタスクに分解できると仮定し，各サブタスクの終端状態に基づいて分節化を行う手法である．CQL と MAXQ がこれに相当する．CQL はタスクの最小単位 (elemental task) を考え，それらの時系列によって解くべきタスク (compositional task) が構成されていると仮定した上で定式化が行われている．下位層には各 elemental task を解決する状態行動価値関数が，そして上位層にはそれらを切り替える gating module と真の価値との差分を学習する bias module が配置されている．一旦，各 elemental task に対する状態行動価値関数の学習を行えば，compositional task の構成に変化があっても上位層の学習のみで対応でき，かつ上位層の学習結果は保持されるという特徴がある．そして下位層の価値関数と上位層の bias module の出力を足し合わせることで，compositional task に対する真の価値関数を表現できる利点がある．しかしながらそれぞれの elemental task，compositional task を識別するための信号を gating module に入力する必要があり，また適応できるタスクは報酬の割引が無い場合に限られる．MAXQ も CQL と同じく，各サブタスク毎にモジュールを担当させる手法であるが，異なる時間解像度毎にサブタスクを仮定している点で異なる．そしてサブタスク間の関係を task graph と呼ばれる木構造であらわしており，木構造の頂点に近い層では長い時間間隔，下位の層では短い時間間隔でタスクの分割行うものとしている．各層において有効になっているモジュールの組み合わせによって複数の文脈を識別でき，各モジュールが獲得している価値関数の和により各文脈に対する真の価値関数を表現できる特徴がある[10]．しかしながらタスクの分割方法や task graph など必要とする事前知識は多い．

次に状態空間を分割する 4 つの手法 (SSRL，Morimoto 方式，Option，Feudal RL) を報酬の扱いと制御器の選択に関して比較してみる．それぞれの手法は環境からの報酬は最上位層にのみ入力され，上位層が選択した局所的な目標の達成度に応じて下位層への報酬が決定する点で共通している．また達成度の評価方法に

[10]Dietterich は “taxi problem” という簡単な問題を例として紹介している．上位層が客を「迎えに行く」と「目的地まで運ぶ」という 2 つのサブタスクを選択し，下位層が実際に環境への行動出力を行う．客を乗せているかどうか 2 つの文脈が存在するが，上位層が持つ 2 つの価値関数と下位の価値関数を組み合わせることにより，混同することなく個別に学習が行える．

よって下位の方策が決定されるため，その評価方法が全体の性能を大きく左右すると言う点も共通している．よってタスク毎に設計者が評価方法を作りこむ必要があるが，SSRL は下位の方策を改善する枠組みが提案されているためその様な作業を軽減できる期待がある．SSRL は初期の評価方法から導かれる下位の価値関数を線形和する事で新たな価値関数を作り出し，環境からの報酬が最大となるようにその重み付けを求める．

| | 分節化の対象 | 非定常性への対応 | 最適性 | 必要な事前知識 |
|---|---|---|---|---|
| SSRL | ダイナミクス | ダイナミクス | 真の価値関数を近似可能 | 下位層への報酬関数 |
| Morimoto 方式 | ダイナミクス | × | 分節化・下位層への報酬関数の初期設定に依存 | 分節化の方法<br>下位層への報酬関数 |
| Option | ダイナミクス | × | 分節化・下位層への報酬関数の初期設定に依存 | 分節化の方法<br>下位層への報酬関数 |
| Feudal RL | ダイナミクス | × | 分節化・下位層への報酬関数の初期設定に依存 | 分節化の方法<br>下位層への報酬関数 |
| CQL | 報酬関数 | 報酬関数 | 真の価値関数を獲得可能 | サブタスクの終端条件 |
| MAXQ | 報酬関数 | 報酬関数 | 真の価値関数を獲得可能 | サブタスクの終端条件<br>サブタスク同士の有向グラフ |

表 3.2 階層型強化学習手法の比較表.

# 6. 本章の議論

本章では連続システムを対象とした階層型強化学習手法の提案を行った．下位層が環境への制御出力と感覚フィードバックの入出力特性に基づいて連続システムの抽象化を行い，上位層がその抽象化された空間で報酬予測を行うと言う構造になっている．抽象化された空間での状態遷移はSMDPの枠組みで記述され，上位層は粗い時間幅で行動選択を行える．上位層が大域的な探索を行い，下位層が局所的な探索を行う事で効率よく学習が行える．下位層の状態予測モデル群が上位層の抽象化を実現しており，ダイナミクスが非定常な場合でも適応的に抽象化を行える事が新規性のひとつである．しかし上位層のダイナミクスと報酬関数の分布は下位の連続状態と連続行動によって変化するため，実際は隠れ状態があるSMDPとして定式化を行わなければならなず今後の課題のひとつである．

Suttonらはoptionと言う概念を導入する事によりMDP環境をSMDP環境で抽象表現する枠組みを定式化したが，今回我々は連続システムをSMDP環境に抽象化する手法を定式化できたと言える．Morimotoらの方式[29]もSSRLと同じ様に上位層が離散空間で局所的な目標を学習し，下位層はその達成度に応じた報酬を得る．Morimoto方式において状態空間の離散化は設計者が行わなければならないが，SSRLではダイナミクスの入出力データから自動的に分節化が行え，またシミュレーションにより実際に自動的に学習できる事が示せた．またMorimoto方式もSSRLも下位報酬関数の形状によって性能が左右されるが，SSRLでは複数の下位状態価値関数の重み付和によりある程度最適化を行う枠組みが定式化されており，シミュレーションによる検証を行えた．

連続変数で記述される事象と離散変数で記述される事象が混在するシステムはHybrid Dynamical Systemと呼ばれ，非線形で大規模な制御対象をHybrid Dynamical Systemとしてモデル化することで制御器の性能が向上することが知られている．近年，システム制御理論や計算機科学の分野で強い関心を集めており，モデル予測制御や画像認識への応用が行われている[51, 50]．Hybrid Dynamical Systemは解析や安定化のための研究が制御の分野でさかんに行われている．本稿では連続な状態と離散な抽象化状態が混在したHybrid Dynamical Systemを階層型強化学習により制御する枠組みを提案したと言え，MorimotoらによるRobust

Reinforcement Learning[30] を適用する事で学習の面からも安定性などを論じられると期待できる．

下位層は上位層が選択した下位報酬関数に対応した下位価値関数を学習してその勾配をもとに制御出力を行う．上位層は報酬を最大化するような下位報酬関数を選択するため，それに対応する下位価値関数の勾配は最適出力を行うためのものとなっている．しかし上位層の行動価値関数は下位報酬関数によって離散化されているため，制御出力は与えられた下位報酬関数に対してのみ最適化が行われている．真に最適な出力を行うためには下位報酬関数の更新則を確立しなければならず，今後の大きな課題である．

# 第4章　階層型モジュール強化学習を用いたコミュニケーションのモデル化

## 1. 運動能力とコミュニケーション

言語に代表されるようなシンボルを生み出したり操作する能力はヒトにのみ備わっており，ヒト以前の霊長類とはわずかな連続性もなく大きな隔たりが存在する [9, 21]．連続な環境をシンボル表現する事で高度な予測や推論，そして他者と知識の共有が可能となりヒトの進化を支えてきたと思われる．ある集団内で共通な意味を持つシンボルはどのように発生するのであろうか．中野はエージェント間で共通に体験された事象に対してシンボルを割り当てていく事が最も自然な考えだとしている [45] [1]．相手が送信してきたシンボルと環境の観測を繰り返す事でシンボルと環境の事象が同時に出現する確率を推定する事で対応関係を獲得できるのであるが，実世界の様に時間・状態・行動が連続値を持ちかつ状態空間の次元数が高い場合は観測された変数をそのまま扱うのは現実的ではない．観測値から何らかの特徴量を推定してシンボルの獲得に必要なサンプル数を減らす作業が必要となる．我々の日常においても観測された他者の運動軌道からその裏に隠れた高次の情報を推定しながら自分の行動を決定しており，ヒトのコミュニケー

[1] ある 2 人のエージェントが異なる部屋に閉じ込められており，お互いに相手を見る事ができない状況を考える．お互いになんらかのシンボルを送信しあう事は出来るが，その意味付けや文法構造は統一されていないとする．この様な状況では共通なシンボルはいつまで経っても生成されない事は容易に想像できる．しかしその部屋には窓が取り付けられており，2 人が同じ景色を見られるとすると共通なシンボルを生成する機会が生まれる．例えば鳥が飛んでいた時に送られてきたシンボルは鳥を意味すると推定が出来る．この様にして共通なシンボルの生成には共通体験が必要なのである．

ションの機構を探る上でもこの推定問題は興味深い．しかし一般に．運動軌道のみからその裏に隠れた高次の特徴量の推定は不良設定問題となる．本章ではエージェントの制御アルゴリズムとして MOSAIC モデルに基づいたモジュール構造を仮定し，過去の経験によって獲得された運動能力がこの不良設定性を解決する鍵となる事を示す．

エージェントの制御アルゴリズムが MOSAIC モデルに基づいているとすると，ここでの問題は観測された運動軌道から相手のモジュール構造を推定する問題となる．相手のモジュール構造は観測不可能でありまた個々のエージェントのモジュール構造には個体差があるため，この推定は不良設定問題となる．そこで自分が獲得しているモジュールのうち，観測の各時刻において相手の運動軌道を最も良く説明するものを推定する事を考える．言い換えると，自分のモジュール構造を用いて相手の運動軌道を模擬するのである．我々の日常を考えるとこのアプローチはごく自然であり，Rizzolatti らにより発見されたミラーニューロン [34] の存在も正当性を支持していると言える．まず 2 節にてモジュール構造の推定手法を，そして 3 節にて共通なシンボルの生成手法の一案を定式化する．

## 2. 他者が使用しているモジュールの推定

この節では他者の運動軌道を観測し，その裏に隠れた高次の情報を推定する手法について説明する．その手法を大まかに述べると，観測された他者の軌道を自分が獲得しているモジュール群を用いて模擬し，そのなかでもっとも良く観測軌道を説明しているモジュールを他者が用いているモジュールの推定値として採用するのである．それを可能とするため，いくつかの仮定を設ける．

- 両者のダイナミクスの差はある程度小さい．
- 観測できるのは運動軌道 $\mathbf{x}^{\dagger}$ のみであり，観測に時間遅れは無いとする．
- 両者とも制御アルゴリズムとして SSRL を用いており，モジュール構造はある程度似通っている．

