NAIST-IS-DD1161005

博士論文

線形化マルコフゲーム理論によるロバスト制御

金城　健

2014 年 3 月 13 日

奈良先端科学技術大学院大学
情報科学研究科 情報システム学専攻

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。

金城　健

審査委員：

池田　和司　教授	（主指導教員）
小笠原　司　教授	（副指導教員）
銅谷 賢治　客員教授	（副指導教員）
吉本　潤一郎　客員准教授	（副指導教員）

線形化マルコフゲーム理論によるロバスト制御*

金城　健

内容梗概

ロボットなどの自律システムが周囲の環境に合わせて適切に振る舞うための制御則を求めるための手法として，最適制御理やロバスト制御がある．最適制御では Hamilton-Jacobi-Bellman (HJB) 方程式，ロバスト制御理論では Hamilton-Jacobi-Isaacs (HJI) 方程式を解くことで制御則が得られるが，これらは非線形偏微分方程式であり，解析解を求めることが一般に困難である．これに対し，線形可解 Markov 決定過程 (Linearly solvable Markov Decision Process; LMDP) と呼ばれる HJB 方程式を線形微分方程式に変換できる問題のクラスが提案され，HJI 方程式を同様に線形化する線形可解 Markov ゲーム (Linearly solvable Markov Game; LMG) に拡張された．この線形化によりあるクラスの非線形制御問題に対しても解を効率良く求めることができるようになった．しかし従来の研究では環境のダイナミクスは既知であることを仮定しているが，実際にはこれらは未知の場合が多く，ロボットは環境との相互作用を通して学習する必要がある．また LMDP や LMG は低次元の問題でのみ検証され，実ロボットへ適用した例はほとんどない．

そこで本研究では LMDP や LMG を実システムに適用する際の問題点を明確にするために，モデルの近似誤差が制御則の制御性能に与える影響を調査する．まず LMDP と環境モデルを学習するモジュールを組み合わせたシステムを提案し，環境モデルの近似誤差が LMDP の学習に及ぼす影響を調査する．車輪型の移動ロボットを用いた視覚誘導課題を通して，提案したシステムは実システムに対して有効であることを示す．次に LMG の外乱の強さを調整するパラメータと近似誤

*奈良先端科学技術大学院大学 情報科学研究科 情報システム学専攻 博士論文, NAIST-IS-DD1161005, 2014 年 3 月 13 日.

差の関係を明らかにするために様々な学習環境とテスト環境のもとで，振り子の振り上げや格子状迷路課題のシミュレーションを実施し，制御性能がどのように変化するかを調査する．実験結果より，格子状迷路のような離散問題では許容する外乱の強さに対するパラメータを単に最大に設定することでロバストな制御器が獲得されるのに対し，振り子のような連続問題では，外乱の強さに対するパラメータは適切に調整しなければならないことが示す.

キーワード

最適制御, ロバスト制御,Bellman 方程式, 倒立振子制御, 移動ロボット

Robust control of autonomous systems based on the linearly solvable Markov Game*

Ken Kinjo

Abstract

Optimal control and robust control are the methods to obtain a control policy for an autonomous system, such as a robot, to act appropriately in adapting to surrounding environment. A control policy is derived by solving the Hamilton-Jacobi-Bellman (HJB) equation for optimal control and the Hamilton-Jacobi-Isaacs (HJI) equation for robust control, respectively. However, it is generally difficult to solve those equations analytically because they are nonlinear partial differential equations. A class of problem called Linearly solvable Markov Decision Process (LMDP), in which the HJB equation can be converted to a linear differential equation, was proposed and it was extended to a class called Linearly solvable Markov Game (LMG) to make the HJI equation linear. This linearization allows us to solve a certain class of nonlinear control problems efficiently. Previous studies on LMDP and LMG assumed that the environmental dynamics are given. However, a robot is often required to construct the model of the environmental dynamics through interaction with the environment. Moreover, the LMDP and LMG have been evaluated only in low dimensional problems, and there are only few applications to real systems.

In this research, to test the applicability of the LMDP and LMG to real systems, we investigate how the approximation error of the model affects the performance of the obtained control policy. First, we propose a framework which integrates LMDP and a

*Doctoral Dissertation, Department of Information Systems, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DD1161005, March 13, 2014.

module which learns the environmental model. Through the experiments of visually-guided navigation task using a wheeled-type mobile robot, we investigate how the approximation error of the environmental model affects the learning process of LMDP and show the proposed method is effective to the real system. Next, to clarify the relationship between the approximation error and the parameter which moderates the strength of the permissible disturbance, we conduct simulations using a grid-world task and a swing-up pendulum task under a variety of training and test environments and investigate the differences of the performances among the simulation conditions. The results show LMG gives a robust control policy in the discrete state problems such as the grid-world task when the above parameter is maximum, which means the worst disturbance is considered. On the other hand, we find that the parameter should be set appropriately to get a robust control policy in the continuous state problems such as the pendulum task.

Keywords:

Optimal Control, Robust Control, Bellman Equation, Inverted Pendulum, Mobile robot

目 次

図 目 次

表 目 次

第1章　はじめに

ロボットなどの自律システムにおける制御問題は，各関節の角度やカメラからの視覚情報などセンサー情報やアクチュエータへの出力値が実数となることから，状態行動空間は連続であることが望まれる．また制御対象のダイナミクスは非線形でノイズを含んだものである．このことから実機の制御は連続状態行動空間の非線形確率系の制御問題となる．

状態や制御に依存して与えられるコスト (報酬) が定義され，現在の状態と制御によって制御対象がどのように振る舞うかが定義された環境のダイナミクスが利用可能であるとする．この制御対象に対する最適な制御器 (方策) を設計するためのに，積算 (報酬) コストや単位時間当たりの平均コスト (報酬) などの評価関数を最小 (最大) にする制御則を求める．このような枠組みは最適制御 [1] やモデルベース強化学習 [2] と呼ばれる手法である．両者は，使用される評価関数の符号が異なるが本質的には同質の問題となっている．どちらも連続時間系では Hamilton-Jacobi-Bellman (HJB) 方程式を，離散時間系では Bellman 方程式を解くことが課題となる．即ち，これらの方程式を解くことによって得られる価値関数から最適制御則が導出される．

強化学習は神経科学分野において広く利用され [3]，近年，モデルベース強化学習は意思決定に関する研究者の間においても高い関心を集めている [4, 5].

しかしながら HJB 方程式および Bellman 方程式は非線形方程式であるために解を求めることは容易ではない．

さらに，最適制御やモデルベース強化学習の適用の条件として環境のダイナミクスが既知である必要があるが，実ロボットにおいてこの条件を満たすことは極めて稀である．そのため実際は，環境のダイナミクスを近似したモデルを仮定もしくは学習し，そのモデルを環境のダイナミクスの代替として使用する [6, 7].

1. 線形化 Bellman 方程式による学習制御

近年，Linearly solvable Markov decision process (LMDP) と呼ばれる，コスト関数と制御入力がダイナミクスに及ぼす影響に対して制約を置くことで，非線形な Bellman 方程式を線形に変換することが出来る新たなマルコフ決定過程の枠組みが提案された [8, 9].

LMDP は Bellman 方程式を線形化することで得られる興味深い特性により [8], ロボット工学や機械学習の分野において近年，注目を集めている [10, 11]．LMDP には経路積分に基づく手法 [12, 13] と価値関数を基づく手法 [8] の 2 種類が存在しており，この 2 つの手法は理論的な密接に関わりが存在することが報告されている [10]．しかしながら，実用において両者は大きく異なる．経路積分に基づく手法では，線形化 Bellman 方程式をベースにしたサンプリング手法を用いて始点と終点が固定の経路を評価する．このことから経路積分を基づく手法は，予め初期位置と目標位置が設定されるような経路計画の最適化に適した手法である．既に経路積分に基づく手法から得られる確率的な方策を用いて多自由度を持つロボットを制御する研究が存在する [14, 15, 16]．しかしながら，この手法では新たな初期位置と目標位置が与えられた場合には再度学習が必要となってくる．価値関数を基づく手法では価値関数を指数変換した指数価値関数 (*desirability function*) を定義し，指数価値関数が線形化 Bellman 方程式の解として得られる．線形化 Bellman 方程式は離散状態空間では固有値問題へと帰着され [17]，離散状態空間では固有関数問題へと帰着される [18, 19]. 指数価値関数に基づく手法の利点として Compositionality と呼ばれる特性が挙げられる．Bellman 方程式が線形化されたことにより以前に学習した課題を複合したような新たな課題の最適方策は，既に得られている指数価値関数の線形和にから求められるという特性である [11, 20]. さらに，一般的に不良設計問題となっている状態と行動の履歴から報酬関数を推定する逆強化学習 (Inverse Reinforcement Learning) と呼ばれる問題 [21, 22] において，LMDP を導入することで不良設計を排除することが可能となることが分かっている [23].

以上のように LMDP の価値関数に基づく手法は実機の学習制御においておもしろい特質を持った枠組みである．しかしながら，LMDP に限らず最適制御やモ

デルベースの強化学習では，制御対象のダイナミクスが既知という前提条件があり，実機においてこの前提条件を満たすことは極めて稀である．

そのため本研究では，この問題点を解決するためにランダムな方策のもとで集めたサンプルから制御対象のダイナミクスを関数近似器を用いて学習し，制御対象のダイナミクスを学習モデルで代替する．即ち，学習モデルを用いたLMDPに基づく学習制御機構を提案した．本研究では，この提案手法をシミュレーションと実機の実験を行いLMDPの価値関数に基づく手法の実機に対する実現可能性に関して評価した．

2. モデルの近似誤差に対するロバスト性

先に提案した手法で用いる学習モデルには必ず近似誤差が含まれておりその近似誤差による制御性能の劣化の恐れがある．この問題の一つの解決策として，たとえ学習モデルに近似誤差が存在していたとしても，課題を成功に導くことができるだけのロバスト性を持った方策を獲得する手法が考えられる．ロバストな制御則の獲得するための代表的な手法としてH^∞制御 [24] が挙げられ，線形行列不等式 (Linear Matrix Inequality) による制御器の設計法などが知られている [25]．また，強化学習の分野でもロバストな方策の獲得する手法が提案されている [26]．どちらの手法においてもエージェントの行動を妨げる敵対者 (Adversary) や妨害 (Disturber) を想定し，エージェントにとっても最も不利益になるように敵対者らが振舞っているもとでの最適な方策の獲得を目指す．即ち，獲得した方策は最悪な状況を踏まえた上での方策となっていることからロバストな方策となるのである．このような枠組みは2プレイヤーのMarkov Gamesとして知られ [27] サッカーゲームなどのマルチエージェント課題への適用例がある [28]．Markov Gamesにおいて，最適方策と価値関数は連続時間系ではHamilton-Jacobi-Isaacs (HJI) 方程式を方程式を解くことで得られる．また，離散時間系ではBellman-Isaacs方程式を解くことで得られる．これらの方程式もまた非線形方程式であるために解を求めることは容易ではない．

近年，Linearly Markov Game (LMG) と呼ばれるRisk sensitive Markov決定過

程 [29] の Bellman 方程式を線形化する枠組みが Dvijotham らにより提案された [30]．Risk sensitive Markov 決定過程は通常の最適制御の用いる目的関数を指数変換されたものを目的関数として定義したマルコフ決定過程であり，得られる制御則は制御則は通常の最適制御と比較してリスクに対してより忌避的な行動を実現する．この Risk sensitive Markov 決定過程と Markov Games の間には関連性があることが Fleming らにより示されており [31]，LMG もまた Markov Games として解釈可能である．このときの HJI 方程式および Bellman-Isaacs 方程式は線形となり，解析解を導出することが可能である．このため，LMG を利用することでロバストな制御則が効率的に得られることが期待される．

さらに LMG は LMDP の拡張として捉えることが出来ることから，先節で提案した学習モデルを利用した LMDP に基づく学習制御とも親和性が高く，学習モデルの近似誤差に対してもロバストな制御則を獲得できると予測される．このことから先の提案手法の LMDP の代わりに LMG 利用することでより実機に対して好ましい学習制御機構に改善されることが期待される．しかしながら，LMG によって得られた制御則はあくまで Risk sensitive Markov 決定過程に基づく制御則としてのみ議論され，モデルの近似誤差に対するロバスト性については十分に議論されていなかった．そのため本研究では LMG におけるモデルの近似誤差の影響を検証するため離散状態行動空間および連続状態行動空間のにおけるシミュレーション実験を行った．

3. 本研究の目的と論文の構成

本研究では実機への応用に適した学習制御器の構築を念頭に，LMDP 及び LMG を用いた学習制御機構の提案を行う．2 章では先行研究で提案されている学習モデルを用いた LMDP に基づく学習制御機構を振り子の振り上げ課題のシミュレーション実験と車輪型ロボット Spring Dog を用いた実機による実験を行い．提案手法の実現可能性と有効性に関して議論する．3 章では学習モデルの有する近似誤差による制御性能の劣化が，LMG を用いることによりどれほど抑えられるかを検証する．離散空間の問題として Grid-world with risky state 課題を，連続空間系

として振り子とアクロボットの振り上げ課題を行い．LMG のパラメータ α とモデルの近似誤差の関係について獲得した制御則を通して議論する．最後に 4 章にて本研究のまとめと今後の発展について述べる．

第2章　学習モデルを用いた Linearly Solvable MDP に基づく学習制御

LMDP を最適制御則の獲得のために使用するには前提条件として，制御対象に関するモデルが既知である必要がある．しかしながら，実機においてその前提条件が満たされることは極めて稀である．この問題点を解決するため本研究では実際のダイナミクスの代替として，サンプルデータから学習したモデルを用いて LMDP による学習制御を行う手法を提案する．類似する研究として Burdelis と池田により同様の手法 [32] が挙げられるが，この手法は離散状態行動空間に対する手法であり，連続空間に適用することはできない．

提案手法では学習モデルを制御則の導出に利用することから，本研究ではまずモデルの近似誤差が得られた制御則の性能に対してどれだけの影響を与えるかを振り子の振り上げ課題のシミュレーションを通して検証した．また実機に対する提案手法の実現性を検証するためにカメラの搭載された車輪型ロボットを用いた視覚誘導課題の実験を行った．

1. 学習モデルを用いたLMDPに基づく学習制御

本節では提案手法である学習モデルを用いた LMDP に基づく学習制御について述べる．本節は根幹となる連続状態空間における LMDP による Bellman 方程式の線形化について述べた 1.1 節，指数価値関数の関数近似とそれを用いた線形化 Bellman 方程式の解の導出手法について述べた 1.2 節と 1.3 節，モデルの学習手法について述べた 1.4 節により構成される．

1.1 LMDP による Belllman 方程式の線形化

この節では Todorov によって示された Bellman 方程式の厳密な線形化について説明する [8].

本論文では連続な状態と行動の空間をそれぞれ，$\mathcal{X} \subseteq \mathbb{R}^{N_{\mathbf{x}}}$，$\mathcal{U} \subseteq \mathbb{R}^{N_{\mathbf{u}}}$ とする．$N_{\mathbf{x}}$ と $N_{\mathbf{u}}$ はそれぞれ空間と行動の次元数を表している．

各時刻 t において制御対象は現在の状態 $\mathbf{x}(t) \in \mathcal{X}$ と実行した制御 $\mathbf{u}(t) \in \mathcal{U}$ を観測する．さらに制御対象は即時コスト $\ell(\mathbf{x}(t), \mathbf{u}(t))$ が与えられる．また，制御対象のダイナミクスが次のような確率微分方程式

$$d\mathbf{x} = \boldsymbol{a}(\mathbf{x})dt + \boldsymbol{B}(\mathbf{x})(\mathbf{u}\,dt + \sigma d\omega), \tag{2.1}$$

で記述されているとする．ここで $\boldsymbol{\omega} \in \mathbb{R}^{N_{\mathbf{u}}}$ と σ はそれぞれブラウンノイズとノイズのスケールを表している．$\boldsymbol{a}(\mathbf{x}) \in N_{\mathbf{x}}$ はシステムの受動的なダイナミクスを示し，$\boldsymbol{B}(\mathbf{x}) \in N_{\mathbf{x}} \times N_{\mathbf{u}}$ は入力ゲインを示している．式 (2.1) は状態 $\mathbf{x}$ 対して非線形であるが制御量 $\mathbf{u}$ に関して線形であること，ブラウン $\boldsymbol{\omega}$ は制御と同一空間上で作用していることに注意されたい.

以降の説明のためダイナミクス (2.1) を微小時間幅 h で区切ることで時間に関して離散化すると，以下のような状態遷移確率として得られる．

$$\begin{aligned} p^{\mathbf{u}}(\mathbf{x}_{k+1} \mid \mathbf{x}_k) &= \mathcal{N}(\mathbf{x}_{k+1} \mid \boldsymbol{\mu}(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{x}_k, \boldsymbol{\Sigma}(\mathbf{x}))h \\ \boldsymbol{\mu}(\mathbf{x}, \mathbf{u}) &= (\boldsymbol{a}(\mathbf{x}) + \sigma\boldsymbol{B}(\mathbf{x})\mathbf{u})h \end{aligned} \tag{2.2}$$

ここでは表記を簡略化するために時刻 $t = hk$ の状態と制御をそれぞれ $\mathbf{x}_k = \mathbf{x}(hk)$，$\mathbf{u}_k = \mathbf{u}(hk)$ と表現している．$\mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma})$ は平均 $\boldsymbol{\mu}$ と共分散行列 $\boldsymbol{\Sigma}$ であるようなガウス分布であり，

$$\boldsymbol{\mu}(\mathbf{x}, \mathbf{u}) = h(\boldsymbol{a}(\mathbf{x}) + \boldsymbol{B}(\mathbf{x})\mathbf{u}), \tag{2.3}$$

となっている．$\boldsymbol{\mu}(\mathbf{x}, \mathbf{u})$ は決定論的な状態遷移であると捉えることが可能である．

制御入力が与えられていないとき，即ち $\mathbf{u} = \mathbf{0}$ ときにおける状態遷移確率を uncontrolled probability と呼び，$p^{\mathbf{0}}(\mathbf{x}' \mid \mathbf{x})$ と表す．また，制御入力が与えられたときの状態遷移確率を controlled probability と呼び，$p^{\boldsymbol{u}}(\mathbf{x}' \mid \mathbf{x})$ と表す.

方策または制御則 $\pi(\mathbf{u} \mid \mathbf{x})$ は各状態 $\mathbf{x}$ における制御量 $\mathbf{u}$ を定義している．例えば，制御対象を吸収状態 $\mathbf{x}_{\mathcal{A}} \in \mathcal{X}_{\mathcal{A}} \subseteq \mathcal{X}$ へと導くことを目標とした課題，即ち，目標状態と吸収状態が同一である課題における制御則 $\pi^*(\mathbf{u} \mid \mathbf{x})$ を得るための目的関数は以下のような積算コストの期待値から定義される．

$$v^{\pi(\mathbf{u}|\mathbf{x})}(\mathbf{x}) = \mathbb{E}\left[\sum_{k=1}^{T_{\mathcal{A}}-1} \ell(\mathbf{x}_k, \pi(\mathbf{x}_k)) + g(\mathbf{x}_{T_{\mathcal{A}}})\right]$$

$g(\mathbf{x})$ は目標状態 (吸収状態)$\mathbf{x}_{\mathcal{A}}$ の終端コスト，$T_{\mathcal{A}}$ は吸収状態に初めて到達した時刻である．この目的関数は制御則 $\pi(\mathbf{u} \mid \mathbf{x})$ に対する cost-to-go 関数 (価値関数) とも呼ばれている．

この cost-to-go 関数の方策に関する最小値が最適 cost-to-go 関数となり以下のように導出され，

$$v^*(\mathbf{x}) = \min_{\pi(\mathbf{u}|\mathbf{x})} v^{\pi(\mathbf{u}|\mathbf{x})}(\mathbf{x}),$$

これに対応する制御則が最適制御則 $\pi^*(\mathbf{u} \mid \mathbf{x})$ となる．

$$\pi^*(\mathbf{u} \mid \mathbf{x}) = \arg\min_{\pi(\mathbf{u}|\mathbf{x})} v^{\pi(\mathbf{u}|\mathbf{x})}(\mathbf{x})$$

以下に示される Bellman 方程式が最適 cost-to-go 関数の必要十分条件を与えることが知られている，

$$v^*(\mathbf{x}) = \min_{\mathbf{u}} \left(\ell(\mathbf{x}, \mathbf{u}) + \mathbb{E}_{\mathbf{x}' \sim p^{\mathbf{u}}(\cdot|\mathbf{x})} v^*(\mathbf{x}')\right), \tag{2.4}$$

$$v^*(\mathbf{x}_{\mathcal{A}}) = g(\mathbf{x}_{\mathcal{A}}), \quad \mathbf{x}_{\mathcal{A}} \in \mathcal{X}_{\mathcal{A}},$$

式 (2.4) は最小化演算子を含んだ非線形方程式であるため，一般的に最適 cost-to-go 関数を解くことは困難である．しかしながら，LMDP では即時コストが状態依存のコスト $q(\mathbf{x})$ と controlled probability と uncontrolled probability の[1]KL ダイバージェンスによって定義される制御コストの和であると仮定することにより Bellman 方程式は線形化される．

$$\ell(\mathbf{x}, \mathbf{u}) = hq(\mathbf{x}) + \mathrm{KL}(p^{\mathbf{u}}(\mathbf{x}' \mid \mathbf{x}) \parallel p^{\mathbf{0}}(\mathbf{x}' \mid \mathbf{x})) \tag{2.5}$$

[1]Kullback-Leibler(KL) ダイバージェンスは 2 つの異なる分布の距離を測る尺度である．2 つの分布が全く同じであるとき KL ダイバージェンスは 0 となる．LMDP において， 制御量 $\mathbf{u}$ に関する制御コストはどれだけ状態遷移確率に影響を与えたかによって定義される．

仮定したコスト (式 (2.5)) を Bellman 方程式 (式 (2.4)) の右辺に代入すると，右辺の最小化を実現する最適な controlled probability $p^{\mathbf{u}^*}(\mathbf{x}' \mid \mathbf{x})$ が以下のように得られる．

$$p^{\mathbf{u}^*}(\mathbf{x}' \mid \mathbf{x}) = \frac{p^0(\mathbf{x}' \mid \mathbf{x}) \exp(-v^*(\mathbf{x}'))}{\int p^0(\mathbf{x}' \mid \mathbf{x}) \exp(-v^*(\mathbf{x}'))d\mathbf{x}'}$$

ここで以下のように積分演算子 $\mathcal{G}$ と指数価値関数あるいは好適度関数 (desirability function) $z(\mathbf{x})$ を導入する．

$$\mathcal{G}[f](\mathbf{x}) = \int p^0(\mathbf{x}' \mid \mathbf{x}) f(\mathbf{x}')d\mathbf{x}' \tag{2.6}$$

$$z(\mathbf{x}) = \exp(-v^*(\mathbf{x})). \tag{2.7}$$

これによって最適な controlled probability は，

$$p^{\mathbf{u}^*}(\mathbf{x}' \mid \mathbf{x}) = \frac{p^0(\mathbf{x}' \mid \mathbf{x}) z(\cdot)}{\mathcal{G}[z](\mathbf{x})} \tag{2.8}$$

と表され，この結果を再度，Bellman 方程式に代入する．その結果，以下のように Bellman 方程式は指数価値関数 $z(\mathbf{x})$ 関して厳密に線形化される．

$$\begin{aligned} z(\mathbf{x}) &= \exp(-hq(\mathbf{x}))\mathcal{G}[z](\mathbf{x}) \\ z(\mathbf{x}_g) &= \exp(-q(\mathbf{x}_{\mathcal{A}})), \quad \mathbf{x}_{\mathcal{A}} \in \mathcal{X}_{\mathcal{A}}, \end{aligned} \tag{2.9}$$

以降では上式のことを線形化 Bellman 方程式と呼ぶ．

ここで制御コストとして定義した controlled probability と uncontrolled probability の KL-ダイバージェンスは式 (2.2) の状態遷移確率の定義より，以下のような制御量の 2 次形式となる．

$$\mathrm{KL}(p^{\mathbf{u}}(\mathbf{x}' \mid \mathbf{x}) \parallel p^{\mathbf{0}}(\mathbf{x}' \mid \mathbf{x})) = \frac{h}{2\sigma^2} \mathbf{u}^{\mathrm{T}} \left(\boldsymbol{B}(\mathbf{x})^{\mathrm{T}} \boldsymbol{B} \boldsymbol{B}(\mathbf{x}) \right)^{-1} \mathbf{u} \tag{2.10}$$

このためノイズのスケール σ が 0 へと近づくと制御コストが発散してしまい，導出される制御則が課題を成功させるために必要な制御量を出力できなくなる恐れがある．

1.2 指数価値関数の関数近似

線形化 Bellman 方程式 (2.9) は指数価値関数 $z(\mathbf{x})$ を固有関数とするような固有関数方程式となっている．固有関数を解くことは一般的に困難である．Todorov は放射基底関数 (RBF) を用いた固有関数の関数近似を行うことでこの問題を解決している [18].

指数価値関数 $z(\mathbf{x})$ を N_z 個の基底関数 $\{f_i(\mathbf{x})\}$ を用いて以下のように線形近似する．

$$z(\mathbf{x};\boldsymbol{w},\boldsymbol{\theta}) = \sum_{i=1}^{N_z} w_i f(\mathbf{x},\boldsymbol{\theta}_i) = \boldsymbol{w}^\top \boldsymbol{f}(\mathbf{x},\boldsymbol{\theta}), \tag{2.11}$$

ここで w_i は線形パラメータ，$\boldsymbol{w}$ は線形パラメータを要素に持つベクトル，$[w_1,\ \ldots,\ w_{N_z}]^{\mathrm{T}}$，を表している．また，$f(\mathbf{x},\boldsymbol{\theta}_i)$ は $\boldsymbol{\theta}_i$ をパラメータに持つ基底関数，$\boldsymbol{f}(\mathbf{x};\boldsymbol{\theta})$ は基底関数を要素に持つベクトル，$[f(\mathbf{x};\theta_1),\ \ldots,\ f(\mathbf{x};\theta_{N_z})]^{\mathrm{T}}$，を表している．

基底関数 f_i は Todorov が [18] にて提案しているように，非正規化ガウス関数を採用した．

$$f(\mathbf{x};\boldsymbol{\theta}_i) = \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{m}_i)^{\mathrm{T}}\boldsymbol{S}_i(\mathbf{x}-\boldsymbol{m}_i)\right), \quad \boldsymbol{\theta}_i = \{\boldsymbol{m}_i,\boldsymbol{S}_i\} \tag{2.12}$$

ここで $\boldsymbol{m}_i$ と $\boldsymbol{S}_i$ はそれぞれ i 番目の基底関数の中心 $\boldsymbol{m}_i$ と共分散行列の逆行列 $\boldsymbol{S}_i$ を表している．

ガウス関数を利用する利点の一つとして，積分演算子 (2.6) が以下のように解析的に得られる．

$$\mathcal{G}[f_i](\mathbf{x}) = |\boldsymbol{V}_i|^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{y}-\boldsymbol{m}_i)^{\mathrm{T}}\boldsymbol{H}_i(\mathbf{y}-\boldsymbol{m}_i)\right), \tag{2.13}$$

ここで $\mathbf{y}$ は制御入力を $\mathbf{u}=\mathbf{0}$ としたときの次状態の期待値 $\mathbf{y}=\mathbf{x}+h\boldsymbol{a}(\mathbf{x})$ であり，

$$\boldsymbol{H}_i = \boldsymbol{S}_i - \boldsymbol{S}_i\boldsymbol{C}\boldsymbol{V}_i^{-1}\boldsymbol{C}^{\mathrm{T}}\boldsymbol{S}_i, \qquad \boldsymbol{V}_i = \boldsymbol{I} + \boldsymbol{C}^{\mathrm{T}}\boldsymbol{S}_i\boldsymbol{C},$$
$$\boldsymbol{C} = \sigma h^{1/2}\boldsymbol{B}$$

である．ここで $\boldsymbol{y},\boldsymbol{H}_i,\boldsymbol{V}_i,\boldsymbol{C}$ はすべて状態 $\mathbf{x}$ の関数であるが，表記を簡略化するために引数 $(\mathbf{x})$ を省略している．LMDP では明示的に最適な制御則 $\mathbf{u}^*(\mathbf{x})$ は

線形化 Bellman 方程式から得ることができない．しかしながら，状態遷移確率 $p^{\mathbf{u}^*}(\mathbf{x}' \mid \mathbf{x})$ を実現するための最適制御則 $\mathbf{u}^*(\mathbf{x})$ は，遷移後の状態 $\mathbf{x}'$ の期待値と最適状態遷移 確率 $p^{\mathbf{u}^*}(\mathbf{x}' \mid \mathbf{x})$ の期待値が一致するという条件から，

$$\mathbf{u}^*(\mathbf{x}) = \sigma \sum_{i=1}^{N_{z(\mathbf{x})}} \frac{w_i \mathcal{G}[f_i(\mathbf{x})]}{\sum_{k=1}^{N_{z(\mathbf{x})}} w_k \mathcal{G}[f_k(\mathbf{x})]} \boldsymbol{d}_i(\mathbf{x}) \tag{2.14}$$
$$\boldsymbol{d}_i(\mathbf{x}) = \boldsymbol{V}_i^{-1} \boldsymbol{C}^{\mathrm{T}} \boldsymbol{S}_i \left(\mathbf{m}_i - \mathbf{x} - h\boldsymbol{a}(\mathbf{x})\right)$$

となる．

1.3 指数価値関数の学習

指数価値関数の近似関数もまた線形化 Bellman 方程式 (式 (2.9)) を満たす必要がある．式 (2.11) で示した指数価値関数の近似関数の定義より線形化 Bellman 方程式 (2.9) は

$$\sum_{i=1}^{N_{z(\mathbf{x})}} w_i f_i(\mathbf{x}) = \sum_{i=1}^{N_{z(\mathbf{x})}} w_i \exp(-hq(\mathbf{x}))\mathcal{G}[f_i](\mathbf{x}) \tag{2.15}$$

と変形できる．

状態空間 $\mathcal{X}$ から任意の状態を節点（collocation state）[2]として選び，全ての節点で式 (2.15) を満たすような線形パラメータ $\mathbf{w}$ と基底関数のパラメータ $\boldsymbol{\theta}$ を求めることで指数価値関数を近似する．ここで節点の番号を n として以下のような要素を持つ行列 $\boldsymbol{F}(\boldsymbol{\theta})$, $\boldsymbol{G}(\boldsymbol{\theta})$ を定義する。

$$\boldsymbol{F}(\boldsymbol{\theta})_{ni} = f_i(\mathbf{x}_n) \tag{2.16}$$
$$\boldsymbol{G}(\boldsymbol{\theta})_{ni} = \exp(-hq(\mathbf{x}_n))\mathcal{G}[f_i](\mathbf{x}_n) \tag{2.17}$$

これらの行列を用いることにより，式 (2.15) の行列表記

$$\boldsymbol{F}(\boldsymbol{\theta})\boldsymbol{w} = \boldsymbol{G}(\boldsymbol{\theta})\boldsymbol{w} \tag{2.18}$$

[2]節点は一般的な関数近似の手法では学習点とも呼ばれたりする．

が得られる．

以上を考慮して目的を関数を

$$\begin{aligned} e &= \|\boldsymbol{r}(\boldsymbol{\theta}, \boldsymbol{w})\|^2, \\ \boldsymbol{r}(\boldsymbol{\theta}, \boldsymbol{w}) &= (\boldsymbol{F}(\boldsymbol{\theta}) - \boldsymbol{G}(\boldsymbol{\theta}))\, \boldsymbol{w} \end{aligned} \tag{2.19}$$

と定義する．式 (2.19) は $\boldsymbol{w}$ の二次関数で $\boldsymbol{\theta}$ の非線形関数である．即ち，e を $\boldsymbol{w}, \boldsymbol{\theta}$ に関して最小化することで指数価値関数を近似解を得ることが出来る．それぞれのパラメータの最適化は付録 2 にて記述する．

また目標状態 $\mathbf{x}_g$ では $z(\mathbf{x}_g) = \exp(-q(\mathbf{x}_g))$ が明らかであることから，関数近似に対して以下の条件を加える．

$$\boldsymbol{f}(\mathbf{x}_g; \boldsymbol{\theta})^{\mathrm{T}} \boldsymbol{w} = \exp(-g(\mathbf{x}_g)) \tag{2.20}$$

1.4 モデルの学習手法

1.2 節で紹介した指数価値関数及び制御則の導出は制御対象のダイナミクス (式 (2.1)) が既知であると仮定した手法であることから，制御対象のダイナミクスが与えられていない場合には何らかの手法で制御対象のダイナミクスを近似したモデルが必要である．ダイナミクスを近似するモデルの学習には様々な手法が提案されている [33, 34].

