第 1 章

序論

1.1 背景

人間による操縦を必要とせず、自律的に活動するロボットに対する社会の期待が高まっている []。自律ロボットの活躍が期待される場所は多岐に渡り、従来のロボットのように工場のラインだけでなく、住宅や商業施設、あるいは被災地のような極限環境も含まれる。ロボットには様々な種類が存在するが、中でも自律移動ロボットは、病院内の搬送や空港の警備など、すでに社会実装されたものも多く存在する []。

これらの環境で移動ロボットが自律的に活動するためには、自己位置や周辺環境の把握と行動計画を行う必要がある。実世界で活動するロボットは、自身の位置を直接知ることはできない。したがって、搭載したセンサを用いて自身の位置や周辺環境を把握する必要がある。そして、ロボットは把握した自己位置や環境の情報を頼りに、目標地点へと移動するための行動を計画し動作する。

しかしながら、移動ロボットの活躍が求められている環境の多くは複雑で、不確かな要素が多く存在する。人間の生活する空間は、工業の組み立てラインのように不確かさや誤差がなるべく小さくなるように精密に構成されてはいない。また、センサが知覚できる情報には限りがありノイズが含まれているため、ロボットが環境の情報を完全に知覚することはできない。アクチュエータにおいても、制御ノイズや消耗のような要因から誤差が存在し、モデル化やアルゴリズムの近似による誤差も存在する。したがって、ロボットが実世界で自律的に活動するためには、これら多くの不確かさに対処していく必要がある。

不確かさを考慮するためには、確率論に基づく方法が有効である。ロボティクスにおいて、確率論を用いて不確かさに対処する試みは「確率ロボティクス (Probabilistic Robotics)」と呼ばれ、盛んに研究が行われている [**?**, **?**]。確率ロボティクスは、ロボットの知覚と制御に確率・統計を駆使することで、ロボット技術において避けて通ることができない不確かさを陽に表現することを可能としている。移動ロボットにおいて重要な自己位置推定や行動計画についても、不確かさを考慮した様々なアルゴリズムが提案されている。

この確率論に基づく自己位置推定には、Baysian Filter が有効であると示されている。

確率ロボティクスでは、ロボットの自己位置を決定論的な 1 つの位置ではなく、全ての位置からなる空間中における確率密度関数として表現する。Bayesian Filter による自己位置推定では、ベイズの定理に基づき、ロボットの自己位置を事後確率の分布として推定する。これらの分布は信念分布と呼ばれ、ロボットが主観的に得られる情報を基に自身の位置をどのように信じているかを表している。

現在、多くの移動ロボットにおいて、確率論に基づく自己位置推定が取り入れられている。中でもとくに、Monte Carlo Localization (MCL) という手法が多く利用されている [?][?]。これは、Bayesian Filter によるロボットの自己位置推定を実装する方法の 1 つである。MCL では、ロボットの信念分布を位置と姿勢からなる空間中に分布させた重みを持つ粒子 (パーティクル) で近似的に表現する。これにより、ロボットの信念を複雑な確率分布として表現することを可能としている。同じく Bayesian Filter を理論的背景としたカルマンフィルタによる自己位置推定では、信念は正規分布でしか表現できない [?]。対して、MCL では一様や多峰性の分布などを表現することが可能であるため、複雑な環境で活動することが求められる移動ロボットに適している。ロボット開発用のミドルウェアである ROS では、LiDAR と専有格子地図を用いた MCL による自己位置推定のプログラムが、標準のナビゲーションパッケージとして用意されており、多くの開発者や研究者に利用されている [?, ?]。また、実環境において移動ロボットに自律移動を行わせる技術チャレンジであるつくばチャレンジでは、多くのチームが LiDAR と専有格子地図用いた MCL を用いている。[?] これらのことからも、MCL は実環境における自律移動ロボットの自己位置推定の手法として、有効であることが分かる。

現在、様々な自律移動ロボットにおいて確率的な自己位置推定手法が取り入れられている一方で、行動決定においては不確かさについて考慮されないことがある。一般的に多くの自律移動ロボットは、MCL やカルマンフィルタにより自身の位置を確率分布として推定する。そして、最も確率の高い位置を自身が存在する真の位置だと仮定し、行動計画を行う。しかし、このような方法では、ロボットの自己位置推定により得られた確率的な情報が、行動決定に反映されない。ロボットは信念分布が大きい場合も、分布が推定誤差が小さい場合も、同様の行動を行うことになる。