SSRL を用いた他者の内部状態の推定は，観測された軌道から他者の上位層の状態と行動の組を推定する問題である．簡単のため，2.3 節で説明した下位価値関数の拡張は行わないものとする．観察者は自分が獲得しているモジュール群を用いて他者の状態変化を予測し，その予測誤差が小さなモジュールほど他者の状態変化を良く説明するとする．ある状態予測モデル $f_i$ と下位報酬関数 $r_{ij}$ の組み合わせに対する評価方法は以下の通りである．観察者はまず相手の状態 $\mathbf{x}^{\dagger}(t)$ [2]を評価対象の組み合わせに対する制御器に入力し，出力の推定値 $\hat{\mathbf{u}}_{ij}(t)$ を得る．次に状態予測モデル $f_i$ に相手の状態 $\mathbf{x}^{\dagger}(t)$ と出力の推定値 $\hat{\mathbf{u}}_{ij}(t)$ を入力して状態変化を予測する．制御入力として他者が実際に出力した値ではなく自分のコントローラの出力値を用いる事で，自分のダイナミクスとモジュール構造に適した予測が行える．

$$\hat{\dot{\mathbf{x}}}_{ij}(t) = f_i(\mathbf{x}^{\dagger}(t), \hat{\mathbf{u}}_{ij}(t)) \tag{4.1}$$

自分のモジュールによって予測された状態変化 $\hat{\dot{\mathbf{x}}}_{ij}(t)$ と相手の状態変化 $\mathbf{x}^{\dagger}(t)$ の差が小さいほど，相手の上位層の状態と行動を説明する組み合わせとして尤もら

[2]区別のため披観察者の変数は † で修飾されるものとする．

しいと言える．誤差の分布が分散 $\sigma^2$ がガウス分布に従うとすると，組み合わせに対する評価は以下の様になる．

$$\lambda_{ij}(t) = \frac{G_{ij}(t)}{\sum_{i'=1}^{Nf}\sum_{j'=1}^{Nr} G_{i'j'}(t)} \tag{4.2}$$

$$G_{ij}(t) = \frac{1}{(2\pi)^{1/2}\sigma}\exp\left[-\frac{1}{2\sigma^2}\|\hat{\mathbf{x}}_{ij}(t) - \mathbf{x}^{\dagger}(t)\|^2\right] \tag{4.3}$$

ここで $\lambda_{ij}(t)$ は $0 \sim 1$ の値をとり，1 に近いほど評価が高い組み合わせとなる．他者の上位層の状態行動対 $(\hat{i}, \hat{j})$ は，最も単純には以下の様に推定できる．

$$\hat{i}, \hat{j} = \underset{i,j}{\mathrm{argmax}}\, \lambda_{ij}(t) \tag{4.4}$$

**見まね学習**　この節では学習済みのエージェントを教示者とし，その成功軌道を観測する事により短時間で解を得る見まね学習を定式化する．教示者と見まね学習者の間にダイナミクスの差異がある場合，教示者の軌道が見まね学習者にとっての最適軌道になるとは限らない．また一般的な状況を考えると観測軌道にはノイズや時間遅れなどが生じるため観測軌道をそのまま再現する見まね学習は得策ではない．ある程度大まかなレベルで見まねを行い，その後自分の環境に合わせて細かなレベルでの学習を行うべきである．

前節の手法で推定される上位層の状態行動対 $(\hat{i}, \hat{j})$ は，他者の軌道を自分のダイナミクスとモジュール構造のもとで再現しようとした時にもっとも妥当な組み合わせとなっている．そこで推定された状態行動対の時系列から求められる条件付分布を事前知識として用いる手法が考えられる．見まね学習者はまず観測時系列から状態行動対の同時出現頻度 $S(i,j)$ を求める．その後自律学習を行う際の事前知識は時刻 $t$ における状態予測モデルの責任信号を $\lambda_i^f(t)$ とすると，頻度 $S(i,j)$ を責任信号 $\lambda_i^f(t)$ で重み付けしたもの

$$P(j|i) = \frac{\sum_{i=1}^{Nf} S(i,j)}{\sum_{i=1}^{Nf}\sum_{j=1}^{Nr} S(i,j)} \tag{4.5}$$

となる．この事前知識を用いた場合の行動選択確率は，例えば Boltzmann 分布に

従うとすると式 (3.6) を以下の様に変更する事で求められる．

$$P(j|\lambda^f(t)) = \frac{P(j|i)\exp\left[\beta\sum_{i=1}^{Nf}\lambda_i^f(t)Q(i,j)\right]}{\sum_{j'}P(j'|i)\exp\left[\beta\sum_{i=1}^{Nf}\lambda_i^f(t)Q(i,j')\right]} \tag{4.6}$$

このような事前知識を用いる事で，状態行動価値関数 $Q$ が未学習であっても課題達成のための行動選択が行える．これにより，教示者と見まね学習者のダイナミクスとモジュール構造が等しい場合は即座に課題が達成できる．またそれらに差異がある場合でも探索の候補を絞れるため高速に学習が行える．

**協調作業**　複数のエージェントが相互作用を与えながら行動する場合，自分の状態だけでなく他のエージェントの状態や行動にも応じて行動を出力しなければならない．しかし観測された軌道をそのまま用いると状態空間が広くなり過ぎてしまい現実的ではない．また状態や行動が連続な場合を考えると筋指令やモータートルクが行動に相当するが，特別なセンサを用いない限り観測不可能である．他者の状態と行動をよく抽象化している信号を取り出し，それに応じて自分の行動を決定しなければならない．協調作業にはその様な困難が伴うが，SSRL を用いて推定される上位層の状態行動対 $(\hat{i},\hat{j})$ は相手の状態と行動をよく抽象化している信号と言える．そこで上位層の行動価値関数を拡張し，自分の上位層の状態 $i$ に加えて推定された状態行動対 $(\hat{i},\hat{j})$ を価値関数の状態変数に組み込む事を考える．3 者の組を $\mathbf{z}=(i,\hat{i},\hat{j})$ とすると，式 (3.5) は以下の様に書きかえられる．

$$Q^*(\mathbf{z},j) = \sum_{\mathbf{z}'}P(\mathbf{z}'|\mathbf{z},j)\int_0^\infty P(l|\mathbf{z}',\mathbf{z},j)\left\{\int_t^{t+l} e^{-\frac{s-t}{\tau}}R(s)ds + e^{-\frac{l}{\tau}}\max_{j'}Q^*(\mathbf{z}',j')\right\}dl \tag{4.7}$$

# 3. エージェント間で共通なシンボルの生成

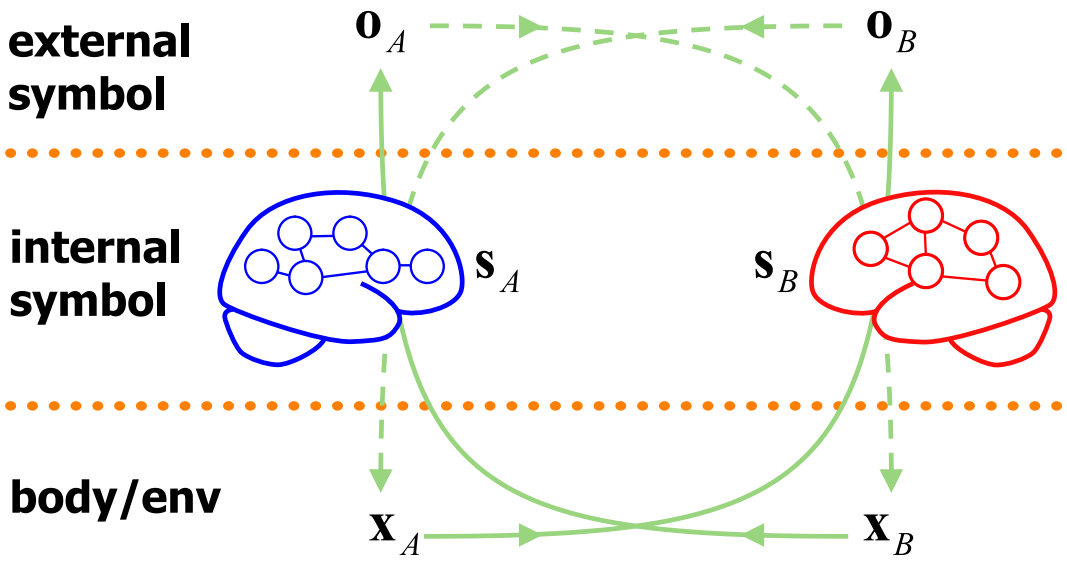



図 4.1 信号の性質に基づき 3 種類の層が考えられる．上から外部シンボル，内部シンボル，連続な運動軌道の層を表している．一般に内部シンボルの構造は個体差があるため，この層での通信は不可能となる．

複数のエージェント間で共通なシンボルが生成される一案をこの節では定式化する．これ以降では区別のため，エージェントが内的に持っているシンボルを内部シンボル (変数 $\mathbf{s}$ で表す)，そして他者に対して送信されるものを外部シンボル (変数 $\mathbf{o}$ で表す) と呼ぶことにする．図 4.1 はエージェント間における情報の伝達経路を模式化したものであり，情報の性質に応じて 3 種類 (下から順に連続軌道 $\mathbf{x}$，内部シンボル $\mathbf{s}$，外部シンボル $\mathbf{o}$) に大別できる．そしてエージェントは自分の内部シンボル $\mathbf{s}$ の状態に応じて運動軌道 $\mathbf{x}$ や外部シンボル $\mathbf{o}$ を生成し，それらをお互いに観察し合えるものとする．連続軌道 $\mathbf{x}$ がエージェントのモジュール構造を通して動的に離散化されながら外部シンボル $\mathbf{o}$ との対応付けが行われる点がこのモデルの大きな特徴である．

エージェント制御アルゴリズムとして SSRL を用いた場合について話を進めると上位層における離散状態と離散行動の対が内部シンボル $\mathbf{s}$ に相当する．エージェ