本研究の提案手法では，学習モデルの予測性能を上げることに焦点を当てるのではなく，近似誤差を含む場合であっても LMDP によって得られた方策がどれだけその性能を維持できるかに焦点を当てることから，単純な最小二乗法に基づいて式 (2.2) の決定論的な状態遷移である $\boldsymbol{\mu}(\mathbf{x}, \mathbf{u})$ を近似する手法を採用した．ここで一般的に未知であるノイズの尺度 σ は既知であるという事に注意されたい．なぜなら式 (2.14) でも示されているように，指数価値関数に基づく最適制御則の制御量はノイズの尺度に比例することから，現実的な制御量を得るためにも任意で設定する必要があったためである．

提案手法では以下のように式 (2.2) の決定論的な状態遷移 $\boldsymbol{\mu}(\mathbf{x}, \mathbf{u})$ を N_φ 個の基底関数 $\varphi_i(\mathbf{x}, \mathbf{u})$ の線形和によって近似する．

$$\boldsymbol{\mu}(\mathbf{x}, \mathbf{u}; \boldsymbol{W}) = \boldsymbol{W}^{\mathrm{T}} \boldsymbol{\varphi}(\mathbf{x}, \mathbf{u}) \tag{2.21}$$

ここで $\boldsymbol{W}$ は重み行列で $\boldsymbol{\varphi}(\mathbf{x}, \mathbf{u})$ は基底関数をまとめた基底関数ベクトルである．

訓練データとして適当な制御則のもとで得られた状態と制御の時系列データ，$\{\mathbf{x}_1, \mathbf{u}_1, \ldots, \mathbf{x}_{N_s}, \mathbf{u}_{N_s}, \mathbf{x}_{N_s+1}\}$，が与えられたとする．モデル学習における目的関数は以下の二乗誤差和で定義される．

$$E = \frac{1}{2} \sum_{k=1} \left\{ \Delta \mathbf{x}_k - \boldsymbol{W}^{\mathrm{T}} \boldsymbol{\varphi}(\mathbf{x}_k, \mathbf{u}_k) \right\}^2, \tag{2.22}$$

ここで $\Delta \mathbf{x}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$ である．

目的関数 (2.22) を最小化するような重み行列は，$\partial E / \partial \boldsymbol{W} = \mathbf{0}$ から，

$$\boldsymbol{W} = (\boldsymbol{\Phi}^{\mathrm{T}} \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^{\mathrm{T}} \Delta \boldsymbol{X} \tag{2.23}$$

となることが分かる．ここで，$\Delta \boldsymbol{X}$ と $\boldsymbol{\Phi}$ はそれぞれ訓練データの差分 $\Delta \mathbf{x}_k$ から構成される差分行列と基底関数 $\boldsymbol{\varphi}(\mathbf{x}_k, \mathbf{u}_k)$ から構成される行列であり，以下のように構築されている．

$$\Delta \boldsymbol{X} = \left[\Delta \mathbf{x}_1, \ \cdots, \ \Delta \mathbf{x}_{N_s} \right]^{\mathrm{T}}, \quad \boldsymbol{\Phi} = \left[\boldsymbol{\varphi}(\mathbf{x}_1, \mathbf{u}_1), \ \cdots, \ \boldsymbol{\varphi}(\mathbf{x}_{N_s}, \mathbf{u}_{N_s}) \right].$$

2. シミュレーション実験: 振り子の振り上げ課題

振り子はダイナミクスが非線形であることから，非線形な制御対象の制御システムの基本的な理解を得るための問題として多くの研究で用いられており，エネルギー法を用いた制御手法 [35, 36] やニューラルネットワークを用いた学習制御手法 [37] など様々な制御手法が提案されている．

本研究で用いる LMDP は制御対象が非線形なダイナミクスであっても最適な Cost-to-go 関数及び対応する最適制御則を導出すること可能な手法である．このことから振り子の振り上げ課題のシミュレーション実験を行った．実験では様々な関数近似器を用いた学習モデルから得られる制御則と真のダイナミクスから得られた制御則の比較を行い，近似誤差に対してどれだけの制御性能を維持できるかを調査した．

2.1 課題設定

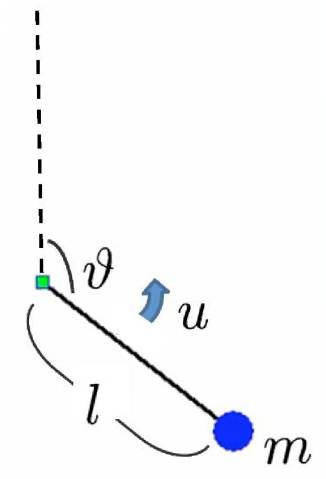


図 2.1 実験に使用した振り子のイメージ図

図 2.1 はシミュレーションで想定した振り子のイメージ図である．シミュレーションで用いた振り子の状態変数は角度 ϑ と角速度 $\dot{\vartheta}$ によって定義され，以下のような運動方程式に従うものとした．

$$\begin{aligned} d\vartheta &= \dot{\vartheta} dt \\ d\dot{\vartheta} &= \left(m\frac{g}{l}\sin(\vartheta) - k\dot{\vartheta} \right) dt + u dt + \sigma d\omega \end{aligned} \tag{2.24}$$

ここで一次元の制御量 u はシステムに対してトルクとして働く．l は振り子の長さ，g は重力加速度，k は摩擦係数である．実際のシミュレーションで微小時間 $t = 0.01[s]$ で離散化を行った．また，振り子の長さを $l = 1[m/s]$，質量を $m = 1[kg]$，重力加速度を $g = 9.8[m/s^2]$，摩擦係数を $k = 0.05$ とした．ノイズのスケール σ に設定については，LMDP による制御則の性能を大きく左右することから以降の節で述べることとする．

状態 $\mathbf{x}$ は $\left[\vartheta,\ \dot{\vartheta}\right]^{\mathrm{T}}$ となり，1.1 節で用いた状態方程式 (2.1) の 形式に従えば，

$$\boldsymbol{a}(\mathbf{x}) = \left[\dot{\vartheta},\ m\tfrac{g}{l}\sin(\vartheta) - k\dot{\vartheta}\right]^{\mathrm{T}}, \qquad B = \left[0,\ 1\right]^{\mathrm{T}}$$

となる．$\boldsymbol{a}(\mathbf{x})$ は $\mathbf{x}$ に関する非線形のベクトル関数であるが，B は定数ベクトルであることに注意されたい．振り子が鉛直方向に振り上がり，倒立制止状態を目標状態として設定し，$-3.6[\mathrm{deg}] \leq \vartheta \leq 3.6[\mathrm{deg}]$，$-14.4[\mathrm{deg}/s] \leq \dot{\vartheta} \leq -14.4[\mathrm{deg}/s]$ に到達するとタスクが終了する吸収状態 $\mathbf{x}_{\mathcal{A}} \in \mathcal{X}_{\mathcal{A}}$ とした．目標状態 (吸収状態) でコストが 0 となるように，状態コスト関数 $q(\mathbf{x})$ を非正規ガウス関数を用いて

$$q(\mathbf{x}) = \alpha(1 - \exp(\mathbf{x}^{\mathrm{T}}\Sigma_{\mathrm{cost}}^{-1}\mathbf{x})) \tag{2.25}$$

のように定義した．$\boldsymbol{\Sigma}_{\mathrm{cost}}$ と α はそれぞれガウス関数の共分散行列とコストのスケールとなっている．今回のシミュレーションでは $\mathrm{diag}(\boldsymbol{\Sigma}_{\mathrm{cost}}) = [\pi/4,\ 4\pi/4]^2$, $\alpha = 1$ とした．

タスクでは角度速度成分に関しては上限と下限を設定し，遷移可能な状態空間は $\dot{\vartheta} \in [-4\pi,\ 4\pi]$ とする．角度の範囲は $\vartheta \in [-\pi,\ \pi]$ とし，$\vartheta = -\pi$ と $\vartheta = \pi$ は同一である．

ダイナミクスの学習には各時刻の状態と次時刻の状態との差分 $\Delta\mathbf{x}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$ を用いた．しかし，角度 ϑ の境界をまたぐ状態変化は連続していることを考慮して差分を与えた．

サンプルの過度な偏在を抑えるために，推定のための状態行動系列 $\{\mathbf{x}_\cdot, u_\cdot\}$ の取得期間では，何度か始点を変えてサンプルを取得する．状態系列のサンプルの中から無作為に 1000 サンプルを選び出しモデル学習のための訓練データ $\mathcal{D}$ として用いる．シミュレーションでは表 2.2 に示す 2 つの関数近似器，Linear model, Linear-NRBF model を用意してダイナミクスの学習を行った．Linear-NRBF model に用いる正規化動径基底関数関数 (Normalized radial basis function，NRBF) は以下のように定義されている．

$$\psi_n(\mathbf{x}) = \frac{\exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_n)^{\mathrm{T}}\boldsymbol{\Sigma}_{\psi_n}^{-1}(\mathbf{x} - \boldsymbol{\mu}_n)\right)}{\sum_k \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_k)^{\mathrm{T}}\boldsymbol{\Sigma}_{\psi_k}^{-1}(\mathbf{x} - \boldsymbol{\mu}_k)\right)} \tag{2.26}$$

ここで訓練データ $\mathcal{D}$ の状態 $\mathbf{x}$ に関して K-means クラスタリングを行った結果得られたクラスターの中心を NRBF の中心 $\boldsymbol{\mu}_i$ として用いた．共分散行列 $\boldsymbol{\Sigma}_{\psi_i}$ は状態空間の領域と基底関数の数に応じてハンドチューニングでその値を決定した．

指数価値関数の近似には節点行列 $\mathbf{F}, \mathbf{G}$ が必要になるため，以下のように節点と指数関数の基底関数 $f(\mathbf{x})$ のパラメータの初期値を設定した．まず，節点は状

態空間 $\mathcal{X}$ に対して一様に分布するように格子状に $N_c = 441$ 個配置した．指数価値関数の基底関数 $f(\mathbf{x})$ のパラメータの初期値は，基底関数の中心 $\boldsymbol{m}_{\cdot}$ が節点と同じように状態空間に対して格子状に $N_f = 441$ 個配置し，基底関数の共分散 $\mathbf{S}_{\cdot}$ はハンドチューニングよって同一の共分散行列 $\mathrm{diag}(\mathbf{S}) = [2\pi/25,\ 8\pi/25]^{-2}$ とした．今回のシミュレーションでは付録 2 の手法を用いて，指数価値関数の基底関数の中心 $\boldsymbol{m}_{\cdot}$ のみを最適化した．ここで得られた指数価値に基づき最適制御則 $\mathbf{u}^*(\mathbf{x})$ を式 (2.14) を利用して求めた．

2.2 実験結果

節 1.1 において説明したように，ノイズのスケールによってはタスク達成に必要なだけの制御量を出力しない制御則が導出される恐れがある．そのことから近似誤差による得られた制御則の性能を調査する前に，コストの大きさを $\alpha = 1$ で固定した状態で様々なスケールのノイズ，$\sigma = [1, 1.5, 2, 3, 4, 5, 8, 10, 12]$ を仮定し，式 (2.24) によって定義される振り子のダイナミクス (以下では真のダイナミクスと表記) を用いて LMDP に基づく制御則を導出し，以降の実験で用いるノイズのスケールを決定することとした．

異なるスケールのノイズ $\sigma = [1, 4, 10]$ における指数価値関数と対応する最適制御則を図 2.2 は示している．図 2.2 のそれぞれのパネルは x 軸上に振り子の角速度 $\vartheta[\mathrm{rad}]$，y 軸上に角速度 $\dot{\vartheta}[\mathrm{rad}/s]$ を取るように描画している．一番右上のパネルはシミュレーションで使用したコスト関数を表しており，上から 2 行目以降は右列に指数価値関数，左列に最適制御則とし，σ の値が小さい順に表示している．

表 2.1 振り子のダイナミクスの学習に用いた関数近似器の基底関数

	$\boldsymbol{\phi}(\mathbf{x}, \mathbf{u})$
Linear model	$\begin{bmatrix}\mathbf{x}^{\mathrm{T}} & \mathbf{u} & 1\end{bmatrix}^{\mathrm{T}}$
Linear-NRBF model	$\begin{bmatrix}\mathbf{x}^{\mathrm{T}} & \psi_1(\mathbf{x}) & \psi_2(\mathbf{x}) & \cdots & \psi_M(\mathbf{x}) & \mathbf{u} & 1\end{bmatrix}^{\mathrm{T}}$

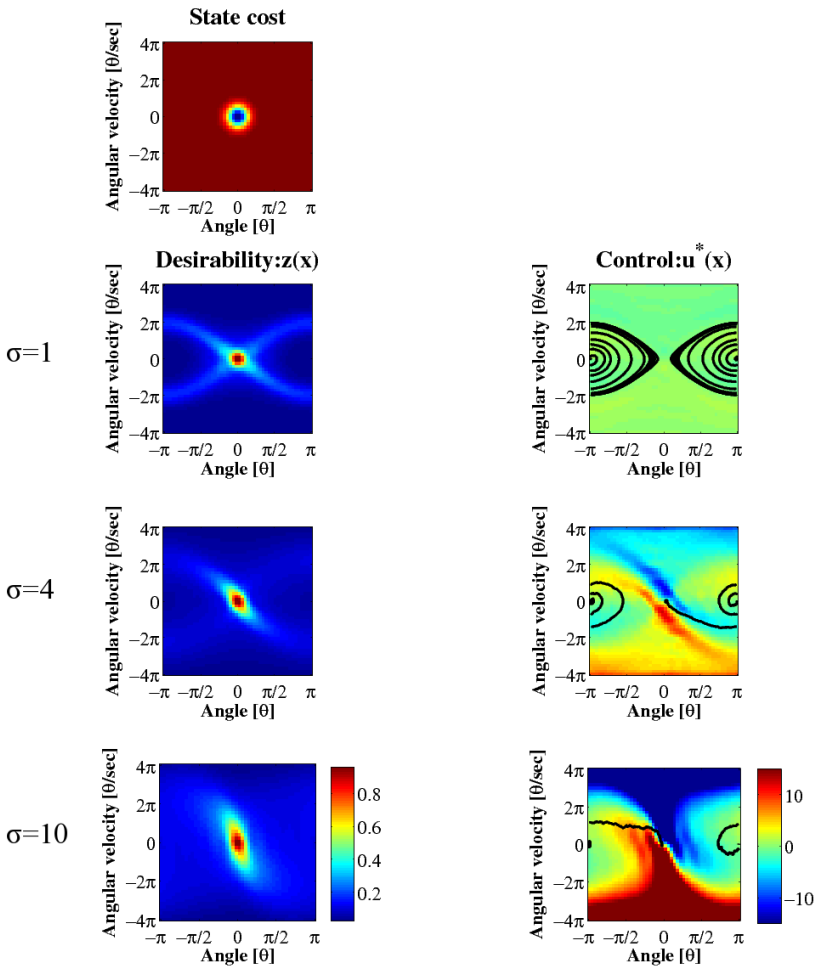


図 2.2 様々なスケールのノイズのもとでの指数価値関数と制御則

また，最適制御則を示すパネル上に描画されている実線は振り子が下がった状態 $\mathbf{x} = [\pi,\ 0]^{\mathrm{T}}$ を初期位置として最適制御則を用いたときの目標状態に到達するまでもしくは 60 秒経過するまでの軌道の一例を示している．全てのスケールのノイズで得られた指数価値関数は，目標状態である原点をピークに図中で対角線を描くように指数価値関数の値が高い領域が存在している．この図中の対角線の領域は制御入力がなくとも振り子の慣性で目標状態へと到達することができる領域付近となっている．即ち，制御によるコストが少なくても目標状態へ辿り着く領域の指数価値関数の値が大きくなっていると推察される．この推察と一致するように，どのノイズから得られた最適制御則も指数価値関数の値が高くなっている領域における制御量はほぼ 0 になっている．その一方でノイズのスケールが大きくなるに従い指数価値関数の値の高い領域は広範囲になりなだらかになっていく．また値の高い領域が描く図中の対角線の傾きは急になり，負の傾きをとる対角線上の領域の方が正の傾きを対角線上のより高い値へと変化している．最適制御則はノイズのスケールが大きくなるに従って全体的に大きな値をとるようになっている．その結果，振り子を下で振る回数は少なくなり，振り子の慣性を用いるのではなく大きな力で目標状態へと導く制御則へと変化している．また $\sigma = 1$ のときには十分な大きさの制御量が出力できず，60 秒経過後も目標状態に到達することができなかった．

図 2.3 は振り子が下がった状態 $\mathbf{x} = [\pi,\ 0]^{\mathrm{T}}$ を初期位置として最適制御則を用いたときの目標状態に到達するまでもしくは，60 秒経過するまでの制御量の絶対値の最大値と平均値の変化をノイズのスケール σ を x 軸上にとりプロットしたものである．絶対値の最大値と平均値ともにノイズのスケール σ が大きくなるに従い増加している．この結果は最適制御則の導出に用いた式 (2.14) とも一致している．この結果から，過大な制御量を出力せずに振り子の慣性を利用しながら振り上げを実現する制御則を導出することができるノイズのスケール $\sigma = 4$ を使用して以降の実験を行った．

節 2.1 でも述べたように，本実験では振り子のダイナミクスを Linear モデルと NRBF-Linear モデルの 2 種類の関数近似器を使用して学習した．まずは学習モデ

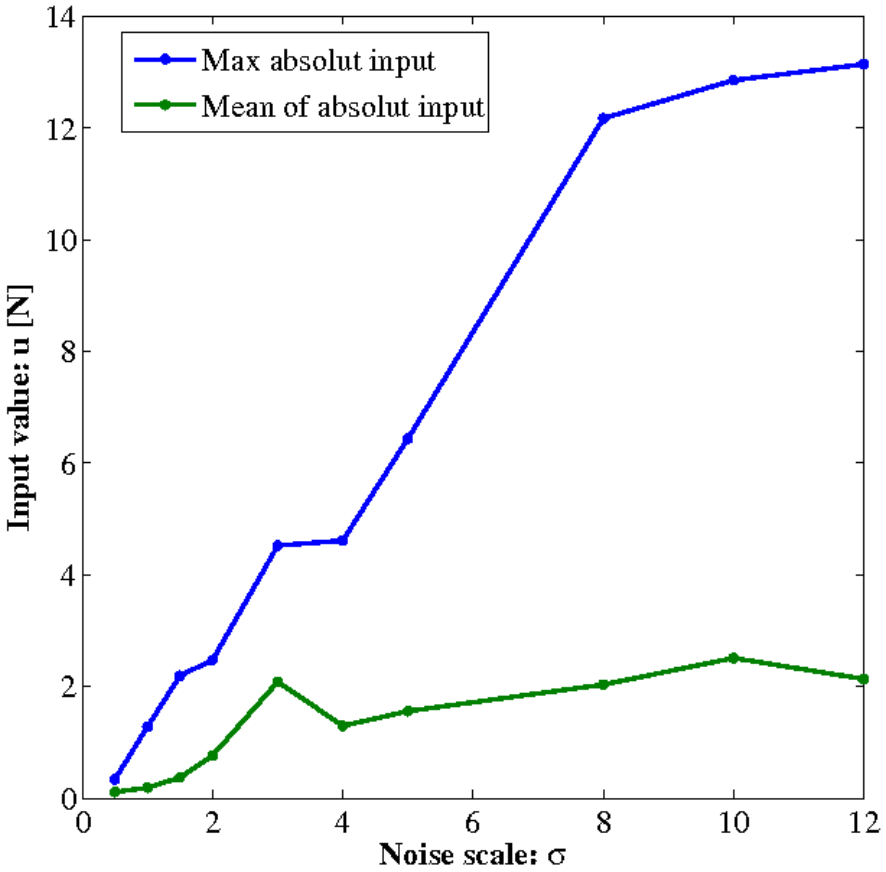


図 2.3 振り子が振りあがるまでの制御量の絶対値の最大値と平均値

ルの予測性能を評価するために，訓練データ $\mathcal{D}$ の抽出元であった系列データのサンプルから無作為に 500 個をテストデータとして抽出し，学習モデルの近似誤差の平均 2 乗誤差を求めた．

図 (2.4) は各成分の平均 2 乗誤差とその信頼区間をプロットした図である．振り子の角度成分の状態遷移はノイズの影響を受けないことから決定論的でかつ線形であることから，Linear model においても誤差なく推定ができている．その一方で，各速度成分の状態遷移はノイズの影響を受けるため近似誤差が発生している．しかしながら，Linear-NRBF model における予測の二乗誤差平均はノイズによる分散 ($(h\sigma)^2 = 1.6 \times 10^{-2}$) と一致していおり，モデルによる推定誤差はほとんど生じていないと考えられる．その一方でノイズによる分散以上の 2 乗誤差が現れてい

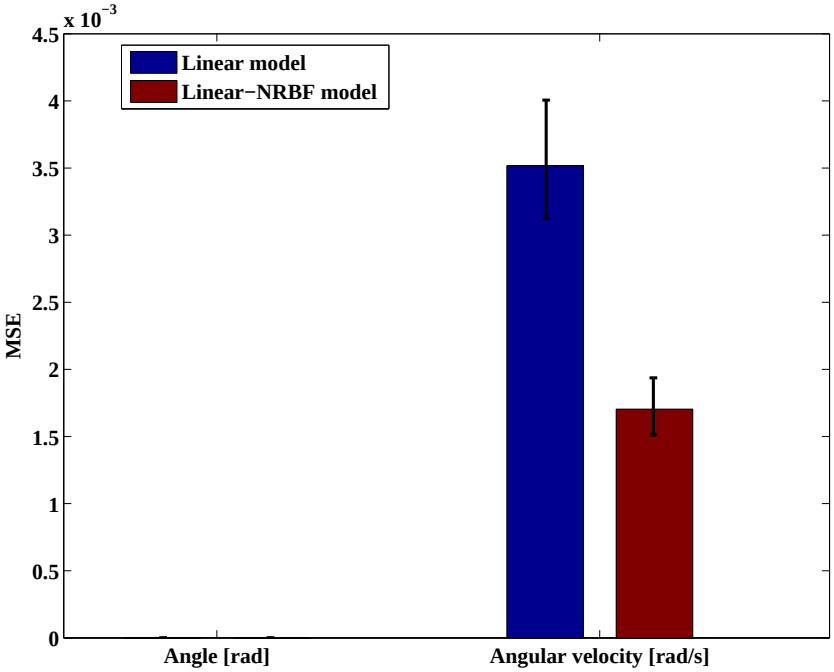


図 2.4 学習モデルの各成分における近似誤差

ることから何らかの近似誤差が発生していると考えられる．両者のモデルの近似誤差の違いは角速度成分の状態遷移に含まれる非線形な三角関数を Linear model では近似することができなかった一方で，Linear-NRBF model では近似することができたためであると考えられる．特に，今回のシミュレーションでは鉛直方向に振り上がったときを原点としていることから，振り子の下の方である $\vartheta = -\pi$ と $\vartheta = \pi$ 付近では Linear model の近似誤差は大きくなっている．

今回の実験で用意したモデルの関数近似器はどちらのモデルを用いたとしても，制御量が状態遷移に与える影響を定める $\mathbf{B}(\mathbf{x})$ は定数の形で表現されるように定義されている．実際の振り子のダイナミクスでも $\mathbf{B} = [0,\ 1]^{\mathrm{T}}$ と定義されており，Linear model および Linear-NRBF model における $\mathbf{B}(\mathbf{x})$ に対応する重みを取り出すと，

$$\boldsymbol{B}_{\mathrm{linear}} = \begin{bmatrix} 0.0000 \\ 1.0392 \end{bmatrix}, \quad \boldsymbol{B}_{\mathrm{linear-NRBF}} = \begin{bmatrix} 0.0000 \\ 1.0106 \end{bmatrix}.$$

となり，どちらも真のダイナミクスに非常に近い値となっている．

以降では，真のダイナミクスから得られた指数価値及び最適制御則と学習モデルから得られた最適制御則を比較する．図 2.5 は指数価値関数 $z(\mathbf{x})$ を左列，最適制御則 $\pi^*(x)$ を右列にまとめて図示したものである．上から順に真のダイナミクスを用いたとき，Linear model，Linear-NRBF model を用いた学習モデルを使用したときに対応している．また，最適制御則を示すパネル上に描画されている実線は振り子が下がった状態 $\mathbf{x} = [\pi,\ 0]^{\mathrm{T}}$ を初期位置として最適制御則を用いたときの軌道の一例を示している．

いずれの指数価値関数も対角線上にピークを取るようになっており，またその付近での最適制御則は 0 となっている．Linear model によって近似した学習モデルから得られた指数価値関数と制御則は真のダイナミクスから得られたものと若干の違いが見られるものの大きな違いは見られない．しかしながら，Linear-model によって近似した学習モデルから得られた制御則の下での軌道と他の軌道を比べると，振り子を振る回数が多くなっている．

獲得した最適制御則の性能をより詳しく評価するため，振り子が下がった状態 $\mathbf{x} = [\pi,\ 0]^{\mathrm{T}}$ を初期位置として目標状態に到達もしくは開始から 20[s] (2000step) 経過するまでを 1 回の試行とした課題を 50 回繰り返した．各制御則のもとでの総コストの平均値を図 2.6 で示している．コストの計算の際には制御コストとして定義した controlled probability と uncontrolled probability の KL-ダイバージェンスが制御の二次形式となることから (式 (2.10))，即時コストを $\ell(\mathbf{x}, \mathbf{u}) = h\left(q(\mathbf{x}) + \frac{1}{2\sigma^2}\| \mathbf{u} \|^2\right)$ として計算を行った．

真のダイナミクスから得られた制御則が最も良い性能となり，近似誤差の非常に小さかった Linear-NRBF model によって関数近似した学習モデルから得られた制御則も真のダイナミクスから得られた制御則に匹敵するほどの性能を持っている．その一方で，近似誤差の大きかった Linear model によって関数近似した学習モデルから得られた制御則は振り上がりそのものは出来たものの総コストの平均値は他の 2 つの制御則と比較すると有意な差が現れた．

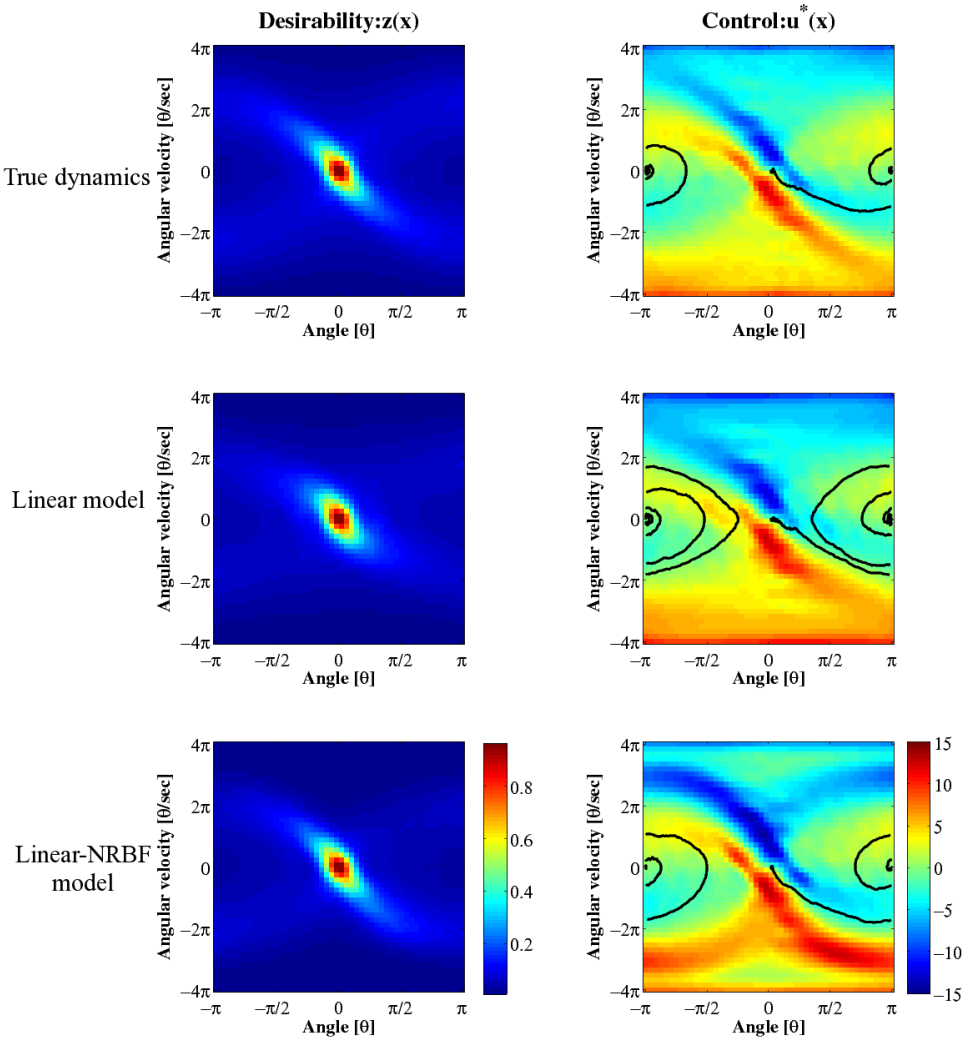


図 2.5 真のダイナミクスおよび学習モデルから得られた指数価値関数および方策

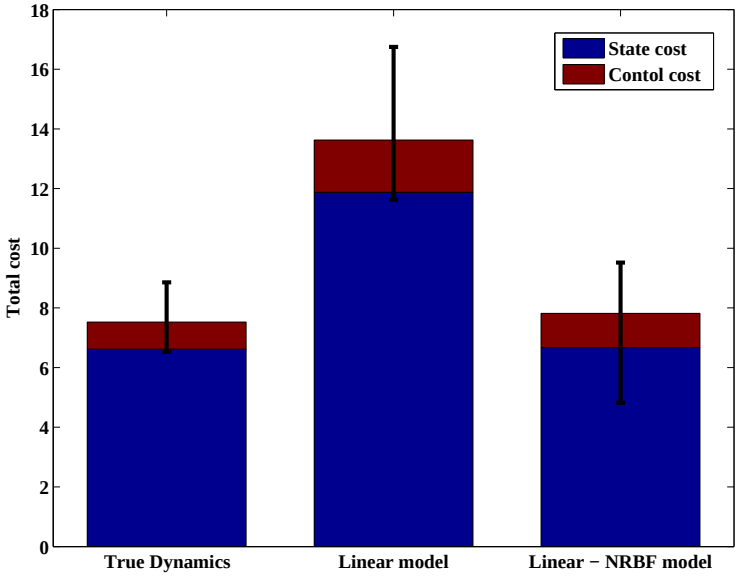


図 2.6 1 試行あたりの総コストの平均値

最後にノイズのスケールについての考察を述べる．本実験のようなシミュレーションではノイズのスケールを適当に決めることができるが，実機においては制御対象のノイズは任意に決定する事は出来ない．さらに，制御対象によってはノイズが殆ど 0 となることも考えられる．シミュレーションで用いる振り子もその一つである．しかしながら，ノイズが殆ど 0 となると式 (2.10) からも分かるように制御コストが発散してしまい，適切な制御則の導出が困難となってくる．この問題の解決策の一つとして，仮にシステムのノイズが無視できるほど小さいものであるならば，ノイズは制御対象固有のものとして捉えるのではなくスケールを任意に決定することが可能なランダムな制御入力 $\tilde{\mathbf{u}}$ によって発生していると仮定する方法がある．この仮定により制御対象への制御入力 $\mathbf{u}$ は，以下の式のように最適制御則もしくは何かの方策に従う制御入力 $\mathbf{u}^{\pi}$ とランダムな制御入力 $\tilde{\mathbf{u}}$ によって構成されるようになる．

$$\mathbf{u} = \mathbf{u}^{\pi} + \tilde{\mathbf{u}}$$

これにより制御対象の制約はあるものの任意のスケールのノイズを設定すること

が可能となり，ノイズのスケールが既知のものとして扱うこと出来るだけでなく，制御コストが発散する事を防ぐことが出来る．この知見を用いて以降の実験では制御対象固有のノイズのスケールは考慮せず，制御量が確率的なノイズを持つものとして以降の実験を行った．

