特別研究報告

題目

無線ネットワークを経由した移動ロボットの遠隔操作のための ベイズ推定を用いた環境同定手法の提案と実装

指導教員
村田 正幸 教授

報告者
松田 拓己

2018 年 2 月 14 日

大阪大学 基礎工学部 情報科学科

平成 29 年度 特別研究報告

無線ネットワークを経由した移動ロボットの遠隔操作のための ベイズ推定を用いた環境同定手法の提案と実装

松田 拓己

内容梗概

近年、ドローン、災害救助ロボット、無人搬送車など、ネットワーク経由で制御可能なロボットが注目を集めており、様々な手法が研究されている。それらの機器では、ネットワークを経由することにより、作業を行う領域に人が立ち入ることなく、作業が可能となる。

ネットワーク経由で操作する機器においては、ネットワークの遅延が大きな問題となる。そのような機器を効率的に動作させるためには、遅延分先の実際に制御が行われる状況を予知し、その状況に合わせた制御を行うことが求められる。しかしながら、遠隔制御機器は、車輪がスリップする等により、制御時の意図通りの状態に達することができるとは限らない。その結果、前の時刻の制御で想定される状態にロボットが達するものとして、遅延分先の状態を予測したうえで制御を行うと、予測した状態と実際の状態の誤差が大きく異なり、事前に想定した制御ができなくなるという問題が生じる。

そのため、ロボットの制御の際に生じる誤差を把握し、誤差を考慮して制御を行うことが考えられる。ロボットの制御の際に生じる誤差の大きさは環境により大きく異なる。たとえば、車輪がスリップしすい路面で遠隔操作機器を動作させる場合と、車輪がスリップしにくい路面で動作させる場合では、制御の際に生じる誤差は大きく異なる。そのため、ロボットを遠隔制御する際には、ロボットの現在の環境を同定し、その環境に応じた誤差を考慮した制御を行うことが求められる。このような環境の同定は、遠隔制御ロボットを用いたタスクを実行しながら、行った制御とその制御後に遠隔制御ロボットが到達する状態をもとに行うことができる。しかしながら、この方法では、行った制御、制御後に到達したロボットの状態に関する情報が少ない、タスク実行初期の段階では、適切な環境同定ができずに、遠隔制御ロボットに対して不適切な制御を行ってしまう可能性がある。

本報告では、上記のような遠隔操作ロボットに対して、得られた観測結果が少ないタスク実行初期の段階であっても、適切な制御が可能となるような制御を確立する。文献 [1] では、生物が少ない経験であっても、適切な認知を行うことができる仕組みを持つように進化をすることが示唆されており、進化の結果、自身が持つ事前分布をもとに観測情報を用いてベイ

ズ推定を行い、かつ、発生しうる状況を認知するのに適した事前分布を持つような進化が行われる可能性を明らかにしている。そこで、本報告における機器の遠隔制御においても、ベイズ推定の認知を行うとともに、その事前分布を進化計算により得ることにより、少ない観測情報しか得られていないタスク実行初期の段階であっても、現在の環境下で生じる誤差を適切に予測し、制御を行うことが期待できる。

本報告では、対向二輪ロボットを例として、遠隔制御のための、ベイズ推定を用いた環境同定手法を提案する。本手法では、ロボットを遠隔から制御するコントローラにおいて、(1) ロボットから得られる観測情報と、過去にロボットに送出した制御コマンドをもとに、当該コマンドに該当する制御を行った際に生じた誤差を計算、(2) 得られた誤差をもとに、逐次ベイズ推定により、現在の環境において生じる誤差モデルを更新するということを繰り返すことにより、現在の環境において生じる誤差を把握する。そして、把握された誤差を考慮して、対向二輪ロボットに対して送る制御コマンドを計算する。さらに、本報告では、ベイズ推定を用いた環境同定手法における、初期事前分布を進化計算によって得ることにより、想定しうる様々な環境下において、適切な制御が可能な事前分布を定める。具体的には、事前分布のパラメータ候補を生成する。各事前分布のパラメータ候補について、当該事前分布が与えられた際のロボットの制御について、シミュレーションを行い、当該制御により達成される、移動ロボットの目標軌跡からのずれや目標地点に到達するまでの時間をもとに評価をする。そして、評価結果をもとに、選択、淘汰、交叉を行うことにより、適切な事前分布を定める。

本報告では、進化計算により求めた事前分布から逐次ベイズ推定を行うことにより、現在の環境で生じる誤差を推定しながら制御を行うコントローラを実装し、シミュレーションにより有効性を評価する。評価結果より、タスクを実行しながら、観測された制御誤差の情報のみから、現在の環境において生じる誤差を推定して制御に用いる手法と比べ、目標軌跡からのずれを 30%削減できることを示す。

主な用語

キーワード
ネットワーク制御システム、ベイズ推定、遠隔操作ロボット

目 次

1 はじめに **5**

2 関連研究 **7**

2.1 遠隔制御 7

2.2 ベイズ推定を用いた状態予測の進化 8

3 移動ロボットの遠隔制御における環境同定手法 **9**

3.1 移動ロボットのダイナミクス 9

3.2 移動ロボットの遠隔制御 10

3.2.1 現在の状況の認知 11

3.2.2 将来の状況の予測 12

3.2.3 制御 12

3.3 ベイズ推定を用いた環境同定 14

3.4 進化計算による事前分布の算出 15

4 移動ロボットの遠隔制御における環境同定手法の実装 **16**

4.1 移動ロボットの遠隔制御 16

4.1.1 現在の状況の認知 16

4.1.2 将来の状況の予測 16

4.1.3 制御 17

4.2 ベイズ推定を用いた環境同定 18

4.3 環境同定の進化 18

5 シミュレーション **21**

5.1 評価環境 21

5.1.1 制御を行う環境 21

5.1.2 ロボットのタスク 21

5.2 比較対象 21

5.3 結果 22

6 おわりに **27**

謝辞 **28**

参考文献 **29**

図 目 次

1 移動ロボットの概要図 10
2 コントローラとロボットで行われる通信 11
3 $\sigma^{=}$0.0025 の環境でベイズ推定を用いた制御でのロボットの速度 23
4 $\sigma^{=}$0.04 の環境でベイズ推定を用いた制御でのロボットの速度 24
5 誤差の分散値 $\sigma^2 = 0.0025$ の環境において各手法を用いた場合の制御の経路からのずれの最大値の累積補分布 25
6 誤差の分散値 $\sigma^2 = 0.04$ の環境において各手法を用いた場合の制御の経路からのずれの最大値の累積補分布 25
7 誤差の分散値 $\sigma^2 = 0.0025$ の環境において各手法を用いた場合の制御の時間の累積補分布 26
8 誤差の分散値 $\sigma^2 = 0.04$ の環境において各手法を用いた場合の制御の時間の累積補分布 26

1　はじめに

近年、ドローン、災害救助ロボット、無人搬送車など、ネットワーク経由で制御可能なロボットが注目を集めており、様々な手法が研究されている [2, 3]。それらの機器では、ネットワークを経由することにより、作業を行う領域に人が立ち入ることなく、作業が可能となる。

ネットワーク経由で操作する機器において、ネットワークの遅延は意図した制御を行う上で大きな問題となる。そのような機器を効率的に動作させるためには、遅延時間分たけ先のロボットの状態を予測し、実際に制御が行われる状況を予知した上で、その状況に合わせた制御を行うことが求められる。しかしながら、遠隔制御機器は、車輪がスリップする等により、制御時の意図通りの状態に達することができるとは限らない。その結果、前の時刻の制御で想定される状態にロボットが達するものとして、遅延分先の状態を予測したうえで制御を行うと、予測した状態と実際の状態の誤差が大きく異なり、事前に想定した制御ができなくなるという問題が生じる。

そのため、ロボットの制御の際に生じる誤差を把握し、誤差を考慮して制御を行うことが考えられる。[4, 5] ロボットの制御の際に生じる誤差の大きさは環境により大きく異なる。たとえば、車輪がスリップしすい路面で遠隔操作機器を動作させる場合と、車輪がスリップしにくい路面で動作させる場合では、制御の際に生じる誤差は大きく異なる。そのため、ロボットを遠隔制御する際には、ロボットの現在の環境を同定し、その環境に応じた誤差を考慮した制御を行うことが求められる。このような環境の同定は、遠隔制御ロボットを用いたタスクを実行しながら、行った制御とその制御後に遠隔制御ロボットが到達する状態をもとに行うことができる。しかしながら、この方法では、行った制御、制御後に到達したロボットの状態に関する情報が少ない、タスク実行初期の段階では、適切な環境同定ができずに、遠隔制御ロボットに対して不適切な制御を行ってしまう可能性がある。

本報告では、上記のような遠隔操作ロボットに対して、得られた観測結果が少ないタスク実行初期の段階であっても、適切な制御が可能となるような制御を確立する。文献 [1] では、生物が少ない経験であっても、適切な認知を行うことができる仕組みを持つように進化をすることが示唆されており、進化の結果、自身が持つ事前分布をもとに観測情報を用いてベイズ推定を行い、かつ、発生しうる状況を認知するのに適した事前分布を持つような進化が行われる可能性を明らかにしている。そこで、本報告における機器の遠隔制御においても、ベイズ推定の認知を行うとともに、その事前分布を進化計算により得ることにより、少ない観測情報しか得られていないタスク実行初期の段階であっても、現在の環境下で生じる誤差を適切に予測し、制御を行うことが期待できる。