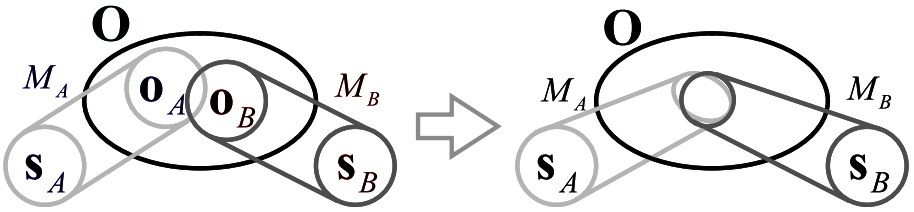



図 4.2　初期状態においては異なった外部シンボルが各エージェントの内部シンボルと対応しているが，相互作用を繰り返す事で共通な外部シンボルへの対応が学習される．

ントは内部シンボル $\mathbf{s}$ から外部シンボル $\mathbf{o}$ への写像関数 $\mathbf{o} = M(\mathbf{s})$ を持っており，図 4.2 は 2 人のエージェント (A,B) についてその関係を模式的に表したものである．コミュニケーションの初期段階においては外部シンボルの共有が行われていないが，写像関数 $M$ をある法則に従って更新を繰り返す事で共有信念が形成される．

共有信念を形成する為の更新方法について述べる．図 4.3 はエージェント A の写像関数 $M_A$ を更新する場合の流れを表したものである．エージェント A はエージェント B の運動軌道 $\mathbf{x}_B$ を観測し，エージェント B の内部シンボルの推定値 $\hat{\mathbf{s}}_B$ を得る．そして自分のシンボル写像 $M_A$ を用いて外部シンボルの推定値 $\hat{\mathbf{o}}_B = M_A(\hat{\mathbf{s}}_B)$ を求める．この時，運動軌道 $\mathbf{x}_B$ に対する意味づけが両者で等しければ発せられた外部シンボル $\mathbf{o}_B$ と推定された外部シンボル $\hat{\mathbf{o}}_B$ は等しくなるはずである．そこで，その差分が減少するようにエージェント A は自分のシンボル写像 $M_A$ を更新する．エージェント B も同様の手続きによりシンボル写像 $M_B$ を更新する．

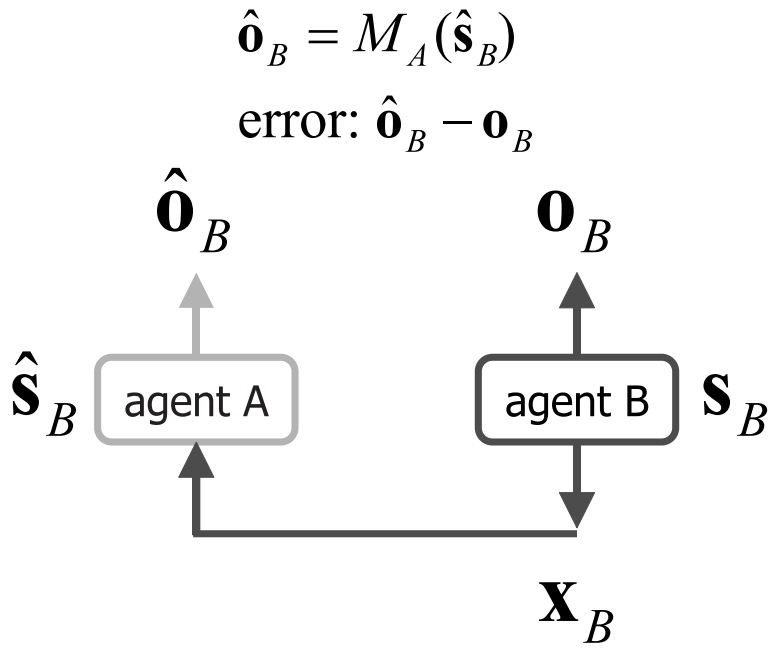



図 4.3 エージェント A はエージェント B の軌道と自分のシンボル写像 $M_A$ を用いて推定される外部シンボル $\hat{\mathbf{o}}_B$ とエージェント B が発した外部シンボル $\mathbf{o}_B$ を比較し，その誤差が減少するようにシンボル写像 $M_A$ を更新する．エージェント B も同様の更新を行う．

# 4. 実験

## 4.1 見まね学習

学習済みのエージェントを観察する事でタスクに対する知識を得る見まね学習の実験を行う．制御対象は前章のシミュレーションで用いた図 3.5 の様な倒立振子を採用する実験に参加するエージェントは教示者，見まね者 A，見まね者 B の 3 人である．教示者と見まね者は同じ物理パラメータ ($M = m = 1$[kg]，$l = 1$[m]) を持ち，見まね者 B は少し重い物理パラメータ ($M = m = 1.2$[kg]，$l = 1$[m]) を持っている．物理パラメータに個体差がある場合でも提案手法によって見まねが行える事を示す．

各エージェントは $\theta - \dot{\theta}$ 空間に対して格子状に $4 \times 7$ 個の線形な状態予測モデルを持ち，それぞれのパラメータは運動方程式を偏微分する事により求めた．また下位報酬関数の個数は 5 個とした．エージェントには振子の頭の高さに応じた報酬が与えられる．

$$R(\theta(t), \mathbf{u}(t)) = \cos(\theta(t)) - S(\mathbf{u}(t)) \tag{4.8}$$

ここで $S$ は出力に関するコストを表している．

実験はまず最初に教示者が自律学習を行った．次に 2 人の見まね者は教示者の学習が十分に収束した後の試行を観察し，教示者の上位層の状態と行動の時系列を推定した．そしてその時系列から教示者の行動選択確率 $P(j \mid i)$ を求め，それを自分の行動選択確率の事前分布として学習を開始した．教示者，見まね者共に行動選択確率は Boltzmann 分布に従うものとし，見まね者は式 (4.6) と同じ様に推定した事前分布を自分の行動選択確率に組み込んで自律学習を行った．

### 結果

図 4.4 に見まね学習の結果を表す．最初の 1000 試行は教示者のみが学習を行っており，収束までに約 400 試行を要している事が見て取れる．1000 試行以降は 2 人の見まね者による学習結果を表しているが，開始と同時に高い性能を示している事が分かる．特に見まね者 B は異なる物理パラメータを持っているにもかかわ

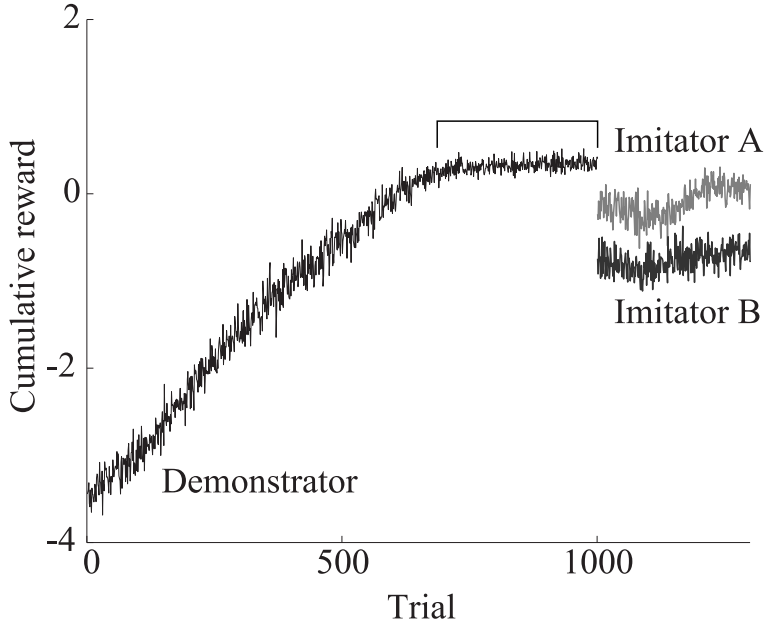



図 4.4　見まね学習の結果．最初の 1000 試行は教示者が自律学習を行い，2 人の見まね者はその後半部分 (鉤括弧で示した部分) を見まねする．その後，2 人の見まね学習者は見まねで得た行動選択確率を事前知識として自律学習を行う (1000 試行以降)．

らず見まねが成功しており，提案する見まね学習の手法はダイナミクスの個体差があっても成立する事を示せた．教示者に比べて性能が低いのは教示者に比べて重い自重を持っている為と考えられる．また見まね者 A の性能がわずかに低い原因として，推定や観測の誤差が考えられる．

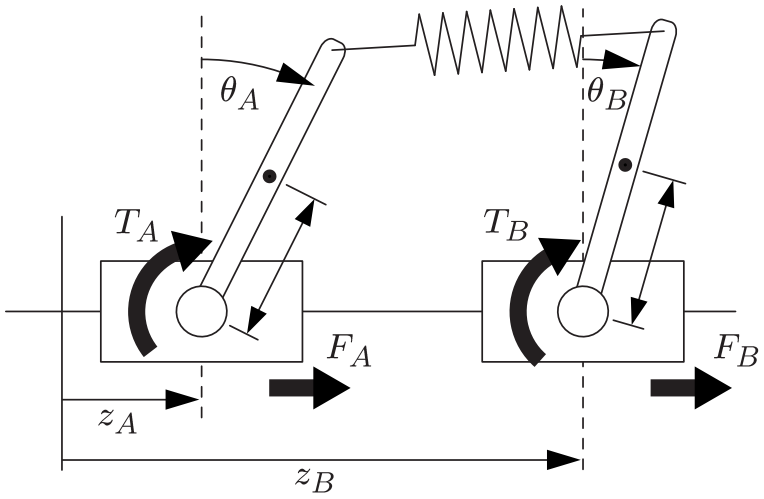



図 4.5 2 台の倒立振子を 2 人のエージェントがそれぞれ制御する．振子の先端はばね定数が 0.1[N/m] のばねで連結されており，2 人が協力しないと振り上げる事ができない．

## 4.2 協調作業

図 4.5 のような 2 台の倒立振子を用いて協調作業の実験を行う．2 人のエージェント A，B がそれぞれ 1 台の倒立振子を制御するが振子の先端がばねで連結されているため，お互いに協調しないと振り上げる事ができない．各エージェントの状態変数は

$$\mathbf{x}_A(t) = [z_A(t)\ \ \theta_A(t)\ \ \dot{z}_A(t)\ \ \dot{\theta}_A(t)]^T \tag{4.9}$$

$$\mathbf{x}_B(t) = [z_B(t)\ \ \theta_B(t)\ \ \dot{z}_B(t)\ \ \dot{\theta}_B(t)]^T \tag{4.10}$$