3. ロボット実験: SpringDog による視角誘導課題

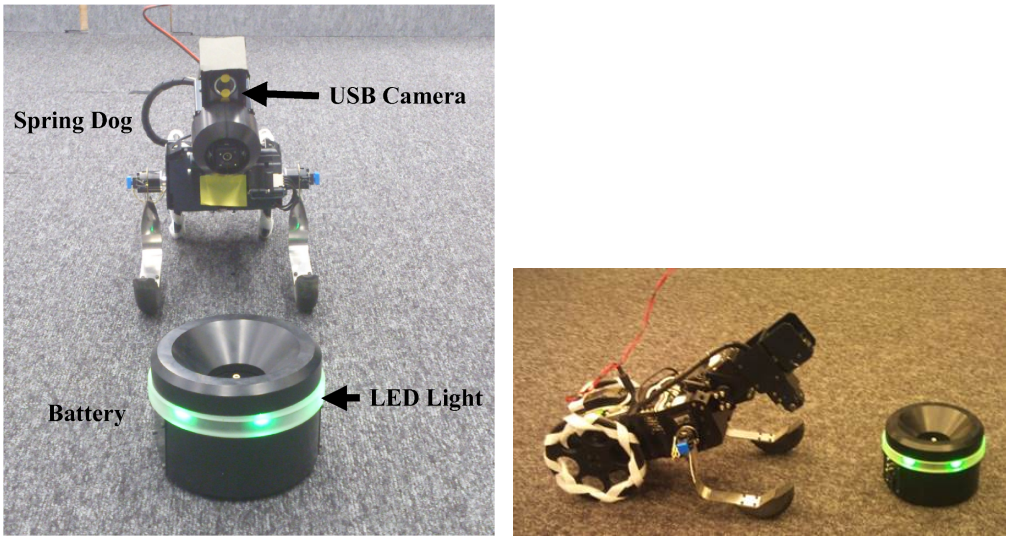
自身と環境と相対情報を観測するために，触覚センサーや力覚センサー，超音波等を用いた距離センサーなどがある．その中でも視覚センサーは対象物体と接触することなく，観測値の特徴量の動的な変化を捉えることができることから非常に有用である．このため視覚情報の特徴量を利用した視覚サーボ [38, 39] や視覚誘導技術の研究 [40, 41] が盛んに行われている．とりわけ視角誘導技術は，自動車の自動運転などの応用が期待され近年注目を浴びている．

本研究では，状態変数であるロボット自身の関節角度と視覚情報における目標物体の特徴量が制御量である車輪スピードの応じてどのように変化するかを表した視覚-運動ダイナミクスをサンプルから学習し，得られた学習モデルに対する最適な制御則を LMDP を用いて導出することで視覚誘導制御を実現した．

3.1 課題設定

図 2.7 は SpringDog と視角誘導課題の目標として利用したバッテリーパックを実験に用いたフィールド中で撮影したものである．SpringDog の自由度は左右両前足，後部の左右両車輪，頭部のカメラの上下方向と左右方向に対応する 2 つの首関節の計 6 自由度である．また，SpringDog には 3 次元加速度センサー，3 次元ジャイロセンサー，そして頭部に装着された USB カメラ等の複数のセンサーが搭載されている．目標として利用したバッテリーパックの上部周囲には 3 色の LED ライトが装着されている．

図 2.8 は実験で使用した Spring Dog の制御ダイアグラムを示しており，制御ダイアグラムには 3 つの制御器が実装されている．第 1 の制御器は目標であるバッテリーパックをトラッキングし続けるために頭部のカメラの上下方向と左右方向



(a) 正面図　　(b) 側面図

図 2.7 実験にて使用した車輪型ロボット Spring Dog とバッテリーパック

に定める関節に働く視覚サーボ制御器で，図 2.8 では青色のブロックで表されている．ただし，上下方向と左右方向に定める首関節にはそれぞれ可動範囲が設定され，その範囲内でのみ視覚サーボ制御器による制御が行われる．第 2 の制御器はバッテリーパックへと誘導するための後部の両車輪に対して働く視覚誘導制御器であり，図 2.8 では緑色のブロックで表されている．この制御器の入力はロボット自身の関節角度および目標物体の視覚特徴量であり，入力に応じて左右車輪の目標速度を制御することでロボットを目標物体まで導く．この制御器が提案手法によって得られた最適方策からなる制御器となり，左右両車輪の目標速度が式 (2.1) における制御入力 $\mathbf{u}$ となる．第 3 の制御器は探索行動を実現するための方策となっており，図 2.8 では紫色のブロックで表されている．カメラ画像の中にバッテリーパックが写っていない場合には，探索制御器によりロボットが旋回するように左右両車輪が制御されバッテリーパックの探索が実現される．視覚誘導制御器を除いた視覚サーボ制御器と探索方策は事前に実装した．

画像特徴量を抽出するために，撮像されたカメラ画像に対して 2 値化処理を適用することでバッテリーパックの装着された LED ライトに対応する緑色のピクセルと背景を分離した．図 2.9 は画像の処理結果と画像特徴量を示している．

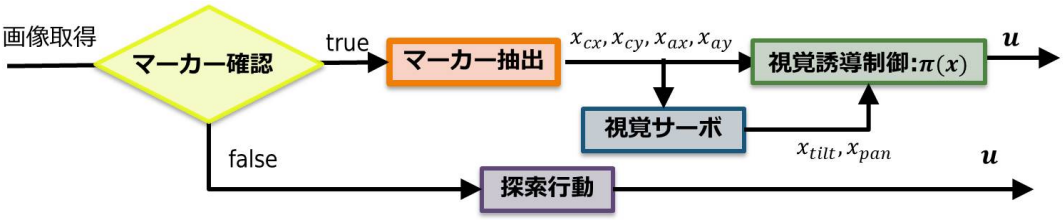


図 2.8 Spring Dog の制御ダイアグラム

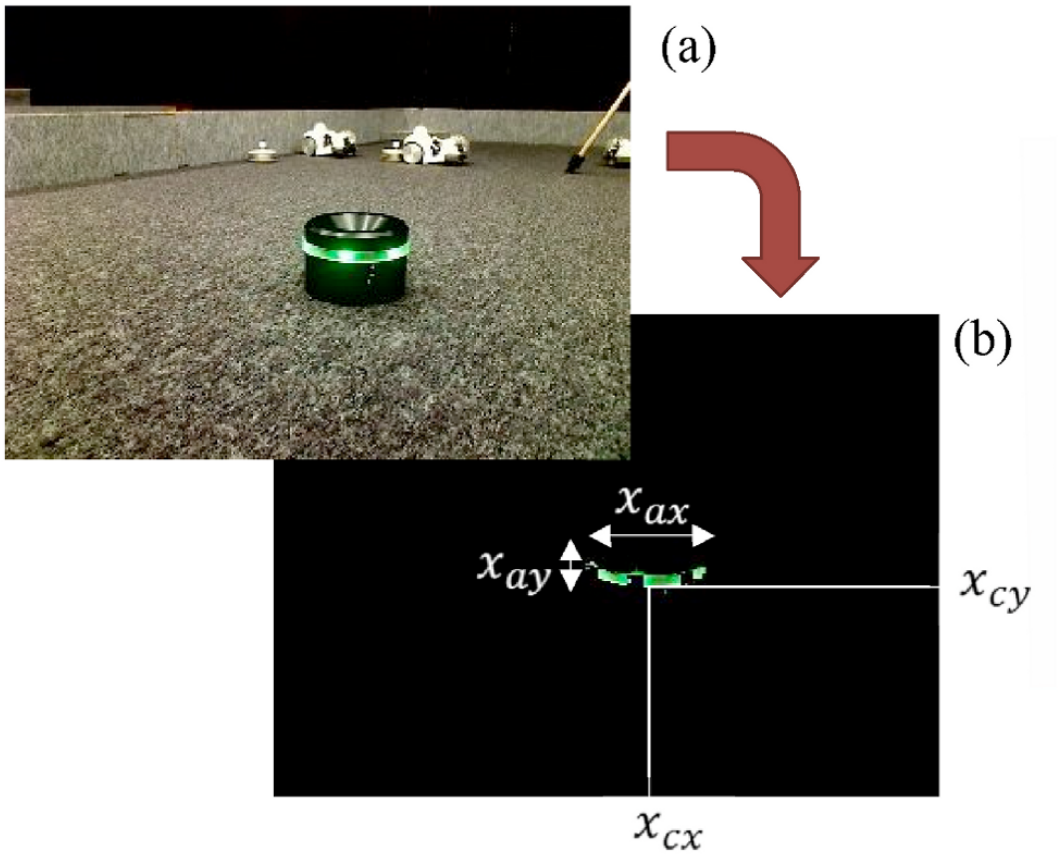


図 2.9 二値化画像と画像特徴量．(a): 撮像された原画像 (b): 二値化処理後の画像と画像特徴量

画像平面上のバッテリーパック (抽出されたピクセル) の重心 $(x_{cx},\ x_{cy})$，バッテリーパックの重心周辺の水平及び鉛直方向の絶対値平均 $(x_{ax},\ x_{ay})$，そして既に述べた視覚サーボ制御器によって定められる首関節の現在の目標角度 $(x_{\mathrm{tilt}}, x_{\mathrm{pan}})$，以上の 6 変数によって状態空間は構成される．これらを状態変数と制御入力である左右車輪の目標速度をまとめると

$$\mathbf{x} = [x_{cx}, x_{cy}, x_{ax}, x_{ay}, x_{\mathrm{tilt}}, x_{\mathrm{pan}}]^{\mathrm{T}}, \quad \mathbf{u} = [u_{\mathrm{left}}, u_{\mathrm{right}}]^{\mathrm{T}}.$$

となる．ここでそれぞれの変数は以下のようにスケールを変換し正規化されていることに注意されたい．

$$\begin{aligned} -1 \leq & x_{cx}, x_{cy}, x_{\mathrm{tilt}}, x_{\mathrm{pan}} \leq 1, \\ 0 \leq & x_{ax}, x_{ay} \leq 1, \\ -1 \leq & u_{\mathrm{left}}, u_{\mathrm{right}} \leq 1. \end{aligned}$$

目標状態は Spring Dog が十分にバッテリーパックに対して近づいた状態のときの姿勢と場所にとなって居るときに対応するように設定した．具体的には，図 2.7 のように顔を真正面にし少しうつむいた姿勢で前足の間にバッテリーパックがあるような状態となっている．今回の実験では目標状態に辿り着いたときに課題は終了することから，目標状態と吸収状態 $\mathbf{x}_{\mathcal{A}}$ が同一となるような課題である．そのことから今回の実験では目標状態でコスト関数が 0 となる以下で示されている 2 種類の状態依存のコスト関数 $q_1(\mathbf{x})$ と $q_2(\mathbf{x})$ を設定し，コスト関数の違いによる導出される制御則の違いについても検討した．

$$q_1(\mathbf{x}) = \alpha \left(\mathbf{x} - \mathbf{x}_g\right)^{\mathrm{T}} \boldsymbol{\Sigma}_{\mathrm{cost}}^{-1} \left(\mathbf{x} - \mathbf{x}_g\right) \tag{2.27}$$

$$q_2(\mathbf{x}) = \alpha \left(1 - \exp\left(-\left(\mathbf{x} - \mathbf{x}_g\right)^{\mathrm{T}} \boldsymbol{\Sigma}_{\mathrm{cost}}^{-1} \left(\mathbf{x} - \mathbf{x}_g\right)\right)\right), \tag{2.28}$$

ここでの α はコスト関数をスケーリングする定数である．

次にモデル学習の手順について説明する．まず訓練データの取得のため行動の滑らかさを考慮したランダムな制御則の下で SpringDog を操作し状態変数と制御入力の時系列データを取得した．実験では制御サイクルを $h = 300 \pm 60$ [ms] と

表 2.2 視覚-運動ダイナミクスの学習に用いた関数近似器の基底関数

	$\boldsymbol{\varphi}(\mathbf{x}, \mathbf{u})$
linear model	$[(\mathbf{x} - \mathbf{x}_g)^{\mathrm{T}} \ \mathbf{u}^{\mathrm{T}}]^{\mathrm{T}}$
bilinear model	$[(\mathbf{x} - \mathbf{x}_g)^{\mathrm{T}} \ u_{\mathrm{left}}(\mathbf{x} - \mathbf{x}_g)^{\mathrm{T}} \ u_{\mathrm{right}}(\mathbf{x} - \mathbf{x}_g)^{\mathrm{T}} \ \mathbf{u}^{\mathrm{T}}]^{\mathrm{T}}$

定めたが，他のプロセスの干渉によって制御サイクルの遅延が変則的に発生していた．このようなデータはモデル学習に悪影響を及ぼすことが考えられたため，対応するサンプルデータは全て除去した．加えて，目標であるバッテリーパックが撮像画像に存在しないときのデータや首関節角度が可動範囲の限界に達してしまっているデータもモデル学習の訓練データから取り除いた．このような手順を経てサンプルデータのセット，$\mathcal{D}_{\mathrm{origin}} = \left[\left[\mathbf{x}_1^{\mathrm{T}} \ \mathbf{u}_1^{\mathrm{T}} \right]^{\mathrm{T}}, \ \ldots, \ \left[\mathbf{x}_{N_{\mathcal{D}}}^{\mathrm{T}} \ \mathbf{u}_{N_{\mathcal{D}}}^{\mathrm{T}} \right]^{\mathrm{T}} \right]$，が構成された．このサンプルデータのセット $\mathcal{D}_{\mathrm{origin}}$ に対して正規化処理を行ったものを訓練データ $\mathcal{D}$ として，1.4 節で述べた推定手法に基づき SpringDog の視覚-運動ダイナミクスのモデルを学習した．ダイナミクスの学習に用いた関数近似機の基底関数 $\boldsymbol{\varphi}(\mathbf{x}, \ \mathbf{u})$ を表 2.2 は示しており，2 種類の関数近似器を用いて実験を行った．表 2.2 の linear model をダイナミクスの学習に用いて，式 (2.27) の二次形式の状態コストを使用した場合は，線形二次レギュレーター (LQR) と同一の問題設定となる．このことから，このときの LMDP の最適制御則は LQR と同一になると予想される．その一方で bilinear model を用いたモデルは非線形方程式であるために LQR を適用することはできない．このためダイナミクスの近似が良ければ LQR よりも性能の良い制御則の導出が期待される．

指数価値関数の導出は 1.3 節で示した手法に従って行った．このとき $\boldsymbol{F}, \boldsymbol{G}$ を導出するために節点と基底関数を設定する必要がある．節点 $\{\mathbf{x}_1, \ldots, \mathbf{x}_{N_s}\}$ は状態空間全体をカバーするように格子状に配置した．基底関数を陽に用いた制御則の計算はロボットのオンボードでリアルタイム行われため，限られたロボット上の計算リソースを考慮すると節点と同じ数の基底関数を用意するのは望ましくない．このため指数価値関数の基底関数 $f_i(\mathbf{x})$ の中心 $\boldsymbol{m}_i$ の初期値はダイナミクス

Procedure 1 関数近似器の初期値 $\mathbf{M}_{init}$ の決定方法

Input: The date set of state, $\mathcal{D}_{\mathbf{x}}$.

Output: The set of initial center positions, $\mathbf{M}_{init}$

$\mathbf{M}_{init} \leftarrow \emptyset$
while $\mathcal{D}_{\mathbf{x}} \neq \emptyset$ **do**
$\mathbf{x} = \mathrm{ChooseSample}(\mathcal{D}_{\mathbf{x}})$
$\mathcal{D}_{\mathbf{x}} \leftarrow \mathbf{X}_{\mathcal{D}} - \{\mathbf{x}\}$
if $\forall i\ f_i(\mathbf{x}; \boldsymbol{m}_i) < \tau$ or $\mathbf{M}_{init} = \emptyset$ **then**
$\mathbf{M}_{init} \leftarrow \mathbf{M}_{init} \bigcup \{\mathbf{x}\}$
end if
end while
return $\mathbf{M}_{init}$

の学習に使用した訓練データの状態変数だけを取り出した $\mathcal{D}_{\mathbf{x}} = [\mathbf{x}_1, \ \cdots \ , \mathbf{x}_{N_{\mathcal{D}}}]$ の中からいくつかを状態変数に対応するデータを選んだ．具体的にはアルゴリズム 1 で示しているような手順を用いて，$\mathcal{D}_{\mathbf{x}}$ のどのデータを関数近似器に $\boldsymbol{f}(\mathbf{x})$ に代入しても，すくなくとも一つの基底関数が閾値 τ 以上であるように基底関数の中心 $\boldsymbol{m}_i$ を決定した．共分散 $\boldsymbol{S}_i$ に関してはハンドチューニングで決定した．ただし，今回の実験では指数価値関数の基底関数のパラメータは中心のみを更新し，共分散は固定とした．最適制御則 $\mathbf{u}^*(\mathbf{x})$ には式 (2.14) より求めた．

実験で評価した条件はモデル学習に用いた関数近似器とコスト関数の定義から以下の四つにまとめることが出来る．

1. linear model + 二次形式コスト,
2. bilinear model + 二次形式コスト,
3. linear model + Gaussian based state cost (非二次形式),
4. bilinear model + Gaussian based state cost (非二次形式),

先に述べたように 1. の条件は LMDP だけでなく LQR においても適用可能な条件であることから，LQR に基づく制御器との比較も行った．

3.2 実験結果

3 節において記述したように，2 種類に関数近似器，linear model，bilinear model を使用してダイナミクスを学習した．ランダムの制御則のもと $N = 9509$ 個の状態と制御の系列データを取得したあと，そのうち $N_{\mathcal{D}_{\text{test}}} = 2500$ 個のデータをテストデータとして無作為に抽出し，残りを訓練データ $\mathcal{D}$ として用いた．

振り子の振り上げ課題のシミュレーション課題と同様に，式 (2.23) を用いて関数近似器の重み行列を導出し，テストデータを用いて平均二乗誤差を計算し，学習モデルの近似誤差を評価した．図 2.10 はその結果を示している．図から分かるように linearmodel と bilinear model の近似誤差はどの状態変数においても有意な差は存在しなかった．この結果からどちらの学習モデルもほぼ同定程度の近似誤差を有している．それぞれの状態変数ごとの推定精度を比較すると，どちらの学

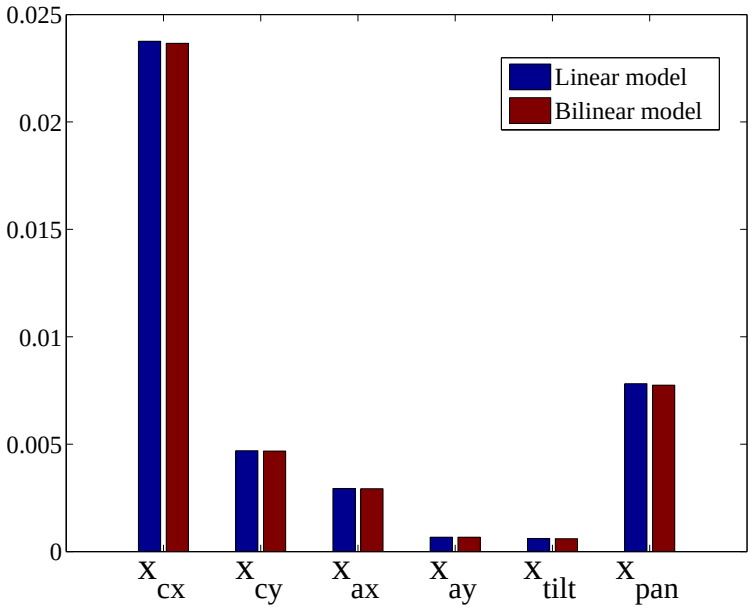


図 2.10 ロボット実験:各状態変数の二乗誤差平均

習モデルも x_{cx} と x_{pan} の平均二乗誤差が大きい結果となった．その理由として，サンプル取得時のロボットの動きが並進する動きは比較的ゆったりとしていた一方で，旋回する動きは比較的素早かったことが考えられる．推定精度の低かった2つの要素はどちらもロボットが旋回する動きに依存して変化する状態変数であり，旋回する動きが素早かったことでサンプルされたデータが大きくバラつき推定精度の低下させた推察される．

次に学習モデルとコスト関数の違いが，得られた指数価値関数及び方策に対してどのような影響を与えたかを評価した．図 2.11 は二次形式のコスト関数 (式 (2.27)) を利用したときの指数価値関数と左右車輪の制御量を示している．図 2.11 は，ロボットの状態変数の中から x_{tilt} と x_{pan} のそれぞれ x 軸と y 軸にとり，その他の状態変数は目標状態の値をとるとして図示されている．上段の行はダイナミクスの関数近似器として linear model を用いて学習したモデルに対して LQR を適用して得られた指数価値関数と左右の車輪の制御量を示している．中段と下段の行は指数価値関数および制御量の導出に LMDP を用いているが，中段は linear model を下段は bilinear model をダイナミクスの関数近似器に用いたときの結果となっている．指数価値関数の関数近似に用いた基底関数の中心 $\boldsymbol{m}_i$ が中段と下段の行の図に描画されている青いドットでプロットされている．LMDP と LQR と比較すると指数価値関数のピークの範囲が広くはなっているものの，得られた制御則はほとんど似たような傾向であることが分かる．

さらに得られた制御則を用いて目標物体であるバッテリーパックへと接近するテスト課題を実行し，それぞれの制御則の性能評価を行った．テストでは，目標であるバッテリーパックから 1.5 メートルは離れた地点を初期位置とした．バッテリーパックの位置は図 2.12 に示されているように，ロボットの正面と正面から左右 15 度ずつずれた地点の 3 つからランダムに選んでテストを行った．テストの時間は制御サイクルで 50 回，およそ 15 秒を目安に 1 回の試行とした．図 2.13 は二次形式コストを用いたときに得られた制御則において，30 回のテストの累積コストの平均を示した図である．ここで各時刻における即時コストは $\ell(\mathbf{x}\,,\mathbf{u}\,) = h\left(q_1(\mathbf{x}\,) + \frac{1}{2\sigma^2}\|\mathbf{u}\,\|^2\right)$

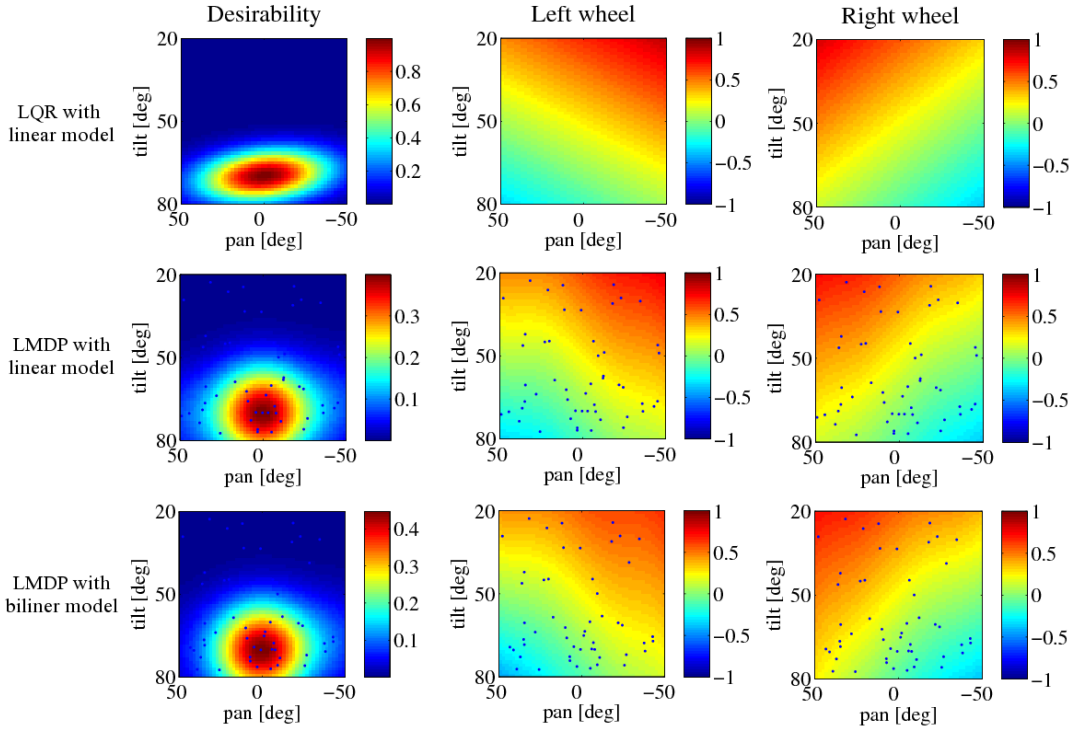


図 2.11 学習モデルから得られた指数価値関数と左右車輪の制御則

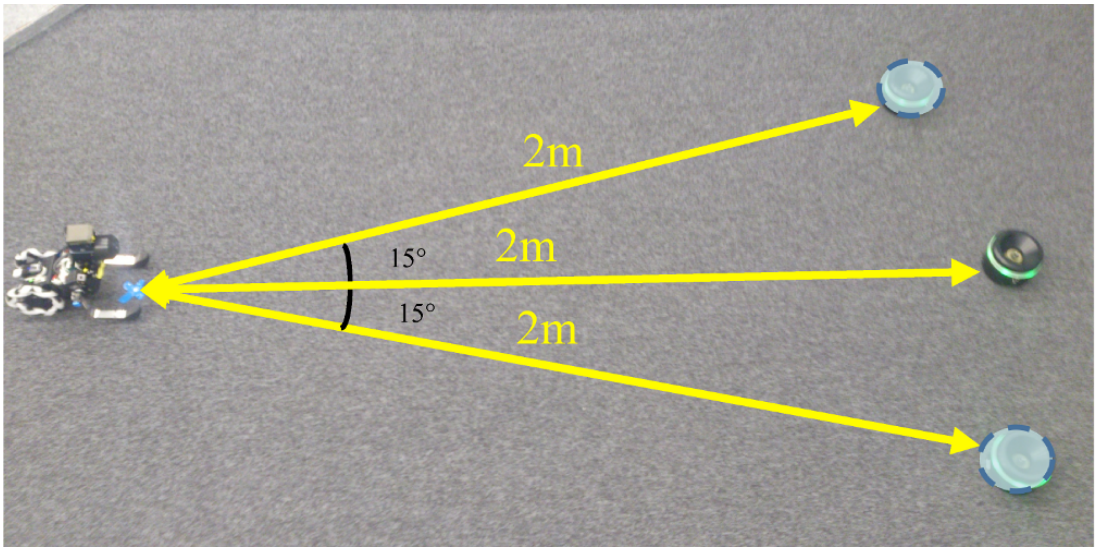


図 2.12 ロボットの制御性能の評価に用いたフィールドと目標の初期位置

とし，目標であるバッテリーパックが視界から外れてしまったときは評価からは除外した． 図 2.13 で示されている二次形式コストを用いた 3 つの制御則の性能を比較すると，モデルの関数近似器に linear model を用いた LQR と LMDP から得られた 2 つの制御則はほぼ同程度の制御性能を持っていることが分かる．この 2 つの制御則は LQR と LMDP の解き方の違いはあるものの両者が同じ問題を解いていることから当然の結果であると考えられる．また，それぞれの制御則を用いたときの課題中の x_{tilt} と状態コストの時間変化をした図 2.14 で両者を比較すると両者に大きな違いは存在しない．その一方で，モデルの関数近似器に bilinear model を用いた場合には，他の二つの制御器と比較するとその性能は若干悪くなり，時間変化も異なったものとなっている．この結果は bilinear model を用いた学習モデルでは過学習が発生しているために制御性能の劣化を引き起されている考えられる．

今回の実験では 2 種類の異なる定義のコスト関数を用いて制御則を獲得しているために，獲得した全ての制御則の性能を比較に累積コストを用いることは好ましくない．このことから，全ての制御則の評価をするための指標として目標状態