本報告では、対向二輪ロボットを例として、遠隔制御のための、ベイズ推定を用いた環境同定手法を提案する。本手法では、ロボットを遠隔から制御するコントローラにおいて、(1)

ロボットから得られる観測情報と、過去にロボットに送出した制御コマンドをもとに、当該コマンドに該当する制御を行った際に生じた誤差を計算、(2) 得られた誤差をもとに、逐次ベイズ推定により、現在の環境において生じる誤差モデルを更新するということを繰り返すことにより、現在の環境において生じる誤差を把握する。そして、把握された誤差を考慮して、対向二輪ロボットに対して送る制御コマンドを計算する。さらに、本報告では、ベイズ推定を用いた環境同定手法における、初期事前分布を進化計算によって得ることにより、想定しうる様々な環境下において、適切な制御が可能な事前分布を定める。具体的には、事前分布のパラメータ候補を生成する。各事前分布のパラメータ候補について、当該事前分布が与えられた際のロボットの制御について、シミュレーションを行い、当該制御により達成される、移動ロボットの目標軌跡からのずれや目標地点に到達するまでの時間をもとに評価をする。そして、評価結果をもとに、選択、淘汰、交叉を行うことにより、適切な事前分布を定める。

本報告では、進化計算により求めた事前分布から逐次ベイズ推定を行うことにより、現在の環境で生じる誤差を推定しながら制御を行うコントローラを実装し、シミュレーションにより有効性を評価する。

本報告の構成は、以下の通りである。2 章で関連研究について述べる。3 章で移動ロボットの遠隔制御における環境同定手法を提案し、4 章で移動ロボットの遠隔制御における環境同定手法におけるコントローラの実装について述べる。5 章でシミュレーションによる評価を行い、6 章でまとめと今後の課題について述べる。

2　関連研究

2.1　遠隔制御

近年、ドローン、災害救助ロボット、無人搬送車など、ネットワーク経由で制御可能なロボットが注目を集めている。そのため、機器を遠隔から制御する様々な手法が研究されている [2, 3, 6, 7]。

ネットワーク経由での制御において、大きな課題となるのは、ネットワークの遅延である。制御対象機器と制御コントローラの間には、遅延が生じるため、制御コントローラが制御対象機器から取得した情報は、遅延時間前の情報であり、また、制御コントローラが送出した制御コマンドが制御対象機器に到達するのも遅延時間後となる。

この遅延に対応する方法として、遅延を予測し、予測された遅延を考慮して、現在コントローラが送出する制御コマンドが制御機器に到達する際の制御対象機器の状況を予測、予測された機器の状況に合わせて制御コマンドを計算する方法が考えられる。文献 [8] では、カルマンフィルターを用いて、ネットワーク遅延を予測し、予測した遅延に合わせて移動ロボットを制御する手法を提案している。しかしながら、遅延の予測値には誤差が含まれ、予測遅延の誤差が制御に大きく影響を与える。

遅延の予測誤差の影響を受けない方法として、将来の各時刻における制御コマンドと当該制御コマンドを実行する時刻の組をコントローラから、制御対象機器に送出する手法が考えられる。この手法では、遅延時間より十分に先の時刻の制御対象機器の状況を予測し、その状況に合わせた制御コマンドを計算、制御対象機器に送出する。これにより、遅延の予測誤差の影響を受けず、制御を行うことが可能となる。さらに、このような制御では、制御コマンドを送出する各時刻において、一つの時刻の制御コマンドのみではなく、将来にわたる K 個の制御コマンドを計算して送出することも可能である。この手法では、コントローラは、各時刻 t において、遅延時間先の時刻 $t+d$ から $t+d+K$ までの各時刻における制御コマンドを計算して、制御対象機器に送出する。制御対象機器では、制御時刻と対応する制御コマンドを保存し、当該時刻となった際に、対応する制御コマンドを実行する。また、制御対象機器において、同一制御時刻のコマンドを何度も受信することとなるが、新しい制御コマンドが到達した際に上書きする。これにより、新しい情報をもとに計算された新しい制御コマンドを適用することができる。また、複数時刻の制御コマンドを送信することにより、以降の制御コマンドが遅延の大幅な増大等により、当該時刻までに到達しない場合であっても、適切な制御を行うことができる。文献 [9] や文献 [10] では、上記の、将来の複数の時刻における制御を、制御対象機器の状況を予測しながら定め、まとめて送出することを繰り返す手法が、ネットワーク遅延やパケットロスが発生する環境において効果的であることが示されている。

これらの手法を適切に動作させるためには、将来の制御コマンドを実行する際の制御対象機器の状況を予測することが必要となる。しかしながら、制御対象機器の動作には、誤差が入る。例えば、車輪による移動ロボットの場合、車輪のスリップにより、制御コマンドにより意図した位置や角度まで到達できないといった事象が発生する。しかも、車輪のスリップのしやすさは、ロボットが移動する床面のすべりやすさや、ロボットの荷物の積載の有無等の条件により大きく変わってくる。そのため、上述の手法によるネットワークの遅延の考慮に加えて、制御対象機器がタスクを実行している現在の環境を把握し、現在の環境にあわせて制御誤差を予測して、制御することが必要となる。本報告では、対向二輪ロボットを対象とし、現在の環境を同定し、現在の環境に合わせた制御誤差を考慮しつつ、ネットワーク経由の制御を行う手法について検討を行う。

2.2 ベイズ推定を用いた状態予測の進化

生物が少ない経験であっても、適切な認知を行うことにより、生存してきたと考えられる。文献 [1] では、ベイズ推定により状態の認知を行っている生物において、ベイズ推定における事前分布を進化の過程によって獲得できる可能性を検証している。本検証では、ベルヌーイ試行を対象として、各個体は、事象 A の発生確率 p_A の予測を行う。その際、各個体は、ベイズ推定により予測を行うものとし、事前分布としてパラメータ $[\alpha, \beta]$ のベータ分布を保持している。各個体は、事象 A のベルヌーイ試行を n 回行い、事象 A が得られた回数 k をもとに、以下のように確率分布を更新し、その後の事象 A の発生確率の予測に用いる。

$$\phi_x(p_A) = \frac{k+\alpha}{n+\alpha+\beta} \tag{1}$$

本検証においては、上記の予測を行う各個体について、以下の適応度関数 $f_s(x)$ をもとに評価を行い、変異、淘汰により、個体の進化をシミュレーションしている。

$$f_s(x) = \frac{1}{1+\sum_{p_i \in S}[p_i - \phi_x(p_i)]^2} \tag{2}$$

$f_s(x)$ は、個体 x の予測と実際の事象 A の発生が一致すれば、高くなる。すなわち、本値が高い個体が生き残るような淘汰、変異を行うことにより、適切に事象 A の発生を予測できるような進化が行われる。

文献 [1] では、上記の手順の進化により獲得した事前分布をもとにベイズ推定を行う個体が、自身が観測した観測値のみをもとに頻度主義的に事象の発生を予測する個体よりも高い精度で事象の発生を予測することができることを示している。

そこで、本報告においても、遠隔制御における環境の同定を、上記の検証と同様、進化により獲得した事前分布を用いたベイズ推定により行う手法について検討する。

3　移動ロボットの遠隔制御における環境同定手法

3.1　移動ロボットのダイナミクス

まず、本報告において使用する対向二輪型移動ロボットのダイナミクスについて述べる。図 1 は対向二輪型移動ロボットの概要図を表している。本ロボットの車輪の半径を r、2 つの車輪の間隔を W とする。移動ロボットの状態はロボットの重心が位置する 2 次元の座標と向きで表される。すなわち、時刻 t におけるロボットの状態を X_t は以下のように定義される。

$$X_t = \begin{pmatrix} x_t \\ y_y \\ \theta_y \end{pmatrix}$$

ただし、x_t、y_t はロボットの重心の座標であり、θ_t はロボットの進行方向の X 軸方向に対する角度である。

対向二輪型移動ロボットには、2 つの駆動輪の速度を入力として与える。つまり、ロボットへの制御入力 B_t は次式のように定義される。

$$B_t = \begin{pmatrix} w_t^{\mathrm{right}} \\ w_t^{\mathrm{left}} \end{pmatrix}$$

ただし、w_t^{right} は右車輪の速度、w_t^{left} は左車輪の速度である。

ロボットが状態 X_t で、制御入力 B_t を受け取った際、制御誤差が生じない場合は、次のロボットの状態 X_{t+1} は、以下のようになる。

$$X_{t+1} = X_t + \begin{pmatrix} v_t \cos(\theta + \frac{w_t}{2}) \\ v_t \sin(\theta + \frac{w_t}{2}) \\ w_t \end{pmatrix} \tag{3}$$