とし，また制御変数は

$$\mathbf{u}_A(t) = [F_A\ \ T_A]^T \tag{4.11}$$

$$\mathbf{u}_B(t) = [F_B\ \ T_B]^T \tag{4.12}$$

とした．エージェントはお互いに相手の状態のみを観測する事ができる．1 試行の長さは 20[sec] とし，シミュレーションの間隔は 0.02[sec] とした．

各エージェントの制御アルゴリズムにはSSRLを用い，エージェントAの上位層における状態を$i_A$，行動を$j_A$，状態行動価値関数を$Q_A$とし外部シンボルを$\mathbf{o}_A$とした．状態予測モデルは$\theta - \dot{\theta}$平面に$5 \times 5$個を格子状に配置し，下位報酬関数は10個用意した．またエージェントBに関してはそれぞれ$i_B$，$j_B$，$Q_B$，$\mathbf{o}_B$とし，状態予測モデルは$\theta - \dot{\theta}$平面に$5 \times 7$個を格子状に配置し，下位報酬関数は10個用意した．両エージェントの状態予測モデルと強化学習コントローラは学習済みのものを用い，上位層の状態行動価値関数の学習のみ行った．比較のため，以下の3種類の実験を行った．

**お互いに相手の状態を無視して行動選択**　相手の状態を考慮せずに行動を決定する非協力的な場合について実験を行う．各エージェントは相手を考慮しない状態行動価値関数$Q_A(i_A, j_A)$，$Q_B(i_B, j_B)$を学習する．

**推定された相手の内部シンボルに応じて行動選択**　エージェントは観測された相手の状態からその内部シンボルを推定し，それを組み込んだ状態行動価値関数を学習する．制御の各時刻においてエージェントAはエージェントBの状態$\mathbf{x}_B(t)$とその変化$\dot{\mathbf{x}}_B(t)$を観測し，エージェントBの内部シンボルの推定値$\hat{i}_A$，$\hat{j}_A$を得る[3]．そしてエージェントAの上位層はその推定値を組み込んだ状態行動価値関数$Q_A(\{i_A, \hat{i}_A, \hat{j}_A\}, j_A)$を学習する．エージェントBに関しても同様に，エージェントAの状態$\mathbf{x}_A(t)$とその変化$\dot{\mathbf{x}}_A(t)$からそれを生成している内部シンボルの推定値$\hat{i}_B$，$\hat{j}_B$を求め，その推定値を組み込んだ状態行動価値関数$Q_B(\{i_B, \hat{i}_B, \hat{j}_B\}, j_B)$を学習する．

**送信された外部シンボルに応じて行動選択**　エージェントは相互作用を通して共通の外部シンボルを生成し，それを組み込んだ状態行動価値関数を学習する場合の実験を行う．制御の各時刻においてエージェントAはエージェントBが発した外部シンボル$\mathbf{o}_B$を組み込んだ状態行動価値関数$Q_A(\{i_A, \mathbf{o}_B\}, j_A)$を学習する．それと同時に，自分の内部シンボルに応じた外部シンボル$\mathbf{o}_A = M_A(i_A, j_A)$をエー

[3]この推定値はエージェントBの内部シンボルに対するものであるが，エージェントAの内部シンボルを代替とした推定値であるため“A”を修飾してある．

ジェントBに向けて送信する．シンボル写像$M_A$は各内部シンボルに対して1つの外部シンボルが対応するものとした．初期状態においてシンボル写像$M_A$は，各内部シンボルから20000個の外部シンボルへのランダムな写像であるが，以下の様にして更新を行った．

- エージェントBの状態$\mathbf{x}_B(t)$とその変化$\dot{\mathbf{x}}_B(t)$からエージェントBの内部シンボルの推定値$\hat{i}_A$，$\hat{j}_A$を得る．
- 自分のシンボル写像に上記の推定値を入力し，推定値$\hat{\mathbf{o}}_B = M_A(\hat{i}_A, \hat{j}_A)$を得る．
- エージェントBが発した外部シンボル$\mathbf{o}_B$を観測する．
- $\mathbf{o}_B$と$\hat{\mathbf{o}}_B$が異なる場合，1/10の確率でシンボル写像$M_A$を$\mathbf{o}_B = M_A(\hat{i}_A, \hat{j}_A)$となるように書き換える．

エージェントBに関しても同様に，エージェントAが発した外部シンボル$\mathbf{o}_A$を組み込んだ状態行動価値関数$Q_B(\{i_B, \mathbf{o}_A\}, j_B)$の学習と外部シンボル$\mathbf{o}_B = M_B(i_B, j_B)$の送信，そしてシンボル写像$M_B$の更新を行う．

**結果**

図4.6に実験の結果を表す．それぞれの場合について10回ずつシミュレーションを行い，その平均値を表示した．共通なシンボルを用いる場合が最も良い性能を示せている．

2人のエージェント間で共有信念が形成されていたかどうかを調べる．図4.7は発せられた外部シンボルと推定された外部シンボルの一致率を表示している．青色がエージェントA，赤色がエージェントBに関する一致率である．エージェントのモジュール構造に差異があるため完全に一致はしないが，最終的にはほぼ一致しており共有信念が形成されていたと言える．

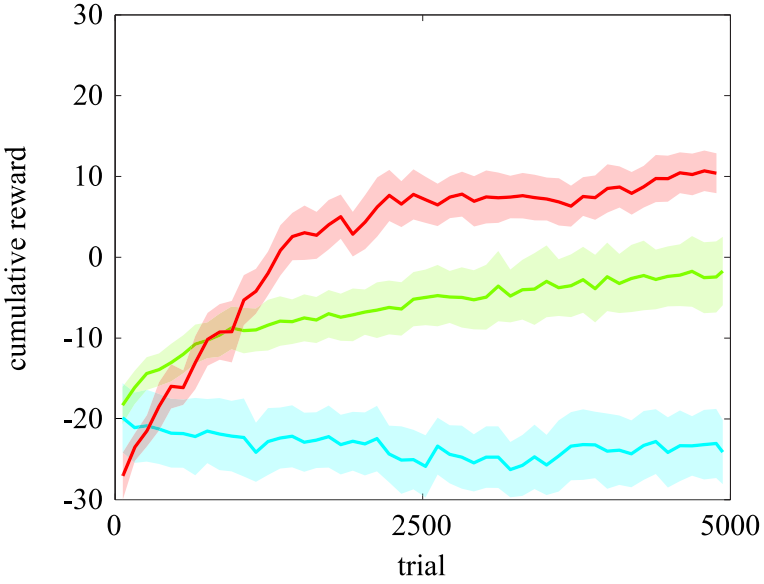



図 4.6　3 種類の実験の結果を表す．横軸が試行数，縦軸が 1 試行における累積報酬値を示している．A が「お互いに相手の状態を無視して行動選択」した場合，B が「推定された相手の内部シンボルに応じて行動選択」した場合，そして C が「送信された外部シンボルに応じて行動選択」した場合の結果を表す．

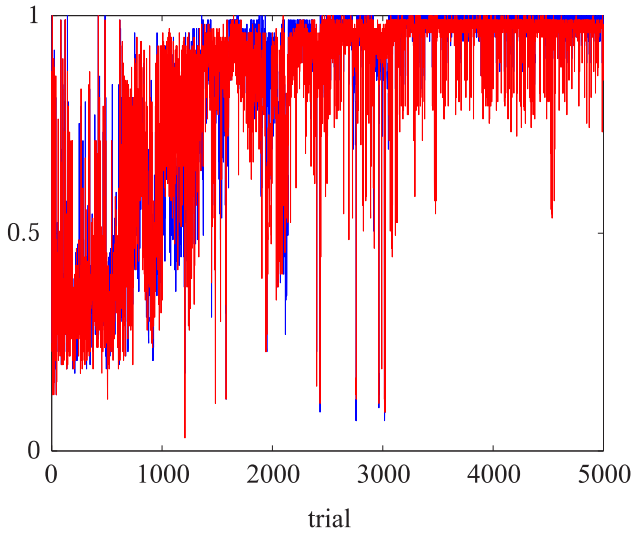



図 4.7　外部シンボルの意味付けの一致率を表示した．縦軸は各試行における一致率の平均値である．青色はエージェント B が発した外部シンボル $\mathbf{o}_B(t)$ とエージェント A が推定した外部シンボル $\hat{\mathbf{o}}_B(t) = M_A(\hat{\mathbf{s}}_B(t))$ の一致率，赤色はエージェント A が発した外部シンボル $\mathbf{o}_A(t)$ とエージェント B が推定した外部シンボル $\hat{\mathbf{o}}_A(t) = M_B(\hat{\mathbf{s}}_A(t))$ の一致率を表す．

## 4.3 第3者による協調作業の見まね

前節では外部シンボルの共有により学習が高速化されることを示したが，外部シンボルの有用性は3人以上のエージェントが存在する時にも発揮される．その例として，ある2人のエージェントが協調作業を学習した後で片方のエージェントが3人目のエージェントと交代した場合を考える．本稿で提案している手法では相手の内部シンボルを推定する事で協調作業が可能になるが，エージェントのモジュール構造には個体差があるため相手が代わると内部シンボルの推定結果も変わってしまい，その変化が大きい場合には学習をほとんどやり直さなければならない可能性もある．そこで複数のエージェント間で共通なシンボルを用いる事で，協調作業の相手が代わってもそれ以前に得た学習結果を用いる事が可能になる．この節ではその様な状況をシミュレーションにより検証する．

2人のエージェントA,Bが協調作業を行い，その後に3人目のエージェントCがエージェントBと交代するような状況を考える．実験の手順は以下の通りである．

- エージェントAとBが共通なシンボルを用いながら協調作業の学習を行う．エージェントAは$\theta - \dot{\theta}$平面に$5 \times 5$個，エージェントBは$\theta - \dot{\theta}$平面に$5 \times 7$個の状態予測モデルを配置している．各エージェントは写像関数$M_A$, $M_B$に従って外部シンボルの送信を行い，またその更新も行う．

- エージェントCはエージェントBを観察し，写像関数と行動選択確率の2つを見まねする．観察はエージェントA,Bの学習が十分に収束した後で行う．エージェントCの状態予測モデルは$\theta - \dot{\theta}$平面に$7 \times 7$個が配置されている．

- エージェントAとCが協調作業を行う．エージェントAは交代前の学習結果をそのまま用い，エージェントCは見まねで得た写像関数を用いて外部シンボルの発生，また行動選択確率を事前知識として式(4.6)に従い行動選択を行う．両者とも上位層の状態行動価値関数，写像関数の更新を行う．