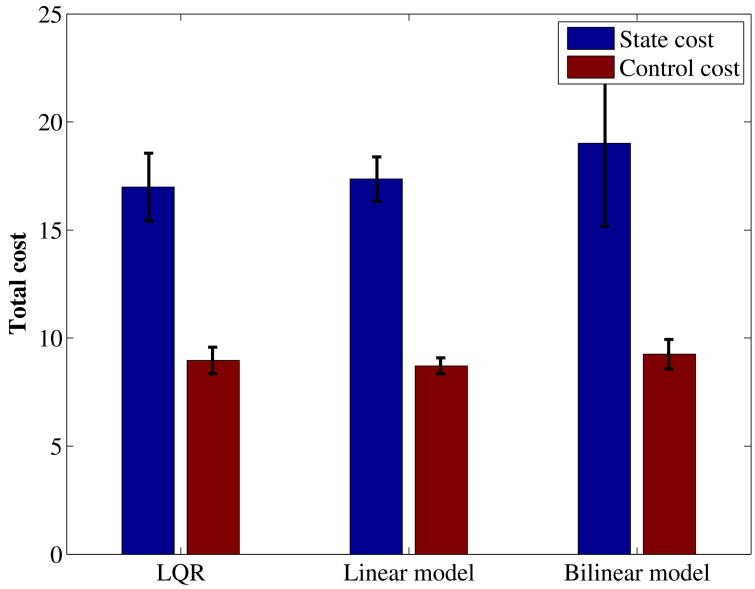


図 2.13 二次形式コストから得られる制御則のテスト課題中の平均累積コスト

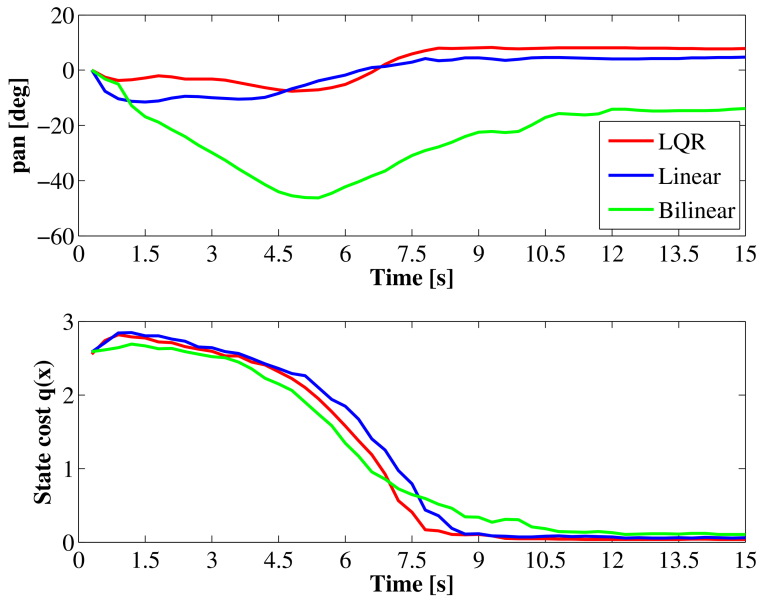


図 2.14 二次形式コストから得られる制御則を用いた課題中の x_{tilt} と状態コストの時間変化

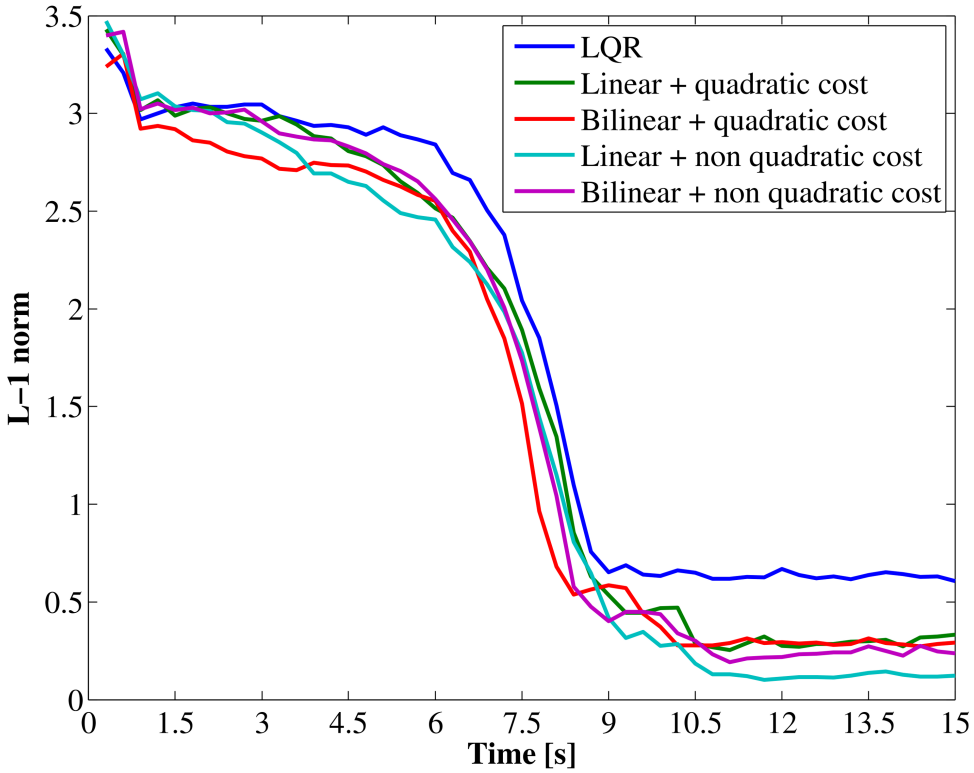


図 2.15 獲得した全ての制御則による L-1 ノルムの時間変化

からの L1 ノルム[3]を計算した．図 2.15 はその時間変化を示している．すべての制御器は 10 秒以内に Spring Dog を目標状態に十分に近い場所まで導くことに成功してる．その中でも特に非二次形式のコストを用いて得られた制御則は他の制御則と比較すると目標状態のより近くまで導いていることが分かる．これは，非二次形式のコストのコストの低い領域が二次形式のコストに比べて狭かったことが原因であると考えられる．

[3]L-1 ノルム $\mathbf{x} = (x_1, \ldots, x_n)^{\mathrm{T}}$ のそれぞれの要素の絶対値の和 $\|\mathbf{x}\|_1 = \sum_i |x_i|$ から計算できる．

4. 本章のまとめ

振り子のシミュレーションにおける実験では，学習モデルの近似誤差により制御性能の劣化が確認されたが，そのような制御則のもとでも課題の目標である振り上げが達成されることが確認できた．

また実機における実験で用いた 2 つの関数近似器はどちらも， x_{cx} と x_{pan} の成分で推定精度が悪くなる結果を示した．この結果はロボットは次状態のバッテリーパックの左右方向の位置を正確に推定できていないことを示している．それでもなお，学習した制御則によってロボットはバッテリーパックに近づくことができた．この結果から，ダイナミクスの関数近似がそれほど正確ではなくとも，学習したモデルを用いた LMDP による学習制御はある程度の制御性能があることを示すことができた．

二次形式のコスト関数と linear model を用いてモデルの学習を行った場合では，LMDP だけでなく LQR からも最適な指数価値関数と対応する最適制御則を解析的な手法を用いて求めることが可能であった．本研究では LMDP だけでなく LQR 用いてそれぞれの数価値関数と対応する最適制御則を求めた．LMDP によって得られた指数価値関数及び対応する制御則は，LQR によって得られたそれらと全く同じものではなかった．しかしながら，得られた制御則の性能に関してはほぼ同程度であることが実験より確認された．

LMDP はこれまで Bellman 方程式を線形化したことにより最適な方策を従来のモデルベース強化学習手法よりも効率的に求めるられることが示されていた．しかしながら，LMDP には制御対象のダイナミクスを事前に知っている必要があった．本研究では関数近似器によるモデル学習を組み合わせることにより，LMDP が実機において適用可能であることを示した．節 1 でも述べた通り，本研究と同様に LMDP から導出される線形化 Bellman 方程式を利用する手法として経路積分法を用いた手法が存在し，既にいくつかの実機の適用した研究も存在する [14, 15, 16]．しかしながら，本研究における提案手法のように指数価値関数に基づく手法で実機の学習制御に LMDP を適用した例は研究は本研究が初である．この理由として指数価値関数を明示的に利用するためには，定義された状態空間上

で指数価値関数を解く必要があることが挙げられる．即ち，このため指数価値関数を明示的に利用する手法は，状態空間の次元の増加に伴って計算コストが指数的に増加する “次元の呪い”[42] の影響を受けてしまうからである．今回の実機における実験でも指数価値関数の近似に用いる基底関数の取り方に注意したにも関わらず関数近似の学習に非常に計算時間を要した．その一方で，経路積分法を用いた手法はサンプルとして得られた経路付近のみで計算を行うことで “次元の呪い” を回避している．

本研究で提案した手法はモデルベース強化学習として見なすことができるが，モデルベース強化学習の手法として，本研究で提案した手法と同様にモデル学習を含めたモデルベース強化学習の手法が既にいくつかの提案されている [43, 44]．しかし，それらの手法は非線形な Bellman 方程式から最適価値関数及び制御則を求めている．その一方で，本研究における提案手法は実機のような確率的でかつ非線形なダイナミクスを持つ制御対象に対しても適用可能であることを本研究の実験結果が示していることから，これらの手法をさらに改善させることが可能であると期待される．

最後に，本研究の提案手法はコスト関数は既知のものであるとしてコスト関数の推定は行っていない．しかしながら，コスト関数の学習はモデル学習と同様の教師有り学習であることから，コスト関数の学習はモデルの学習を同時に行うことは可能である．また，強化学習を実機に適用した多くの研究で報酬の計算は設計者によって事前定義されていることから，コスト関数が既知であるという仮定はそれほど大きな問題ではない．

第3章　Linearly Solvable Markov Game の近似誤差に対するロバスト性の検証

Linearly Solvable Markov Game (LMG) は Markov Game の最適解を導出するために用いられる Bellman-Isaacs 方程式を線形化しすることができる問題クラスであり，その解からロバストな制御則が導出されると期待される．しかしながら，LMG は Risk sensitive 制御で用いられる Risk sensitive Markov 決定過程の Bellman 方程式を線形化することを本来の目的として提案されたために，モデルの近似誤差に対するロバスト性の議論が十分に行われていなかった．このことから，本章では LMG およびその特別な場合として捉えられる LMDP のモデルの近似誤差に対するロバスト性について調査をするため，離散状態行動空間の問題である格子状迷路課題と連続状態行動空間の問題である振り子およびアクロボットの振り上げ課題の実験を行った．

1. LMG による ロバストな制御則の導出

離散時間系における Markov Game の最適方策と価値関数を導出するために用いられる Bellman-Issac 方程式が LMG によって線形化されることについて論文 [30] に基づき記述する 1.1 節と LMG と一般的なロバスト制御で用いられる H^{∞} 制御との関連について説明する 1.2 節から本節は構成される．

LMG は本来の目的である Risk sensitive 制御との関連も存在するが，本論文における本筋では無いことから特に議論しない．その詳細については論文 [30] を参照されたい．

1.1 LMG による Bellman-Issac 方程式の線形化

Markov Games には制御則 (方策) による制御入力 $\mathbf{u}^c$ と外乱もしくは Adversary による妨害入力 $\mathbf{u}^a$ が制御対象の入力として与えられる．これらの入力と現在の状態 $\mathbf{x}$ によって次状態 $\mathbf{x}'$ が導かれる状態遷移確率 $\mathbf{x}' \sim p(\cdot \mid \mathbf{x}, \mathbf{u}^c, \mathbf{u}^a)$ と即時コスト $\ell(\mathbf{x}, \mathbf{u}^c, \mathbf{u}^a)$ 与えられているものとする．

このときの目的関数に関するいくつかの定義が存在するが，今回は目標状態の $\mathbf{x}_g \in \mathcal{X}$ に到達するまでの累積コストの期待値を制御則の目的関数として定義する．

$$\mathcal{J}^\pi \quad = \quad \mathbb{E}_{\mathbf{x}' \sim p(\cdot \mid \mathbf{x}_t, \mathbf{u}_t^c, \mathbf{u}_t^a)} \left[\sum^{\mathcal{T}_g} \ell(\mathbf{x}_t, \mathbf{u}_t^c, \mathbf{u}_t^a) \right]$$

ここでの $\mathcal{T}_g$ は目標状態に初めて到達した時刻であり，目標状態は吸収状態となっている．

この目的関数における最適方策を導く Bellman-Issac 方程式は

$$v(\mathbf{x}) \quad = \quad \min_{\mathbf{u}^c} \max_{\mathbf{u}^a} \{ \ell(\mathbf{x}, \mathbf{u}^c, \mathbf{u}^a) + \mathbb{E}_{\mathbf{x}' \sim p(\cdot \mid \mathbf{x}_t, \mathbf{u}_t^c, \mathbf{u}_t^a)} [v(\mathbf{x}')] \} \tag{3.1}$$

となる． Bellman-Issac 方程式の解は最大化演算子と最小化演算子の順番により，上限と下限が存在する．ここでは最悪のケースを想定した方策を得ることが目的であることから上記の順番の式を利用する．

この Bellman-Issac 方程式を線形化のために，制御量 $\mathbf{u}$ が制御対象の状態遷移確率に及ぼす影響を表す関数として $g^{\mathbf{u}}(\mathbf{x}' \mid \mathbf{x}) \in \mathbb{R}^+$ を定義する．これによって制御量 $\mathbf{u}$ が与えられたときの制御対象の次状態の分布である Controlled probability $p^{\mathbf{u}}(\mathbf{x}' \mid \mathbf{x})$ は $g^{\mathbf{u}}(\mathbf{x}' \mid \mathbf{x})$ を用いて以下のように表されるものとする．

$$p^{\mathbf{u}}(\mathbf{x}' \mid \mathbf{x}) \quad = \quad g^{\mathbf{u}}(\mathbf{x}' \mid \mathbf{x}) \otimes p^0(\mathbf{x}' \mid \mathbf{x})$$

ここでの $p^0(\mathbf{x}' \mid \mathbf{x})$ は Uncontrolled probability と呼ばれる $\mathbf{u} = \mathbf{0}$ の制御量が与えられたときの制御対象の次状態の分布であり，$\otimes$ は以下のような計算を表している．

$$f(\mathbf{s}') \otimes p(\mathbf{s}' \mid \mathbf{s}) \quad = \quad \frac{f(\mathbf{s}') p(\mathbf{s}' \mid \mathbf{s})}{\int f(\mathbf{s}') p(\mathbf{s}' \mid \mathbf{s}) d\mathbf{s}'}$$

制御量 $\mathbf{u}^c$ と妨害入力 $\mathbf{u}^a$ のそれぞれの制御対象の状態遷移確率に対する影響を $g^{\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x}, \mathbf{u}^a)$，$g^{\mathbf{u}^a}(\mathbf{x}' \mid \mathbf{x})$ で表すと，それぞれが与えられたときの制御対象の状態遷移確率は以下のようになる．

$$p^{\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x}) = g^{\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x}) \otimes p^0(\mathbf{x}' \mid \mathbf{x}) \tag{3.2}$$

$$p^{\{\mathbf{u}^a, \mathbf{u}^c\}}(\mathbf{x}' \mid \mathbf{x}) = g^{\mathbf{u}^a}(\mathbf{x}' \mid \mathbf{x}, \mathbf{u}^c) \otimes g^{\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x}) \otimes p^0(\mathbf{x}' \mid \mathbf{x}) \tag{3.3}$$

ここでは明確に妨害入力 $\mathbf{u}^a$ が制御入力 $\mathbf{u}^c$ の後に制御対象に与えるものと考えている．これによって妨害入力 $\mathbf{u}^a$ がシステムの状態遷移に与える影響 $g^{\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x}, \mathbf{u}^a)$ は制御入力 $\mathbf{u}^c$ にも依存するようになっている．

先に定義されている Uncontrolled probability と 2 種類の Controlled probability (式 (3.2)，式 (3.3)) を用いて即時コスト $\ell(\mathbf{x}, \mathbf{u}^c, \mathbf{u}^a)$ を以下のように定義する．

$$\begin{aligned}\ell(\mathbf{x}, \mathbf{u}^c, \mathbf{u}^a) = \ell(\mathbf{x}) &+ \frac{1}{\alpha}\mathcal{D}_\alpha\left(p^0(\mathbf{x}' \mid \mathbf{x}) \| p^{\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x})\right) \\ &- \frac{1}{\alpha}\mathrm{KL}\left(p^{\{\mathbf{u}^a, \mathbf{u}^c\}}(\mathbf{x}' \mid \mathbf{x}) \| p^{\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x})\right)\end{aligned} \tag{3.4}$$

ここで使用されている $\mathcal{D}_\alpha(p \mid q)$ は Renéyi-ダイバージェンスと呼ばれる二つの分布の距離の尺度であり，同じ二つの分布の距離の尺度である KL-ダイバージェンスのより一般的な定義となっている．定義などの詳細については付録 1 を参考にされたい．ここで定義された式 (3.4) の即時コストを式 (3.1) の Bellman-Issac 方程式に代入すると，Bellman-Issac 方程式は以下のように線形化される．

$$z^\alpha(\mathbf{x}) = \exp((\alpha - 1)\ell(\mathbf{x}))\mathbb{E}_{p^0(\mathbf{x}' \mid \mathbf{x})}[z(\mathbf{x})^\alpha] \tag{3.5}$$

この線形化方程式の解は $z(\mathbf{x})^\alpha = \exp((\alpha - 1)v(\mathbf{x}))$ となっている．この線形化の詳細については付録 3 を参照されたい．また $\alpha = 1$ のときは例外的に以下のように価値関数 $v(\mathbf{x})$ に対して線形になる．

$$v(\mathbf{x}) = \ell(\mathbf{x}) + \mathbb{E}_{p^0(\mathbf{x}' \mid \mathbf{x})}[v(\mathbf{x}')]$$

これらの結果から Bellman-Issac 方程式が離散状態空間では固有関数問題，連続状態空間では固有関数問題に帰着される．固有関数問題の解の導出は一般的に困難であるとされている．しかしながら，式 (3.5) の結果は節 2.1.1 の Bellman 方

程式の線形化の結果の右辺の係数が変化しただけであることから，節 2.1.2 で紹介している非正規化ガウス関数による関数近似の手法を拡張することでその近似解を得ることが可能である．

ここで得られた価値関数 (もしくは指数価値関数) $v(\mathbf{x}) \left(= \frac{1}{\alpha-1}\log\left(z^{\alpha}(\mathbf{x})\right)\right)$ に基づく最適な Controlled probability は以下のようになる．

$$p^{*\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x}) = \exp(-v(\mathbf{x}'))p^0(\mathbf{x}' \mid \mathbf{x}) \tag{3.6}$$

$$p^{*\{\mathbf{u}^a,\mathbf{u}^c\}}(\mathbf{x}' \mid \mathbf{x}) = \exp((\alpha-1)v(\mathbf{x}')) \otimes p^0(\mathbf{x}' \mid \mathbf{x}) \tag{3.7}$$

この結果より即時コストの定義 (式 3.4) で用いられている Réneyi-ダイバージェンスのパラメータである α は，導出される解およびそれに対応する最適な Controlled probability を特徴づけていることが分かる．即ち，LMG のパラメータ α は妨害入力の強さに対応している．例えば，$\alpha = 0$ のとき得られる最適な Controlled probability は $p^{*\{\mathbf{u}^a,\mathbf{u}^c\}}(\mathbf{x}' \mid \mathbf{x}) = p^{\mathbf{u}^{*c}}(\mathbf{x}' \mid \mathbf{x})$ となる．この結果から $\alpha = 0$ の LMG は妨害入力が全くない場合に相当し，先の章で用いた LMDP と等価な問題クラスとなる．その一方で，$\alpha = 1$ のとき得られた最適な Controlled probability は $p^{*\{\mathbf{u}^a,\mathbf{u}^c\}}(\mathbf{x}' \mid \mathbf{x}) = \Pi^0(\mathbf{x}' \mid \mathbf{x})$ となる．この結果から $\alpha = 1$ の LMG は妨害入力が制御と同じだけの強さで働く場合に相当する．

ここで導出される Controlled probability $p^{*\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x})$ はあくまでも最適な制御入力 (もしくは行動) が与えられたときの次状態の分布であるため，現状態での最適な制御量や行動は得られた Controlled probability $p^{*\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x})$ や導出された価値関数 (もしくは指数価値関数) に基づいて別途求める必要がある．

1.2 LMG と H^{∞} 制御の関係

LMG と H^{∞} 制御の関連について説明するために，図 3.1 のように連続な状態 $\mathbf{x}$，制御入力 $\mathbf{u}_c$ そして妨害入力 $\mathbf{u}_a$ に対して線形なダイナミクスをもつ制御対象を観測値 $\mathbf{y}$ によって定められる制御器 π によって制御するフィードバック制御機構を考える．ただし，ここでは簡単のため状態 $\mathbf{x}$ は観測値 $\mathbf{y}$ と同一であるとする．

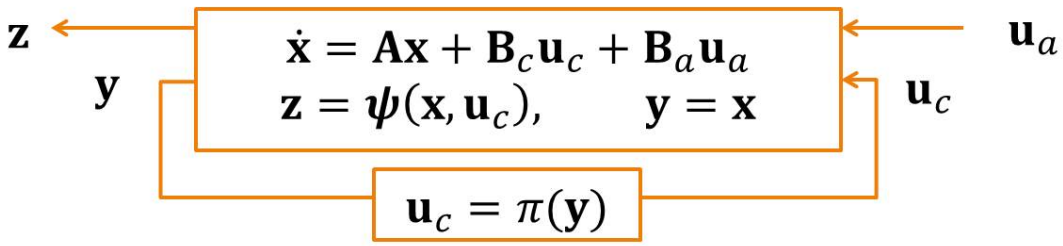


図 3.1 妨害入力が含まれるようなフィードバック制御

状態と制御入力に関するコスト出力 $z = \psi(\mathbf{x}, \mathbf{u}_c)$ が定められているとしたとき，H^∞ 制御では以下のように妨害入力 $\mathbf{u}_a$ とコスト出力 z の H^∞-ノルムが以下の制約条件を満たすような制御器の設計を目指す [24].

$$\sup_{\mathbf{u}_a} \frac{\|z\|_2^2}{\|\mathbf{u}_a\|_2^2} \leq \gamma^2, \quad \gamma > 0 \tag{3.8}$$

H^∞ 制御では一般的に γ が小さいほど妨害入力に対してロバストな制御則となるが，このような制約を満たす γ には下限が存在する．ここでの $\|z\|_2^2$ と $\|\mathbf{u}_a\|_2^2$ はそれぞれ,

$$\|z\|_2^2 = \int_0^\infty z(t)^2 dt \tag{3.9}$$

$$\|\mathbf{u}_a\|_2^2 = \int_0^\infty \mathbf{u}_a^{\mathrm{T}} \mathbf{u}_a dt \tag{3.10}$$

と定義されている．即ち，先ほどの制約条件 (式 (3.8)) は,

$$\sup_{\mathbf{u}_a} \int_0^\infty (z(t)^2 - \gamma^2 \mathbf{u}_a^{\mathrm{T}} \mathbf{u}_a) dt \leq 0 \tag{3.11}$$

と書きなおすことが出来る．このような制約下の最適制御 $\mathbf{u}_c^*$ を求めることになることから，最適価値関数は

$$\begin{aligned} v(\mathbf{x}(t=0)) &= \max_{\mathbf{u}_c} \min_{\mathbf{u}_a} \int_0^\infty (z(t)^2 - \gamma^2 \mathbf{u}_a^{\mathrm{T}} \mathbf{u}_a) dt \\ &= \max_{\mathbf{u}_c} \min_{\mathbf{u}_a} \int_0^\infty (\mathbf{x}^{\mathrm{T}} \mathbf{Q} \mathbf{x} + \mathbf{u}_c^{\mathrm{T}} \mathbf{R} \mathbf{u}_c - \gamma^2 \mathbf{u}_a^{\mathrm{T}} \mathbf{u}_a) dt \end{aligned} \tag{3.12}$$

となる．ここで二段目の右辺はコスト出力を状態と制御に 2 次形式となるような $z(\mathbf{x}, \mathbf{u}_c)^2 = \mathbf{x}^{\mathrm{T}} \mathbf{Q} \mathbf{x} + \mathbf{u}_c^{\mathrm{T}} \mathbf{R} \mathbf{u}_c$ に置きかえている．この結果，即時コスト

$\ell(\mathbf{x}, \mathbf{u}_c, \mathbf{u}_a)$ が以下のようになる．

$$\ell(\mathbf{x}, \mathbf{u}_c, \mathbf{u}_a) \quad = \quad \mathbf{x}^{\mathrm{T}}\mathbf{Q}\mathbf{x} + \mathbf{u}_c^{\mathrm{T}}\mathbf{R}\mathbf{u}_c - \gamma^2\mathbf{u}_a^{\mathrm{T}}\mathbf{u}_a \tag{3.13}$$

さらに制御対象は状態と制御に関して線形なダイナミクスであると仮定していることから，この問題は線形ロバスト制御の問題として解くことが可能であり，可解条件から仮定した γ のもとで式 (3.8) を満たす制御則が存在するかを確認することが可能である [45].

一方，LMG では制御と妨害入力およびガウシアンノイズが同一空間で作用する以下のようなダイナミクスもつ制御対象を考える．

$$\Delta\mathbf{x} = \mathbf{A}\mathbf{x} + \mathbf{B}(\mathbf{u}_c + \mathbf{u}_a + \sigma\xi)\Delta t, \quad \xi \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

このときの uncontrolled probability 及び controlled probability は,

$$\begin{aligned} p^0(\mathbf{x}' \mid \mathbf{x}) &= \mathcal{N}(\mathbf{y}, \sigma^2\mathbf{B}\mathbf{B}^{\mathrm{T}}) \\ p^{*\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x}) &= \mathcal{N}(\mathbf{y} + \mathbf{B}\mathbf{u}_c, \sigma^2\mathbf{B}\mathbf{B}^{\mathrm{T}}) \\ p^{*\{\mathbf{u}^a, \mathbf{u}^c\}}(\mathbf{x}' \mid \mathbf{x}) &= \mathcal{N}(\mathbf{y} + \mathbf{B}(\mathbf{u}_c + \mathbf{u}_a), \sigma^2\mathbf{B}\mathbf{B}^{\mathrm{T}}) \end{aligned}$$

となる．このときの即時コスト $\ell(\mathbf{x}, \mathbf{u}_c, \mathbf{u}_a)$ は LMG における即時コストの定義より (式 (3.4)),

$$\ell(\mathbf{x}, \mathbf{u}_c, \mathbf{u}_a) = q(\mathbf{x}) + \frac{1}{2\sigma^2}\mathbf{u}_c^{\mathrm{T}}\mathbf{u}_c - \frac{1}{2\alpha\sigma^2}\mathbf{u}_a^{\mathrm{T}}\mathbf{u}_a \tag{3.14}$$

となる．この結果を H^∞ 制御の即時コスト (式 (3.13)) と比較すると，即時コストの妨害入力 $\mathbf{u}_a$ に関する項はどちらも二次形式であり，その係数から以下のような関係が成立している．

$$\gamma^2 = \frac{1}{2\alpha\sigma^2} \tag{3.15}$$

この関係はあくまでも限られた制御対象のダイナミクスを仮定した場合の数学的な関係に過ぎない．さらに，両者には様々な違いが存在する．例えば，H^∞ 制御の制御対象のダイナミクスは決定的である一方で，LMG の制御対象のダイナミクスは確率的である．さらに，H^∞ 制御はレギュレーターを想定している一方

で，LMG は状態空間全てにおける制御則を想定している．しかしながら，この関係式から LMG のパラメータ α も H^∞ 制御における γ と同様に導出される制御則のロバスト性を定める役割があると予測され，さらに α には最適値や制御則を導出可能な領域が存在すると考えられる．

2. 離散状態行動空間における LMG： Grid-world task with risky state 課題

連続状態行動空間における LMG のモデルの近似誤差に対するロバスト性について検証する前に，大きなコストが与えられるような状態を含んだ格子状迷路課題のシミュレーション実験を通して，離散状態行動空間における LMG のモデルの近似誤差に対するロバスト性について検証する．実験では方策の導出に用いたモデルと異なるテスト環境を用意しすることで，モデルの近似誤差を意図的に生じさせ，そのときの獲得した方策の性能を評価した．

2.1 課題設定

本課題においてエージェントの目標はスタートからゴールへと最小コストで導く方策 (経路) を獲得することである．格子の中には大きなコストが与えられる沼地の領域が存在するため，最適方策 (経路) は状態遷移確率だけでなく沼地とそれ以外の格子のコストの比にも依存する．

図 3.2.a は Grid-world task with risky state 課題で実際に使用した状態の配置図である．図 3.2.a にて緑で塗りつぶされた格子はゴールを示しており，エージェントがゴールに到達するとタスクは終了する．このときエージェントは $\ell(x_g) = 0$ のコストを受け取る．図 3.2.a にて濃い青で塗りつぶされた格子は沼地を示しエージェントがこのグリッドに訪問すると $\ell(x_g) = 200$ のコストが与えられる．課題はゴールに到達した場合にのみ終了し，エージェントが沼地に踏み入れたとしても課題は続行する．

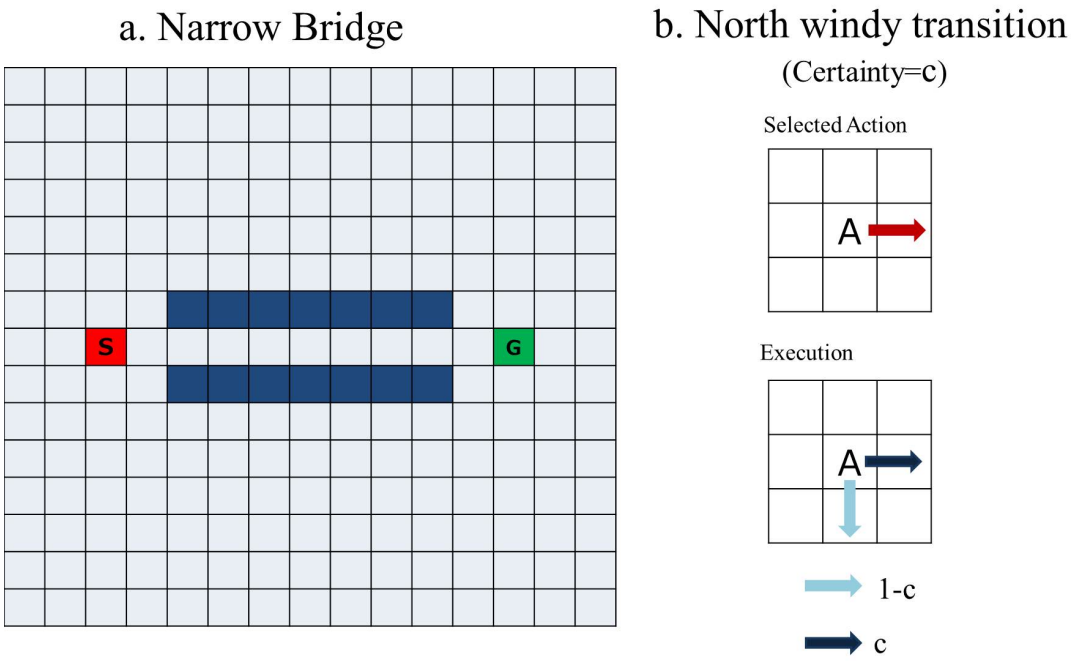


図 3.2 a. 格子状迷路課題の状態の配置 b. エージェントの状態遷移

エージェントは上下左右の格子へと移動する 4 つの行動を選択できるが，図 3.2.b で示されるように図中の上の方から下へ吹く北風の影響を受けて状態遷移が特徴付けられる．北風の強さは Certainty $c = [0.5,\ 1.0]$ によって表現され，エージェントの状態遷移確率はこの Certainty によってパラメタライズされている．具体的に c は選択した行動に対応するグリッドへの移動とそれ以外のグリッドへの移動の確率の差分であり，選択したグリッドへは c の確率で移動し，それ以外のグリッドへは $1 - c$ の確率で移動する．即ち，エージェントの状態遷移は c が 1.0 に近づくほど状態遷移は決定的になる一方で，c が 0.5 に近づくほど確率的な状態遷移となる. 従って $c = 0.5$ のときはエージェントは選択した方向もしくは下方向どちらかを完全にランダムに移動する．以上のように状態遷移確率が 1 変数でパラメタライズされるように設計したことで，テスト環境と方策の導出に用いたモデルの間の誤差を意図的に生み出せるようにした．方策の導出に用いるモデルとして $c = 1.0$ として状態遷移が選択した行動の通りに実現される deterministic model と　$c = 0.9$ として状態遷移が北風の影響を受けて確率的となる windy model の

2 種類を用意した.