ロボット同様に、実世界で活動する我々にとっても不確かさは避けることができない問題だが、人間や一部の動物は、得られる情報が制限されている状況下でも、適切な行動を選択する。たとえば、人間は壁沿いを移動することで暗闇の中でも寝室に向かうことが可能であり、明るく視界に制限がないときとは異なる動作を行うことで、不確かさに対処している。このように状況に合わせた自律的な行動をロボットに行わせるためには、行動決定に不確かさを考慮させることが必要である。

1.2 先行研究

不確かさに応じた柔軟な行動決定をロボットに行わせることができると、得られる情報が制限された状況下でもタスクを実行させることが可能となる。ロボット工学では、部分

観測マルコフ決定過程 (POMDP) という枠組みで研究されているが、計算量の観点から最適な解を導くことができないと知られている [?]。そこで、近似的に POMDP を解く手法が複数提案されている。

暗闇の中、壁伝いに寝室へ向かうような行動については、Roy らによって提案された手法によりロボットへと実装されている [?]。書き途中。

Littman らによって提案された Q-MDP 値法 [?]。書き途中。

Q-MDP 値法は、Ueda らによって実際のロボットへと実装された [?]。書き途中。

Ueda によって提案された PFC 法では、移動ロボットがゴールを探索するような動作を可能にしている [?]。この手法では、Q-MDP 法に少し変更を加えることで、ゴールに近いパーティクルが行動決定に及ぼす影響を大きくする。これにより、ロボットは自身の位置がほぼ分からない状況から、パーティクルにより近似された信念を徐々にゴールに流し込むような動作を行う。分布が徐々にゴールに吸い込まれるように移動していき、ロボットが分布の中にいる場合、ロボットもその流れに従いゴールへと到達する。また、ゴール付近のパーティクルが行動決定に及ぼす影響の大きさを変更した際に、Q-MDP 値法で問題となっていたローカルミニマムによる行動の停滞が起こる頻度が低下することが示されている [?]。

PFC 法は、障害物が存在しない空間において、自己位置推定が極めて不確かな移動ロボットのナビゲーションに有効であることが、シミュレーション実験において示されている。しかし、障害物が存在する環境での動作については、有効性は確認されていない。PFC 法では、ゴール付近のパーティクルが行動決定に及ぼす影響を大きくする一方で、ゴールから遠いパーティクルや障害物内に存在するパーティクルは、軽視されることになる。つまり、ロボットが障害物を回避しようとする行動は反映されにくいという性質がある。一般的に、自律移動ロボットが活動することが求められている環境には、動的なものや静的なものを含め、多くの障害物が存在する。移動ロボットのナビゲーションでは、障害物の回避について考えることが必要であると言える。

目的への繋ぎ方検討中。問題提起のような節を分けるべきか。

1.3　目的

そこで本研究では、自己位置推定の不確かさを考慮した障害物の回避行動を自律移動ロボットに行わせるための手法を提案する。PFC 法に少し変更を加えることで、信念を近似するパーティクルクラウド全体が障害物を回避するような動作を生成する。これにより、自律移動ロボットに自己位置推定の不確かさを考慮した障害物回避の行動をとらせつつ、ゴールを探索するような動作を行わせることを可能にする。また、その有効性について通常の PFC 法と比較することで検証する。

1.4　本論文の構成

まず、第 1 章では、本研究の背景と先行研究を述べ、問題と目的を設定した。第 2 章では、不確かさを考慮した行動決定について述べ、第 3 章では、提案する手法について述べる。第 4 章では、提案手法を用いた実験を行い、最後に第 5 章で本論文のまとめを述べる。

第 2 章

移動ロボットの自己位置推定

本章では、Bayesian Filter による移動ロボットの自己位置推定について定式化する。また、Bayesian Filter の実装方法の 1 つである Monte Carlo Localization (以下 MCL) について述べる。本論文においても、移動ロボットの自己位置推定には MCL が利用されるものとする。