ただし、

$$\begin{pmatrix} v_t \\ w_t \end{pmatrix} = \begin{pmatrix} \frac{r}{2} & \frac{r}{2} \\ \frac{r}{W} & -\frac{r}{W} \end{pmatrix} \begin{pmatrix} w_t^{\mathrm{right}} \\ w_t^{\mathrm{left}} \end{pmatrix} \tag{4}$$

しかしながら、実際には、ロボットの車輪はスリップ等により、意図した状態にまで到達できない。本報告では、右車輪、左車輪が、それぞれ、意図した移動距離の $1 - \epsilon_t^{\mathrm{right}}$ 倍、$1 - \epsilon_t^{\mathrm{left}}$ 倍となる誤差が生じるものとし、$\epsilon_t^{\mathrm{right}}$、$\epsilon_t^{\mathrm{left}}$ は正規分布に従うものとした。

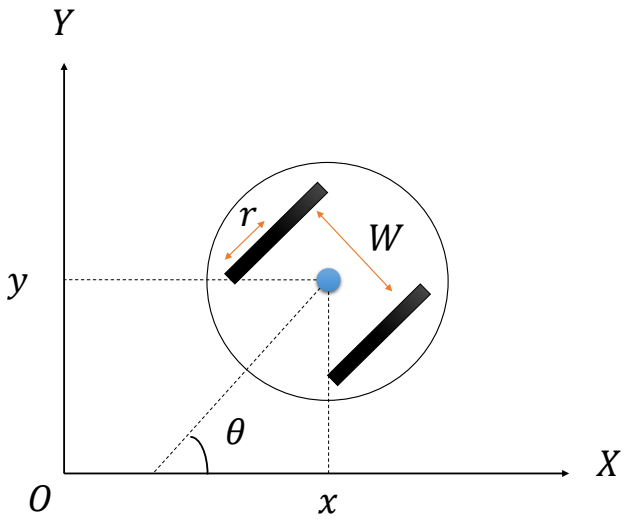

図 1: 移動ロボットの概要図

この場合、ロボットが状態 X_t で、制御入力 B_t を受け取った際、次のロボットの状態 X_{t+1} は、以下のようになる。

$$X_{t+1} = X_t + \begin{pmatrix} v'_t \cos(\theta + \frac{w_t}{2}) \\ v'_t \sin(\theta + \frac{w_t}{2}) \\ w'_t \end{pmatrix} \tag{5}$$

ただし、

$$\begin{pmatrix} v'_t \\ w'_t \end{pmatrix} = \begin{pmatrix} \frac{r}{2} & \frac{r}{2} \\ \frac{r}{W} & -\frac{r}{W} \end{pmatrix} \begin{pmatrix} w_t^{\text{right}}(1 - \epsilon_t^{\text{right}}) \\ w_t^{\text{left}}(1 - \epsilon_t^{\text{left}}) \end{pmatrix} \tag{6}$$

この予測を $t+1$ から逐次的に行うことにより、$\hat{P}(X_{t+d})$ を得ることができる。

3.2 移動ロボットの遠隔制御

図 2 に本報告で想定する移動ロボットの制御の概要を示す。本報告では、時刻 t において、ロボットが自身のセンサーや周囲のセンサー等から情報を取得し、観測値 A_t を得る。そして、A_t をコントローラに送出する。コントローラでは、片方向遅延 d_1 後の時刻 $t + d_1$ に観測値 A_t を受け取り、A_t から時刻 t におけるロボットの状態 X_t を推定し、遅延分先の時刻

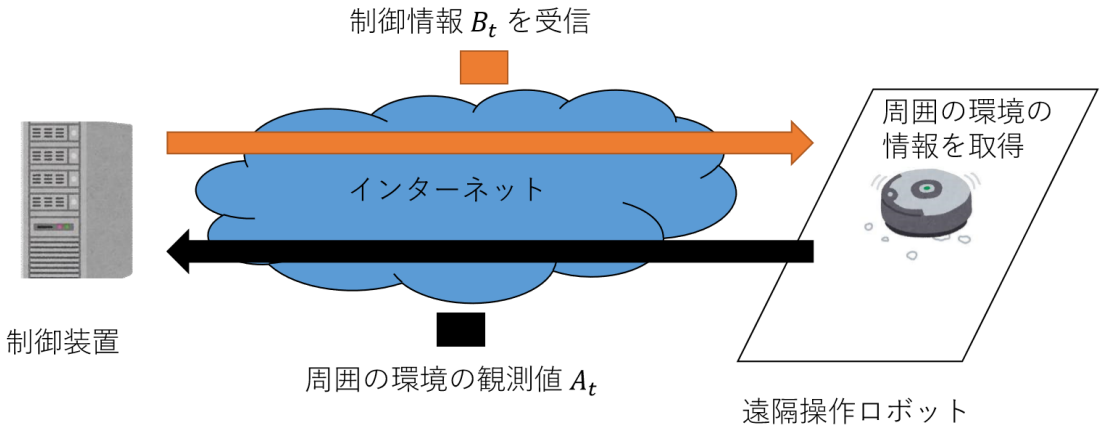

図 2: コントローラとロボットで行われる通信

$t + d_1 + d_2$ の状態を予測する。そして、予測した状態に対する制御入力 $B_{t+d_1+d_2}$ を決定し、ロボットに送出する。ロボットに送出する際には、当該制御入力を投入する時刻も含めて送り、操作機器では、受け取った制御入力について、当該入力を投入する時刻まで当該入力を保持し、当該時刻になった際に入力を投入する。この制御を繰り返すことにより、遠隔地にある機器の制御を行う。

上記環境下でコントローラが行うロボットの状態の認知、将来の状態予測、制御方を以下に述べる。

3.2.1 現在の状況の認知

コントローラでは、A_t の情報を受信後、以下のように逐次ベイズ推定により、時刻 t の状況を推定する。

$$P(X_t) = \alpha P(A_t|X_t)\hat{P}(X_t)$$

ただし、$\hat{P}(X_t)$ は、前の時刻までに把握されていた状況 X_t の確率分布であり、$P(A_t|X_t)$ は状況 X_t の場合に得られる観測値 A_t の確率分布であり、事前に与えられるものであるとする。また、α は定数である。

3.2.2 将来の状況の予測

時刻 $t+d$ における制御入力を計算するためには、ロボットが制御入力を受け取る時刻 $t+d$ における状態を予測する必要がある。

時刻 $t+1$ のロボットの状態の確率分布 $\hat{P}(X_{t+1})$ は以下のように予測できる。

$$\hat{P}(X_{t+1}) = \int_{X_t,\epsilon_t} P(X_{t+1}|X_t, B_t, \epsilon_t)\hat{P}(X_t)P(\epsilon_t) \tag{7}$$

ただし、$\epsilon_t = (\epsilon_t^{\mathrm{right}}, \epsilon_t^{\mathrm{left}})$ であり、$P(\epsilon_t)$ はコントローラが把握している誤差の分布、$P(X_{t+1}|X_t, B_t, \epsilon_t)$ は、時刻 t におけるロボットの状態が X_t で、制御入力 B_t が与えられた際に誤差 ϵ_t が発生したときに次状態の分布を示す。

本報告における対向二輪型移動ロボットにおいては、$P(X_{t+1}|X_t, B_t, \epsilon_t)$ は、以下のように定まる。

$$P(X_{t+1}|X_t, B_t, \epsilon_t) = \begin{cases} 1 & (X_{t+1} = X_{t+1}^{\mathrm{pred}}) \\ 0 & (\mathrm{Others}) \end{cases} \tag{8}$$

ただし、

$$X_{t+1}^{\mathrm{pred}} = X_t + \begin{pmatrix} v'_t \cos(\theta + \frac{w_t}{2}) \\ v\text{'}_t \sin(\theta + \frac{w_t}{2}) \\ w\text{'}_t \end{pmatrix} \tag{9}$$

ただし、

$$\begin{pmatrix} v'_t \\ w\text{'}_t \end{pmatrix} = \begin{pmatrix} \frac{r}{2} & \frac{r}{2} \\ \frac{r}{W} & -\frac{r}{W} \end{pmatrix} \begin{pmatrix} w_t^{\mathrm{right}}(1-\epsilon_t^{\mathrm{right}}) \\ w_t^{\mathrm{left}}(1-\epsilon_t^{\mathrm{left}}) \end{pmatrix} \tag{10}$$

3.2.3 制御

本報告では、各時刻において、目標状態 X^{target} 与え、目標状態 X^{target} へ移動するための制御入力 B_{t+d} を与えることを続けることにより、目標軌道に沿ってロボットを移動させる。その際、ロボットの制御の際に生じる誤差を考慮する。式 (10), (9) より、本報告で用いる移動ロボットにおいては、ロボットの車輪速度が大きくなるほど、誤差も大きくなりやすい。そのため、大きな誤差が生じる可能性のある環境下においては、車輪速度を抑制することが、誤差を抑えるためには有効である。

本報告では、上記の目標を達成する制御入力を計算するにあたり、まず、目標状態 X^{target} に到達するための制御入力 $B\text{'}_{t+d}$ を求め、誤差を考慮し、ロボットの状態 X_{t+d+1} の取りうる範囲が閾値以下となるように、$B\text{'}_{t+d}$ をスケールしたものを B_{t+d} とする。以下に、詳細について述べる。