式 (3.5 または式 (3.6) を利用して，指数価値関数や価値関数そしてそれらに対応する最適な Controlled probability を導出するためには Uncontrolled probability $p^0(x' \mid x)$ が必要となる．Uncontrolled probability 定義するために，全ての方向への行動をとる確率が一様な方策を初期方策 $\pi^0(a \mid x)$ として設定し，想定するモデルに基づいて Uncontrolled probability $p^0(x' \mid x)$ を以下のように求めた.

$$p^0(x' \mid x; c) = \sum_{a \in \mathcal{A}(x)} \pi(a \mid x) p(x' \mid x, a; c)$$

ここでの $\mathcal{A}(x)$ は現在の格子で取りうる行動全てを表している.

1.1 節でも述べたように，LMG からは最適方策が直接求めることはできず，あくまで最適な次状態の分布である $p^{*\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x})$ が求められる．本論文の実験では適切な行動方策を獲得するために，以下の二次形式問題を全ての状態において解くことでこの問題を解決した.

$$\min_{\pi(a|x)} \left(\sum_{a \in \mathcal{A}(x)} \pi(a \mid x) p(x' \mid x, a; c) - p^{*\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x}) \right)^2 \tag{3.16}$$

$$\text{s.t.} \quad \sum_{a \in \mathcal{A}(x)} \pi(a \mid x) = 1, \;\; \pi(a \mid x) \leq 1, \;\; \pi(a \mid x) \geq 0 \;\; \forall a \in \mathcal{A}(x)$$

2.2 実験結果

先の節 2.1 で述べたように実験では 2 つのモデルのパラメータ c と 5 つの LMG のパラメータ α の組み合わせで計 10 個のパラメータの設定があり，それぞれのパラメータの組み合わせから指数価値関数および最適方策を導出した．ここで得られた方策の性能を評価するために状態遷移確率 $p(x' \mid x, a; c)$ のパラメータ c を $[0.5,\ 1.0]$ の範囲で変化させたテスト環境を用意し，全てのテスト環境において獲得した方策を用いた試行を 1000 回繰り返した.

図 3.3 は Deterministic model を，図 3.4 は Windy model をモデルに使用した結果をまとめたものである．両図ともそれぞれの行は α の値に対応し，上から順に 0.0，0.5，0.95，0.99，1.0 となっている．左端列の導出される価値関数 (Cost-to-go

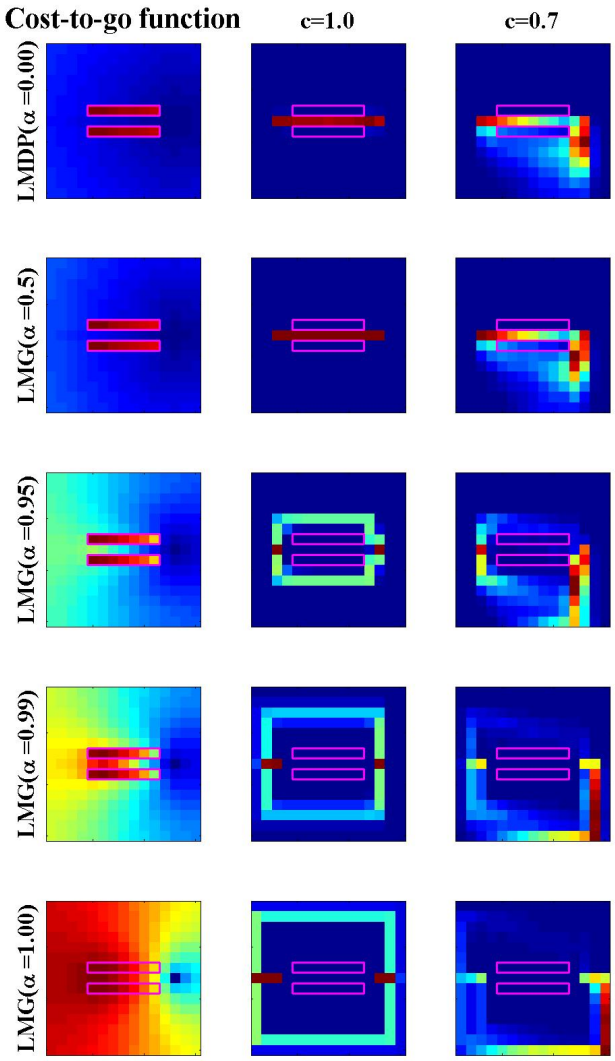


図 3.3 Deterministic model を用いた際の価値関数 (Cost-to-go 関数) と定常分布

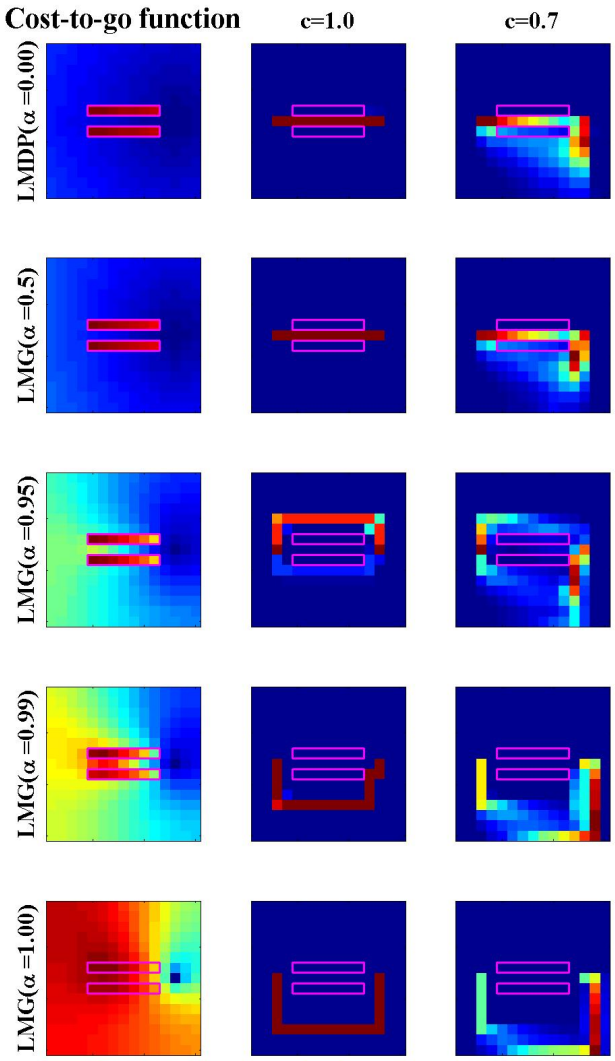


図 3.4 Windy model を用いた際の価値関数 (Cost-to-go 関数) と定常分布

関数) を示し, 中列，右端列のパネルはそれぞれテスト環境の状態遷移のパラメータを $c = 1.0, 0.7$ としたときの 1000 回の試行中の訪問頻度を図示したものである．これらはゴールに到達するとすぐに 1 の確率でスタートへに戻るとしたときの定常分布として解釈することができる．特に中列パネルで示されているテスト環境のパラメータが $c = 1$ のときの定常分布は，エージェントの状態遷移は北風の影響を受けず選んだ行動通りの経路が実現されるため，獲得した方策による経路を明確に伺うことができる．

Deterministic model を用いた際に得られる価値関数は $\alpha = 0.0, 0.5$ のとき沼地の間の狭い経路上のが低くなっている．これによって得られた方策も沼地の間を通りぬけ最短距離でゴールへと導くものとなっていることが分かる．この経路は北風が強く吹くようなテスト環境においては沼地に踏み込んでしまう場合が存在する．その一方，α が 1.0 に近いときの価値関数は沼地の間の入口付近は周囲と比較すると高いの値となる．さらに，α が 1.0 に近づくにつれてその範囲も広く，値も高くなっていく．その結果，得られた方策による経路は α が 1.0 に近づくにつれて沼地の避ける度合いが増していくことが確認できる．これら経路は北風が強く吹くようなテスト環境においても沼地に踏み込む確率を抑えることができている．

Windy model を用いた際に得られる価値関数は $\alpha = 0.0, 0.5$ のときは Deterministic model と同様に沼地の間の狭い経路上のが低くなっている．これによって得られた方策も沼地の間を通りぬけ最短距離でゴールへと導くものとなっていることが分かる．その一方，α が 1.0 に近いときの価値関数は沼地の間の入口付近の値は高くなるだけでなく，グリッドの上下で非対称な値となっている．その結果，得られた方策による経路は α が 1.0 に近づくにつれて沼地の避ける度合いが増していくだけでなく沼地の上を通過する経路と下を通過する経路の二つに別れる．$\alpha = 0.95$ の経路は沼地の避けて上を通過する大多数の経路と沼地のすぐ下を通過する少数の経路が確認できる．この経路は沼地を避けつつも北風による影響を利用しながらゴールへと近づく経路となっている．$\alpha = 0.99, 1.0$ は沼地の下を通過するのみが経路選択されどれだけ風が強い環境下でも沼地に踏み込むことのない経路となっている．

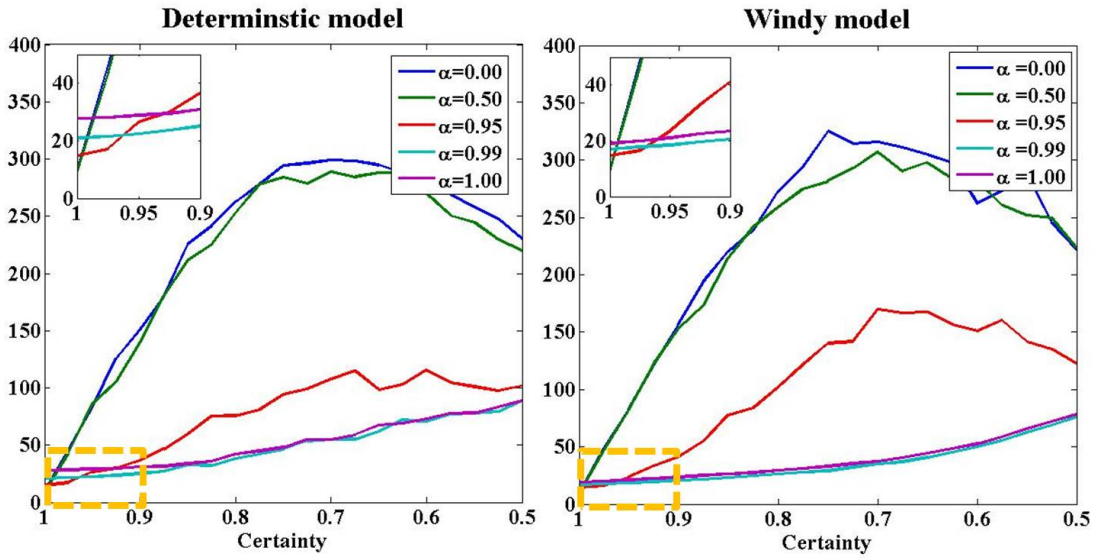


図 3.5 Grid-world with risky state 課題における LMG 方策の性能変化

図 3.5 はテスト環境のパラメータ c を x 軸にとり，それぞれの方策の一回の試行当たりの累積コストの平均の変化を示している．LMDP と同様の設定となる ($\alpha = 0.0$) や α の値の低い $\alpha = 0.5$ と設定した場合における Deterministic model を用いて得られた方策はテスト環境が決定論的であるときは最適な方策として機能している．しかしながら，テスト環境が確率的になり Deterministic model とテスト環境の差異が生じるとするすぐに累積コストの平均が増加しており，制御性能の劣化が確認される．その一方で，α の値を 1.0 に近い値に設定した場合における Deterministic model を用いて得られた方策はテスト環境が確率的になり Deterministic model とテスト環境の差異が生じたとしても累積コストの平均が増加が抑えられており，制御性能の劣化を抑えられていることが分かる．

Windy model を用いた際の最適方策の結果も Deterministic model を用いた結果と同様の傾向が見られるが，$\alpha = 0.99, 1$ と設定した場合，累積コストの平均の増加が Deterministic model 用いた方策の結果以上に抑えられており，制御性能の劣化のさらなる抑制が確認できる．その一方で Windy model はモデルとテスト環境一致する場合において，$\alpha = 0$ であっても方策の性能は最も良い性能を示さなかった．この一つの原因は LMG および LMDP は行動方策ではなく最適な状態遷移で

ある Controlled probability $p^{\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x})$ を最適化しているためであると考えられる．これは即ち，最適な Controlled probability $p^{\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x})$ は Uncontrolled probability $p^0(\mathbf{x}' \mid \mathbf{x}) > 0$ であるならば，与えられている状態遷移確率では実現不可能な次状態 $\mathbf{x}'$ の分布になる可能性があること示している．このために，得られた方策が必ずしも与えられている状態遷移確率の上で最適ではなくなってしまう．最適方策 $\pi(\mathbf{u}^c \mid \mathbf{x})$ を得られた指数価値関に基づいて直接導く手法は存在するが [46]，この手法は $\alpha = 1$ のときのみで適用可能である．

3. 連続状態行動空間における LMG：振り子の振り上げ課題

連続状態空間における LMG を用いた制御則のモデルの近似誤差に対するロバスト性を検証するために，本節では節 2.2.1 と同様の振り子の振り上げ課題のシミュレーション実験を行った．

3.1 課題設定

節 2.2.1 における実験と同様に，微小時間 $t = 0.01[s]$ で時刻を離散化してシミュレーションを実行した．また振り子のパラメータについても節 2.2.1 と同様に，長さを $l = 1[m/s]$，質量を $m = 1[kg]$，重力加速度を $g = 9.8[m/s^2]$，摩擦係数を $k = 0.05$ とした．ただし，今回のシミュレーションでは振り子固有のノイズのスケールは 0 とし，ノイズのスケール $\sigma = 4$ のランダムな制御ノイズが制御入力に対して付加されているものとした．さらに，コスト関数は式 (2.25) のガウス型の関数を用いて定義し，スケールを $\alpha = 2.5$，共分散行列を $\mathrm{diag}(\mathbf{\Sigma}_{\mathrm{cost}}) = [\pi/4,\ 4\pi/4]^2$ と設定した．

指数価値関数の近似に用いる基底関数 $f_i(\mathbf{x})$ のパラメータである，基底関数の中心 $\boldsymbol{m}_i$ は状態空間に対して一様に分布するように格子状に配置し ($N_f = 441$)，共分散 $\mathbf{S}_i$ はハンドチューニングによって適当な同一の共分散行列を使用した ($\mathrm{diag}(\mathbf{S}) = [\pi/20,\ \pi/20]^{-2}$)．その一方で指数価値関数の近似に用いる節点は基底関数の中心

とその中心の間を埋めるように格子状に配置した ($N = 1806$). 指数価値関数の関数近似の際，基底関数のパラメータはである中心と共分散は固定とし，線形パラメータ $\boldsymbol{w}$ のみを最適化することとした．また積分演算子 $\mathcal{G}[f_i](\mathbf{x})$ の計算は角速度成分が境界に接する場合であっても式 (2.13) の結果をそのまま用いることとした．

線形化された Bellman-Isacc 方程式である式 (3.5) と式 (3.6)) から得られる解はそれぞれ $z^\alpha(\mathbf{x})$ と $v(\mathbf{x})$ である．このため前章で用いた式 (2.14) による最適制御則を簡単に利用することができない．このことから本実験では連続時間系の問題において用いられる指数価値関数 $z^\alpha(\mathbf{x})$ および価値関数 $v(\mathbf{x})$ の勾配を明示的に利用した以下のような制御則を使用する．

$$\begin{aligned} u^*(\mathbf{x}) &= -\sigma^2 \boldsymbol{B}(\mathbf{x})^{\mathrm{T}} \frac{1}{(\alpha-1)z^\alpha(\mathbf{x})} \frac{\partial z^\alpha(\mathbf{x})}{\partial \mathbf{x}} \\ u^*(\mathbf{x}) &= -\sigma^2 \boldsymbol{B}(\mathbf{x})^{\mathrm{T}} \frac{\partial v(\mathbf{x})}{\partial \mathbf{x}} \end{aligned} \tag{3.17}$$

学習モデルを使用する場合は，$\boldsymbol{B}(\mathbf{x})$ ではなく学習モデルから推定される $\hat{\boldsymbol{B}}(\mathbf{x})$ を使用する．

3.2 学習モデルの近似誤差と LMG のパラメータ α

節 2.2.1 のシミュレーションと同様に，ランダムの制御則のもとで得られたサンプルから訓練データを抽出し，Linear-model と Linear-NRBF model を用いてモデルを学習した．

振り子のモデルの近似誤差は図 3.6 で示されていように，Linear-model を用いた学習モデルは角速度成分の近似誤差が大きくなっている．その一方で，Linear-NRBF model を用いた学習モデルの近似誤差はほとんど見られていない．これは，NRBF によって振り子の速度成分の存在する非線形な状態遷移がよく近似できたことに加え，今回の実験では振り子の自体のノイズを 0 としたためである．

図 3.7 は 2 つの異なる関数近似器を用いた学習モデルと真のダイナミクスから得られた指数価値関数 $z^\alpha(\mathbf{x})$ をまとめて図示したものである．いずれのモデルでも異なる値の $\alpha = [0, 0.50, 0.75, 0.9, 0.95, 0.99, 1]$ を設定して指数価値関数を導出

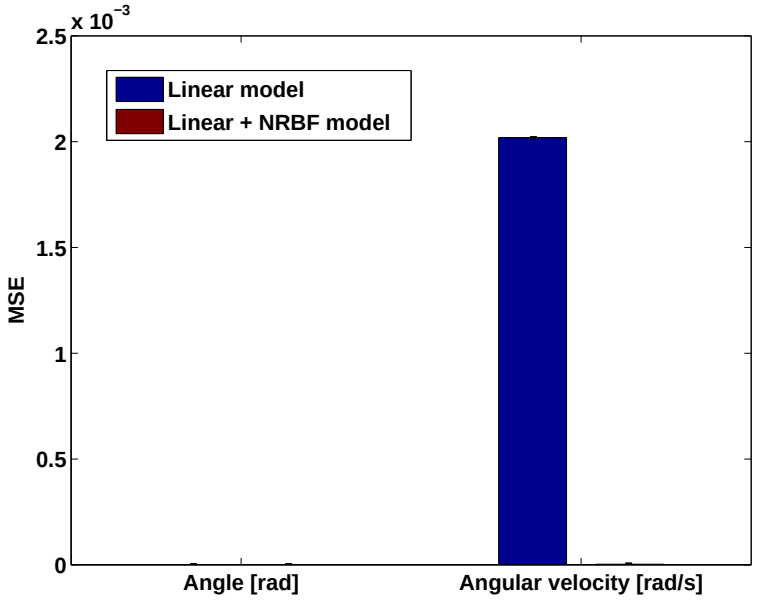


図 3.6 各成分と学習モデルにおける近似誤差

した．α が 1 に近づくにつれて得られた指数関数 $z^{\alpha}(\mathbf{x})$ は全体的に平坦になっていくこれは，式 (3.5) のコスト関数の係数が α が 1 に近づくに連れて小さくなって行くことと一致している．真のダイナミクスと Linear-NRBF model を用いた学習モデルから得られた指数価値関数に大きな差異は見られないが，Linear model を用いた学習モデルから得られた指数価値関数は，対角線上に存在する指数価値関数の値が高い ($\alpha = 1$ に関しては価値関数 (Cost-to-go 関数) が低い) 領域が他の指数関数と違った傾きで分布をしている．

図 3.8 では導出された指数関数 ($\alpha \neq 1$) 及び価値関数 ($\alpha = 1$) に対応する制御則とそれぞれの制御則のもとでの軌道を描画している．いずれの制御則も存在する指数価値関数の値が高い (もしくは価値関数が低い) 領域屁と導くような制御則となっている．制御量の全体の大きさは $\alpha = 0.75$ を境に増減をしている．LMDP と同一の問題設定となる $\alpha = 0$ と設定した場合には，真のダイナミクスから得られた制御則と Linear model を用いた学習モデルから得られた制御則の軌道は下で振る回数など異なったものになっているが，$\alpha = 0.75, 0.95$ では似たような軌道となっている．

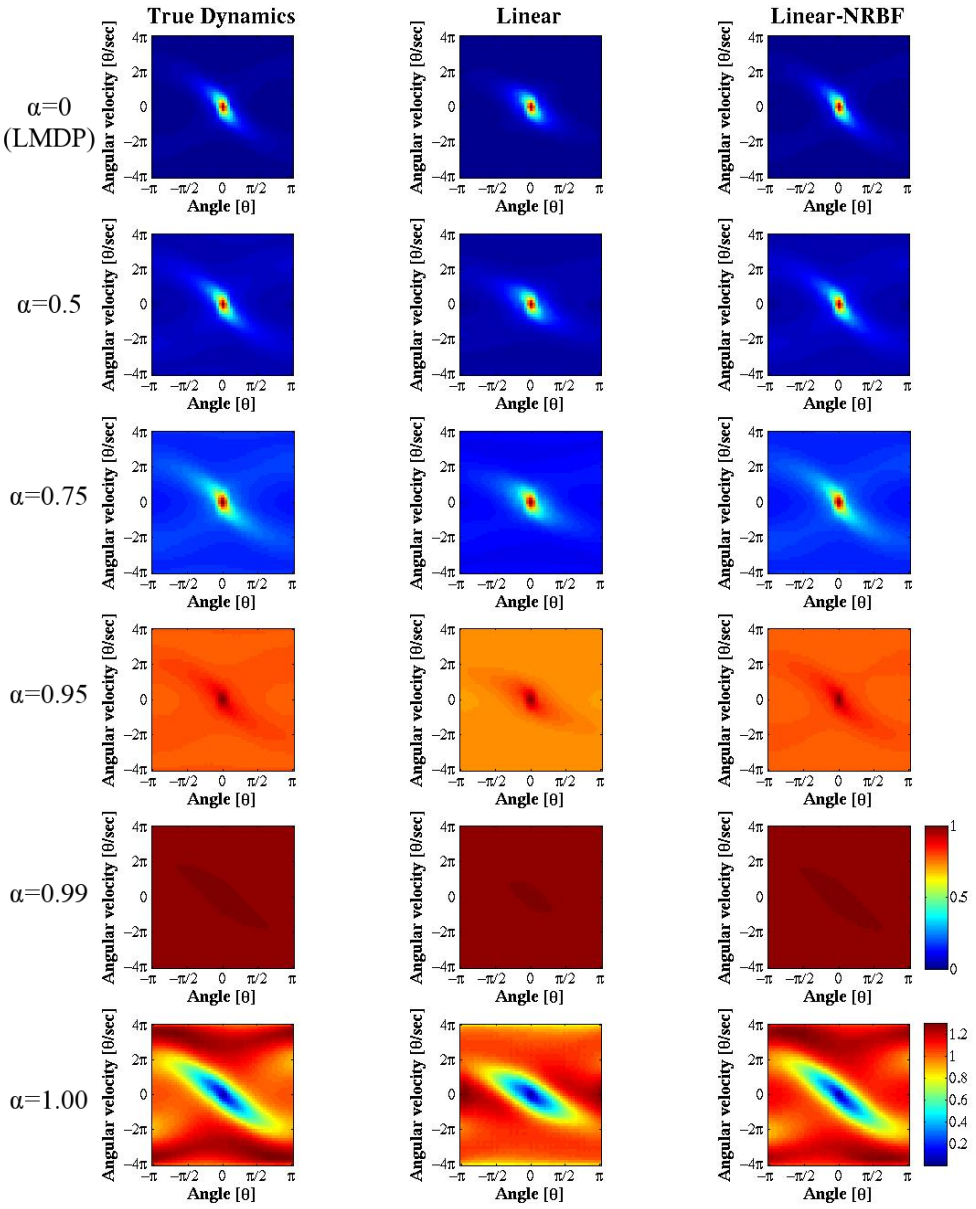


図 3.7 LMG によって得られた指数価値関数 $z^{\alpha}(\mathbf{x})$ 及び価値関数 $v(\mathbf{x})$

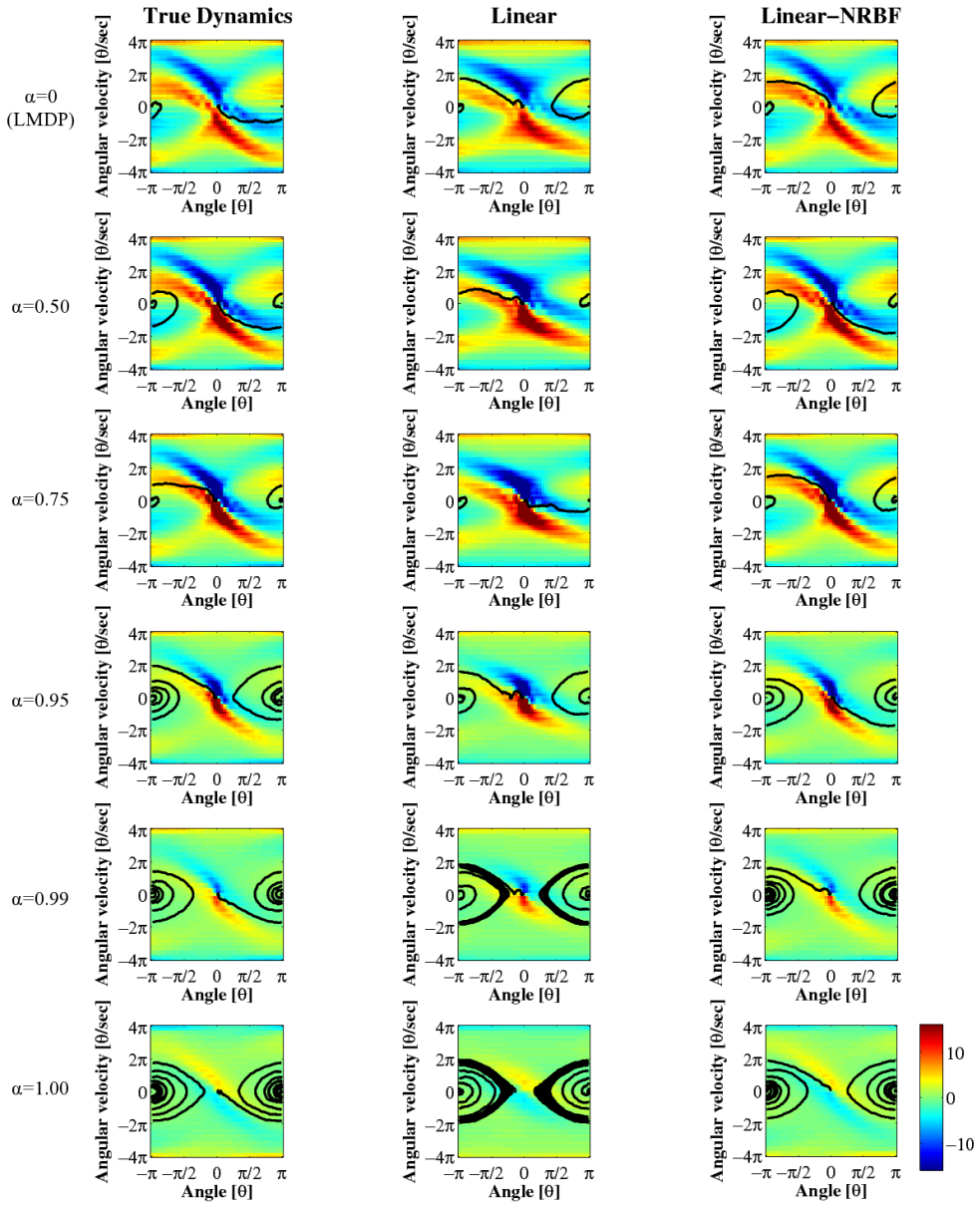


図 3.8 LMG によって得られた制御則と獲得した制御則のもとでの軌道

α の変化によってそれぞれの学習モデルから導出された制御則の性能がどのように変化するかを評価するために，全ての制御則を用いて振り上げ課題を 50 回実行した．それぞれの試行は振り子が目標状態に到達したもしくは $100[s]$ (10000step) 経過したときに終了した．図 3.9) の上図は α を x 軸にとり，振り上げまでの総コストを y　軸にとり価値関数及び制御則の導出に利用したモデル毎にプロットしたものである．下図は要した時間を y　軸にとりプロットしたものである．

LMDPP と同一の問題設定となる $\alpha = 0$ と設定した場合では，近似誤差の大きかった Linear-model を用いた学習モデルから得られた制御則の総コスト及び振り上げまでの時間のどちらも他のモデルから得られた制御則の結果と比較すると良くない結果を示しており，得られた制御則の性能が学習モデルの近似誤差によって劣化していることが確認できる．しかしながら，この制御性能のギャップは α の値が大きくなるに従って小さくなり，$\alpha = 0.75, 0.90$ ではいずれの関数近似器を使用した学習モデルであっても同程度の制御性能をもった制御則が得られていることが確認できる．ただし，$\alpha = 0.90$ 以上となると全ての制御則の性能が悪化してしまい，モデル間の制御性能のギャップも再び増加してしまう結果となった．

3.3 テスト環境の変化と LMG のパラメータ α

先の節では近似誤差の存在する学習モデルから得られた制御則の性能について議論したが，本節では制御則を導出するために用いた環境と異なるテスト環境を用意することで，意図的に学習モデルの近似誤差を発生させたときの制御則の性能について検証する．

はじめに学習環境として真のダイナミクスを用意し，異なる値の $\alpha = [0, 0.50, 0.75, 0.9, 0.95, 0.991]$ の LMG のパラメータのもとで制御則を導出した．