2.1 Bayesian Filter による自己位置推定

Bayesian Filter による自己位置推定では、式 (2.1) に示すベイズの定理を用いて、移動ロボットの位置を確率的に推定する。ベイズの定理は、事象 a, b が起きる確率をそれぞれ $p(a), p(b)$ としたときに、

$$y = a \tag{2.1}$$

2.2 Monte Carlo Localization (MCL)

Monte Carlo Localization (MCL) は、Bayesian Filter を理論的背景としたロボットの自己位置推定手法の実装方法の 1 つである。MCL では、表現したい任意の空間上 $\mathcal{X}$ 上に存在する確率密度関数を、空間上に散布した標本により表現することで、ロボットの姿勢を確率的に推定する。この標本はパーティクル（粒子）と呼ばれ、ロボットの信念分布を表す確率密度関数 $b_t(\boldsymbol{x})$ を近似するように配置される。パーティクルはそれぞれパラメータとしてロボットと同次元の姿勢 $\boldsymbol{x}$ と重み w の情報を持っており、N 個のパーティクルの集合は

$$\Xi_t = \xi_t^{(i)} = (\boldsymbol{x}_t^{(i)}, w_t^{(i)} | i = 1, 2, \ldots, N) \tag{2.2}$$

のように定義される。基本的に全パーティクルの重みの合計は、常に

$$\sum_{i=1}^{N} w_t^{(i)} = 1 \tag{2.3}$$

を保つように実装される。

ロボットの状態を表すパラメータについては様々存在するが、自己位置推定ではロボットの姿勢（位置と向き）を推定する。本論文では、二次元平面を低速で移動する対抗二輪型の移動ロボットを想定し、ロボットの二次元平面上における位置 (x, y) と向き θ をあわせた 3 次元の状態 $\boldsymbol{x} = (x, y, \theta)$ の推定について扱う。したがって、ロボットの姿勢と同様に、パーティクルの姿勢はそれぞれ $\boldsymbol{x}^{(i)} = (x^{(i)}, y^{(i)}, \theta^{(i)})$ となる。

MCL のアルゴリズムは、主に次の 4 ステップからなる。2 から 4 を繰り返し行うことで逐次ロボットの姿勢を推定する。

1. Initilize
2. Motion Update
3. Sensor Update
4. Resampling

2.2.1 Initialize

Initialize のステップでは、パーティクルの初期化を行う。N 個のパーティクルの初期姿勢 $\boldsymbol{x}^{(i)}$ を決定し、重み $w^{(i)}$ を $\frac{1}{N}$ とする。パーティクルのばらまき方は、ロボットの初期の信念分布に従うように行う。一般的には、以下のようなパーティクル初期姿勢の決定方法が用いられる。

ロボットを人間が自由に配置する場合など、初期姿勢が予め分かっている場合は、ロボットの信念分布をロボットが存在すると考えられる位置を平均とした正規分布と考え、パーティクルの初期姿勢 $\boldsymbol{x}^{(i)}$ を平均 μ、分散 Σ の正規分布に従うように決定する。本論文における移動ロボットの自己位置推定では、状態 $\boldsymbol{x}^{(i)}$ が位置と向きからなる 3 次元のため、μ と Σ はそれぞれ 3 次元ベクトル、3×3 の分散共分散行列として扱う。多くの場合 μ と Σ はヒューリスティックに決定する。

一方、ロボットの初期位置が完全に不明なとき、ロボットの初期信念分布は状態空間 $\mathcal{X}$ 全体の一様分布と考えられる。その場合は、パーティクルの姿勢は一様分布に従うように配置する。

2.2.2 Motion Update

Motion Update のステップでは、Bayesian Filter における予測ステップを、各パーティクルに対して行う。移動ロボットの状態遷移にはノイズが伴うため、同様の制御入力でも試行ごとに異なる状態遷移結果となる。

2.2.3 Sensor Update

Sensor Update のステップでは、Bayesian Filter における計測更新ステップを行う。各パーティクルの重み $w^{(i)}$ を、ベイズの定理を用いて式 (2.4) のように決定する。センサ

により得られた情報をパーティクルの重みに反映するための処理は、式 (2.4) に示すように、現在の重みに尤度をかけあわせることで行える。

$$w_t^{(i)} = q(\boldsymbol{z}_t|\boldsymbol{x}_t^{(i)})\hat{w}_t^{(i)} \tag{2.4}$$