最初に行われるエージェントA,Bの学習は前節における，“送信された外部シンボルに応じて行動選択” を行う実験とまったく同じである．そこでエージェント

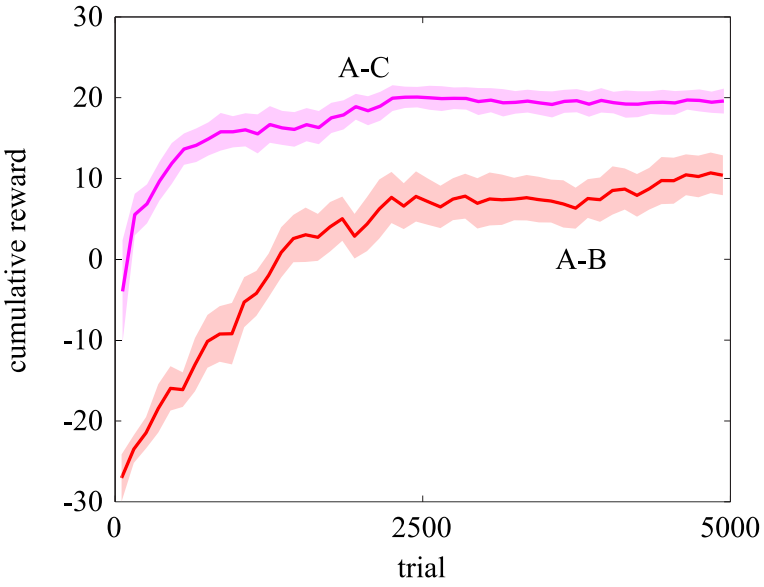



図 4.8 エージェント B と C が交代した後，エージェント A,C による学習結果を紫色で示す．比較のため，前節における学習結果を赤色で表示した．

C の観察対象は，前節にて 10 回行った実験のうち最も結果が良かったもののエージェント B とした．エージェント C のモジュール構造はエージェント A,B と異なっており，その様な設定でも共通なシンボルがうまく働くかどうかを検証する．

**結果**

図 4.8 に累積報酬の変化を示す．グラフの左端がそろえられているが，実際には赤色で示されている学習曲線の後にエージェント A,C が協調作業を開始している事に注意して欲しい．図より共通なシンボルを見まねにより獲得する事で，内部構造が異なるエージェント C が課題に参加した場合でも即座に高い性能が得られている事が分かる．エージェント A,B による協調作業はうまく働く場合とそうでない場合のばらつきが多いのに比べ，エージェント A,C による協調作業は 10 回中全てで高い累積報酬を得られており，平均すると図 4.8 の様な結果となった．エージェント C が観察の対象とした実験における学習曲線とエージェント A,C による結果を比較した所，最終的な性能に大差はないものの学習初期における学習

曲線の立ち上がりが後者のほうが勝っている結果となっていた.

# 5. 本章の議論

コミュニケーションや言語に関する研究は離散な環境を扱ったものが多い．しかし我々を取り巻く環境は常に連続なシステムである．高次元かつ連続な環境が持つ膨大な情報の中から必要に応じてラベルを割り当てる能力，つまり記号接地問題 [17] を解く能力こそがコミュニケーションや言語の扱う際に必要となるが，実験者が予め環境を離散化する様な設定では本当にその様な問題を解決したとは言いがたい．本章では連続な環境を制御対象とし，エージェント同士が相互作用を通して共通なシンボルを生成する枠組みを示す事に成功し，真に記号接地問題を解くための枠組みを示せた．

MOSAIC モデルに基づいた制御アルゴリズムによるコミュニケーションの研究は MMRL や CMRL を用いた見まね学習の例がある．MMRL を用いた例 [47] は他者の運動軌道からその生成に用いられているモジュール系列を推定し，自律制御の際はその系列を自身のモジュール選択の事前分布とする手法である．他者の連続的な軌道を観測する場合，制御出力は観測できないばかりか，一般的には物理パラメータが異なるため観測軌道を再現する制御出力は異なったものとなる．そこで観測軌道を最も良く説明する自己のモジュール系列を推定するアプローチが有用であるとしている．CMRL を用いた見まねの例 [48] は MMRL の様に状態予測モデルではなく報酬予測モデルの系列に重点を置いて行われている．状態予測モデルと報酬予測モデル，強化学習コントローラのそれぞれに観測された他者の運動軌道を入力する事で，観測軌道を最も良く説明する 3 者の組み合わせを推定する手法である．教示者が複数の目標，すなわち報酬関数を切り替えながら行動を行っている場合，MMRL では観測軌道を再現できないが CMRL では可能となっている．見まね者と教示者の物理パラメータが異なるため観測軌道をそのまま再現しようとすると破綻してしまうような設定であっても，見まね者は自身が過去に獲得している制御則の中から適切なものを選択する事で定性的に同じ運動軌道を再現できるとしている．MMRL や CMRL による見まねがモジュールの時系列の推定と再生であるのに対して，本研究で定式化を行った SSRL による見まねは状態空間の各局所領域において使用するべきモジュールの推定である．状態行動価値関数の見まねとも言い換える事ができ，より実用的な見まねの手法であ

ると言える．また銅谷ら [53] によって提案されているコミュニケーションの枠組みは観測された相手の運動軌道の意図やメッセージに応じて自身の行動則を決定すると言うものである．この枠組みは本章 4.2 節にて行った実験のうち，“推定された相手の内部シンボルに応じて行動選択” する事に相当する．しかしこの枠組みは相手が自分の事を正しく理解しているかが保障されていない．一方，共通なシンボルを生成する実験ではエージェントは自分の内部シンボルに応じた連続軌道と外部シンボルを生成でき，ある連続軌道に対して外部シンボルの割り当てがお互いに同じものであるかどうかを評価する機構が組み込まれている．その機構により両者が同じ状況に面した時，お互いに同じ外部シンボルを送信する事が実現される．

# 第5章　議論

## 1. 議論

見まねや協調作業といったコミュニケーションを行う際，他者の運動軌道の裏に隠れた高次の情報を推定する必要があるが，その問題は一般に不良設定性をもち容易には解決できない．本研究ではエージェントが過去の経験により獲得した制御則やモジュール構造がその不良設定性を解決すると言う視点から，運動制御と他者行動の認知の両方を同じ数学モデルを用いて行える制御アルゴリズムの開発を行った．

Inamura らも同じ観点から研究を行っており，連続な下位層と離散的な表現を持つ上位層と間で双方向な情報変換を行う事によりシンボルが創発される枠組みを提案している [22]．Inamura らの手法を用いた他者の運動軌道の認識はまず運動軌道を分節化し，さらに分節の切り替わりに対する隠れ状態を HMM を用いて推定している．その事により上位層は複数の行動時系列を認識できると言う点で興味深い．SSRL における他者の運動軌道の認識は状態行動価値関数の推定として行われるが，上位層はひとつの状態行動価値関数しかもっていないため，複数の行動を認識したり見まねする事は出来ない．今後は上位層のモジュール化したり，Inamura らの手法を取り入れる事で改善を行いたい．

マルチエージェントを用いた進化の研究では，優れたエージェントの学習結果を次世代へ受け渡す作業が行われる．しかし現実的な問題を考えるとエージェントの内部構造やヒトの脳のシナプス結合には個人差があるため，結合荷重などの学習パラメータをそのまま他者に移す事は不可能である．もし仮に可能だとしても，身体の物理パラメータにも個人差はあるため他者の学習結果をそのまま用いる事は出来ない．他者の学習結果を自分の内部構造や物理パラメータに応じて見

まねする作業が必要となる．本研究ではその問題を解決する見まねの枠組みを定式化しており，モジュール構造や物理パラメータが異なるエージェント間で見まねが行える事をシミュレーションにより検証した．

他者運動軌道から内部シンボルの推定と内部シンボルから外部シンボルへの写像の学習を繰り返す事でエージェント間に共通なシンボルが生成される枠組みを定式化した．今回の実験では各時刻にて有効になっている内部シンボルを写像関数への入力としたが，例えば将来に渡る計画や他者への要望を入力とする事も考えられる．今後は生成された外部シンボルがコミュニケーションにどのていど貢献しているかの評価基準を定め，シンボル生成の研究を深めていきたい．

# 謝辞

本研究の全般においてご指導してくださった銅谷賢治客員助教授，および川人光男客員教授に深く感謝いたします．銅谷賢治客員助教授には主に理論的な面の全般において貴重な助言をして頂きました．川人光男客員教授には特に，提案手法の脳のモデルとしての発展に関してご指導をして頂きました．お二方に改めて感謝いたします．

研究発表などの場面において貴重なご意見をしてくださった石井信教授，柴田智広助教授，及び日々の議論を通して研究を支えてくれた論理生命学分野の先輩方に深く感謝いたします．

最後に論文審査を快く引き受けてくださった小笠原司教授に感謝いたします．

# 参考文献

## 参考文献


[1] Arita, T. and Koyama, Y.: Evolution of Linguistic Diversity in a Simple Communication System, *Artificial Life*, Vol. 4, No. 1, pp. 109–124 (1998).

[2] Barto, A. G., Bradtke, S. J. and Singh, S. P.: Learning to act using real-time dynamic programming, *Artificial Intelligence*, Vol. 72, No. 1-2, pp. 81–138 (1995).

[3] Barto, A. G., Sutton, R. S. and Anderson, C. W.: Neuronlike adaptive elements that can solve difficult learning control problems, *IEEE Transactions on Systems Man and Cybernetics*, Vol. 13, pp. 834–846 (1983).

[4] Bertsekas, D. P.: *Dynamic programming and optimal control*, Athena Scientif (1995).

[5] Bradtke, S.: *Incremental Dynamic Programming for On-line Adaptive Optimal Control*, PhD Thesis (1994).

[6] Bradtke, S. J. and Duff, M. O.: Reinforcement Learning Methods for Continuous-Time Markov Decision Problems, *Advances in Neural Information Processing Systems* (Tesauro, G., Touretzky, D. and Leen, T.(eds.)), Vol. 7, The MIT Press, pp. 393–400 (1995).

[7] Cangelosi, A.: Evolution of communication and language using signals, symbols and words, *IEEE Transactions on Evolutionary Computation*, Vol. 5, No. 2, pp. 93–101 (2001).

[8] Dayan, P. and Hinton, G. E.: Feudal Reinforcement Learning, *Advances in Neural Information Processing Systems* (Giles, C. L., Hanson, S. J. and Cowan, J. D.(eds.)), San Mateo, CA, Morgan Kaufmann (1993).

[9] Deacon, T. W.: *The Symbolic Species: The Co-Evolution of Language and the Brain*, W. W. Norton & Company (1997).