学習時と異なるテスト環境として振り子の質量 $m[kg]$ を $[1,\ 2.5][kg]$ の間で $0.1[kg]$ 刻みで変更した振り子を用意した．用意されたテスト環境において先に得られた制御則による試行を 50 回行った．それぞれの試行は振り子が目標状態に到達もしくは $200[s]$(20000step) 経過したときに終了した．図 3.10 の上図は振り上げまでに要した時間を y 軸，質量を x 軸として制御則の導出に用いた α ごと

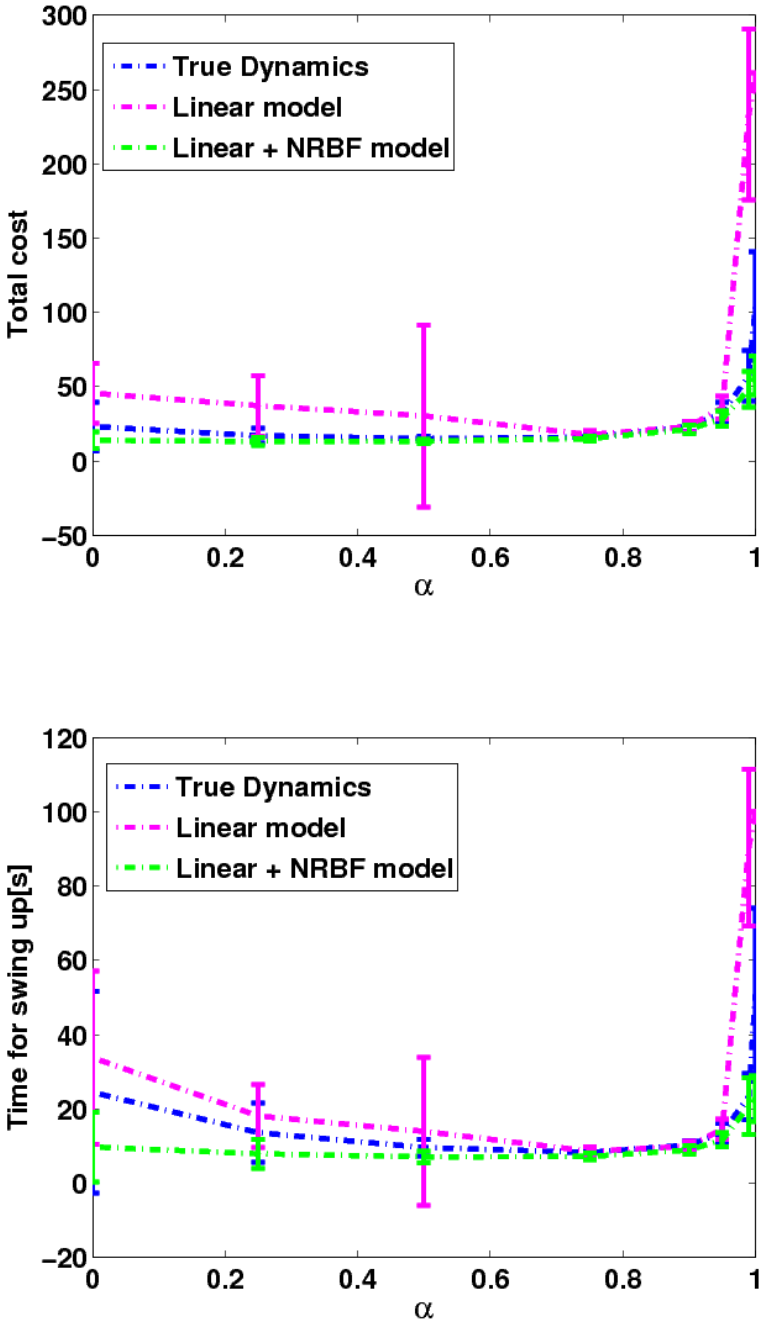


図 3.9 各制御則の振り上げまでの総コストと時間の α による変化

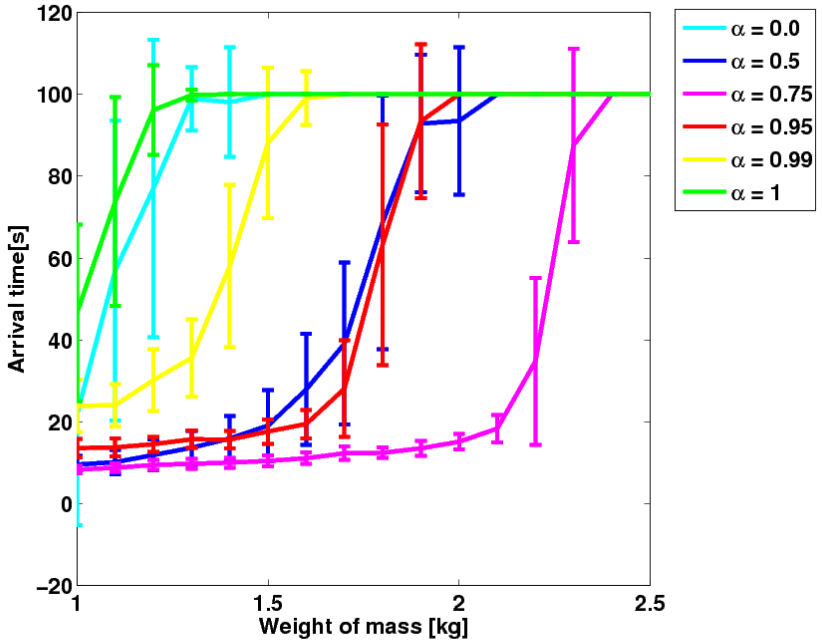
にプロットしている．また下図はと振り上げ成功率を y 軸としてプロットしたものである．

どのパラメータ α を用いても振り子の質量が増えるにつれて振り上げに必要な時間は増加し成功率も低下した．即ち，制御性能の劣化が見られた．しかしながら，制御性能の劣化速度は α の値によって異なっている．$\alpha = 0.00$ と設定し LMDP 同一の問題設定となるときの制御則は，学習時より振り子の質量が少しでも増すと振り上げまでに必要な時間が増加し，成功率も低下している．即ち，テスト環境の変化に対応できない制御則となっている．その一方で，$\alpha = 0.75$ までは α の値が大きくなるに従い振り上げまでに必要な時間の増加や成功率の低下の速度がより抑えられている．即ち，α の値が大きくなるに従いテスト環境の変化に対してよりロバストな制御則となった．しかしながら，$\alpha = 0.75$ 以降ではそれまでと全く逆に α が大きくなるにつれて振り上げまでに必要な時間の増加や成功率の低下の速度が増す結果となり，テスト環境の変化に対するロバスト性は低下する結果となった．

4. 連続状態行動空間における LMG： アクロボットの振り上げ課題

本節では，さらなる連続状態行動空間における LMG の適用例として，図. 3.11 で示されているようなアクロボットの振り子の振り上げ課題を採り上げる．アクロボットは制御トルクは 2 関節の中でリンク間の関節のみで働くが，このように自由度よりも制御入力の低いシステムは列駆動システムと呼ばれ，アクロボットは代表的な列駆動システムとして知られている．列駆動システムは，制御入力が少なく済ませることができることから，コストや重量の観点から優れた側面を持つ．その一方でそのダイナミクスは強い非線形を持つことから，その制御はより複雑となる [47]．例えば，振り子の場合は一定だった制御トルクが状態遷移に及ぼす影響は，アクロボットでは状態 (リンク 2 の関節角度 q_2) に依存して非線形に変化する．

アクロボットの 2 リンクが倒立している安定領域付近では，LQR を用いること



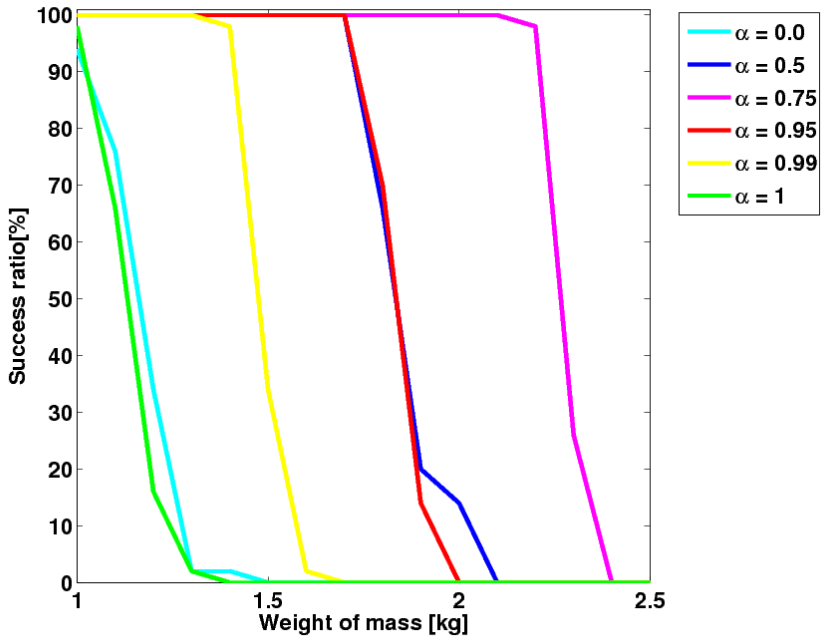


図 3.10 学習時と異なる実行環境における振り上げまでの時間と振り上げの成功率

で安定化が実現される．しかしながら，LQR をだけで振り上げを実現することはできないため，アクロボットの振り上げは非線形制御の課題として多くの研究がなされている [48, 49]

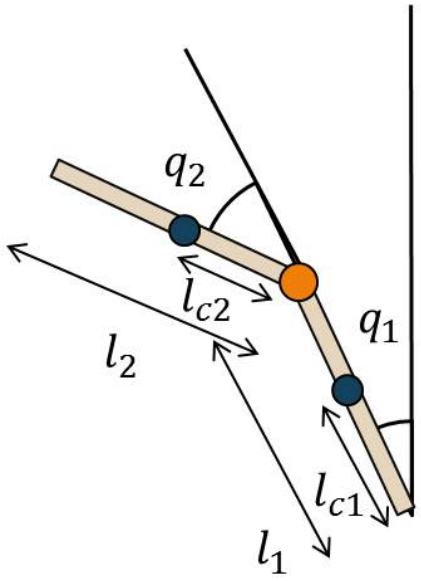


図 3.11 アクロボットのイメージ図

4.1 課題設定

アクロボットの状態 $\mathbf{x}$ は，リンク 1 の角度 q_1, 角速度 $\dot{q}_1$，リンク 2 の角度 q_2, 角速度 $\dot{q}_2$ から構成された 4 次元のベクトルで表現され，リンク 1 とリンク 2 の間で働くトルクを制御 u で表現する．その状態方程式は以下のように，

$$\begin{aligned} \dot{\mathbf{x}} &= (\boldsymbol{a}(\mathbf{x}) + \boldsymbol{b}(\mathbf{x})u)dt, \\ \mathbf{x} &= \begin{bmatrix} q_1 & q_2 & \dot{q}_1 & \dot{q}_2 \end{bmatrix}^{\mathrm{T}} \end{aligned} \tag{3.18}$$

状態依存の状態遷移 $\boldsymbol{a}(\mathbf{x})$ と制御入力に線形な状態遷移 $\boldsymbol{b}(\mathbf{x})u$ の 2 つ和として表現される．それぞれ $\boldsymbol{a}(\mathbf{x})$ と $\boldsymbol{b}(\mathbf{x})$ の詳細については付録 4 を参照されたい．

以上のように定義されたダイナミクスに従いアクロボットは振る舞うこととし，節 2 における実験と同様に，微小時間 $t = 0.01[s]$ で離散化してシミュレーションを

実行した．ただし，シミュレーションでは各関節角速度はそれぞれ $\dot{q}_1 \in [-4\pi,\ 4\pi]$，$\dot{q}_2 \in [-8\pi,\ 8\pi]$ と上限と下限を設けた．また，各関節角度の範囲は $q_1,\ q_2 \in [-\pi,\ \pi]$ とし，$q_1,\ q_2 = -\pi$ と $q_1,\ q_2 = \pi$ はそれぞれ同一とみなした．アクロボットの各リンクの長さや重さ等のパラメータは表 3.1 の値を使用した．

課題の目標状態をアクロボットの 2 つのリンクが倒立し静止した状態とした．この目標状態の基準を定めるために，以下のようなアクロボットの先端の高さ h を定義する．

$$h = \frac{l_1 \cos(q_1) + l_2 \cos(q_1 + q_2)}{l_1 + l_2}$$

アクロボットの先端の高さ h は $[-1,\ 1]$ の間で正規化されており，完全に倒立した状態で $h = 1$ となり，下の方で下がった状態では $h = -1$ となる．目標状態への振り上げを実現されたときに課題は終了すると仮定し，$h > 0.95$，$\dot{q}_1 \in [-\pi/2,\ \pi/2]$，$\dot{q}_2 \in [-\pi/2,\ \pi/2]$ となる領域を吸収状態 $\mathbf{x}_{\mathcal{A}}$ と定めた．

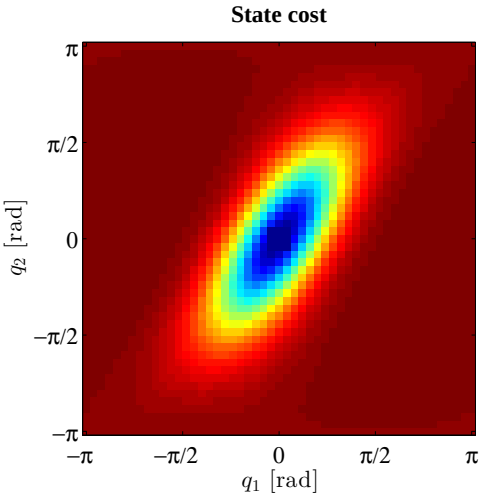
また，目標状態でコスト関数 $q(\mathbf{x})$ が 0 となるように，状態コスト関数 $q(\mathbf{x})$ を非正規ガウス関数を用いて

$$q(\mathbf{x}) = c_{scale}(1 - \exp\left(k_1\left(1 - h\right)\right)\exp(-k_2\dot{q}_1^2)\exp(-k_3\dot{q}_1^2)) \tag{3.19}$$

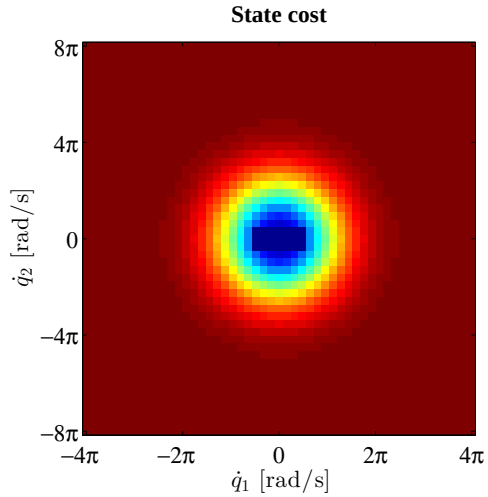
のように定義した．ここで，$k_1 = 8$，$k_2 = 0.14$，$k_3 = 0.36$，そして $c_{scale} = 2$ とした．また，吸収状態 $\mathbf{x}_{\mathcal{A}}$ では $q(\mathbf{x}_{\mathcal{A}}) = 0$ としている．図 3.12 はここで定義したコスト関数を示している．

表 3.1 アクロボットのシミュレーション実験に用いたパラメータ

リンク 1 の長さ：$l_1[m]$	1.0	リンク 1 の重さ：$l_1[kg]$	1.0
リンク 2 の長さ：$l_2[m]$	1.0	リンク 2 の重さ：$l_2[kg]$	1.0
リンク 1 の重心位置：$l_{c1}[m]$	0.5	リンク 1 の慣性モーメント：$I_1[kg \cdot m^2]$	0.083
リンク 2 の重心位置：$l_{c2}[m]$	0.5	リンク 2 の慣性モーメント：$I_2[kg \cdot m^2]$	0.083
重力：$[m \cdot kg/s^2]$	9.8	リンク 1 の摩擦係数：k_1	0.00
		リンク 2 の摩擦係数：k_2	0.00



(a) 関節角度 q_1, q_2 に対するコスト関数 ($\dot{q}_1, \dot{q}_2 = 0$)



(b) 関節角速度 $\dot{q}_1, \dot{q}_2$ に対するコスト関数 ($q_1, q_2 = 0$)

図 3.12 アクロボットのシミュレーションで使用したコスト関数

モデルの学習には現時刻での状態と次時刻の状態との差分 (状態遷移)$\Delta \mathbf{x}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$ と現時刻の状態 $\mathbf{x}_k$ 及び制御量 u_k を用いた．ここで，関節角度の境界をまたぐ状態遷移は連続していることを考慮し差分 $\Delta \mathbf{x}_k$ を与えた．モデルの学習のために $u \sim \mathcal{N}(1, 5^2)$ となるようなランダムな制御則のもとでアクロボットの各時刻での状態遷移と状態，そして制御量からなるデータ $\mathcal{D}_k = [\Delta \mathbf{x}_k^{\mathrm{T}}, \mathbf{x}_k^{\mathrm{T}}, u]$ を取得した．ただし，学習に用いるデータの過度な偏在を抑えるために，10 秒ごとに初期位置を $h > 0.7$，$\dot{q}_1,\ \dot{q}_2 = 0$ となるように変えながらデータの取得を行った．ここで得られたデータの中から無作為に $N_{\mathcal{D}} = 10000$ のサンプルを抽出し，モデル学習の訓練データ $\boldsymbol{\mathcal{D}} = [\mathcal{D}_1,^{\mathrm{T}} \cdots,\ \mathcal{D}_{N_{\mathcal{D}}}^{\mathrm{T}}]^{\mathrm{T}}$ として用いた．

モデルの学習には表 3.2 に示す，Linear-Constant，NRBF-Constant，Linear-NRBF，NRBF-NRBF の以上 4 つの関数近似器を用いた．これらの関数近似器の名前は式 (4.10) における $\boldsymbol{a}(\mathbf{x})$ と $\boldsymbol{b}(\mathbf{x})u$ に対応する基底関数 $\boldsymbol{\phi}_a(\mathbf{x})$,$\boldsymbol{\phi}_b(\mathbf{x}, u)$ にに基づいて名付けられている．例えば，linear-constant　model では基底関数 $\boldsymbol{\phi}_a(\mathbf{x})$ は状態に関して線形とし，基底関数 $\boldsymbol{\phi}_b(\mathbf{x}, u)$ は制御に関して線形となるような状態に依存しない定数の基底関数となっている．モデル学習の基底関数として用いる正規化動径基底関数関数 (Normalized radial basis function，NRBF) は節 2.2.1 と同様の計算式 (式 (2.26)) を用いた．ここで，共分散行列 $\boldsymbol{\Sigma}_{\psi_i}$ はすべての基底関数で同一の $\boldsymbol{\Sigma}_{\psi}$ となるようにハンドチューニングにより適当に設定した．また，中心 $\boldsymbol{\mu}_i$ はモデル学習の訓練データ $\boldsymbol{\mathcal{D}}$ の中から，

$$\min_j \left(\boldsymbol{\mu}_j - \boldsymbol{\mu}_i\right)^{\mathrm{T}} \boldsymbol{\Sigma}_{\psi} \left(\boldsymbol{\mu}_j - \boldsymbol{\mu}_i\right) \geq c_{t1}, \quad c_{t1} > 0$$

となるように状態 $\mathbf{x}$ を抽出した．この結果，NRBF の中心は共分散行列 $\boldsymbol{\Sigma}_{\psi_i}$ としたときの中心間のマハラビノス距離が最短の値は必ず $\geq c_t$ 以上になるように定められる．この結果，$N_b = 97$ の基底関数を NRBF として使用した．

指数価値関数 $z^{\alpha}(\mathbf{x})$ の関数近似に用いる節点行列 $\mathbf{F}, \mathbf{G}$ の作成には，関数近似のための基底関数と学習点を定める必要がある．しかしながら，状態変数が 4 次元で表現されるアクロボットにおいて先の節の振り子と同様に状態空間に上で一様な密度で基底関数の中心および学習点を配置すると，計算コストが単純計算で振り子のの 2 乗となってしまう．この問題点を解決するため，今回の実験では基底関数の中心と学習点はそれぞれ，格子状に配置した状態データの中からモ

表 3.2 アクロボットのダイナミクスの学習に用いた関数近似器の基底関数セット

	$\boldsymbol{\phi}_a(\mathbf{x})$	$\boldsymbol{\phi}_b(\mathbf{x}, u)$
Linear-Constant	$\begin{bmatrix}\mathbf{x}^{\mathrm{T}} & 1\end{bmatrix}^{\mathrm{T}}$	u
NRBF-Constant	$\begin{bmatrix}\psi_1(\mathbf{x}) & \cdots & \psi_{N_b}(\mathbf{x}) & 1\end{bmatrix}^{\mathrm{T}}$	u
Linear-NRBF	$\begin{bmatrix}\mathbf{x}^{\mathrm{T}} & 1\end{bmatrix}^{\mathrm{T}}$	$\begin{bmatrix}\psi_1(\mathbf{x})u & \cdots & \psi_{N_b}(\mathbf{x})u & u\end{bmatrix}^{\mathrm{T}}$
NRBF-NRBF	$\begin{bmatrix}\psi_1(\mathbf{x}) & \cdots & \psi_{N_b}(\mathbf{x}) & 1\end{bmatrix}^{\mathrm{T}}$	$\begin{bmatrix}\psi_1(\mathbf{x})u & \cdots & \psi_{N_b}(\mathbf{x})u & u\end{bmatrix}^{\mathrm{T}}$

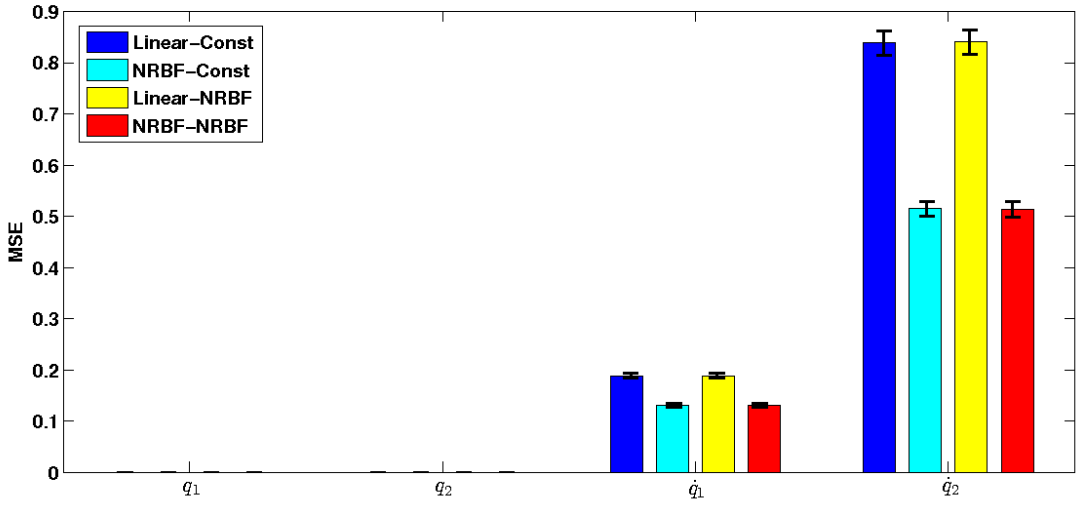
デル学習の訓練データ $\boldsymbol{\mathcal{D}}$ の状態 $\mathbf{x}$ の近傍のみを採用することとした．具体的には，先に基底関数 $f(\mathbf{x})$ の共分散行列 $\mathrm{diag}(\mathbf{S})^{-1}$ をハンドチューニングによって $\mathrm{diag}(\mathbf{S})^{-1} = [0.07,\ 0.07,\ 5.00,\ 20.0]$ と適当に定め，格子状に配置した状態 $\mathbf{x}$ と訓練データ $\boldsymbol{\mathcal{D}}$ の状態 $\mathbf{x}_{\mathcal{D}}$ の共分散行列 $\mathrm{diag}(\mathbf{S})^{-1}$ としたときのマハラビノス距離がある値 c_{t2} 以上であれば基底関数の中心もしくは学習点として採用した．この結果，$N_f = 4960$ の基底関数と $N = 9571$ の学習点を用いて節点行列 $\mathbf{F}, \mathbf{G}$ の作成して指数価値関数を近似した．ただし，指数価値の関数近似は基底関数の線形パラメータ $\boldsymbol{w}$ のみを最適化し，基底関数 $f(\mathbf{x})$ のパラメータは固定で行った．

以上の学習によって得られたモデルと指数価値 $z^{\alpha}(\mathbf{x})$ に基づき最適制御則 $\mathbf{u}^{*}(\mathbf{x})$ を式 (3.17) を利用して求めた．

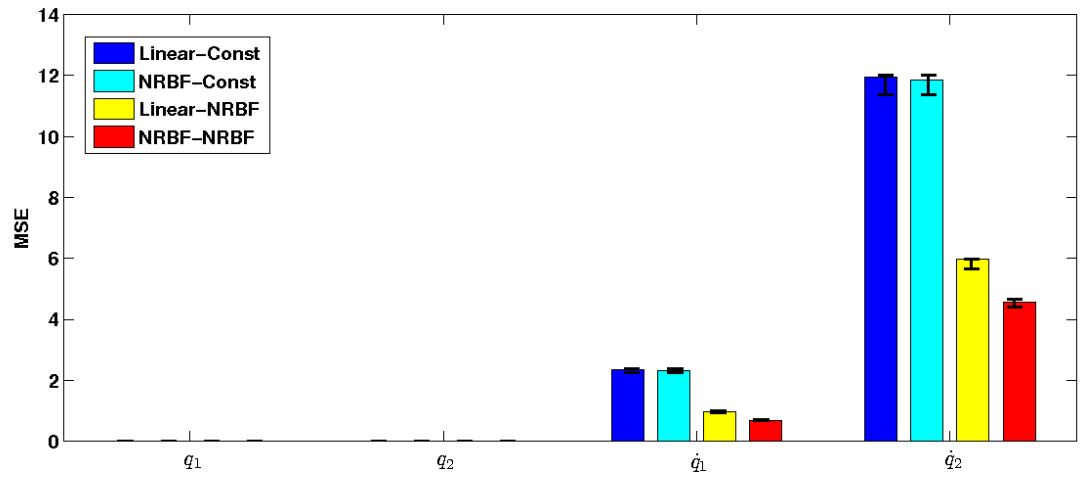
4.2 学習モデルの近似誤差と LMG のパラメータ α

節 4.1 で記述したように今回のシミュレーションでは，アクロボットの状態遷移を Linear-constant，NRBF-constant，Linear-NRBF，そして NRBF-NRBF の計 4 つの関数近似器を用いて学習を行った．

これら関数近似器による学習モデルの近似誤差を比較するために，実際の状態遷移 $\Delta\mathbf{x}$ と学習モデルによって予測される状態遷移 $\Delta\hat{\mathbf{x}}$ の平均二乗誤差を計算した．その結果を図 3.13.(a) は示している．エラーバーは平均二乗誤差の信頼区間



(a) 制御入力 $u = 0$ における状態遷移 $\Delta\mathbf{x}$ の各学習モデルの近似誤差 ($N = 9671$)



(b) $\boldsymbol{b}(\mathbf{x})$ の各学習モデルの近似誤差 ($N = 9671$)

図 3.13 各成分の学習モデルにおける近似誤差

を示している．平均二乗誤差の導出のために，テストデータとして制御入力 $u = 0$ のときの指数価値関数の関数近似の学習点 ($N = 9671$) における状態遷移 $\Delta\mathbf{x}$ を式 (4.10) から計算した．また，学習モデルから予測される状態遷移 $\Delta\hat{\mathbf{x}}$ もテストデータ同様に制御入力 $u = 0$ とした．どの学習モデルも関節角度成分 q_1，q_2 の状態遷移は十分に予測が出来ている一方で，強い非線形をもつ関節角速度成分 $\dot{q}_1$，$\dot{q}_2$ の状態遷移の予測が悪くなっている．特に，$\boldsymbol{a}(\mathbf{x})$ の近似を行う基底関数 $\boldsymbol{\phi}_a(\mathbf{x})$ に線形な基底関数を使用した学習モデルが NRBF を使用した学習モデルより近似誤差が大きくなっていることが分かる．

今回のシミュレーションでは制御 u が状態遷移にどれだけ影響を及ぼすかを定める行列 $\mathbf{B}(\mathbf{x})$[1]は状態に依存するだけでなく，制御の導出 (式 (3.17)) にも用いられれるため，その予測は非常に重要となってくる．このことから，ベクトル $\boldsymbol{b}(\mathbf{x})$ の近似誤差も調査した．具体的にはベクトル $\boldsymbol{b}(\mathbf{x})$ の各成分の値の平均二乗誤差を求めた，その結果を図 3.13.(b) は示している．エラーバーは平均二乗誤差の信頼区間を示している．テストデータには状態遷移のときと同様に指数価値関数の関数近似の学習点 ($N = 9671$) における $\boldsymbol{b}(\mathbf{x})$ を式 (4.10) から計算し，予測値は $\boldsymbol{\phi}_b(\mathbf{x}, u)$ の $u = 1$ としたときの値から計算した．状態遷移と同様にの学習モデルも関節角度 q_1，q_2 に対応する成分は十分に良く予測が出来ている一方で，強い非線形性をもつ関節角速度成分 $\dot{q}_1$，$\dot{q}_2$ に対応する成分の予測が悪い．特に，$\boldsymbol{b}(\mathbf{x})$ が状態に依存しない基底関数に Constant を使用した学習モデルが NRBF を使用した学習モデルより近似誤差が大きくなっていることが分かる．

以上の結果から，NRBF-NRBF を関数近似器として用いた学習モデルが状態遷移 $\Delta\mathbf{x}$ と $\boldsymbol{b}(\mathbf{x})$ の近似誤差が小さく，最も予測性能の高いモデルとなっている．その一方で，Linear-Constant を関数近似器として用いた学習モデルが近似誤差が大きく，最も予測性能の低いモデルとなっている．

ここで得られた 4 つの学習モデルおよびと真のダイナミクスを用いて得られた制御則の性能と LMG のパラメータ α の関係を調査するため，LMG のパラメータを $\alpha = [0.00, 0.25, 0.50, 0.65, 0.80]$ と設定し，それぞれのモデルで指数価値関数およびそれに基づいた制御則を導出した．得られたそれぞれの制御則のもとで目標

[1]アクロボットは制御が 1 次元であるため行列 $\mathbf{B}(\mathbf{x})$ は実際にはベクトル $\boldsymbol{b}(\mathbf{x})$ となる．