目標状態へ到達するための制御入力の計算　時刻 $t+d$ のロボットの状態の予測結果をもとに、時刻 $t+d+1$ に目標状態 X^{target} に到達することができる入力 B_{t+d} を求める。ここで、予測された時刻 $t+d$ の状態の期待値を $X^{\mathrm{ave}}_{t+d} = (x^{\mathrm{ave}}_{t+d}, y^{\mathrm{ave}}_{t+d}, \theta^{\mathrm{ave}}_{t+d})$ とする。B_{t+d} は以下のように計算できる。

1. 時刻 $t+d$ のロボットの状態の期待値 $[x^{\mathrm{ave}}_{t+d}, y^{\mathrm{ave}}_{t+d}, \theta^{\mathrm{ave}}_{t+d}]$ と目標状態 $[x^{target}, y^{target}, \theta^{target}]$ との差 $[e_x, e_y, e_\theta]$ を計算する。

$$\begin{pmatrix} e_x \\ e_y \\ e_\theta \end{pmatrix} = \begin{pmatrix} \cos\theta_C & \sin\theta_C & 0 \\ -\sin\theta_C & \cos\theta_C & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x^{target} - x \\ y^{target} - y \\ \theta^{target} - \theta \end{pmatrix} \tag{11}$$

2. 目標地点に到達するために、時刻 $t+d$ のロボットが出すべき速度と角速度 (v, w) を導出する。

$$v = K_{(t+d)} \tag{12}$$

$$w = 2A_{(t+d)}K_{(t+d)} \tag{13}$$

ただし、

$$A_{(t+d)} = sign(e_x)\frac{e_y}{e_x^2} \tag{14}$$

$$K_{(t+d)} = sign(e_x)\frac{\alpha}{1+|A_{(t)}|} \tag{15}$$

ただし、α は正の定数であり、値が大きくなるほど ロボットが速く移動する。

3. 速度と角速度から入力 $B'_{t+d} = (w^{\mathrm{left}}_{t+d}, w^{\mathrm{right}}_{t+d})$ を計算する。計算式は次のようになる。

$$w^{\mathrm{right}}_{t+d} = \frac{v}{r} + \frac{Ww}{2r} \tag{16}$$

$$w^{\mathrm{left}}_{t+d} = \frac{v}{r} - \frac{Ww}{2r} \tag{17}$$

誤差を考慮した速度の抑制　時刻 $t+d$ に入力 B_{t+d} が与えられた際、時刻 $t+d+1$ のロボットは、式 (9) により求まる。入力 B_{t+d} の絶対値が大きくなればなるほど、式 (9) における $\epsilon^{\mathrm{left}}, \epsilon^{\mathrm{right}}$ の影響が大きくなる。本報告では、時刻 $t+d+1$ のロボットの状態の取りうる範囲が、一定以下となるように、上記の手順で計算された入力 B'_{t+d} をスケールすることにより、$\epsilon^{\mathrm{left}}, \epsilon^{\mathrm{right}}$ の影響が大きくなることを防ぐ。すなわち、変数 s を導入し、以下の最適化問題を解き、入力 $B_{t+d} = sB'_{t+d}$ を定める。

目的関数：

$$minimize|s-1|$$

制約条件：

$$\forall X_{t+d+1}(sB_{t+d}) \in \{X_{t+d+1}(sB_{t+d}) | P\left(X_{t+d+1}(sB_{t+d})\right) < P^{\text{threshold}}\},$$
$$|X_{t+d+1}(sB_{t+d}) E[X_{t+d+1}(sB_{t+d})]| \leq X^{\text{threshold}}$$

ただし、$P^{\text{threshold}}$、$X^{\text{threshold}}$ は閾値であり、$P\left(X_{t+d+1}(sB_{t+d})\right)$ は、

$$P\left(X_{t+d+1}(sB_{t+d})\right) = \int_{X_{t+d},\epsilon_t} P(X_{t+d+1}|X_{t+d}, sB_{t+d}, \epsilon_{t+d}) \hat{P}(X_{t+d}) P(\epsilon_{t+d})$$

である。

これにより、誤差が大きく、大きい ϵ_{t+d} に対する $P(\epsilon_{t+d})$ が大きな環境下においては、s が小さな値に設定され、速度を抑制することが可能となる。また、$\hat{P}(X_{t+d})$ の分布が大きい場合には、s を 0 として、ロボットの状態の不確実性がさらに高まることを防止することも可能となる。

3.3 ベイズ推定を用いた環境同定

移動ロボットにおいて、ロボットを制御した際に生じる誤差 ϵ^{left}、ϵ^{right} は、環境により大きく異なる。そのため、ロボットを遠隔制御する際には、ロボットの現在の環境を同定し、その環境に応じた誤差を考慮した制御を行うことが求められる。

本報告では、遠隔制御ロボットを用いたタスクを実行しながら、行った制御とその制御後に遠隔制御ロボットが到達する状態をもとに $P(\epsilon)$ を推定することにより、環境同定を行う。タスクを実行しながら、$P(\epsilon)$ の推定を行う場合、推定に用いる観測結果が少ない場合についても考慮することが求められる。すなわち、タスク実行の初期段階の、観測結果が少ない状況においても、ロボットに対して、誤差の影響が大きくなってしまうような不適切な制御入力を与えることは防ぐ必要がある。

本報告では、逐次ベイズ推定により、$P(\epsilon)$ の推定を行う。これにより、事前分布を適切に定めることができれば、観測結果が不十分な場合であっても、不適切な制御入力を与えてしまうことを避けることができる。

本報告における、コントローラが $P(\epsilon)$ を逐次推定する手順を以下に示す。

1. ロボットから受信した観測値 A_t から、X_t を推定する
2. X_t、X_{t-1}、B_{t-1} から以下の連立方程式を解き、$\epsilon^{\text{left}}_{t-1}$、$\epsilon^{\text{right}}_{t-1}$ を得る

$$X_t - X_{t-1} = \begin{pmatrix} \cos\theta_{t-1} & 0 \\ \sin\theta_{t-1} & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \frac{r}{2} & \frac{r}{2} \\ \frac{r}{W} & -\frac{r}{W} \end{pmatrix} \begin{pmatrix} w^{\text{right}}_{t-1}(1-\epsilon^{\text{right}}_{t-1}) \\ w^{\text{left}}_{t-1}(1-\epsilon^{\text{left}}_{t-1}) \end{pmatrix} \tag{18}$$

3. ϵ のモデル、$P(\epsilon|A_{0:t-1})$ をベイズ推定により更新する。

$$P(\epsilon|A_{0:t}) = \alpha P(\epsilon_{t-1}^{\text{right}}, \epsilon_{t-1}^{\text{left}}|\epsilon)P(\epsilon|A_{0:t-1})$$

3.4 進化計算による事前分布の算出

ベイズ推定を用いた環境同定では、誤差の観測結果が少ない、タスク実行の初期段階において、事前分布の影響が大きく、事前分布を適切に定めていないと、誤差の影響が大きくなるような不適切な制御入力を与えてしまうことも考えられる。また、逐次ベイズ推定による $P(\epsilon)$ の収束に必要な観測数も、事前分布の影響を受ける。そのため、タスクを実行しながら、ベイズ推定により環境の同定を行う手法においては、適切な事前分布を定めることが必要となる。

生物が進化の過程でベイズ推定における適切な事前分布を獲得した可能性が示唆されている [1]。そこで、本報告においても、進化計算により、事前分布を定めるものとする。

本報告では、$P^{\text{prior}}(\epsilon)$ の分布の形状は既知であるとし、そのパラメータを進化させる。具体的な手順は、以下の通りである。

1. 初期化： $P^{\text{prior}}(\epsilon)$ の分布を定めるパラメータをランダムに設定与えた個体を N 個生成する。

2. 評価： 各個体について、当該個体が保持するパラメータの $P^{\text{prior}}(\epsilon)$ を事前分布として持つコントローラにより、移動ロボットを遠隔制御するタスクを想定しうる環境下で実行した場合の挙動をシミュレーションにより得る。そして、そのタスクの正確性、速度といった指標で、各個体を評価する。

3. 選択・交叉・突然変異： 評価値をもとに選択、交叉、突然変異を行い、新たな個体を生成する。

4. 2 へ戻る

上記の手順では、適切にタスクを実行するのに有用な事前分布を持つように選択・交叉が行われる。そのため、上記の手順による進化を行うことで、適切な事前分布を得ることができると考えられる。

4　移動ロボットの遠隔制御における環境同定手法の実装

本章では、3 章で述べた移動ロボットの遠隔制御における環境同定手法を制御コントローラに実装する。

4.1　移動ロボットの遠隔制御

4.1.1　現在の状況の認知

コントローラでは、まず、ロボットから通知された観測値 A_t をもとに、X_t を推定する。本報告で実装した制御コントローラでは、簡単のため、A_t から正確に X_t を推定することができるものとした。すなわち、

$$P(X_t) = \begin{cases} 1 & (X_t = F(A_t)) \\ 0 & (\text{otherwise}) \end{cases} \tag{19}$$