[10] Dempster, A., Laird, N. and Rubin, D.: Maximum Likelihood from Incomplete Data via The EM Algorithm, *Journal of Royal Statistical Society*, Vol. 39, pp. 1–38 (1977).

[11] Dietterich, T. G.: Hierarchical Reinforcement Learning with the MAXQ Value Function Decomposition, *Journal of Artificial Intelligence Research*, Vol. 13, pp. 227–303 (2000).

[12] Doucet, A., de Freitas, N. and Gordon, N.: *Sequential Monte Carlo Methods in Practice*, Springer-Verlag (2001).

[13] Doya, K.: Reinforcement Leaning in continuous time and space, *Neural Computation*, Vol. 12, pp. 219–245 (2000).

[14] Doya, K. and Morimoto, J.: Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning, *Robotics and Autonomous Systems*, Vol. 36, pp. 37–51 (2001).

[15] Doya, K., Samejima, K., Katagiri, K. and Kawato, M.: Multiple Model-Based Reinforcement Learning, *Neural Computation*, Vol. 14, pp. 1347–1369 (2002).

[16] Ghahramani, Z. and Hinton, G. E.: Variational Learning for Switching State-Space Models, *Neural Computation*, Vol. 12, No. 4, pp. 831–864 (2000).

[17] Harnad, S.: The Symbol Grounding Problem, *Physica D*, Vol. 42, pp. 335–346 (1990).

[18] Haruno, M., Wolpert, D. M. and Kawato, M.: *Multiple paired forward-inverse models for human motor learning and control*, Advances in Neural Information Processing Systems, MIT Press, Cambridge, Massachusetts 11: 31-37 (1999).

[19] Haruno, M., Wolpert, D. M. and Kawato, M.: MOSAIC model for sensorimotor learning and control, *Neural Computation*, Vol. 13, pp. 2201–2220 (2001).

[20] Haruno, M., Wolpert, D. M. and Kawato, M.: *Hierarchical MOSAIC for movement generation*, International Congress Series, Vol. 1250, T. Ono and G. Matsumoto and R.R. Llinas and A. Berthoz and R. Norgren and H. Nishijo and R. Tamura (Eds.), Amsterdam (2003).

[21] Hauser, M. D.: *The Evolution of Communication*, MIT Press (1996).

[22] Inamura, T., Toshima, I., Tanie, H. and Nakamura, Y.: Embodied Symbol Emergence based on Mimesis Theory, *International Journal of Robotics Research*, Vol. 23, No. 4, pp. 363–377 (2004).

[23] Jacobs, R., Jordan, M., Nowlan, S. and Hinton, G.: Adaptive Mixtures of Local Experts, *Neural Computation*, Vol. 3, pp. 79–87 (1991).

[24] Kaelbling, L. P., Littman, M. L. and Moore, A. P.: Reinforcement Learning: A Survey, *Journal of Artificial Intelligence Research*, Vol. 4, pp. 237–285 (1996).

[25] Kalman, R. E.: A New Approach to Linear Filtering and Prediction Problems, *Transactions of the ASME–Journal of Basic Engineering*, Vol. 82, No. Series D, pp. 35–45 (1960).

[26] Mahadevan, S.: Partially-Observable Semi-Markov Decision Processes: Theory and Applications in Engineering and Cognitive Science (1998).

[27] Mahadevan, S., Marchalleck, N., Das, T. K. and Gosavi, A.: Self-improving factory simulation using continuous-time average-reward reinforcement learning, *Proceedings of the 14th International Conference on Machine Learning*, Morgan Kaufmann, pp. 202–210 (1997).

[28] Moody, J. and Darken, C. J.: Fast learning in networks of locally-tuned processing units, *Neural Computation*, Vol. 1, No. 2, pp. 281–294 (1989).

[29] Morimoto, J. and Doya, K.: Acquisition of stand-up behavior by a real robot using hierarchical reinforcement learning, *Robotics and Autonomous Systems*, Vol. 36, No. 1, pp. 37–51 (2001).

[30] Morimoto, J. and Doya, K.: *Robust Reinforcement Learning*, Advances in Neural Information Processing Systems, MIT Press, 13: 1061-1067 (2001).

[31] Murphy, K. P.: Switching Kalman filters (1998).

[32] Parr, R. E.: *Hierarchical Control and Learning for Markov Decision Processes*, PhD Thesis (1990).

[33] Peng, J. and Williams, R. J.: Incremental multi-step Q-learning, *Machine Learning*, Vol. 22, pp. 283–290 (1996).

[34] Rizzolatti, G., Fadiga, L., Gallese, V. and Fogassi, L.: Premotor Cortex and the Recognition of Motor Actions, *Cognitive Brain Research*, Vol. 3, pp. 131–141 (1996).

[35] Singh, S. P.: Scaling Reinforcement Learning Algorithms by Learning Variable Temporal Resolution Models, *Proceedings of the Ninth International Conference on Machine Learning* (1992).

[36] Singh, S. P.: Transfer of Learning by Composing Solutions of Elemental Sequential Tasks, *Machine Learning*, Vol. 8, pp. 323–339 (1992).

[37] Steels, L. and Baillie, J.-C.: Shared Grounding of Event Descriptions by Autonomous Robots, *Robotics and Autonomous Systems*, Vol. 43, No. 2–3, pp. 163–173 (2003).

[38] Sutton, R. S. and Barto, A. G.: *Reinforcement Learning*, MIT Press (1998).

[39] Sutton, R. S., Precup, D. and Singh, S. P.: Intra-Option Learning about Temporally Abstract Actions, *Proceedings of the Fifteenth International Conference on Machine Learning*, Morgan Kaufmann, pp. 556–564 (1998).

[40] Sutton, R. S., Precup, D. and Singh, S. P.: Between MDPs and Semi-MDPs: A Framework for Temporal Abstraction in Reinforcement Learning, *Artificial Intelligence*, Vol. 112, No. 1-2, pp. 181–211 (1999).

[41] Watkins, C. J. C. H. and Dayan, P.: Technical Note Q-Learning, *Machine Learning*, Vol. 8, pp. 279–292 (1992).

[42] Wolpert, D. M., Doya, K. and Kawato, M.: A unifying computational framework for motor control and social interaction, *The Royal Society*, Vol. 358, No. 1431, pp. 593–602 (2003).

[43] Wolpert, D. M. and Kawato, M.: Multiple paired forward and inverse models for motor control, *Neural Networks*, Vol. 11, pp. 1317–1329 (1998).

[44] Wolpert, D. M., Miall, C. and Kawato, M.: Internal Models in the Cerebellum, *Trends in Cognitive Sciences* (1998).

[45] 中野馨: 脳をつくる-ロボット作りから生命を考える, 共立出版 (1995).

[46] 鮫島和行, 片桐憲一, 銅谷賢治, 川人光男: 複数の予測モデルを用いた強化学習による非線形制御, 電子情報通信学会誌 (D-II), Vol. J84-D-II, No. 9, pp. 2092–2106 (2001).

[47] 鮫島和行, 片桐憲一, 銅谷賢治, 川人光男: モジュール結合による運動パターンのシンボル化と見まね学習, 電子情報通信学会誌 (D-II), Vol. J85-D-II, No. 1 (2002).

[48] 杉本徳和, 鮫島和行, 銅谷賢治, 川人光男: 複数の状態予測と報酬予測モデルによる強化学習と行動目標の推定, 電子情報通信学会誌 (D-II), Vol. J87-D-II, No. 2, pp. 683–694 (2004).

[49] 川人光男, 銅谷賢治, 春野雅彦: 多重順逆対モデル（モザイク）ーその情報処理と可能性, 科学, Vol. 70, No. 11, pp. 1009–1017 (2000).

[50] 川嶋宏彰, 堤公孝, 松山隆司: 動的イベントの分節化・学習・認識のための Hybrid Dynamical System, 第 3 回情報科学技術フォーラム, pp. 175–178 (2004).

[51] 藤田政之, 大嶋正裕: モデル予測制御 VI ーハイブリッドモデル予測制御, システム/制御/情報, Vol. 47, No. 3, pp. 146–152 (2003).

[52] 銅谷賢治, 森本淳, 鮫島和行: 強化学習と最適制御, システム制御情報, Vol. Vol.45, No. No.4, pp. 186–196 (2001).

[53] 銅谷賢治, 川人光男, 春野雅彦: 小脳，大脳基底核，大脳皮質の機能分化と統合, 科学, Vol. 70, No. 9, pp. 740–749 (2000).

[54] 野田五十樹: HMM の学習による環境の分節, 人工知能学会全国大会予稿集, pp. 2B3–08 (2002).

# 付録

## A. 二足歩行ロボットの状態予測モデルと報酬予測モデル

学習に用いたデータは $\mathbf{x} = [-\frac{45}{180}\pi \sim \frac{45}{180}\pi \ -\frac{45}{180}\pi \sim \frac{45}{180}\pi \ -\pi \sim \pi \ -\pi \sim \pi]^T$ の範囲で一様にランダムな初期状態 $\mathbf{x}(0)$ から開始し，正弦波の重ね合わせを制御入力として与える事で生成した $(T(t) = \sin(t/0.1 - \Delta) + \sin(t/0.15 - \Delta) + \sin(t/0.2 - \Delta))$．0.5[sec] の試行を計 1000 回い，正弦波の位相 $\Delta$ は試行毎にランダムな定数として与えた．観測は 0.01[sec] 毎に行い，足の角度 $\theta_L$，$\theta_R$ のどちらかが $-60\pi/180 \simeq 60\pi/180$ から外れたデータは除外した．そうして生成したデータを用いて以下の様に状態予測モデル，報酬予測モデルのパラメータ $\phi_i^f$，$\phi_j^r$ を学習した．

### A.1 状態予測モデル

状態予測モデルの中心 $\mathbf{x}_i^f$ の初期値は用意したデータの範囲で一様な乱数，出力の分散 $\sigma_i^{f\,2}$ の初期値は 10 とした．また係数 $A_i$，$B_i$，$c_i$ の各要素の初期値は $-0.1 \sim 0.1$ の間で一様な乱数で与えた．以下で説明する責任信号の推定とパラメータの学習をパラメータが十分に収束するまで繰り返した．