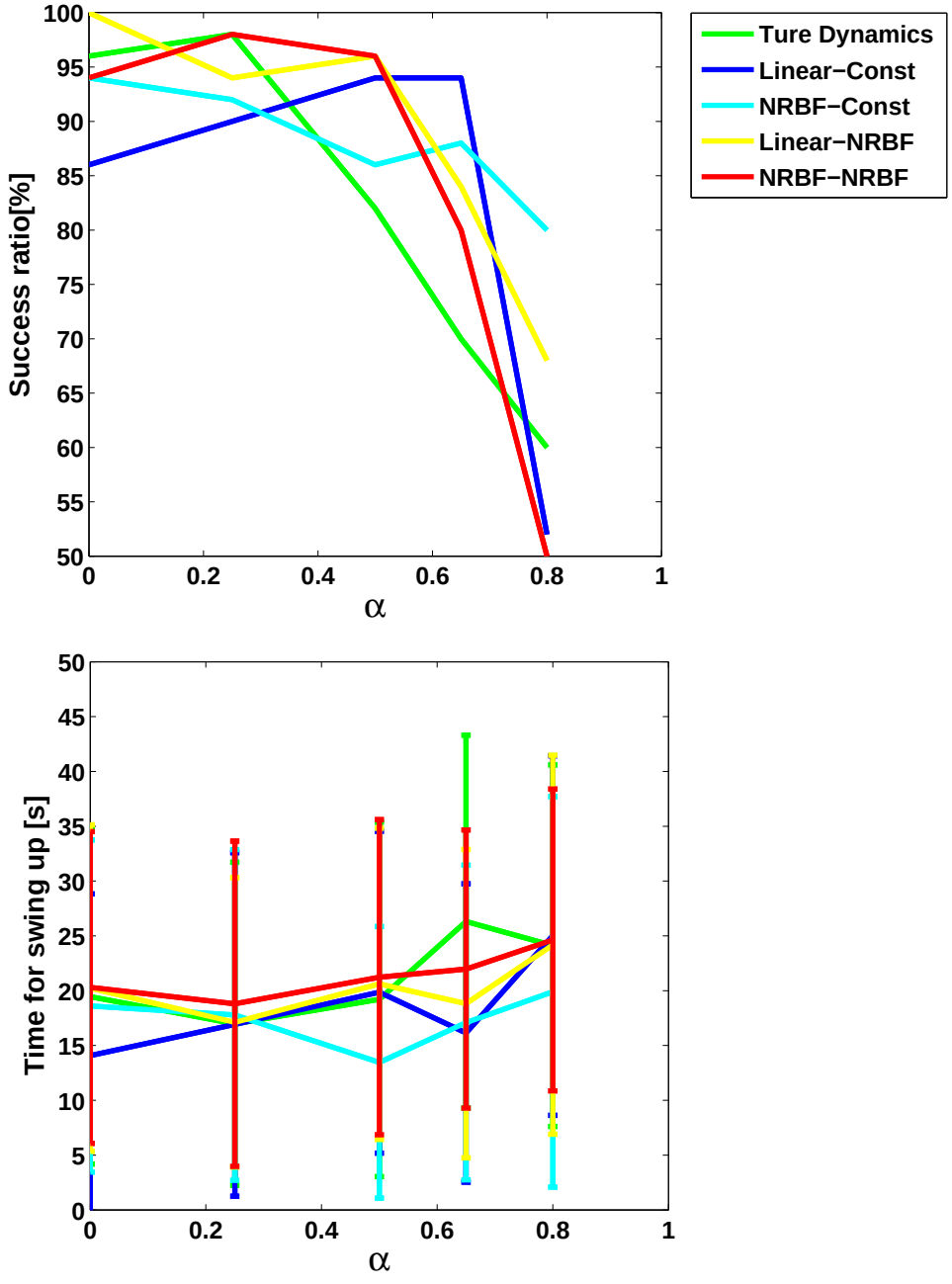


図 3.14 各制御則による振り上げの成功率と時間の α による変化

状態 (吸収状態) に到達もしくは開始から $60[s]$ (6000step) 経過するまでを 1 試行とした試行を 50 回行った．その成功率と振り上げに成功した試行における振り上げまでに必要とした時間を縦軸に横軸に LMG のパラメータ α をとりプロットしたものが図 3.14 である．

まず，いずれの LMG のパラメータ α および学習モデルのもとで得られた制御則は最低でも 50%の試行で振り上げまでに至ることが確認できた，しかしながらどの制御則も倒立状態での安定化までに至ることはなかった．振り上げの様子としては，リンク 2 を最大の速度付近になるほど振り回し，その慣性を利用しリンク 1 も振り上げ高さまで振り上げ，倒立状態に近づくと関節角速度を速やかに抑えるような振り上げとなっている．

個別のモデルにおける結果を見ていくと．図中にて緑の線で表記されている真のダイナミクスによる制御則は LMG のパラメータ α が上昇するに従って成功率が低下した．同様の傾向は Linear-Constant 以外の関数近似器で学習したモデルにおいても見られた．即ち，真のダイナミクス利用した制御則やその他の学習モデルから得られた制御則は十分にアクロボットのダイナミクスを表現できていたため，妨害入力を想定したことで逆にその性能を劣化を引き起こしたと考えられる．その一方で，図中の青線で表記されている Linear-Constant の関数近似器で学習したモデルから得られた制御則は，LMG のパラメータ $\alpha = 0.65$ において成功率が最大となっている．これは最も予測性能の低かった Linear-Constant による学習モデルによる制御則ではの近似誤差が妨害入力として働いたため，妨害入力を想定した $\alpha > 0$ として得られた制御則で制御性能が上昇したと考えられる．ただし，どのモデルから得られた制御則の振り上げまでに必要とした時間は，LMG のパラメータ α が上昇するに従い増加している．これは，妨害入力を想定したことで目標状態から最も離れているアクロボットの両リンクが下の方にあるときは，できるだけ制御出力を抑えるような制御則が導出されていたためであると考えられる．

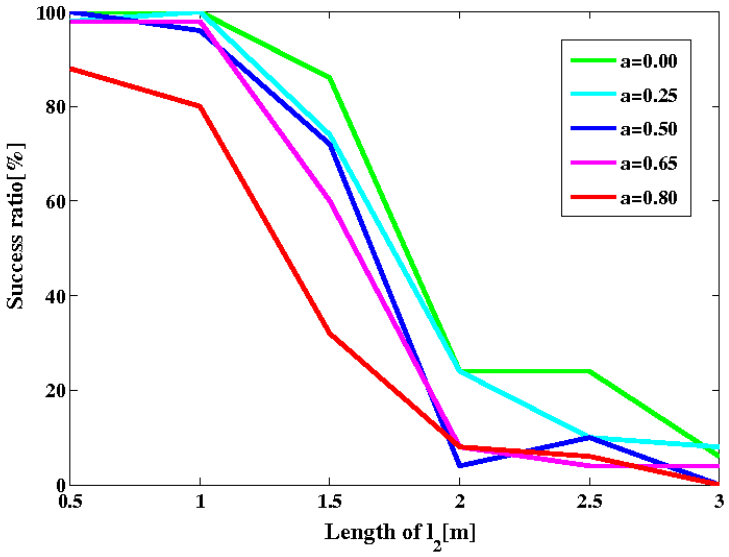
4.3 テスト環境の変化と LMG のパラメータ α

制御則を導出するために用いた環境と実行を変えることで，意図的に学習モデルに近似誤差を発生させたときの制御則の性能を比較した．

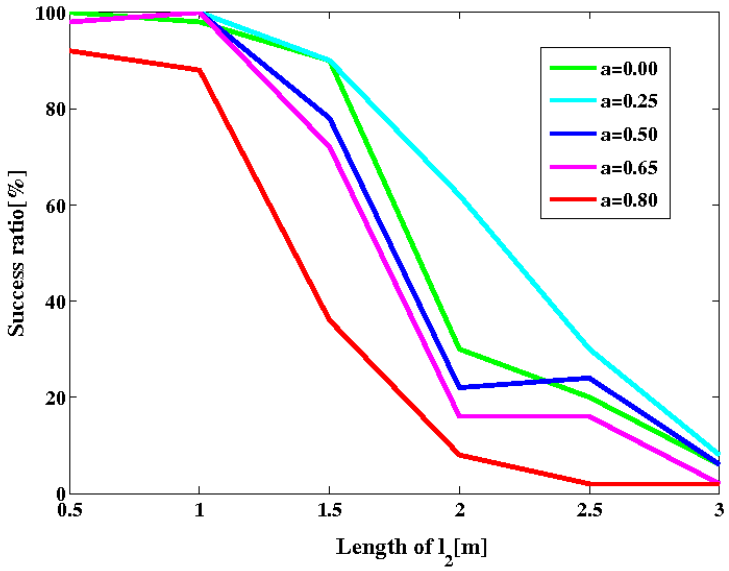
具体的には，学習に用いた環境がにおいて $1[m]$ としているリンク 2 の長さ l_2 を $[0.5,\ 3][m]$ の範囲で $0.5[m]$ 刻みで実行環境を用意した．このリンク 2 の長さ l_2 の変更に伴って，リンク 2 の関節から重心までの距離 l_{c2} もリンク 2 の長さの半分になるように変更した．また，リンク 2 の慣性モーメント I_2 も細長い一様な棒の慣性モーメントの導出式である $I = \frac{1}{12}ml$ を用いて変更した．それぞれの環境で LMG のパラメータを $\alpha = [0.00, 0.25, 0.50, 0.65, 0.80]$ と設定したときの真のダイナミクスと NRBF-NRBF による学習モデルを用いた制御則を用いた試行を 50 回繰り返した．一回の試行は目標状態に到達もしくは開始から $60[s]$ (6000step) 経過するまでを 1 試行とした．図 3.15.(a) は真のダイナミクスを用いて得られた制御則による結果を横軸をリンク 2 の長さ，縦軸を成功率としてプロットしたものである．また，その NRBF-NRBF の関数近似器で学習したモデルによる制御則の結果が図 3.15.(b) となっている．

真のダイナミクスによって得られた全ての制御則はリンク 2 が長くなるにつれて，その成功率は下がっている，また α が上昇するにつれてその成功率が減少する速度も早くなっている．同様の傾向は，NRBF-NRBF の学習モデルによる制御則においても確認できる．しかし，$\alpha = 0.25$ としたときの制御則の成功率の減少は $\alpha = 0.00$ としたときの制御則と比較して緩やかになっていることが確認できる．

今回の課題では実行環境と学習環境が異なることから，真のダイナミクスによる制御則も $\alpha > 0$ としたときに，NRBF-NRBF の学習モデルによる結果と同様に成功率の減少が抑制されると予測したが，そのような結果は現れなかった．その一つの理由として LMG のパラメータ α として想定する妨害入力が強すぎたためであると考えられる．即ち，最適な α は今回のシミュレーションでは取り扱わなかった $\alpha \in (0,\ 0.25)$ の中にあると考えられる．



(a) 真のダイナミクス：異なる α から得られた制御則の l_2 の変化に対する成功率の変化



(b) NRBF-NRBF モデル：異なる α から得られた制御則の l_2 の変化に対する成功率の変化

図 3.15 学習環境と実行環境が異なる場合の成功率の変化

5. 本章のまとめ

本章では，離散状態行動空間である格子状迷路課題と連続状態行動空間である振り子およびアクロボットの振り上げ課題を用いて，LMDP 及び LMG によって得られた制御則のロバスト性の評価を行った．

離散状態空間の実験ではテスト環境を意図的に学習時と変化させることで，獲得した方策のモデルの近似誤差に対するロバスト性の検証を行った．LMDP と同一の問題設定となる $\alpha = 0.00$ とした LMG から得られる制御則は，決定論的なモデルを使用したときは，近似誤差の生じない決定論的なテスト環境で最も良い性能を示したが．近似誤差が生じるとすぐに制御則の劣化がみられた．その一方で，妨害入力の強さが最大となる $\alpha = 1.00$ とした LMG から得られる制御則は，モデルの近似誤差による制御性能の劣化が抑えられていた．確率的なモデルを使用した際もまた，妨害入力の強さが最大となる $\alpha = 1.00$ とした LMG から得られた制御則は，モデルの近似誤差による制御性能の劣化が抑えられていた．しかしながら，モデルとテスト環境が同一となった場合でも LMDP と同一の問題設定となる $\alpha = 0.00$ とした LMG から得られる制御則は，必ずしも最も良い性能であるとは限らなかった．この結果は LMG および LMDP から得られる最適な Controlled probability が必ずしも与えられた状態遷移確率に従うと限らないことが原因であると考えられる．

振り子およびアクロボットの連続状態空間における実験でも LMDP と同一の問題設定となる $\alpha = 0.00$ とした LMG から得られる制御則は，大きな近似誤差が存在する学習モデルを使用した場合と真のダイナミクスを使用した場合で比較すると，近似誤差によって制御性能が劣化していることが確認された．しかしながら，LMG のパラメータ α の取り方次第では近似誤差の大きい学習モデルを使用したとしても，真のダイナミクスを使用した制御則と同程度の制御性能の制御則は得られており，近似誤差に対してロバストな制御則が獲得できることを示した．ただし，$\alpha > 0$ として妨害入力を想定したとしても必ずしも得られた制御則が近似誤差に対してロバストになるとは限らなかった．これは，LMG の妨害入力の強さを定める α にも H^∞ 制御における γ と同様に最適値や制御則を導出できる領域が存在するためであると考えられる．

このことから学習モデルが環境のダイナミクスを十分に近似できる場合には，LMDP もしくは LMDP に相当する $\alpha = 0.00$ の LMG を用いることが有効である一方で，環境のダイナミクスの近似が難しい場合には $\alpha > 0$ の LMG を用いることが有効であることを示した．

本章の実験結果から離散状態行動空間ではモデルの近似誤差による制御性能の劣化を抑えるためにはパラメータ $\alpha = 1.00$ と設定するという方針が立てられる．その一方で連続状態行動空間では適切な大きさの α を設定する必要があり，その設定方針は未だ明らかとなっていない．この問題を解決する端緒として H^∞ 制御における γ の設定方法を参考にすることが考えられる．γ の設定手法として，可解条件から γ の上限と下限を反復的に狭めていく 2 分法と呼ばれる手法が一般的に知られている [45]．しかしながら，LMG における可解条件は明らかとなっていない上に，γ に相当する係数は式 (3.15) より α だけでなくノイズの大きさ σ も含まれているため，2 分法を適用することは容易ではない．更に Belllman-Isaacs 方程式を線形化することが出来るという LMG が持つ最大の利点を考えると，2 分法のような反復的な手法でなく解析的に適切な大きさの α を導出する何らかの方法があるのではないかと考える．

第4章　本研究のまとめ

本研究では実機への適用に適した学習制御器の構築を念頭に，LMDP 及び LMG を用いた学習制御機構の提案を行った．

2 章ではダイナミクスを近似した学習モデルを用いた LMDP に基づく学習制御機構を提案し，振り子の振り上げ課題のシミュレーション実験と車輪型ロボットを用いた視角誘導課題の実験を通して，Uncontrolled probability の決定するノイズのスケールの重要性を指摘し，提案手法の実機における適用可能を示した．

3 章では LMG を用いることにより獲得される制御則がモデルの近似誤差に対してロバストになるという予測をもとに，離散状態行動空間および連続状態行動空間のシミュレーション実験を行った．その結果，LMG のパラメータである α を適切に設定することによりモデルの近似誤差があっても，その制御性能を維持することが出来ることを確認した．

以上の結果から，本研究で提案する学習モデルに用いた LMG による学習制御器は，モデルの近似誤差に対してロバストな制御則を獲得することができることから，より実機への応用が期待される手法であると結論付ける．

1. 今後の発展

LMG がモデルの近似誤差に対してロバスト性を持つとしても，十分に制御対象および環境のダイナミクスが推定できなければ，望ましい制御則を導出できない．特に本手法で用いた手法は単純な最小二乗法によるモデル学習であったこと，さらに LMDP および LMG は制御量とノイズが同一空間上でダイナミクスの線形に働くという制約を満たすならば非線形確率システムへの適用可能であることから，モデル学習の改善の余地は十分に残されている．

本手法と同様に取得したサンプルからダイナミクスの確率分布モデルを推定する手法はいくつか存在し，ガウス過程はすでに広く利用されいる [43, 7]．また杉山らが提案している密度比推定のに基づいて条件付き分布を効率的に求める手法 [50] を用いることで，現在実装されているモデル学習以上に意味のある状態遷移確率を学習が可能であると考えられる．ただし，線形化 Bellman 方程式の期待値を求める積分計算は現在のように解析的に得られなくなってしまう．これについてはモンテカルロ法を利用するなどの解決策が存在する．モデル学習の改善を考えるには，より洗練された積分計算の方法も考慮する必要がある．

価値関数を陽に求める今回の手法のままでは，次元が増えるに従い計算コストが指数的に増加してしまう [42]，その一つの理由は指数価値関数の学習点を状態空間上で一様にとってしまったことがその原因となっている．この問題点を解決するため，学習点を適応的に配置する Aggregation 法と呼ばれる手法が既に提案されていが [19]，LMG においてもこの手法が適用可能であるかについては明らかにする必要がある．

その他の拡張としてモデルフリー学習の導入がある．モデルフリー学習では明示的に環境のダイナミクスを学習する必要がなくなる．離散状態行動空間では，Z-Learning と呼ばれるモデルフリー学習が提案されており，格子状課題のシミュレーションにおいて Q-Learning よりも早く学習することが示されている [17, 8]．連続状態行動空間では，Z-Learning のような手法は確立されていないが，最小二乗法に基づく強化学習アルゴリズム [51, 52] は 将来有望な方向の一つである．しかしながら，付録 2 で説明されているように連続状態空間において境界条件を導入されていなければ，全ての重みパラメータが 0 となってしまう．更に，指数価値関数の定義 $v(\mathbf{x}) = -\log(z(\mathbf{x}))$ より，指数価値関数は必ず $0 \leq z(\mathbf{x}) \leq 1$ という不等式の制約を満たす必要がある．加えて指数価値関数は非常に小さな値になることから境界条件の設定も十分に注意する必要がある．結局のところ，学習中の価値関数は制約条件なしで学習しているにも関わらず，最適な指数価値関数を求めるには制約付き最適化手法を解かねばならない．モデルフリー学習への拡張ためにもこの問題点を解決する必要がある．

謝辞

非常に多くの方のおかげでこの論文を完成させることができました．ここに感謝の意を表します.

はじめに，博士論文の審査をしていただいた池田和司教授，小笠原司教授，銅谷賢治客員教授，吉本潤一郎客員准教授に改めて御礼を申し上げます.

ロボット実験を通した計算論的神経科学の研究を志したきっかけ，沖縄科学技術大学院大学 (OIST) を目指すきっかけ，さらに研究の道へ志すきっかけとなったのは，高校時代に拝聴した銅谷賢治客員教授の講演でした．それから 6 年の歳月を経て私を OIST 神経計算学ユニットへ暖かく迎え入れて頂き，研究全般にわたって御指導いただきました銅谷賢治客員教に深く感謝します.

池田和司教授には基幹研究室として数理情報学研究室に所属させていただいただけでなく，本論文の主査を引き受けて頂き，研究の詰めの甘さを鋭く指摘するだけでなく的確なアドバイスをして頂きました．ありがとうございます.

OIST 神経計算学ユニットの内部博士は，どんなときでも研究で抱えている問題点について一緒になって考えて頂き，常に適切なアドバイスを授けてくださいました．同じく，神経計算ユニットの研究員そして学生の方々には輪講やミーティングで大変貴重な御意見を頂きました．吉本潤一郎准教授には研究に関するアドバイスだけでなく大学の事務手続き等の様々な面でサポートして頂きました．また，伊藤真博士には生物学的な観点から多くの助言だけでなく，博士課程の 3 年間続けることができたこどもかがく教室でも大変お世話になりました. OIST 神経計算学ユニット秘書の安里さんは OIST に赴任する前のインターンシップからの足掛け 4 年半，常にお世話になりました．また，同じく OIST 神経計算学ユニット秘書の松尾さんにも JNNS2011 の若手シンポジウムの主催したときを始め様々な面でお世話になりました．数理情報学研究室秘書の谷本史さんには出張手続き

でいつもお世話になりました．また，OIST 学生支援セクションの方々にも OIST で特別研究学生として活動するために, 様々な事務手続き等をして頂きお世話になりました. OIST 地域連携セクションの方々は様々な形で，地域への貢献活動, 特に学生への講演という貴重な機会を与えて頂きました．改めて，沖縄科学技術大学院大学神経計算学ユニットの皆様，沖縄科学技術大学院大学スタッフの皆様, そして数理情報学研究室の皆様に感謝いたします.

最後に，高校時代からの憧れだった OIST の研究室に所属できたことを誰よりも喜び，OIST に居た 4 年間をはじめ，陰に陽に常に私を支え続けてくれた両親に感謝します.

付録

1. 数学記号

$\mathrm{KL}\left(p(\mathbf{x})\|q(\mathbf{x})\right)$ は，二つの確率分布 $p(\mathbf{x})$ と $q(\mathbf{x})$ の Kullback—Leibler divergence を表し，

$$\begin{aligned}\mathrm{KL}\left(p(\mathbf{x})\|q(\mathbf{x})\right) &= \mathbb{E}_{p(\mathbf{x})}\left[\log\left(\frac{p(\mathbf{x})}{q(\mathbf{x})}\right)\right] \\ &= \int p(\mathbf{x})\log\left(\frac{p(\mathbf{x})}{q(\mathbf{x})}\right)d\mathbf{x} \\ &= \int p(\mathbf{x})\log p(\mathbf{x})d\mathbf{x} - \int p(\mathbf{x})\log q(\mathbf{x})d\mathbf{x} \\ &= -\mathbb{H}[p(\mathbf{x})] - \mathbb{E}_{p(\mathbf{x})}[\log q(\mathbf{x})]\end{aligned}$$

と定義されている．ここで $\mathbb{H}[p(\mathbf{x})]$ は $p(\mathbf{x})$ のエントロピーを表している。

$\mathcal{D}_\beta\left(p(\mathbf{x})\|q(\mathbf{x})\right)$ は，二つの確率分布 $p(\mathbf{x})$ と $q(\mathbf{x})$ の Rényi-divergence を表し，

$$\begin{aligned}\mathcal{D}_\beta\left(p(\mathbf{x})\|q(\mathbf{x})\right) &= \frac{\beta}{\beta-1}\log\left(\mathbb{E}_{p(\mathbf{x})}\left[\left(\frac{q(\mathbf{x})}{p(\mathbf{x})}\right)^{1-\beta}\right]\right) \\ &= \frac{\beta}{\beta-1}\log\left(\int p(\mathbf{x})^\beta q(\mathbf{x})^{1-\beta}d\mathbf{x}\right)\end{aligned}$$

と定義されている．

また KL-divegence と Rényi-divergence の間には以下のように関係が成り立っている．

$$\lim_{\beta\to 0}\frac{1}{\beta}\mathcal{D}_\beta\left(p(\mathbf{x})\|q(\mathbf{x})\right) = \mathrm{KL}\left(q(\mathbf{x})\|p(\mathbf{x})\right)$$

最後に Ψ は以下のような操作を行う演算子である.

$$\Psi^{\theta}_{p(x)}[f(\mathbf{x})] = \frac{1}{\theta}\log\left(\mathbb{E}_{p(\mathbf{x})}\left[\exp(\theta f(\mathbf{x}))\right]\right)$$

2. 指数価値関数の関数近似パラメータの最適化

2.1 $\boldsymbol{w}$ に関する最小化

コスト関数が非負であれば価値関数 $v(\mathbf{x})$ も非負である。よって式 (2.7) の定義より、すべての $\mathbf{x}$ に対して $0 < \hat{z}(\mathbf{x}) \leq 1$ を満足しなければならない。いま基底関数は非正規化ガウス関数 (2.12) と仮定しているので、$0 < f_i(\mathbf{x}) \leq 1$ である。よって $w_i \geq 0$ であれば $\hat{z}(\mathbf{x})$ の不等式制約を満足する。以上より問題は次の二次計画問題

$$\min_{\boldsymbol{w}} e, \quad \text{s.t.} \quad w_i \geq 0,\ \forall i \tag{4.1}$$

に帰着できる。

2.2 θ に関する最小化

基底関数のパラメータに関しても式 (2.19) の二乗誤差を Levenberg-Marquardt 法を用いて最小化することで求めることができる。$\boldsymbol{r}$ の $\boldsymbol{\theta}$ に関するヤコビアンを

$$\boldsymbol{J}(\boldsymbol{\theta}, \boldsymbol{w}) = \partial \boldsymbol{r}(\boldsymbol{\theta}, \boldsymbol{w})/\partial \boldsymbol{\theta}$$

とすると、Levenberg-Marquardt 法による制約なしのパラメータ $\boldsymbol{\theta}$ の更新量 $\boldsymbol{\delta}$ は

$$(\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J} + \gamma \boldsymbol{I})\boldsymbol{\delta} = -\boldsymbol{J}^{\mathrm{T}}\boldsymbol{r} \tag{4.2}$$

を満足する。ここで $\boldsymbol{I}$ は単位行列、γ は Levenberg-Marquardt 法のパラメータである。

ただし式 (4.2) を満足する $\boldsymbol{\delta}$ を使って更新すると、節点 $\mathbf{x}_n$ から離れるように $\boldsymbol{m}_i$ を修正することで e を最小化するために、$\hat{z}(\mathbf{x}_n; \boldsymbol{\theta}, \boldsymbol{w}) \approx 0$ となってしまうことが

報告されている。そこで節点での指数価値関数の値が一定であるという条件

$$\mathbf{1}^{\mathrm{T}}\boldsymbol{F}(\boldsymbol{\theta})\boldsymbol{w} = \sum_{n=1} \hat{z}(\mathbf{x}_n; \boldsymbol{\theta}, \boldsymbol{w}) = \mathrm{const} \tag{4.3}$$

を導入する。これは式 (4.3) の $\boldsymbol{\theta}$ に関する勾配方向 $\boldsymbol{g} = \partial(\mathbf{1}^{\mathrm{T}}\boldsymbol{F}\boldsymbol{w})/\partial\boldsymbol{\theta}$ と直交するように $\boldsymbol{\delta}$ を修正することで実現できる。$\boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{g} = 0$ を制約として Levenberg-Marquardt 法を構成すると、

$$\min_{\delta} \frac{1}{2}\boldsymbol{\delta}^{\mathrm{T}}(\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J} + \gamma\boldsymbol{I})\boldsymbol{\delta} + \boldsymbol{\delta}^{\mathrm{T}}\boldsymbol{J}^{\mathrm{T}}\boldsymbol{r} \qquad \text{s.t.} \quad \boldsymbol{g}^{\mathrm{T}}\boldsymbol{\delta} = 0 \tag{4.4}$$

という制約付きの $\boldsymbol{\delta}$ に関する最小化問題を得る。ラグランジュ未定乗数法を用いることで以下のような $\boldsymbol{\delta}$ の唯一解を得られる連立方程式が得られる。

$$\begin{bmatrix} \boldsymbol{J}^{\mathrm{T}}\boldsymbol{J} + \gamma\boldsymbol{I} & \boldsymbol{g} \\ \boldsymbol{g}^{\mathrm{T}} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\delta} \\ \nu \end{bmatrix} = \begin{bmatrix} \boldsymbol{J}^{\mathrm{T}}\boldsymbol{r} \\ 0 \end{bmatrix} \tag{4.5}$$

これにより $\boldsymbol{\theta}$ の更新は収束を待つことなく一回で終了できる。

$\boldsymbol{w}$ に関する最小化と合わせて、$\boldsymbol{\theta}$ の最小化を二乗誤差が収束するまで行うことで、式 (2.15) を満たすような指数価値関数の近似関数に必要な全てのパラメータの最適化を行うことが可能となる。

3. Bellman-Isaac 方程式の線形化

ここでは，LMG による Bellman-Issac の線形化の詳細について記述する．節 1.1 にて定義されたコスト関数を式 (3.1) の Bellman-Issac 方程式へ代入する．

$$\begin{aligned} v(\mathbf{x}) \;=\; & \min_{\mathbf{u}^c} \max_{\mathbf{u}^a} \{ \ell(\mathbf{x}) + \frac{1}{\alpha}\mathcal{D}_\alpha \left(p^0(\mathbf{x}' \mid \mathbf{x}) \| g^{\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x}) \otimes p^0(\mathbf{x}' \mid \mathbf{x}) \right) \\ & -\frac{1}{\alpha}\mathrm{KL}\left(g^{\mathbf{u}^a}(\mathbf{x}' \mid \mathbf{x}, \mathbf{u}^c) \otimes g^{\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x}) \otimes p^0(\mathbf{x}' \mid \mathbf{x}) \| g^{\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x}) \otimes p^0(\mathbf{x}' \mid \mathbf{x}) \right) \\ & +\mathbb{E}_{\mathbf{x}' \sim g^{\mathbf{u}^a}(\mathbf{x}' \mid \mathbf{x}, \mathbf{u}^c) \otimes g^{\mathbf{u}^c}(\mathbf{x}' \mid \mathbf{x}) \otimes p^0(\mathbf{x}' \mid \mathbf{x})} [v(\mathbf{x}')] \} \\ \Leftrightarrow \alpha v(\mathbf{x}) \;=\; & \min_{\mathbf{u}_c} \max_{\mathbf{u}_a} \{ \alpha\ell(\mathbf{x}) + \mathcal{D}_\alpha \left(p^0(\mathbf{x}' \mid \mathbf{x}) \| g^{\mathbf{u}_c}(\mathbf{x}' \mid \mathbf{x}) \otimes p^0(\mathbf{x}' \mid \mathbf{x}) \right) \\ & -\mathrm{KL}\left(g^{\mathbf{u}_a}(\mathbf{x}' \mid \mathbf{x}, \mathbf{u}_c) \otimes g^{\mathbf{u}_c}(\mathbf{x}' \mid \mathbf{x}) \otimes p^0(\mathbf{x}' \mid \mathbf{x}) \| g^{\mathbf{u}_c}(\mathbf{x}' \mid \mathbf{x}) \otimes p^0(\mathbf{x}' \mid \mathbf{x}) \right) \\ & +\mathbb{E}_{\mathbf{x}' \sim g^{\mathbf{u}_a}(\mathbf{x}' \mid \mathbf{x}, \mathbf{u}_c) \otimes g^{\mathbf{u}_c}(\mathbf{x}' \mid \mathbf{x}) \otimes p^0(\mathbf{x}' \mid \mathbf{x})} [\alpha v(\mathbf{x}')] \} \end{aligned}$$

ここではまず $\mathbf{u}^a$ に関する最大化を行う．付録 3.1 の定理 1 より以下の関係式が導かれ，最大化演算子の中の上限が求められる．

$$-\mathrm{KL}\left(g^{\mathbf{u}^a}(\mathbf{x}'\mid\mathbf{x},\mathbf{u}^c)\otimes p^{\{\mathbf{u}^a,\mathbf{u}^c\}}(\mathbf{x}'\mid\mathbf{x})\|p^{\{\mathbf{u}^a,\mathbf{u}^c\}}(\mathbf{x}'\mid\mathbf{x})\right)+\mathbb{E}_{\mathbf{x}'\sim g^{\mathbf{u}^a}(\mathbf{x}'|\mathbf{x},\mathbf{u}^c)\otimes p^{\{\mathbf{u}^a,\mathbf{u}^c\}}(\mathbf{x}'|\mathbf{x})}\left[\alpha v(\mathbf{x}')\right]$$
$$\leq \log\left(\mathbb{E}_{\mathbf{x}'\sim p^{\{\mathbf{u}^a,\mathbf{u}^c\}}(\mathbf{x}'|\mathbf{x})}\left[\exp(\alpha v(\mathbf{x}'))\right]\right)$$