ただし、$F(A_t)$ は A_t から X_t を推定する関数である。

4.1.2　将来の状況の予測

コントローラは、X_t とコントローラが推定している $P(\epsilon)$ をもとに、時刻 $t+d$ の移動ロボットの状態の確率分布 $\hat{P}(X_{t+d})$ を推定する。移動ロボットの状態の変化は、非線形であり、時刻 $t+d$ における移動ロボットの状態の確率分布を解析的に求めることは困難である。そこで、本報告では、シミュレーションにより、$\hat{P}(X_{t+d})$ を推定する。

$\hat{P}(X_{t+d})$ を求めるため、本報告では、座標を n 刻み、向きを m 刻みとした、$n \times n \times m$ の表を準備した。表の各エントリには、ロボットをシミュレーションした結果、当該エントリの状態となったロボットの割合が保持されている。すなわち、$\hat{P}(X_{t+d})$ は、X_{t+d} に対応する表のエントリの値となる。

$\hat{P}(X_{t+d-1})$ に対応する表から、$\hat{P}(X_{t+d})$ に対応する表は、以下の手順で作成する。

1. 時刻 $t+d$ の状態を予測する元となる時刻 $t+d-1$ のロボットを表の確率に従い選択する。

2. 1 で選択された各ロボットについて、ϵ を $P(\epsilon)$ に従う確率で E 個生成し、各誤差 ϵ が発生した場合に、時刻 $t+d$ に到達するシミュレーションにより求める。

3. 2 で求めた EL 種類のロボットの状態のうち、$\hat{P}(X_{t+d})$ に対応する表の各エントリに対応するものの割合を求め、表に保存する。

4.1.3 制御

コントローラは、予測された時刻 $t+d$ におけるロボットの状態をもとに、制御コマンド B_{t+d} を決める。その際、3.2.2 で述べたように、本報告では、時刻 $t+d$ におけるロボットの状態の期待値から、目標地点へ到達するための制御コマンド B'_{t+d} を求め、誤差を考慮して、B'_{t+d} をスケールすることにより、最終的な制御コマンド B_{t+d} を計算する。

この手順のうち、B'_{t+d} は、3.2.2 の式により、容易に計算できる。それに対して、誤差を考慮した最終的な制御コマンド B_{t+d} の計算は、時刻 $t+d$ におけるロボットの状態の確率分布が解析的に求まらず、また、さらに制御コマンド B_{t+d} を与えた際の時刻 $t+d+1$ におけるロボットの状態も解析的に求まらないため、そのまま最適化問題として解くことは困難である。

そこで、ここでは、式 (**??**) の制約条件を、時刻 $t+d+1$ におけるロボットの角度に対する不確実性と、ロボットの各座標における不確実性の制約に分けて考える。時刻 $t+d+1$ におけるロボットの角度は

$$\theta_{t+d+1} = \theta_{t+d} + \frac{1}{W}\left((1-\epsilon^{\mathrm{left}}_{t+d})w^{\mathrm{left}}_{t+d} - (1-\epsilon^{\mathrm{left}}_{t+d})w^{\mathrm{right}}_{t+d}\right)$$

であり、制御入力 B_{t+d} の線形演算で表すことができる。また、時刻 $t+d+1$ におけるロボットの座標は、ロボットの角度が変化しないとみなすと、以下のように、制御入力 B_{t+d} の線形演算で表すことができる。

$$\begin{pmatrix} \hat{x}_{t+d+1} \\ \hat{y}_{t+d+1} \end{pmatrix} = \begin{pmatrix} x_{t+d} \\ y_{t+d} \end{pmatrix} + \begin{pmatrix} \cos(\theta_{t+d}) \\ \sin(\theta_{t+d}) \end{pmatrix} \left(\frac{1}{2}((1-\epsilon^{\mathrm{left}}_{t+d})w^{\mathrm{left}}_{t+d} + (1-\epsilon^{\mathrm{right}}_{t+d})w^{\mathrm{right}}_{t+d})\right)$$

上記の近似を用い、本報告では、以下の最適化問題により $B_{t+d} = sB'_{t+d}$ を定めた。目的関数：

$$minimize|s-1| \tag{20}$$

制約条件：

$$\forall X_{t+d} \in X^{\mathrm{sample}}_{t+d}, \forall \epsilon \in \epsilon^{\mathrm{sample}}\colon\ |x_{t+d+1}(X_{t+d}, sB_{t+d}, \epsilon)x_{t+d+1}(X_{t+d}, sB_{t+d}, 0)| \leq x^{\mathrm{threshold}} \tag{21}$$

$$\forall X_{t+d} \in X^{\mathrm{sample}}_{t+d}, \forall \epsilon \in \epsilon^{\mathrm{sample}}\colon\ |y_{t+d+1}(X_{t+d}, sB_{t+d}, \epsilon)y_{t+d+1}(X_{t+d}, sB_{t+d}, 0)| \leq y^{\mathrm{threshold}} \tag{22}$$

$$\forall X_{t+d} \in X^{\mathrm{sample}}_{t+d}, \forall \epsilon \in \epsilon^{\mathrm{sample}}\colon\ |\theta_{t+d+1}(X_{t+d}, sB_{t+d}, \epsilon)\theta_{t+d+1}(X_{t+d}, sB_{t+d}, 0)| \leq \theta^{\mathrm{threshold}} \tag{23}$$

ただし、

$$x_{t+d+1}(X_{t+d}, B_{t+d}, \epsilon) = x_{t+d} + \left(\frac{\cos\theta_{t+d}}{2}((1-\epsilon^{\mathrm{left}}_{t+d})w^{\mathrm{left}}_{t+d} + (1-\epsilon^{\mathrm{right}}_{t+d})w^{\mathrm{right}}_{t+d})\right)$$

$$y_{t+d+1}(X_{t+d}, B_{t+d}, \epsilon) = y_{t+d} + \left(\frac{\sin\theta_{t+d}}{2}((1-\epsilon^{\text{left}}_{t+d})w^{\text{left}}_{t+d} + (1-\epsilon^{\text{right}}_{t+d})w^{\text{right}}_{t+d}) \right)$$

$$\theta_{t+d+1}(X_{t+d}, B_{t+d}, \epsilon) = \theta_{t+d} + \frac{1}{W}\left((1-\epsilon^{\text{left}}_{t+d})w^{\text{left}}_{t+d} - (1-\epsilon^{\text{left}}_{t+d})w^{\text{right}}_{t+d}\right)$$

である。また、X^{sample}_{t+d} は、時刻 $t+d$ におけるサンプリングされた予測されたロボットの状態の集合であり、ϵ^{sample} は、コントローラが把握している ϵ の分布に従いサンプリングされた誤差の集合である。本最適化問題は、線形計画問題であるため、容易に解くことが可能である。

4.2 ベイズ推定を用いた環境同定

本報告では、$P(\epsilon)$ は平均 0 の正規分布に従うとし、事前分布として、以下の分布を与えた。

$$\epsilon_i \sim N(0,\sigma)\frac{1}{\sigma^2} \sim Gamma(\alpha,\beta) \tag{24}$$

上記の事前分布が与えられ、誤差の観測結果 ϵ_t が得られた際の事後分布は、以下で与えられる。

$$\epsilon_i \sim N(0,\sigma)\frac{1}{\sigma^2} \sim Gamma(\alpha\text{'},\beta\text{'}) \tag{25}$$

ただし、

$$\alpha\text{'} = \alpha + \frac{1}{2}$$
$$\beta\text{'} = \beta + \frac{\epsilon_t^2}{2}$$

である。そのため、観測結果が得られるたびに、上記の手順で α、β を更新することにより、逐次ベイズ推定を行い、誤差モデルを更新することができる。

4.3 環境同定の進化

本報告では、上述の事前分布のパラメータ α、β を進化計算により定める。本報告では、進化計算アルゴリズムのうち、遺伝的アルゴリズム [11, 12] を用いた。以下に、本報告で用いた、具体的なパラメータの計算手順を示す。本報告では、1 世代あたりの個体数を N、最大世代数を G とした。遺伝的アルゴリズムでは各世代 g で存在する事前分布 N 個から適応度に応じて選択を行う。選択後、交叉、再生、突然変異の動作を行い次世代 $g+1$ における

事前分布を N 個生成する。これを繰り返すことにより、適切な事前分布のパラメータ α、β をも問える。

本報告では、適応度は、各個体に対応する事前分布を持つコントローラにより、遠隔操作のタスクを実行し、その際に目標となる移動軌跡からのずれと、タスク完了までの時間の双方を考慮して以下のように定めた。

$$f_i = \alpha_T \frac{1}{\frac{T_i - M^T}{M^T} + 1} + \alpha_E \frac{1}{\frac{E_i}{M^E} + 1} \tag{26}$$

ただし、M^T はタスクを環境するのに最低限必要となる制御時間、M^E は許容する目標軌跡からのずれ、T_i は個体 i のタスク完了に要した時間、E_i は個体 i の目標とする移動軌跡からのずれの最大値である。また、α_T、α_E は、タスク実行時間、目標軌跡からのずれに対する重みである。本評価値では、許容するタスク実行時間を超える、あるいは、許容する目標軌跡からのずれを超えると、0 となる。本報告では、許容する目標軌跡からのずれが達成できる限りは、タスク実行時間を重視するように、α_T を α_E に対して十分大きな値に設定した。