**責任信号の推定**　責任信号の計算に事前分布は用いなかった．$\hat{\lambda}_i^f(t) = 1/N^f$ として，式 (2.14) に従って全てのデータ毎に責任信号を計算した．

**パラメータの学習**　期待対数尤度を最大とするパラメータは

$$\mathbf{x}_i^f = \langle \mathbf{x} \rangle_i / \langle 1 \rangle_i \tag{5.1}$$

$$W_i^f = \langle \dot{\mathbf{x}} [\mathbf{x}^T \mathbf{u}^T 1] \rangle_i \langle [\mathbf{x}^T \mathbf{u}^T 1]^T [\mathbf{x}^T \mathbf{u}^T 1] \rangle_i^{-1} \tag{5.2}$$

$$\sigma_i^{f\,2} = \frac{1}{D_s} \langle \| \hat{\dot{\mathbf{x}}}_i(t) - \dot{\mathbf{x}}(t) \|^2 \rangle_i / \langle 1 \rangle_i \tag{5.3}$$

として与えられる [10]．ここで $W_i^f$ はパラメータ $A_i, B_i, c_i$ を横に並べたもの ($W_i^f = [A_i\ B_i\ c_i]$)，$\langle \bullet \rangle_i$ は責任信号による重み付き平均を表す ($\langle g(t) \rangle_i = \frac{1}{T} \int_0^T g(t) \lambda_i^f(t) dt$)．

## A.2 報酬予測モデル

出力のコスト $R_j$ は 0.05 で固定とした．係数 $Q_j$，$q_j$ の初期値は $-0.1 \sim 0.1$ の範囲で，また報酬予測モデルの頂点 $\mathbf{x}_j^r$ は用意したデータの範囲で一様な乱数として与えた．出力の分散 $\sigma_j^{r\,2}$ は $0.01^2$ で固定とした．以下で説明する責任信号の推定とパラメータの学習をパラメータが十分に収束するまで繰り返した．

**責任信号の推定**　状態予測モデルの場合と同様，責任信号の計算に事前分布は用いなかった．$\hat{\lambda}_j^r(t) = 1/N^r$ として，式 (2.15) に従って全てのデータ毎に責任信号を計算した．

**パラメータの学習**　まず 2 次形式を

$$\begin{aligned} r_j(\mathbf{x}(t), \mathbf{u}(t)) &= -(\mathbf{x}(t) - \mathbf{x}_j^r)^T Q_j (\mathbf{x}(t) - \mathbf{x}_j^r) - \mathbf{u}(t)^T R_j \mathbf{u}(t) + q_j \\ &= W_j^r [\tilde{\mathbf{x}}(t)^T \mathbf{x}(t)^T 1]^T - \mathbf{u}(t)^T R_j \mathbf{u}(t) \end{aligned} \tag{5.4}$$

$$W_j^r = [-\mathrm{diag} Q_j \ \ s_j \ \ h_j] \tag{5.5}$$

$$\tilde{\mathbf{x}}(t) = [\theta_L(t)^2 \ \ \theta_R(t)^2 \ \ \dot{\theta}_L(t)^2 \ \ \dot{\theta}_R(t)^2] \tag{5.6}$$

と変形し，係数行列 $Q_j$ は対角成分だけを学習の対象とした．この時，各パラメータを以下の様に更新する事で期待対数尤度を最大化できる．

$$W_j^r = \langle r(t)[\tilde{\mathbf{x}}(t)^T \mathbf{x}(t)^T 1] \rangle_j \langle [\tilde{\mathbf{x}}(t)^T \mathbf{x}(t)^T 1]^T [\tilde{\mathbf{x}}(t)^T \mathbf{x}(t)^T 1] \rangle_j^{-1} \tag{5.7}$$

$$\mathbf{x}_j^r = \frac{1}{2} Q_j^{-1} s_j^T \tag{5.8}$$

$$q_j = h_j + \mathbf{x}_j^{rT} Q_j \mathbf{x}_j^r \tag{5.9}$$

# B. ベクトルを含む微分について

本稿ではベクトルに関する微分がしばしば行われる．$a$ をスカラー値，$\mathbf{b} = [b_1\ b_2\ ....\ b_N]^T$，$\mathbf{c} = [c_1\ c_2\ ....\ c_D]^T$ をそれぞれ $N$，$D$ 次元の縦ベクトルとした時，ベクトルの微分またはベクトルによる微分を以下の様に定義しておく．

$$\frac{\mathrm{d}a}{\mathrm{d}\mathbf{b}} = \begin{bmatrix} \frac{\mathrm{d}a}{\mathrm{d}b_1} & \frac{\mathrm{d}a}{\mathrm{d}b_2} & \cdots & \frac{\mathrm{d}a}{\mathrm{d}b_N} \end{bmatrix}$$

$$\frac{\mathrm{d}\mathbf{b}}{\mathrm{d}a} = \begin{bmatrix} \frac{\mathrm{d}b_1}{\mathrm{d}a} \\ \frac{\mathrm{d}b_2}{\mathrm{d}a} \\ \vdots \\ \frac{\mathrm{d}b_N}{\mathrm{d}a} \end{bmatrix}$$

$$\frac{\mathrm{d}\mathbf{c}}{\mathrm{d}\mathbf{b}} = \begin{bmatrix} \frac{\mathrm{d}c_1}{\mathrm{d}b_1} & \frac{\mathrm{d}c_1}{\mathrm{d}b_2} & \cdots & \frac{\mathrm{d}c_1}{\mathrm{d}b_N} \\ \frac{\mathrm{d}c_2}{\mathrm{d}b_1} & \frac{\mathrm{d}c_2}{\mathrm{d}b_2} & \cdots & \frac{\mathrm{d}c_2}{\mathrm{d}b_N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\mathrm{d}c_D}{\mathrm{d}b_1} & \frac{\mathrm{d}c_D}{\mathrm{d}b_2} & \cdots & \frac{\mathrm{d}c_D}{\mathrm{d}b_N} \end{bmatrix}$$

ベクトルによる微分の結果が転置されてる事に注意されたい．

# C. Adaptive Real-Time Dynamic Programming

上位層の学習は離散状態と行動の対 $(i, j)$ のダイナミクスに依存するが，そのダイナミクスは下位層の学習が進む事によって変化してしまう．環境のモデルを

用いない TD 法は一般に収束が遅く，下位層の変化に上位層の学習が追いつかなくなる可能性がある．

そこでモデルの学習を行う事で，下位層の変化への追従を速くする事が考えられる．Adaptive Real-Time Dynamic Programming(Adaptive RTDP) はシステムのモデルををオンラインで学習する事で価値関数の収束を速くする手法である [5, 2]．

上位層の状態遷移確率 $P(I'|I,j)$，累積報酬の分布 $P(\rho|I,I',j)$ および遷移時間の分布 $P(l|I,I',j)$ の推定値が得られている時，ある状態 $I$ に対する各行動 $j$ の価値は以下の様に更新できる．

$$Q(I,j) = \sum_{I'=1}^{N^f} P(I'|I,j)\left[\int_{-\infty}^{\infty} \rho P(\rho|I,I',j)d\rho + \int_{0}^{\infty} e^{-\frac{l}{\tau}} P(l|I,I',j)dl \max_{j'} Q(I',j')\right] \tag{5.10}$$

ここで $\rho = \int_t^{t+l} e^{-\frac{s-t}{\tau}} R(s)ds$ であり，$j$ は状態 $I$ において選択していた行動，$I'$ は次状態を表す．上位層の状態遷移が起こる毎に上位層の状態遷移確率 $P(I'|I,j)$，累積報酬の分布 $P(\rho|I,I',j)$ および遷移時間の分布 $P(l|I,I',j)$ の推定値を更新し，また全ての行動 $j = 1, ..., N^r$ に対して上記の更新を行う．

# D. Linear Quadratic Controller

時刻 $t$ における環境の状態変化 $\dot{\mathbf{x}}(t)$ と報酬 $r(t)$ が状態 $\mathbf{x}(t)$ と制御出力 $\mathbf{u}(t)$ の関数として

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)) \tag{5.11}$$

$$r(t) = r(\mathbf{x}(t), \mathbf{u}(t)) \tag{5.12}$$

の様に決定される場合，最適な状態価値関数 $V^*(\mathbf{x}(t))$ と制御出力則 $\mu^*(\mathbf{x}(t))$ は以下を満たす.

$$\frac{1}{\tau}V^*(\mathbf{x}(t)) = \max_{\mathbf{u}(t)\in U}\left[r(\mathbf{x}(t),\mathbf{u}(t)) + \frac{\partial V^*(\mathbf{x})}{\partial \mathbf{x}} f(\mathbf{x}(t),\mathbf{u}(t))\right] \tag{5.13}$$

$$\mu^*(\mathbf{x}(t)) = \operatorname*{argmax}_{\mathbf{u}\in U}\left[r(\mathbf{x}(t),\mathbf{u}(t)) + \frac{\partial V^*(\mathbf{x})}{\partial \mathbf{x}} f(\mathbf{x}(t),\mathbf{u}(t))\right] \tag{5.14}$$

ここで $\tau$ は価値関数の時定数である．この時，環境のダイナミクスと報酬関数がそれぞれ

$$f(\mathbf{x},\mathbf{u}) = A(\mathbf{x}-\mathbf{x}^f) + B\mathbf{u} + c \tag{5.15}$$

$$r(\mathbf{x},\mathbf{u}) = -(\mathbf{x}-\mathbf{x}^r)^T Q(\mathbf{x}-\mathbf{x}^r) - \mathbf{u}^T R\mathbf{u} + q \tag{5.16}$$

のように線形モデルと 2 次モデルで記述できる場合，最適状態価値関数 $V^*(\mathbf{x}(t))$ と最適制御出力則 $\mu^*(\mathbf{x}(t))$ は以下の様な 2 次モデルと線形フィードバック則となる事が知られている.

$$V^*(\mathbf{x}(t)) = -(\mathbf{x}(t)-\mathbf{x}^c)^T P(\mathbf{x}(t)-\mathbf{x}^c) + p \tag{5.17}$$

$$\mu^*(\mathbf{x}(t)) = -2R^{-1}B^T P(\mathbf{x}(t)-\mathbf{x}^c) \tag{5.18}$$

ここで係数行列 $P$ は価値関数の広がり，$\mathbf{x}^c$ は頂点，$p$ はバイアス項をそれぞれ表している.