尤度関数はそれぞれのセンサの観測モデルをもとに決定される。

2.2.4 Resampling

Resampling ステップでは、

第 3 章

状態推定の不確かさを考慮した行動決定

本章では、不確かさを考慮した行動決定について述べる。まず、状態が既知のロボットの最適な行動決定を扱う枠組みである「マルコフ決定過程」について述べる。その後、ロボットの状態が不確かにしか分からない状況での最適な行動を扱う枠組みである「部分観測マルコフ決定過程」について述べる。最後に、「部分観測マルコフ決定過程」を近似的に解く手法について、いくつかあるうち、本研究と関連のある Q-MDP 法および PFC 法について述べる。

3.1　ロボットの行動決定

本節では、ロボットの行動決定について考える。本論文で扱うような移動ロボットの行動決定は、現在のロボットの位置から目的地までの最短経路を算出して移動する、「経路計画問題」として扱われる。ダイクストラ法や A*法、人工ポテンシャル法といった多くの手法が提案されており、現在でも研究が行われている。一般的にロボットの経路計画問題では、ロボットの観測や移動は決定論的なものとして扱われ、経路の算出が行われる。

しかし、これまで述べたとおり、移動ロボットは多くの不確かさを有している。移動ロボットに最短の経路を算出して移動するだけでなく、これらの不確かさを考慮した上でさらに知的な動作を行わせる方法について考える必要がある。たとえば、人間であれば、最短ではあるが危険でゴールに到達できる可能性が低い道よりも、多少遠回りではあるが高確率で安全にゴールできる道を選択するような、「急がば回れ」が有効な場合も存在する。あるいは、自身の位置が正しく把握できていない場合に、安全を優先し全く動かないでいるよりも、多少の危険を冒しとりあえず行動してみてタスク達成を目指す方が有効であるような場合も考えられる。

このような知的な行動を、単純に経路計画問題として考えることは困難である。そこで本章では、ロボットの移動と経路計画について、より一般化した枠組みである（有限）マルコフ決定過程および部分観測マルコフ決定過程について述べる。

3.2 マルコフ決定過程

本節では、移動ロボットの現在の自己位置 $\boldsymbol{x} \in \mathcal{X}$ が既知という前提での行動決定についての定式化を行う。このような問題は、マルコフ決定過程 (Markov decision process, MDP) という枠組みで議論される。時間やロボットの状態等を連続系ではなく離散系で考える場合は、有限マルコフ決定過程と呼ばれる。

3.2.1 状態と終端状態

章 2 で述べたものと同様に、状態変数 x, y, θ からなる状態空間 $\mathcal{X}$ を定義する。ロボットは自身の現在の状態ベクトル $\boldsymbol{x} \in \mathcal{X}$ が分かっているものとし、観測の不確かさについては考慮しない。

ロボットのタスクには必ず終わりがあるとし、ロボットの状態が、事前に定めたある状態になったときをタスクの終了とする。このタスクが終了する状態は、終端状態と呼ばれ、終端状態の集合は $\mathcal{X}_f \subset \mathcal{X}$ と表現される。終端状態は、移動ロボットの経路計画問題における、ゴールや目標地点などのロボットが目指すべき望ましい状態だけでなく、陥りたくない状態も含まれる。

3.2.2 行動と状態遷移

ロボットは有限子の制御指令の中から、一つを選択することで動作する。MDP ではこの制御指令のことを行動と呼ぶ。ロボットの行動は m 種類存在し、行動の集合は、

$$\mathcal{A} = \{a_1, a_2, \ldots, a_m\} \tag{3.1}$$

と表現される。

時間は離散的に表現され、タスクの開始からは終わりまでは $0, 1, 2, \ldots, t_f \equiv T$ と定義される。ロボットが最初に行動を選択する時刻を $t = 0$ とし、行動が実行されるたびに $t = 1, 2, \ldots$ と次のステップへと進んでいく。ロボットが終端状態に入りタスクが終了する時刻を t_f とする。t_f は固定ではなく、タスク達成までにかかった時間により異なる。また、「時刻 $t-1$ の状態」、「時刻 t に遷移するために選択された行動」、「時刻 t の状態」をそれぞれ $\boldsymbol{x}_{t-1}, a_t, \boldsymbol{x}_t$ と表現する。しかし、MDP で扱うシステムは時不変であるため、多くの場合 t の具体的な値は重要でははない。したがて、今後はこれらをそれぞれ $\boldsymbol{x}, a, \boldsymbol{x}^{'}$ と表記する。