また、本報告では、以下の手順で選択、交叉、再生、突然変異を発生させた。

- 選択

  本報告では、個体の選択方法はルーレット選択を使用する。各世代の N 個の個体に対して、適応度を計算し、それをもとに計算した以下の確率 p_i に基づいて個体を選択する。

  $$p_i = \frac{f_i}{\Sigma_{k=1}^{N} f_k} \tag{27}$$

- 交叉

  本報告では、上述の方法で、親個体を 2 体選択する。そして、選択した個体の子個体の事前分布のパラメータ (α_c, β_c) を次のように生成する。

  $$\alpha_{\mathrm{c}} = \mathrm{random}(\min(\alpha_1, \alpha_2) - aI_\alpha, \max(\alpha_1, \alpha_2) + aI_\alpha) \tag{28}$$

  $$\beta_{\mathrm{c}} = \mathrm{random}(\min(\beta_1, \beta_2) - aI_\beta, \max(\beta_1, \beta_2) + aI_\beta) \tag{29}$$

  $$I_\alpha = |\alpha_1 - \alpha_2| \tag{30}$$

  $$I_\beta = |\beta_1 - \beta_2| \tag{31}$$

  ただし、2 体の親個体の事前分布のパラメータをそれぞれ $(\alpha_1, \beta_1), (\alpha_2, \beta_2)$ とする。また、a は定数であり、$\mathrm{random}(a, b)$ は区間 $[a, b]$ の一様乱数を表している。

- 突然変異

  事前分布のパラメータをランダムに設定し、新たに生成する。新たに生成する事前分

布のパラメータを (α, β) とするとパラメータは以下に示す式のようにして設定する。

$$\alpha = \mathrm{random}(0, MAX_c) \tag{32}$$

$$\beta = \mathrm{random}(0, MAX_c) \tag{33}$$

ただし、MAX_c は定数であり、$\mathrm{random}(a, b)$ は区間 $[a, b]$ における乱数である。

- 再生
  次世代 N における事前分布を導出する際に、現世代 $N-1$ に存在する事前分布のパラメータ (α, β) を 1 つ選択し、次世代にそのまま受け継ぐ。

5　シミュレーション

本章では、シミュレーションにより、遺伝的アルゴリズムにより導出された事前分布を用いたベイズ推定により $P(\epsilon)$ を推定し、制御に用いる手法が、移動ロボットを動作させうる様々な環境下においても、目標軌道から大きくずれることなく、適切にタスクを完了することができることを示す。

本章では、まず、想定する環境、タスクについて説明したのちに、遺伝的アルゴリズムの各世代の各個体が達成しているタスク完了までの時間、目標軌跡とのずれについて確認し、遺伝的アルゴリズムの世代を経るにつれ、制御をする上で適切な事前分布を得ることができていることを示す。

その後、$P(\epsilon)$ を推定する際に、事前分布を用いず、観測された誤差を用いて推定を行い、制御に用いた手法と比較し、ベイズ推定により環境同定を行う手法の優位性を示す。

5.1　評価環境

5.1.1　制御を行う環境

本シミュレーションにおいては、10ms ごとにコントローラはロボットに送出する指示を計算、ロボットは自身の座標、向きの情報をコントローラに送出するものとする。また、簡単のため、ロボットが把握している自身の位置は正確であるものとする。ロボット・コントローラ間には、100ms 程度の往復遅延が存在するとし、コントローラは、ロボットから受け取った情報をもとに、その状態から 100ms 先のロボットの状態を予測し、制御コマンドの計算をする。また、本シミュレーションにおいて、ロボットは、ϵ^{left}、$\epsilon^{\mathrm{right}}$ が平均 0、標準偏差が 0.0 から 0.2 の正規分布に従う環境下で動作するものとした。

5.1.2　ロボットのタスク

本シミュレーションでは、座標位置 (0,0) から (5000,0) までの直線を移動するタスクを課した。本シミュレーションの環境では、ロボットの左右の車輪の移動距離に誤差が生じる。そのため、大きな誤差が生じた際には、ロボットの角度がずれ、意図した直線から離れた箇所を経由する。そのため、本シミュレーションでは、本タスクにおいて、実際にロボットが経由した軌跡の Y 軸からのずれに注目した。

5.2　比較対象

本節では、以下の手法を比較する。

提案手法　進化計算により求めたパラメータを持つ事前分布を、タスク開始時点の事前分布として用い、逐次事前分布により誤差を把握する。本評価においては、各世代 50 個体を生成、50 世代進化を行った結果得られたパラメータをもつ事前分布を初期事前分布として与えた。

統計的手法　事前分布を用いずに、制御中に得られた誤差のみを用いて誤差を学習し、制御に用いる。本手法では、制御開始時点では、誤差についての情報を持たず、制御を行いながら、制御結果をもとに ϵ^{left}、ϵ^{right} を計算し、その平均 μ、分散 σ^2 を計算する。そして、ϵ^{left}、ϵ^{right} が平均 μ、分散 σ^2 の正規分布に従う想定して、ロボットの状態の予測・制御を行う。本手法と比較することにより、事前分布を用いて、逐次ベイズ推定により現在の環境で生じる誤差を推定することの効果を確認する。

固定の誤差モデルを用いる手法　制御開始前に、ロボットの移動時に生じる誤差のモデルに関するパラメータをすべて固定値として与える。具体的には、ϵ^{left}、ϵ^{right} が従う分布を平均 0、分散 σ^2 の正規分布（ただし、σ はパラメータとして事前に与える）としてモデル化し、このモデルを用い、ロボットの状態の予測・制御する。本評価では、$\sigma = 0.05$、$\sigma = 0.2$ の 2 種類のモデルを用いた。本手法との比較により、誤差モデルのパラメータを環境に合わせて学習することの効果を確認する。

5.3　結果

まず、比較対象の手法提案手法により制御されるロボットの挙動を確認する。図 3、4 に提案手法による制御を行った場合に、ロボットに与えられた速度を示す。横軸が時刻、縦軸がロボットの速度を表している。図 3 は、各車輪の移動距離の誤差の分散値が $\sigma^2 = 0.0025$ の環境で動作した場合のロボットの各時刻における速度を示しており、図 4 では誤差の分散値が $\sigma^2 = 0.04$ の環境で動作させた場合のロボットの各時刻における速度を表している。

図より、誤差の分散値が $\sigma^2 = 0.0025$ の環境ではロボットの速度が徐々に加速していることが分かる。これは、観測結果が少ないタスクの初期段階においては、大きな誤差も想定した動作をしていたのに対して、誤差の観測が進むにつれ、実際に生じる誤差の大きさに合わせた $P(\epsilon)$ を推定することができるためである。その結果、大きな誤差を考慮する必要がなくなり、速度を抑制せずに、速い速度での制御が可能となる。

一方、誤差が大きい環境では、タスク完了まで、コントローラはロボットに入力する速度を抑制している。これはベイズ推定により、車輪の誤差が大きい環境であるとコントローラが認知しているためである。その結果、誤差の影響を小さくするために、速度を抑制して制