式 (5.15)(5.16)(5.17)(5.18) を式 (5.13) に代入すると

$$\begin{aligned}
&\frac{1}{\tau}(-(\mathbf{x}-\mathbf{x}^c)^T P(\mathbf{x}-\mathbf{x}^c) + p) \\
&= -(\mathbf{x}-\mathbf{x}^r)^T Q(\mathbf{x}-\mathbf{x}^r) - (\mathbf{x}-\mathbf{x}^c)^T PBR^{-1}B^T P(\mathbf{x}-\mathbf{x}^c) + q \\
&\qquad - 2(\mathbf{x}-\mathbf{x}^c)^T P(A(\mathbf{x}-\mathbf{x}^f) - BR^{-1}B^T P(\mathbf{x}-\mathbf{x}^c) + c) \qquad (5.19) \\
&= -(\mathbf{x}-\mathbf{x}^r)^T Q(\mathbf{x}-\mathbf{x}^r) + (\mathbf{x}-\mathbf{x}^c)^T PBR^{-1}B^T P(\mathbf{x}-\mathbf{x}^c) \\
&\qquad - (\mathbf{x}-\mathbf{x}^c)^T PA(\mathbf{x}-\mathbf{x}^f) - (\mathbf{x}-\mathbf{x}^f)^T A^T P(\mathbf{x}-\mathbf{x}^c) - 2(\mathbf{x}-\mathbf{x}^c)^T Pc + q \qquad (5.20)
\end{aligned}$$

$\mathbf{x}$ の 2 次の項，1 次の項および係数項に関してそれぞれ書き出すと

$$-\frac{1}{\tau}\mathbf{x}^T P\mathbf{x} = -\mathbf{x}^T Q\mathbf{x} + \mathbf{x}^T PBR^{-1}B^T P\mathbf{x} - \mathbf{x}^T(PA + A^T P)\mathbf{x} \tag{5.21}$$

$$-\frac{1}{\tau}\mathbf{x}^T P\mathbf{x}^c = \mathbf{x}^T Q\mathbf{x}^r - \mathbf{x}^T PBR^{-1}B^T P\mathbf{x}^c + \mathbf{x}^T PA\mathbf{x}^f + \mathbf{x}^T A^T P\mathbf{x}^c - \mathbf{x}^T Pc \tag{5.22}$$

$$-\frac{1}{\tau}(\mathbf{x}^{cT} P\mathbf{x}^c + p) = -\mathbf{x}^{rT} Q\mathbf{x}^r + \mathbf{x}^{cT} PBR^{-1}B^T P\mathbf{x}^c - 2\mathbf{x}^{cT} PA\mathbf{x}^f + 2\mathbf{x}^{cT} Pc + q \tag{5.23}$$

式 (5.21) より係数行列 $P$ は以下の様な Riccati 方程式を満たす事が分かる．

$$PA + A^T P - PBR^{-1}B^T P + Q = \frac{1}{\tau}P \tag{5.24}$$

また 2 次形式の中心 $\mathbf{x}^c$ とバイアス $p$ はそれぞれ

$$\mathbf{x}^c = (PA + Q)^{-1}(Q\mathbf{x}^r + PA\mathbf{x}^f - Pc) \tag{5.25}$$

$$\frac{1}{\tau}p = 2\mathbf{x}^{cT} P(A(\mathbf{x}^c - \mathbf{x}^f) + c) + (\mathbf{x}^c + \mathbf{x}^r)^T Q(\mathbf{x}^c - \mathbf{x}^r) + q \tag{5.26}$$

となる．

# E. EM アルゴリズムによる状態予測モデルの学習

学習に用いたデータは $[0 \ \ -\pi \sim \pi \ \ -10 \sim 10 \ \ -2\pi \sim 2\pi]^T$ の範囲で一様にランダムな初期状態 $\mathbf{x}(0)$ からガウスノイズを制御入力（$\mathbf{u}(t) = [10\mathcal{N}(0,1)]$）として与える事で生成した．0.1[sec] の試行を計 1000 回行った．

学習の対象としたパラメータとその初期値は以下の様にして設定した．式 (3.24) の様な線形モデルを状態予測モデルとした場合 $A_i$ の上 2 行が $[O_{D_s/2} \ \ I_{D_s/2}]$ [1]，$B_i$ の上 2 行と $c_i$ の上半分の要素が 0 となる事は明らかであるため，その他の要素のみ学習の対象とし初期値は $-0.1 \sim 0.1$ の間で一様な乱数で与えた．入力の分散の初期値は $0.1I_{D_s}$ で与え台車の横方向 $z$ に関係する要素のみ固定，出力の分散 ${\sigma^f}^2$

[1] $O_\bullet$ は $\bullet$ 次元の正方なゼロ行列

は $0.1^2$ で固定した．状態予測モデルの中心 $\mathbf{x}_i^f$ は用意したデータの範囲で一様な乱数とした．

以下の処理を各パラメータが十分に収束するまで繰り返した．

- E ステップ：各データに対する $i$ 番目の状態予測モデルの責任信号 $\lambda_i^f(t)$ を求める．
- M ステップ：各パラメータを更新する．

$$\mathbf{x}_i^f = \langle \mathbf{x} \rangle_i / \langle 1 \rangle_i \tag{5.27}$$

$$\Sigma_i^f = \langle (\mathbf{x} - \mathbf{x}_i^f)(\mathbf{x} - \mathbf{x}_i^f)^T \rangle / \langle 1 \rangle_i \tag{5.28}$$

$$W_i = \langle \dot{\mathbf{x}} [\mathbf{x}^T \mathbf{u}^T 1] \rangle \langle [\mathbf{x}^T \mathbf{u}^T 1]^T [\mathbf{x}^T \mathbf{u}^T 1] \rangle^{-1} \tag{5.29}$$

E ステップにおいて $W_i$ はパラメータ $A_i, B_i, c_i$ を横に並べたもの（$W_i = [A_i \; B_i \; c_i]$），$\langle \bullet \rangle_i$ は責任信号による重み付き平均を表す（$\langle g(t) \rangle_i = \frac{1}{T} \int_{t=0}^{T} g(t) \lambda_i^f(t) dt$．$T$ は観測を行った時間）．

# 研究業績

## 論文

1. 杉本徳和, 銅谷賢治, 川人光男: MOSAIC モデルにより環境を抽象化する階層型強化学習, 電子情報通信学会論文誌 (D-II), (印刷中).

2. 鮫島和行, 杉本徳和: モジュール強化学習と意図, 人工知能学会誌, Vol.20, No.4, pp.441-448(2005).

3. 杉本徳和, 鮫島和行, 銅谷賢治, 川人光男: 複数の状態予測と報酬予測モデルによる強化学習と行動目標の推定, 電子情報通信学会論文誌 (D-II), Vol.J87-D-II, No.2, pp.683-694(2004).

## 国際会議

1. Sugimoto, N, Doya, K., Kawato, M.: Cooperation by estimating other's internal state, Ninth Neural Computation and Psychology Workshop, modelling language, cognition and action (NCPW9), Plymouth (Sep., 2004)

## 全国大会・研究会

1. 杉本徳和, 銅谷賢治, 川人光男: マルチエージェント環境における共通なシンボルの生成, 電子情報通信学会技術研究報告, Vol.2005, No.44, p45-50 (Oct., 2005).

2. 杉本徳和, 鮫島和行, 銅谷賢治, 川人光男: MOSAIC モデルにより環境を抽

象化する階層型強化学習, 脳と心のメカニズム 第 6 回夏のワークショップ, 松代 (Aug., 2005).

3. 杉本徳和, 鮫島和行, 銅谷賢治, 川人光男: 教示者の行動目標を推定する見まね学習, 脳と心のメカニズム 第 4 回冬のワークショップ, 留寿都 (Jan., 2004).

4. 杉本徳和, 銅谷賢治, 川人光男: 教示者の行動目標を推定する見まね学習, 電子情報通信学会技術研究報告, Vol.2003, No.69, pp.61–66 (Oct., 2003).

5. 杉本徳和, 鮫島和行, 銅谷賢治, 川人光男: ダイナミクスの線形性に基づいて状態空間を分割する階層型強化学習, 電子情報通信学会技術研究報告, Vol.2003, No.16, pp.25–30 (Jun., 2003).

6. 杉本徳和, 鮫島和行, 銅谷賢治, 川人光男: 複数の状態予測と報酬予測モデルによる強化学習と行動目標の推定, 電子情報通信学会技術研究報告, Vol.2001, No.118, pp87-94 (Jan., 2001).

7. 杉本徳和, 鮫島和行, 銅谷賢治, 川人光男: 複数の状態予測と報酬予測モデルによる強化学習と行動目標の推定, 日本神経回路学会第 12 回全国大会, 鳥取 (Sep., 2002).

## その他の研究業績

1. Doya, K., Sugimoto, N., Wolpert, D.M., Kawato, M.: Selecting optimal behaviors based on contexts, International Symposium on Emergent Mechanisms of Communication, Awaji, pp.19–23 (Mar., 2003).

2. Takadama K., Suematsu, Y. L., Sugimoto, N., Nawa, N.E., Shimohara, K.: Cross-Element Validation in Multiagent-based Simulation: Switching Learning Mechanisms in Agents, The Journal of Artificial Societies and Social Simulation (JASSS), Vol.6, No.4 (2003).

3. Takadama, K., Suematsu, Y.L., Sugimoto, N., Nawa, N.E., Shimohara, K.: Towards Verification and Validation in Multiagent-Based Systems and Simulations: Analyzing Different Learning Bargaining Agents, The 4th Workshop on Multi-Agent Based Simulation (MABS'03), pp.18-32 (2003)

4. Takadama, K., Sugimoto, N., Nawa, N.E., Shimohara, K.: Grounding to Both Theory and Real World by Agent-Based Simulation: Analyzing Learning Agents in Bargaining Game, North American Association for Computational Social and Organizational Science (NAACSOS) Conference (2003).

5. Takadama, K., Suematsu, Y.L., Sugimoto, N., Nawa, N.E., Shimohara, K.: Do Computational Models with Different Learning Mechanisms Produce the Same Results?, The Model to Model (M2M) Workshop, pp.5–15 (2003).

6. 杉本 徳和, 高玉 圭樹, N.E. NAWA, 下原 勝憲, 社会科学におけるバーゲニングの感度分析とその展開：Q 学習からの接近, ATR テクニカルレポート, TR-HIS-0011 (2003).

7. Takadama, K., Suematsu, Y.L., Sugimoto, N., Nawa, N.E., Shimohara, K.: X-MAS: Cross-validation in MultiAgent-based Simulation, The Computational Analysis of Social and Organizational System (CASOS) Conference (2002).