ロボットの状態 $\boldsymbol{x}$ は、ある行動 a により状態 $x^{'}$ へと遷移する。状態の遷移にはノイズが含まれているものとし、状態と行動が同一の $(\boldsymbol{x}, a)$ であっても、事後状態 $\boldsymbol{x}^{'}$ は各試行ごとに異なる。ロボットの状態遷移は、マルコフ性を持ち、

$$\boldsymbol{x} \sim p(\boldsymbol{x}|\boldsymbol{x}_{t-1}, \boldsymbol{u}_t)(t = 1, 2, \ldots, t_f) \tag{3.2}$$

に従うものとする。また、ロボットが $\boldsymbol{x}$ から行動 a により $\boldsymbol{x}'$ に遷移する確率を $p^a_{\boldsymbol{x}\boldsymbol{x}'}$ と表現する。この確率 $p^a_{\boldsymbol{x}\boldsymbol{x}'}$ が、$\boldsymbol{x}, a, \boldsymbol{x}'$ のあらゆる組に対して既知であり、時不変であると仮定する。

3.2.3 報酬

状態 $\boldsymbol{x}$ で行動 a を行った場合に、状態が $\boldsymbol{x}'$ に遷移した際に、その状態遷移に対して報酬を与える。たとえば、移動ロボットではこの報酬は、行動一回ごとに消費する電力や時間などを設定する。この報酬は、一つの実数値とし、$r^a_{\boldsymbol{x}\boldsymbol{x}'} \in \mathbb{R}$ と表現される。評価の基準が多次元的の場合も、一つの実数となるように一元化し評価を行う。

3.2.4 評価

ロボットがある状態 $\boldsymbol{x}$ からある終端状態 $\boldsymbol{x}_f \in \mathcal{X}_f$ に到達するまでの一連の状態遷移に対して評価を与える。評価は、

$$J[\boldsymbol{x}_{0:t_f}, a_{1:t_f}] = \sum_{t=1}^{t_f} r^a_{\boldsymbol{x}\boldsymbol{x}'} + V(\boldsymbol{x}_{t_f}) \tag{3.3}$$

で定義される評価値の大きさで行う。$r^a_{\boldsymbol{x}\boldsymbol{x}'}$ は、項 3.2.3 で述べた報酬である。$V(\boldsymbol{x}_{t_f})$ は終端状態の価値とし、事前に目的に応じて与えられるものとする。ロボットが目指すゴールや目標地点とする場合、0 を設定する。逆に、ロボットがその状態になった場合タスクが失敗として終了するような、絶対に陥ってほしくない状態として設定する場合、非常に小さい値を設定する。

本論文では、この評価値 J を最大化する一連の状態遷移を、最適な制御とする。状態遷移にはノイズが伴うため、同じ姿勢 $\boldsymbol{x}_0$ から常に最適となるような行動をとったとしても、試行ごとに評価値 J は異なる。そのため、ある状態から終端状態までの一連の状態遷移に対する評価 J は、期待値として考える。

3.2.5 最適方策

ある状態における、J の期待値を最大化するための行動を与える関数

$$\Pi : \mathcal{X} \to \mathcal{A} \tag{3.4}$$

を定義する。この関数 Π は最適方策と呼ばれる。最適方策 Π が決まると、任意の状態 $\boldsymbol{x}$ でロボットが取るべき行動は、

$$a = \Pi(\boldsymbol{x}) \quad (\forall \boldsymbol{x} \in \mathcal{X}) \tag{3.5}$$

と自動的に決まる。

3.2.6 最適価値関数

また、ある状態に対して評価値の期待値 J を与える関数

$$V : \mathcal{X} \to \mathbb{R} \tag{3.6}$$

は、最適価値関数と呼ばれる。$V(\boldsymbol{x})$ は、$\boldsymbol{x}$ が初期の状態でも、タスクにおける途中の状態でも変わらない。終端状態の価値 $V(\boldsymbol{x}_{t_f})$ は、この最適価値関数に含まれる。

3.3 部分観測マルコフ決定過程

本節では、移動ロボットの現在の状態が不確かな中での行動決定について述べる。このような問題は、部分観測マルコフ決定過程 (partially observable Markov decision process, POMDP) という枠組みで議論される。

3.4 Q-MDP

本節では、Litteman らが提案した手法である、

3.5 Probabilistic Flow Control