御を行う。以上より、コントローラは動作する環境の誤差の分布を正しく把握してロボットの制御を実行していることがわかる。

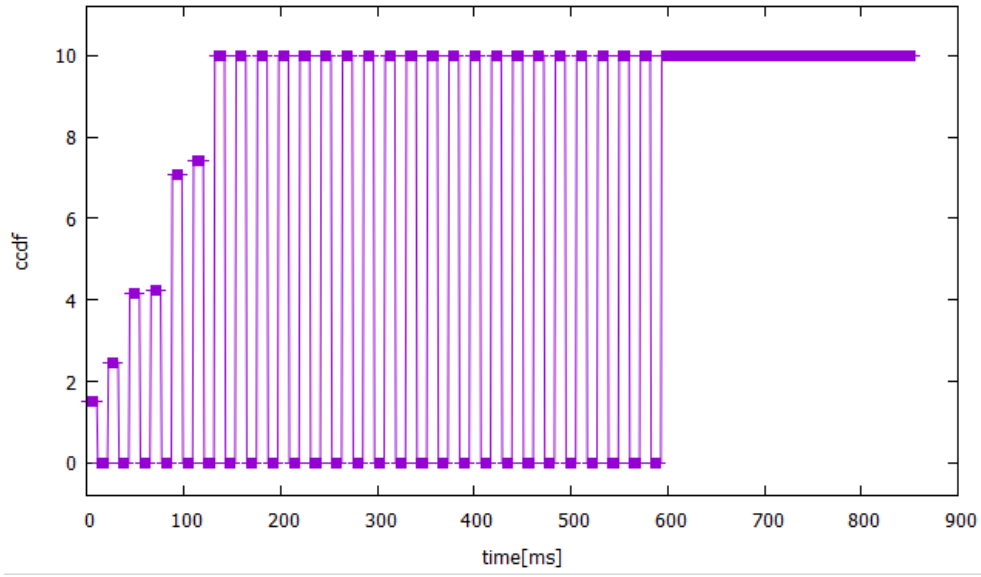

図 3: σ=0.0025 の環境でベイズ推定を用いた制御でのロボットの速度

次に、提案手法でロボットを制御した際と、比較対象の手法でロボットを制御した際の動作について比較する。本評価では、車輪の移動距離の誤差の分散値がそれぞれ 0.00、0.025、0.04 の環境において、各制御手法でロボットを制御するタスクを 30 回行い、各タスクの試行において、目標軌跡からのずれの最大値を調べた。図 5, 6 に、その累積捕分布を示す。図より、いずれの環境下においても、ベイズ推定により誤差を推定する手法は、目標軌跡からのずれを小さく抑えることができていることが分かる。それに対して、$\sigma = 0.05$ の固定の誤差モデルを用いた手法や、誤差が大きい環境（車輪の移動距離の誤差の分散値が 0.04 の環境）下において、目標軌跡からのずれが大きくなっている。これは、コントローラが想定する以上の誤差が生じているためである。同様に、観測した誤差のみから誤差を推定する手法でも、誤差が大きい環境（車輪の移動距離の誤差の分散値が 0.04 の環境）下での目標軌跡からのずれが大きい。これは、観測した誤差のみから誤差を推定する手法では、誤差の推定に十分な観測結果が得られていないタスクの初期段階では、誤って誤差が小さい環境と認識し、高速なロボットの移動を行うような制御入力を行ってしまうことが原因である。それに対して、ベイズ推定を行う手法では、十分な観測結果が得られていない時点では、大きな誤差も想定し、速度を抑制した制御入力を与える。その結果、いずれの環境においても、目標軌跡からのずれを抑えた制御が可能となる。

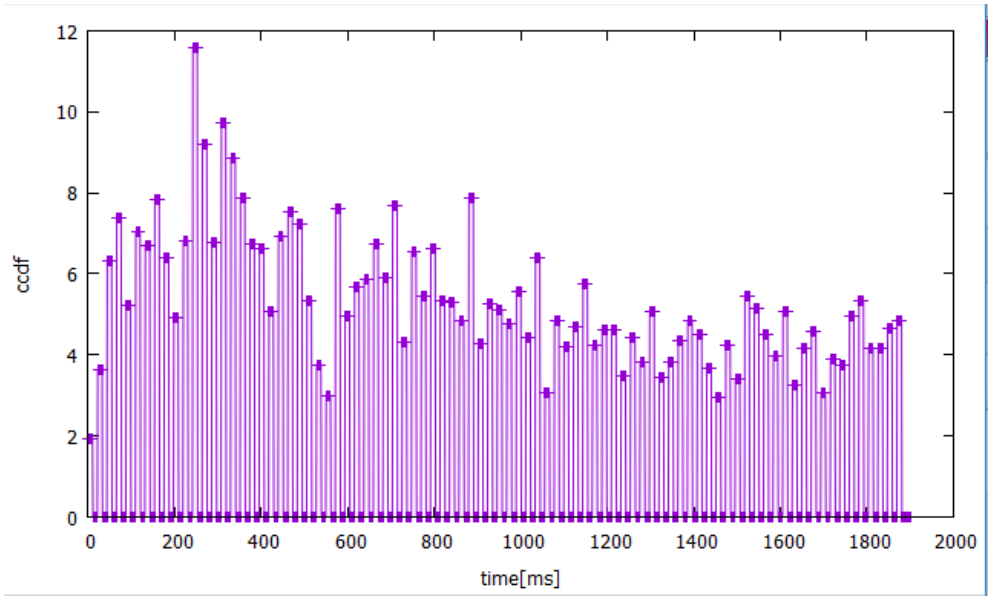

図 4: σ=0.04 の環境でベイズ推定を用いた制御でのロボットの速度

図 7、8 に、各手法を用いて制御を行った際のタスク完了までにかかる時間の累積補分布を示す。横軸はタスク完了までにかかる時間を表しており、縦軸はロボットの累積捕分布を表す。図より、$\sigma = 0.2$ の固定の誤差モデルを用いた手法では、車輪の移動距離の誤差の分散が 0.025 と、小さな誤差しか生じない環境下においても、誤差が大きな環境下と同じく、長い時間を要していることが分かる。これは、固定の誤差モデルを用いた手法では、環境によらず、当該モデルに従い、誤差の影響を抑えるように速度を抑制するためである。つまり、固定の誤差モデルを用いた手法では、そのモデルがタスクを実行する環境と合致していれば、適切な制御を行うことができるものの、タスクを実行する環境が保持している誤差モデルと異なる場合には、適切な制御を実行することができない。それに対して、ベイズ推定を行う手法によるタスク完了までの時間は、車輪の移動距離の誤差の分散値が小さい場合には、短い時間でタスクを完了することができることが分かる。これは、ベイズ推定を行う手法では、タスクを実行しながら観測された誤差から、現在の環境で発生しうる誤差が小さいと認識できた場合には、ロボットの移動速度を速めることができているためである。