また等号条件より，$g^{\mathbf{u}^a}(\mathbf{x}'\mid\mathbf{x},\mathbf{u}^c)=\exp(\alpha v(\mathbf{x}'))$ となる．

次に $\mathbf{u}^c$ に関する最小化を行う．付録 3.1 の定理 3 より以下の関係式が導かれ最小演算子の下限が求められる．

$$\mathcal{D}_\alpha\left(p^0(\mathbf{x}'\mid\mathbf{x})\|g^{\mathbf{u}^c}(\mathbf{x}'\mid\mathbf{x})\otimes p^0(\mathbf{x}'\mid\mathbf{x})\right)+\log\left(\mathbb{E}_{\mathbf{x}'\sim g^{\mathbf{u}^c}(\mathbf{x}'|\mathbf{x})\otimes p^0(\mathbf{x}'|\mathbf{x})}\left[\exp(\alpha v(\mathbf{x}'))\right]\right)\}$$
$$\geq \Psi^{\frac{\alpha-1}{\alpha}}_{p^0(\mathbf{x}'|\mathbf{x})}\left[\alpha v(\mathbf{x}')\right]$$

また等号条件より，$g^{\mathbf{u}^c}(\mathbf{x}'\mid\mathbf{x})=\exp(-v(\mathbf{x}'))$ となる．以上の結果より Bellman-Issac 方程式は,

$$\begin{aligned}
\alpha v(\mathbf{x}) &= \alpha\ell(\mathbf{x})+\Psi^{\frac{\alpha-1}{\alpha}}_{p^0(\mathbf{x}'|\mathbf{x})}[\alpha v(\mathbf{x}')] \\
&= \alpha\ell(\mathbf{x})+\frac{\alpha}{\alpha-1}\log\left(\mathbb{E}_{p^0(\mathbf{x}'|\mathbf{x})}\left[\exp\left(\frac{\alpha-1}{\alpha}\alpha v(\mathbf{x}')\right)\right]\right) \\
\Leftrightarrow z(\mathbf{x})^\alpha &= \exp((\alpha-1)\ell(\mathbf{x}))\mathbb{E}_{p^0(\mathbf{x}'|\mathbf{x})}\left[z(\mathbf{x})^\alpha\right]
\end{aligned}$$

であることが示される．

3.1 定理の諸証明

定理 1

$$\mathrm{KL}\left(g(\mathbf{s})\otimes p(\mathbf{s})\|p(\mathbf{s})\right)-\mathbb{E}_{\mathbf{s}\sim g(\mathbf{s})\otimes p(\mathbf{s})}\left[\theta f(\mathbf{s})\right]\geq-\log\left(\mathbb{E}_{p(\mathbf{s})}\left[\exp\left(\theta f(\mathbf{s})\right)\right]\right)\quad(4.6)$$

等号は $g(\mathbf{s})=\exp\left(\theta f(\mathbf{s})\right)$ のときに成立する．

証明 1

$$\begin{aligned}
\mathrm{KL}\left(g(\mathbf{s})\otimes p(\mathbf{s})\|p(\mathbf{s})\right)-\mathbb{E}_{\mathbf{s}\sim g(\mathbf{s})\otimes p(\mathbf{s})}\left[\theta f(\mathbf{s})\right] &= \mathbb{E}_{g(\mathbf{s})\otimes p(\mathbf{s})}\left[\log\left(\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}\right)-\theta f(\mathbf{s})\right] \\
&= \mathbb{E}_{g(\mathbf{s})\otimes p(\mathbf{s})}\left[-\log\left(\frac{p(\mathbf{s})}{g(\mathbf{s})\otimes p(\mathbf{s})}\exp\left(\theta f(\mathbf{s})\right)\right)\right] \\
&\geq -\log\left(\mathbb{E}_{g(\mathbf{s})\otimes p(\mathbf{s})}\left[\frac{p(\mathbf{s})}{g(\mathbf{s})\otimes p(\mathbf{s})}\exp\left(\theta f(\mathbf{s})\right)\right]\right) \\
&= -\log\left(\mathbb{E}_{p(\mathbf{s})}\left[\exp\left(\theta f(\mathbf{s})\right)\right]\right)
\end{aligned}$$

二段目から三段目への不等式の *Jensen* の不等式より導出した。等号の成立条件は、

$$\begin{aligned}
&\frac{p(\mathbf{s})}{g(\mathbf{s})\otimes p(\mathbf{s})}\exp\left(\theta f(\mathbf{s})\right)=c \\
\Leftrightarrow\quad & p(\mathbf{s})\exp\left(\theta f(\mathbf{s})\right)=c\frac{g(\mathbf{s})p(\mathbf{s})}{\int g(\mathbf{s})p(\mathbf{s})d\mathbf{s}} \\
\Rightarrow\quad & \exp\left(\theta f(\mathbf{s})\right)=g(\mathbf{s})
\end{aligned}$$

となる。また以下のような関係式が成り立っている.

$$\log\left(\mathbb{E}_{p(\mathbf{s})}\left[\exp\left(\theta f(\mathbf{s})\right)\right]\right)=\theta\Psi^{\theta}_{p(\mathbf{s})}[f(\mathbf{s})]=\Psi_{p(\mathbf{s})}[\theta f(\mathbf{s})]$$

定理 2

$$\mathrm{KL}\left(p(\mathbf{s})\|g(\mathbf{s})\otimes p(\mathbf{s})\right)+\Psi_{g(\mathbf{s})\otimes p(\mathbf{s})}[\theta f(\mathbf{s})]\geq\mathbb{E}_{p(\mathbf{s})}[\theta f(\mathbf{s})] \tag{4.7}$$

等号は $g(\mathbf{s})=\exp\left(-\theta f(\mathbf{s})\right)$ のときに成立する.

証明 2

$$\begin{aligned}
\Psi_{g(\mathbf{s})\otimes p(\mathbf{s})}[\theta f(\mathbf{s})]=\log\left(\mathbb{E}_{g(\mathbf{s})\otimes p(\mathbf{s})}\left[\exp\left(\theta f(\mathbf{s})\right)\right]\right) &= \log\left(\mathbb{E}_{p(\mathbf{s})}\left[\frac{\left(g(\mathbf{s})\otimes p(\mathbf{s})\right)\exp\left(\theta f(\mathbf{s})\right)}{p(\mathbf{s})}\right]\right) \\
&\geq \mathbb{E}_{p(\mathbf{s})}\left[\log\left(\frac{\left(g(\mathbf{s})\otimes p(\mathbf{s})\right)\exp\left(\theta f(\mathbf{s})\right)}{p(\mathbf{s})}\right)\right] \\
&= \mathbb{E}_{p(\mathbf{s})}\left[-\log\left(\frac{p(\mathbf{s})}{g(\mathbf{s})\otimes p(\mathbf{s})}\right)+\theta f(\mathbf{s})\right] \\
&= -\mathrm{KL}\left(p(\mathbf{s})\|g(\mathbf{s})\otimes p(\mathbf{s})\right)+\mathbb{E}_{p(\mathbf{s})}\left[\theta f(\mathbf{s})\right] \\
\Leftrightarrow \mathrm{KL}\left(p(\mathbf{s})\|g(\mathbf{s})\otimes p(\mathbf{s})\right)+\Psi_{g(\mathbf{s})\otimes p(\mathbf{s})}[\theta f(\mathbf{s})] &\geq \mathbb{E}_{p(\mathbf{s})}\left[\theta f(\mathbf{s})\right]
\end{aligned}$$

二段目の不等号は *Jensen* の不等式より等号条件は以下のようになる．

$$\begin{aligned} & \frac{(g(\mathbf{s}) \otimes p(\mathbf{s})) \exp(\theta f(\mathbf{s}))}{p(\mathbf{s})} = c \\ \Leftrightarrow \quad & p(\mathbf{s}) \exp(-\theta f(\mathbf{s})) = \frac{g(\mathbf{s}) p(\mathbf{s})}{c \int g(\mathbf{s}) p(\mathbf{s}) d\mathbf{s}} \\ \Rightarrow \quad & \exp(-\theta f(\mathbf{s})) = g(\mathbf{s}) \end{aligned}$$

定理 3

$$\mathcal{D}_\beta(p(\mathbf{s}) \| g(\mathbf{s}) \otimes p(\mathbf{s})) + \Psi_{g(\mathbf{s}) \otimes p(\mathbf{s})}[\theta f(\mathbf{s})] \geq \Psi_{p(\mathbf{s})}^{\frac{\beta-1}{\beta}}[\theta f(\mathbf{s})] \tag{4.8}$$

等号は $g(\mathbf{s}) = \exp\left(-\frac{\theta f(\mathbf{s})}{\beta}\right)$ のときに成立する．

証明 3 付録 *1* の *Rényi-divergence* の定義より

$$\begin{aligned} \mathcal{D}_\beta(p(\mathbf{s}) \| g(\mathbf{s}) \otimes p(\mathbf{s})) &= \frac{1}{\beta - 1} \log \left(\mathbb{E}_{p(\mathbf{s})} \left[\left(\frac{g(\mathbf{s}) \otimes p(\mathbf{s})}{p(\mathbf{s})} \right)^{1-\beta} \right] \right) \\ &= \log \left(\left(\mathbb{E}_{p(\mathbf{s})} \left[\left(\frac{g(\mathbf{s}) \otimes p(\mathbf{s})}{p(\mathbf{s})} \right)^{1-\beta} \right] \right)^{\frac{1}{\beta-1}} \right) \end{aligned}$$

となる．さらに付録 *1* における演算子 Ψ の定義を参照すると，左辺は

$$(\text{左辺}) = \log \left(\left(\mathbb{E}_{p(\mathbf{s})} \left[\left(\frac{g(\mathbf{s}) \otimes p(\mathbf{s})}{p(\mathbf{s})} \right)^{1-\beta} \right] \right)^{\frac{1}{\beta-1}} \mathbb{E}_{g(\mathbf{s}) \otimes p(\mathbf{s})} [\exp(\theta f(\mathbf{s}))] \right)$$

となる．ここで,

$$\left(\mathbb{E}_{p(\mathbf{s})} \left[\left(\frac{g(\mathbf{s}) \otimes p(\mathbf{s})}{p(\mathbf{s})} \right)^{1-\beta} \right] \right)^{\frac{1}{\beta-1}} \mathbb{E}_{g(\mathbf{s}) \otimes p(\mathbf{s})} [\exp(\theta f(\mathbf{s}))] \geq \left(\mathbb{E}_{p(\mathbf{s})} \left[\exp(\theta f(\mathbf{s}))^{\frac{\beta-1}{\beta}} \right] \right)^{\frac{\beta}{\beta-1}} \tag{4.9}$$

であることから，両辺の対数を取ることで定理 *3* が導出される．

式 *(4.9)* は以下のように導出される，ここでは簡単のため $h(\mathbf{s}) = \exp(\theta f(\mathbf{s}))$ と

している．

$$
\begin{aligned}
&\left(\mathbb{E}_{p(\mathbf{s})}\left[\left(\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}\right)^{1-\beta}\right]\right)^{\frac{1}{1-\beta}} \\
&\quad=\left(\mathbb{E}_{h(\mathbf{s})^{\frac{\beta-1}{\beta}}\otimes p(\mathbf{s})}\left[h(\mathbf{s})^{\frac{1-\beta}{\beta}}\left(\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}\right)^{1-\beta}\right]\right)^{\frac{1}{1-\beta}}\left(\mathbb{E}_{p(\mathbf{s})}\left[h(\mathbf{s})^{\frac{\beta-1}{\beta}}\right]\right)^{\frac{1}{1-\beta}} \\
&\quad\leq \mathbb{E}_{h(\mathbf{s})^{\frac{\beta-1}{\beta}}\otimes p(\mathbf{s})}\left[\left(h(\mathbf{s})^{\frac{1-\beta}{\beta}}\left(\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}\right)^{1-\beta}\right)^{\frac{1}{1-\beta}}\right]\left(\mathbb{E}_{p(\mathbf{s})}\left[h(\mathbf{s})^{\frac{\beta-1}{\beta}}\right]\right)^{\frac{1}{1-\beta}} \\
&\quad=\mathbb{E}_{h(\mathbf{s})^{\frac{\beta-1}{\beta}}\otimes p(\mathbf{s})}\left[h(\mathbf{s})^{\frac{1}{\beta}}\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}\right]\left(\mathbb{E}_{p(\mathbf{s})}\left[h(\mathbf{s})^{\frac{\beta-1}{\beta}}\right]\right)^{\frac{1}{1-\beta}} \\
&\quad=\mathbb{E}_{p(\mathbf{s})}\left[h(\mathbf{s})\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}\right]\left(\mathbb{E}_{p(\mathbf{s})}\left[h(\mathbf{s})^{\frac{\beta-1}{\beta}}\right]\right)^{\frac{\beta}{1-\beta}} \\
\Leftrightarrow\quad &\left(\mathbb{E}_{p(\mathbf{s})}\left[\left(\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}\right)^{1-\beta}\right]\right)^{\frac{1}{1-\beta}}\leq \mathbb{E}_{g(\mathbf{s})\otimes p(\mathbf{s})}\left[h(\mathbf{s})\right]\left(\mathbb{E}_{p(\mathbf{s})}\left[h(\mathbf{s})^{\frac{\beta-1}{\beta}}\right]\right)^{\frac{\beta}{1-\beta}} \\
\Leftrightarrow\ &\left(\mathbb{E}_{p(\mathbf{s})}\left[h(\mathbf{s})^{\frac{\beta-1}{\beta}}\right]\right)^{\frac{\beta}{\beta-1}}\leq \mathbb{E}_{g(\mathbf{s})\otimes p(\mathbf{s})}\left[h(\mathbf{s})\right]\left(\mathbb{E}_{p(\mathbf{s})}\left[\left(\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}\right)^{1-\beta}\right]\right)^{\frac{1}{\beta-1}}
\end{aligned}
$$

二段目の変形は *Jensen* の不等式を用いて導出し，等号の成立条件は

$$
\begin{aligned}
h(\mathbf{s})^{\frac{1-\beta}{\beta}}\left(\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}\right)^{1-\beta} &= c \\
\Leftrightarrow g(\mathbf{s})\otimes p(\mathbf{s}) &\propto p(\mathbf{s})h(\mathbf{s})^{-\frac{1}{\beta}} \\
\Leftrightarrow g(\mathbf{s}) &= \exp(-\frac{\theta f(\mathbf{s})}{\beta})
\end{aligned}
$$

となる．また一段目と三段目での変形は以下のようにして得られる．

$$
\begin{aligned}
&\left(\mathbb{E}_{h(\mathbf{s})^{\frac{\beta-1}{\beta}}\otimes p(\mathbf{s})}\left[h(\mathbf{s})^{\frac{1-\beta}{\beta}}\left(\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}\right)^{1-\beta}\right]\right)^{\frac{1}{1-\beta}} \\
&\quad=\left(\int\frac{h(\mathbf{s})^{\frac{\beta-1}{\beta}}p(\mathbf{s})}{\int h(\mathbf{s}')^{\frac{\beta-1}{\beta}}p(\mathbf{s}')d\mathbf{s}'}h(\mathbf{s})^{\frac{1-\beta}{\beta}}\left(\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}\right)^{1-\beta}\right)^{\frac{1}{1-\beta}} \\
&\qquad=\left(\frac{\int p(\mathbf{s})\left(\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}\right)^{1-\beta}}{\int h(\mathbf{s}')^{\frac{\beta-1}{\beta}}p(\mathbf{s}')d\mathbf{s}'}\right)^{\frac{1}{1-\beta}}=\frac{\left(\mathbb{E}_{p(\mathbf{s})}\left[\left(\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}\right)^{1-\beta}\right]\right)^{\frac{1}{1-\beta}}}{\left(\mathbb{E}_{p(\mathbf{s})}\left[h(\mathbf{s})^{\frac{\beta-1}{\beta}}\right]\right)^{\frac{1}{1-\beta}}} \\
&\qquad\Leftrightarrow\left(\mathbb{E}_{p(\mathbf{s})}\left[\left(\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}\right)^{1-\beta}\right]\right)^{\frac{1}{1-\beta}} \\
&\qquad=\left(\mathbb{E}_{p(\mathbf{s})}\left[h(\mathbf{s})^{\frac{\beta-1}{\beta}}\right]\right)^{\frac{1}{1-\beta}}\left(\mathbb{E}_{h(\mathbf{s})^{\frac{\beta-1}{\beta}}\otimes p(\mathbf{s})}\left[h(\mathbf{s})^{\frac{1-\beta}{\beta}}\left(\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}\right)^{1-\beta}\right]\right)^{\frac{1}{1-\beta}}
\end{aligned}
$$

$$
\begin{aligned}
\mathbb{E}_{h(\mathbf{s})^{\frac{\beta-1}{\beta}}\otimes p(\mathbf{s})}\left[h(\mathbf{s})^{\frac{1}{\beta}}\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}\right] &= \int\frac{h(\mathbf{s})^{\frac{\beta-1}{\beta}}p(\mathbf{s})}{\int h(\mathbf{s}')^{\frac{\beta-1}{\beta}}p(\mathbf{s}')d\mathbf{s}'}h(\mathbf{s})^{\frac{1}{\beta}}\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}d\mathbf{s} \\
&= \frac{\int p(\mathbf{s})h(\mathbf{s})^{\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}}d\mathbf{s}}{\int h(\mathbf{s}')^{\frac{\beta-1}{\beta}}p(\mathbf{s}')d\mathbf{s}'} \\
&= \mathbb{E}_{p(\mathbf{s})}\left[h(\mathbf{s})\frac{g(\mathbf{s})\otimes p(\mathbf{s})}{p(\mathbf{s})}\right]\left(\mathbb{E}_{p(\mathbf{s})}\left[h(\mathbf{s})^{\frac{\beta-1}{\beta}}\right]\right)^{-1}
\end{aligned}
$$

4. アクロボットのダイナミクス詳細

アクロボットの状態 $\mathbf{x}$ は，リンク 1 の角度 q_1, 角速度 $\dot{q}_1$，リンク 2 の角度 q_2, 角速度 $\dot{q}_2$ から構成された 4 次元のベクトルで表現される．その状態方程式は以下のように,

$$
\begin{aligned}
\dot{\mathbf{x}} &= (\boldsymbol{a}(\mathbf{x}) + \boldsymbol{b}(\mathbf{x})u)dt, \\
\mathbf{x} &= \begin{bmatrix} q_1 & q_2 & \dot{q}_1 & \dot{q}_2 \end{bmatrix}^{\mathrm{T}}
\end{aligned}
\tag{4.10}
$$

状態依存の状態遷移 $\boldsymbol{a}(\mathbf{x})$ と制御入力に線形な状態遷移 $\boldsymbol{b}(\mathbf{x})u$ の 2 つ和として表現される.

それぞれ $\boldsymbol{a}(\mathbf{x})$ と $\boldsymbol{b}(\mathbf{x})$ の詳細は,

$$\boldsymbol{a}(\mathbf{x}) = \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \frac{-d_{22}(h_1+\phi_1+k_1q_1)+d_{12}(h_2+\phi_2+k_2q_2)}{d_{11}d_{22}-d_{12}^2} \\ \frac{-d_{12}(h_1+\phi_1+k_1q_1)+d_{11}(h_2+\phi_2+k_2q_2)}{d_{11}d_{22}-d_{12}^2} \end{bmatrix}, \quad \boldsymbol{b}(\mathbf{x}) = \frac{1}{d_{11}d_{22}-d_{12}^2} \begin{bmatrix} 0 \\ 0 \\ -d_{12} \\ d_{11} \end{bmatrix} \tag{4.11}$$

$$\begin{aligned} d_{11} &= m_1 l_{c1}^2 + m_2 \left(l_1^2 + l_{c2}^2 + 2 l_1 l_{c2} \cos(q_2) \right) + I_1 + I_2, \\ d_{12} &= m_2 \left(l_{c2}^2 + l_1 l_{c2} \cos(q_2) \right) + I_2, \\ d_{22} &= m_2 l_{c2}^2 + I_2, \\ h_1 &= -m_2 l_1 l_{c2} \left(\sin(q_2) \dot{q}_2^2 + 2 \sin(q_2) \dot{q}_1 \dot{q}_2 \right), \\ h_2 &= -m_2 l_1 l_{c2} \sin(q_2) \dot{q}_1^2, \\ \phi_1 &= -(m_1 l_{c1} + m_2 l_1) g \sin(q_1) - m_2 l_{c2} g \sin(q_1 + q_2), \\ \phi_2 &= -m_2 l_{c2} g \sin(q_1 + q_2) \end{aligned}$$

のようになっている.

参考文献

参考文献

[1] E. Todorov. Optimal control theory. In D. Kenji, S. Ishii, A. Pouget, and R. P. Rao, editors, *Bayesian Brain: Probabilistic Approaches to Neural Coding*, chapter 12, pp. 269–298. MIT Press, Cambridge, MA, 2006.

[2] R. S. Sutton and A. G. Barto. *Reinforcement Learning*. MIT Press, Cambridge, MA, 1998.

[3] Y. Niv. Reinforcement learning in the brain. *Journal of Mathematical Psychology*, Vol. 53, No. 3, pp. 139–154, 2009.

[4] N. D. Daw, S. J. Gershman, B. Seymour, P. Dayan, and R. J. Dolan. Model-based influences on humans' choices and striatal prediction errors. *Neuron*, Vol. 69, No. 6, pp. 1204–1215, 2011.

[5] B. B. Doll, D. A. Simon, and N. D. Daw. The ubiquity of model-based reinforcement learning. *Current Opinion in Neurobiology*, Vol. 22, No. 6, pp. 1075–1081, 2012.

[6] C. G. Atkesonand A. W. Mooreand S. Schaal. Locally weighted learning for control. *Artificial intelligence review*, Vol. 11, No. 1-5, pp. 75–113, 1997.

[7] M. P. Deisenroth and C. E. Rasmussen. PILCO: A Model-Based and Data-Efficient Approach to Policy Search. In *Proceedings of the 28th International Conference on Machine Learning*, Bellevue, WA, 2011.

[8] E. Todorov. Efficient computation of optimal actions. *Proc. Natl. Acad. Sci.*, Vol. 106, No. 28, pp. 11478–11483, 2009.

[9] K. Doya. How can we learn efficiently to act optimally and flexibly? *Proc. Natl. Acad. Sci.*, Vol. 106, No. 28, pp. 11429–30, 2009.

[10] E. Theodorou and E. Todorov. Relative entropy and free energy dualities: Connections to path integral and kl control. In *the 51th IEEE Conference on Decision and Control*, pp. 1466–1473, Maui, Hawaii, USA, 2012.

[11] M. da Silva, F. Durand, and J. Popović. Linear bellman combination for control of character animation. *ACM Trans. Graph.*, Vol. 28, No. 3, pp. 82:1–82:10, July 2009.

[12] H. Kappen. Linear theory for control of nonlinear stochastic systems. *Physical Review Letters*, Vol. 95, pp. 200201–4, 2005.

[13] H. Kappen. Path integrals and symmetry breaking for optimal control theory. *Journal of Statistical Mechanics: Theory and Experiment*, Vol. 11, p. 11011, 2005.

[14] E. Theodorou, J. Buchli, and S. Schaal. A generalized path integral control approach to reinforcement learning. *Journal of Machine Learning Research*, Vol. 11, pp. 3137–3181, 2010.

[15] F. Stulp and O. Sigaud. Path Integral Policy Improvement with Covariance Matrix Adaptation. In *Proc. of the 10th European Workshop on Reinforcement Learning (EWRL 2012)*, Edinburgh, UK, 2012.

[16] N. Sugimoto and J. Morimoto. Phase-dependent trajectory optimization for periodic movement using path integral reinforcement learning. In *Proc. of the 21st Annual Conference of the Japanese Neural Network Society*, Okinawa, Jpapan, 2011.

[17] E. Todorov. Linearly-solvable Markov decision problems. In *Advances in Neural Information Processing Systems 19*, pp. 1369–1376. MIT Press, 2007.

[18] E. Todorov. Eigenfunction approximation methods for linearly-solvable optimal control problems. In *Proc. of the 2nd IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, pp. 161–168, Nashville, TN, USA, 2009.

[19] M. Zhong and E. Todorov. Aggregation Methods for Lineary-solvable Markov Decision Process. In *Proc. of the World Congress of the International Federation of Automatic Control*, Milano, Italy, 2011.

[20] E. Todorov. Compositionality of optimal control laws. In *Advances in Neural Information Processing Systems 22*, pp. 1856–1864. MIT Press, 2009.

[21] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, p. 1. ACM, 2004.

[22] A. Y. Ng and S. J. Russell. Algorithms for inverse reinforcement learning. In *Proc. of the 17th International Conference on Machine Learning*, pp. 663–670, Stanford, California, 2000.

[23] K. Dvijotham and E. Todorov. Inverse optimal control with linearly solvable MDPs. In *Proc. of the 27th International Conference on Machine Learning*, 2010.

[24] T. Başar and P. Bernhard. *H-infinity optimal control and related minimax design problems: a dynamic game approach*. Springer, 2008.

[25] P. Gahinet and P. Apkarian. A linear matrix inequality approach to H^∞ control. *International journal of robust and nonlinear control*, Vol. 4, No. 4, pp. 421–448, 1994.

[26] J. Morimotoand K. Doya. Robust reinforcement learning. *Neural computation*, Vol. 17, No. 2, pp. 335–359, 2005.

[27] L. S. Shapley. Stochastic games. *Proceedings of the National Academy of Sciences of the United States of America*, Vol. 39, No. 10, p. 1095, 1953.

[28] M. L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proc. of the 11th International Conference on Machine Learning*, Vol. 94, pp. 157–163, 1994.

[29] I. M. Steven, F. G. Emmanuel, H. H. Daniel, C. Stefano, and F. Pedram. Risk sensitive markov decision processes. *Progress in Systems and Control Theory*, Vol. 22, pp. 263–280, 1997.

[30] K. Dvijotham and E. Todorov. A unifying framework for linearly solvable control. In *Proceedings of the Twenty-Seventh Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-11)*, pp. 179–186, Corvallis, Oregon, 2011. AUAI Press.

[31] H. H. Fleming and W. M. McEneaney. *Risk sensitive optimal control and differential games*. Springer, 1992.

[32] M. A. P. Burdelis and K. Ikeda. Estimating passive dynamics distributions in linearly solvable markov decision processes from measured immediate costs in reinforcement learning problems. *SICE Jounal of Control,Measument,and System Integration*, Vol. 7, No. 1, pp. 48–54, 2014.

[33] D. Nguyen-Tuong and J. Peters. Model learning for robot control: a survey. *Cognitive Processing*, Vol. 12, No. 4, pp. 319–40, 2011.

[34] O. Sigaud, C. Salaün, and V. Padois. On-line regression algorithms for learning mechanical models of robots: A survey. *Robotics and Autonomous Systems*, Vol. 59, No. 12, pp. 1115–1129, December 2011.

[35] K. Furuta, M. Yamakita, and S. Kobayashi. Swing up control of inverted pendulum. In *Industrial Electronics, Control and Instrumentation, 1991. Proceedings. IECON'91., 1991 International Conference on*, pp. 2193–2198. IEEE, 1991.

[36] K. J. Åström and K. Furuta. Swinging up a pendulum by energy control. *Automatica*, Vol. 36, No. 2, pp. 287–295, 2000.

[37] C. W. Anderson. Learning to control an inverted pendulum using neural networks. *Control Systems Magazine, IEEE*, Vol. 9, No. 3, pp. 31–37, 1989.

[38] F. Chaumette and S. Hutchinson. Visual servo control. i. basic approaches. *Robotics & Automation Magazine, IEEE*, Vol. 13, No. 4, pp. 82–90, 2006.

[39] F. Chaumette, S. Hutchinson, and others. Visual servo control, part ii: Advanced approaches. *IEEE Robotics and Automation Magazine*, Vol. 14, No. 1, pp. 109–118, 2007.

[40] F. F. Bonin, A. Ortiz, and G. Oliver. Visual navigation for mobile robots: A survey. *Journal of intelligent and robotic systems*, Vol. 53, No. 3, pp. 263–296, 2008.

[41] G. N. DeSouza and A. C. Kak. Vision for mobile robot navigation: A survey. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, Vol. 24, No. 2, pp. 237–267, 2002.

[42] W. B. Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*, Vol. 703. John Wiley & Sons, 2007.

[43] M. P. Deisenroth, C. E. Rasmussen, and J. Peters. Gaussian process dynamic programming. *Neurocomputing*, Vol. 72, No. 7-9, pp. 1508–1524, 2009.

[44] T. Hester, M. Quinlan, and P. Stone. Generalized model learning for Reinforcement Learning on a humanoid robot. In *Proc. of IEEE International Conference on Robotics and Automation*, pp. 2369–2374, Anchorage, Alaska, 2010. IEEE.

[45] 劉康志. 線形ロバスト制御. コロナ社, 2002.

[46] K. Dvijotham and E. Todorov. Linearly solvable markov games. In *American Control Conference (ACC), 2012*, pp. 1845–1850. IEEE, 2012.

[47] M. W. Spong. Underactuated mechanical systems. In *Control Problems in Robotics and Automation*, pp. 135–150. Springer, 1998.

[48] M. W. Spong. The swing up control problem for the acrobot. *Control Systems, IEEE*, Vol. 15, No. 1, pp. 49–55, 1995.

[49] J. Yoshimoto, S. Ishii, and M. Sato. Application of reinforcement learning based on on-line em algorithm to balancing of acrobot. *Systems and Computers in Japan*, Vol. 32, No. 5, pp. 12–20, 2001.

[50] M. Sugiyama, I. Takeuchi, T. Suzuki, T. Kanamori, and H. Hachiya. Least-squares conditional density estimation. *IEICE Transactions on Information and Systems*, Vol. E93-D, pp. 583–594, 2010.

[51] J. A. Boyan. Technical Update: Least-Squares Temporal Difference Learning. *Machine Learning*, Vol. 49, No. 2/3, pp. 233–246, 2002.

[52] M. G. Lagoudakis and R. Parr. Least-Squares Policy Iteration. *Journal of Machine Learning Research*, Vol. 4, pp. 1107–1149, 2003.

業績リスト

論文

- K. Kinjo, E. Uchibe, K.Doya. ”Evaluation of linearly solvable Markov decision process with dynamic model learning in a mobile robot navigation task”, Front.Neurorobot. 7:7. doi: 10.3389/fnbot.2013.00007. 2013

国際会議

- K. Kinjo, E. Uchibe, K.Doya. “Robustness of Linearly solvable Markov Games with inaccurate dynamics model”, International Symposium on Artificial Life and Robotics 2014, Oita, 2014 Jan

研究会

- 金城健, 内部英治, 吉本潤一郎, 銅谷賢治. ”運動-視覚ダイナミクス学習と線形ベルマン方程式によるロボット制御”, ニューロコンピューティング研究会 沖縄, 2012 年 6 月
- 金城健, 内部英治, 吉本潤一郎, 銅谷賢治. “線形ベルマン方程式に基づくロボット制御　〜 システム同定と指数価値関数近似〜”, ニューロコンピューティング研究会 玉川, 2011 年 3 月