つまり、ベイズ推定を行うことにより、各環境の誤差を推定し、その環境にあった制御が可能であるといえる。

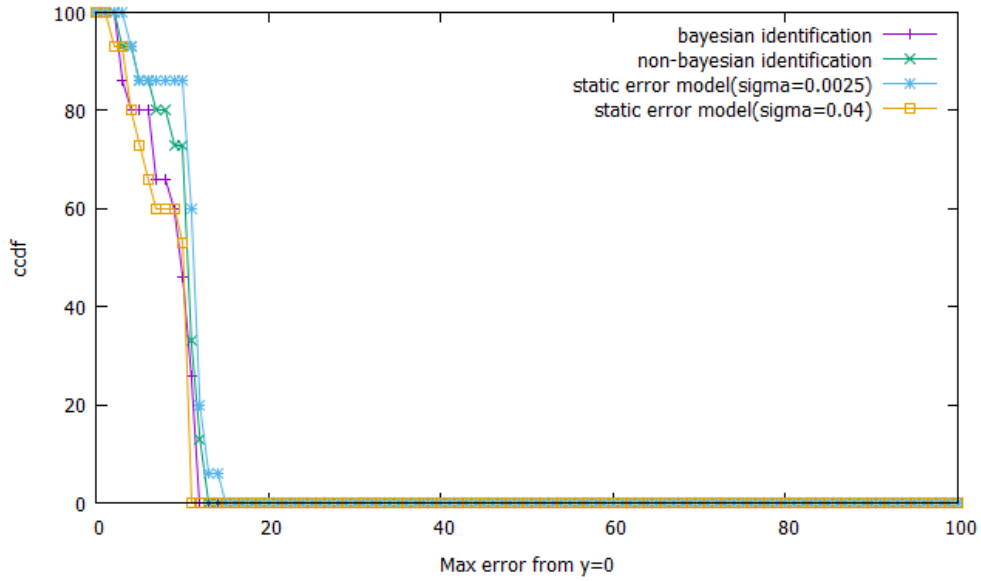

図 5: 誤差の分散値 $\sigma^2 = 0.0025$ の環境において各手法を用いた場合の制御の経路からのずれの最大値の累積補分布

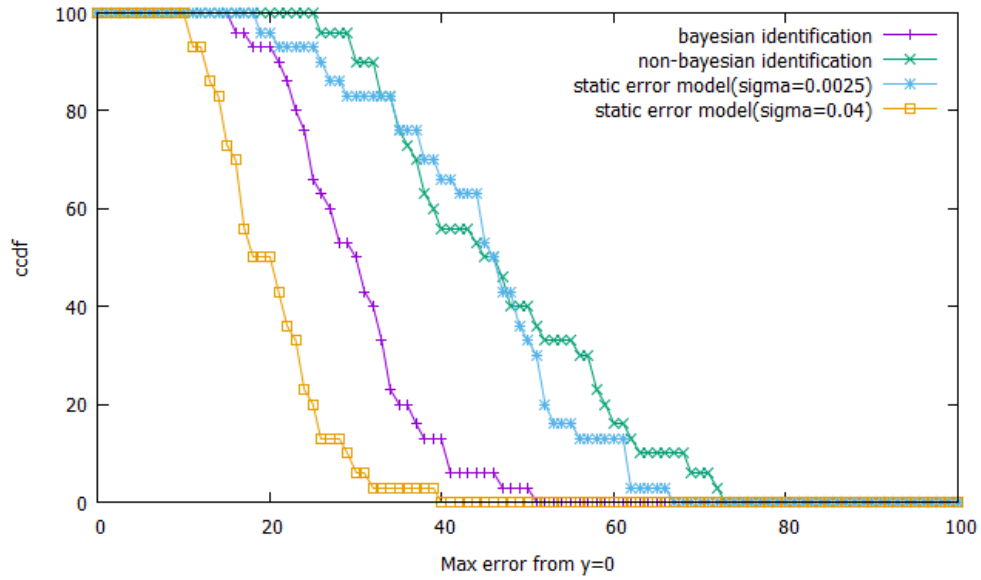

図 6: 誤差の分散値 $\sigma^2 = 0.04$ の環境において各手法を用いた場合の制御の経路からのずれの最大値の累積補分布

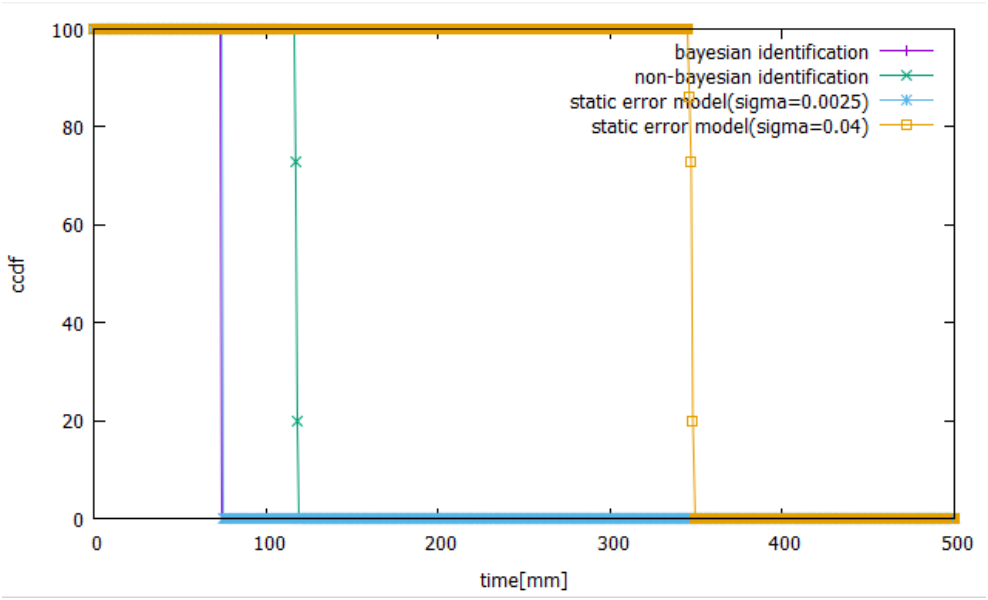

図 7: 誤差の分散値 $\sigma^2 = 0.0025$ の環境において各手法を用いた場合の制御の時間の累積補分布

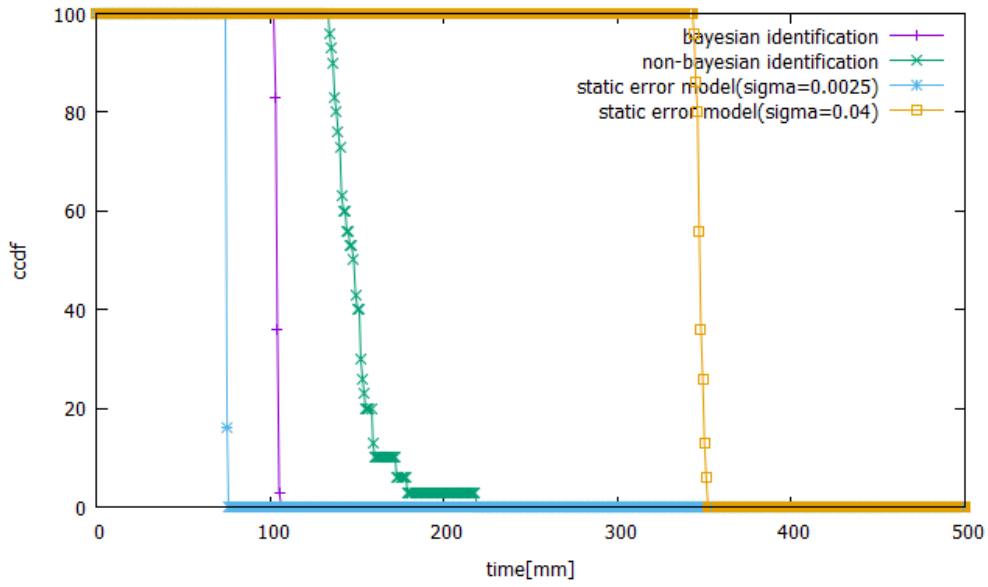

図 8: 誤差の分散値 $\sigma^2 = 0.04$ の環境において各手法を用いた場合の制御の時間の累積補分布

6　おわりに

本報告では、対向二輪ロボットを例として、遠隔制御のための、ベイズ推定を用いた環境同定手法を提案した。本手法では、ロボットを遠隔から制御するコントローラにおいて、(1) ロボットから得られる観測情報と、過去にロボットに送出した制御コマンドをもとに、当該コマンドに該当する制御を行った際に生じた誤差を計算、(2) 得られた誤差をもとに、逐次ベイズ推定により、現在の環境において生じる誤差モデルを更新するということを繰り返すことにより、現在の環境において生じる誤差を把握する。そして、把握された誤差を考慮して、対向二輪ロボットに対して送る制御コマンドを計算する。さらに、本報告では、ベイズ推定を用いた環境同定手法における、初期事前分布を進化計算によって得ることにより、想定しうる様々な環境下において、適切な制御が可能な事前分布を定めた。具体的には、事前分布のパラメータ候補を生成する。各事前分布のパラメータ候補について、当該事前分布が与えられた際のロボットの制御について、シミュレーションを行い、当該制御により達成される、移動ロボットの目標軌跡からのずれや目標地点に到達するまでの時間をもとに評価をする。そして、評価結果をもとに、選択、淘汰、交叉を行うことにより、適切な事前分布を定める。

本報告では、進化計算により求めた事前分布から逐次ベイズ推定を行うことにより、現在の環境で生じる誤差を推定しながら制御を行うコントローラを実装し、シミュレーションにより有効性を評価した。評価結果より、タスクを実行しながら、観測された制御誤差の情報のみから、現在の環境において生じる誤差を推定して制御に用いる手法と比べ、目標軌跡からのずれを 30%削減できることを示した。

今後は、本報告で提案したベイズ推定により環境同定手法を組み込んだコントローラを実際のロボットと接続し、実機実験を通して、提案手法の有効性を検証する予定である。また、本提案手法の移動ロボット以外の遠隔制御機器への適用も今後の課題である。

謝辞

謝辞本研究を進めるにあたりご指導、ご鞭撻いただきました大阪大学大学院情報科学研究科の村田正幸教授に深く感謝申し上げます。また本研究を進めるにあたり日頃から多くの時間を割いていただき、適切なご助言およびご指導をいただきました大阪大学大学院情報科学研究科の大下裕一助教に心より感謝申し上げます。また平素から広くご指導いただきました大阪大学大学院情報科学研究科の荒川伸一准教授、大阪大学大学院経済学研究科の小南大智助教、大阪大学大学院情報科学研究科の大歳達也特任助教に厚く御礼申し上げます。最後に、日頃より様々な面で支えていただきました秋下耀介氏、佐竹幸大氏をはじめ、村田研究室の皆様に感謝の意を表して謝辞といたします。

参考文献

[1] J. C. Ramírez and J. A. Marshall, "Can natural selection encode bayesian priors?," *Journal of Theoretical Biology*, Aug 2017.

[2] J. P. Hespanha, P. Naghshtabrizi, and Y. Xu, "A survey of recent results in networked control systems," in *proceeding of the IEEE*, vol. 95, pp. 138–162, Jan 2007.

[3] M.-F. R. Lee, F.-H. S. Chiu, H.-C. Huang, and C. Ivancsits, "Generalized predictive control in a wireless networked control system," *International Journal of Distributed Sensor Networks*, vol. 9, p. 475730, Jan 2013.

[4] A. Stenman, "Model-free predictive control," in *proceedings of the 38th IEEE Conference on Decision and Control.*, Dec 1999.

[5] J.-B. Mouret, "Micro-data learning: The other end of the spectrum," *ERCIM News*, p. 2, Oct 2016.

[6] Y.-Q. Xia, Y.-L. Gao, L.-P. Yan, and M.-Y. Fu, "Recent progress in networked control system," *International Journal of Automation and Compuing*, vol. 12, pp. 343–367, Aug 2015.

[7] M. Hoy, A. S. Matveev, and A. V. Savkin, "Algorithms for collision-free navigation of mobile robots in complex cluttered environments: a survey," *Robotica*, vol. 33, pp. 463–497, Mar 2015.

[8] C. Lozoya, P. Martı, M. Velasco, and J. Fuertes, "Effective real-time wireless control of an autonomous guided vehicle," *Industrial Electronics, 2007. ISIE 2007. IEEE International Symposium on*, pp. 2876–2881, Nov 2007.

[9] Y. Xia, M. Fu, and P. Shi, *Analysis and synthesis of dynamical systems with time-delays*, vol. 387. Springer Science & Business Media, 2009.

[10] Y. Xia, M. Fu, and G.-P. Liu, *Analysis and synthesis of networked control systems*, vol. 409. Springer Science & Business Media, 2011.

[11] I. Ono, H. Satoh, and S. Kobayashi, "A real-coded genetic algorithm for function optimization using the unimodal normal distribution crossove," *Journal of Japanese Society for Artificial Intelligence*, vol. 14, pp. 1146–1155, nov 1999.

[12] 北野宏明, “遺伝的アルゴリズム,” **人工知能学会誌**, vol. 7, no. 1, pp. 26–37, 1992.