

令和 7 年度 卒業論文
自律移動ロボットのための価値反復と
A*探索を組み合わせた大域経路計画

中村啓太郎
China Institute of Technology

2026 年 1 月 29 日

謝辞

本研究は JSPS 科研費 JP24K15127 の助成を受けたものです。

本論文は，千葉工業大学先進工学部未来ロボティクス学科 自律ロボット研究室（上田隆一研究室）において執筆しました。

研究室では，多くのメンバーと共に過ごし，様々な刺激をいただきました。その中で，私の研究を手伝っていただいた皆様に感謝します。

修士 2 年の吉越さんには，実機実験で使用するロボットの不調の折に大変お世話になりました。長年の経験からアドバイスをいただき感謝します。

また，修士 1 年の佐々木さんには，ロボットの改修でお世話になりました。ハードウェアの知識の足りない私にとってよい学びとなりました。

修士 1 年の永木さんには，同じく価値反復を用いた研究を行っていることから，作成したシミュレータ環境を使わせていただきありがとうございました。

修士 2 年の船井さん，修士 1 年の茂さんと川原さんとザンダーさん，同級生である市東さん，鷺尾さん，鈴木さん，藤野さんには，よく研究室での話し相手になっていただきありがとうございました。私がよく研究に行き詰まると，勝手に話しかけていましたが，快くお相手してくださり幾度も助けられました。

学部 3 年の，辻さん，水牧さん，和田さん，根本さん，平地さん，は，私のくだらない話にも付き合ってくださりありがとうございました。来年度の活躍を楽しみにしています。

最後に，本研究に取り組み，2 度の学会の予稿と本稿を執筆するにあたり，ご指導を頂いた上田隆一教授に感謝します。

目次

謝辞	iii
第 1 章 序論	1
1.1 研究背景	1
1.2 従来研究	7
1.3 研究目的	13
1.4 論文の構成	13
第 2 章 移動ロボットの経路計画問題	15
2.1 問題設定	15
2.2 価値反復アルゴリズム	17
第 3 章 提案手法	21
3.1 価値反復と A^* を組み合わせた大域計画	21
3.2 3D A^* の適用	22
第 4 章 実装	23
4.1 A^* の実装と価値反復への適用	23
第 5 章 実験	25
5.1 シミュレータ実験	25
5.2 実機実験	26
第 6 章 結論	27
参考文献	29

第 1 章

序論

1.1 研究背景

1.1.1 ロボティクスに対する社会的要請

少子高齢化と労働生産性の課題

21 世紀に入り，先進諸国において少子高齢化に伴う生産年齢人口（15～64 歳の人口）の減少が深刻な社会問題となっている．特に日本においては，国立社会保障・人口問題研究所の推計によると，生産年齢人口は 1995 年をピークに減少傾向にあり，2070 年には約 4,500 万人（2020 年比で約 4 割減）まで落ち込むことが予測されている [国立 23]．この人口構造の変化は，経済成長の鈍化のみならず，社会インフラの維持そのものを脅かす要因となっている．

物流業界においては，物流クライシスと呼ばれる状況が顕在化している．電子商取引（E-commerce）の爆発的な普及により，小口配送の需要が急増している．国土交通省の調査によれば，宅配便取扱個数は年間 50 億個（2023 年度）を超え，過去 10 年間で約 1.3 倍に増加している [国土 24]．トラックドライバーや倉庫内作業員の不足は慢性化しており，労働環境の悪化と配送網の維持困難が懸念されている．また，製造業の現場においても，熟練工の引退に伴う技術継承の問題や，単純搬送作業への人員配置の困難さが指摘されている．

さらに，医療・介護の分野では，高齢者人口の増加に対し，介護従事者の数は圧倒的に不足している．厚生労働省の推計では，2040 年度には約 272 万人の介護職員が必要とされているが，現状のままでは数十万人規模の供給不足に陥る可能性が指摘されている [厚生 24]．病院内での検体搬送，リネン類の回収，あるいは介護施設における見守りや配膳など，定型的な業務の負担軽減は，強い需要がある．

ロボットによる業務自動化と限界

こうした労働力不足を補う手段として，工場内物流においては，無人搬送車（Automated Guided Vehicle: AGV）の導入が進められてきた．AGV は，床面に敷設された磁気テー

プや反射テープ、あるいは二次元コードといった物理的なガイドをセンサで読み取りながら走行するロボットである。これにより、従来人の行っていた台車を押すような運搬作業をロボットが代替することが可能になった。AGVは環境が固定されており、かつタスクが定型的である場合に高い効率と信頼性を発揮する。

一方で、AGVの導入と運用には、物理的なガイドを必要とするという制約から以下のような構造的な限界が存在すると指摘されている [Fragapane 21]。これらの限界は、人や有人フォークリフトが頻繁に行き交う物流倉庫や、一般の人々が存在する病院・商業施設といった動的な障害物の多い環境へのロボット導入を阻む大きな障壁となっていた。

1. インフラ敷設コスト: AGVに走行させたい経路のすべてにガイドを設置する必要があり、導入時の工事コストや期間が甚大である。
2. レイアウト変更の柔軟性欠如: 製造ラインや倉庫のレイアウトを変更する際、ガイドの敷設し直しが必要となり、多大なコストとダウンタイムが発生する。
3. 動的環境への非適応: 想定された経路上に障害物が存在した場合、AGVはその場で停止することしかできず、回避して目的地へ向かうことができない。

AGVから自律移動ロボット (Autonomous Mobile Robot: AMR) へ

AGVの課題を克服するため研究、開発が進められてきたのが、自律移動ロボット (AMR) である。AMRは、図 1.1 に示すように、物理的なガイドを必要とせず、LiDAR (Light Detection and Ranging) やカメラといった外界センサを通じて周囲の環境の情報を得る。このセンサ情報と事前に作成した環境地図を照らし合わせることで地図内の自身の位置を推定し、ガイドを必要とせず走行する。

AMRは、AGVと違い、経路上に障害物を検知すると、回避経路を生成し、その経路を追従することでタスクを継続することが可能である。これは、AMRが走行する経路は、物理的なガイドによる経路ではなく、ソフトウェア上の経路を走行するために、動的に経路を変更することが可能であるため可能である。

また、物理的なガイドを必要としないため、導入時の工事が不要であり、ソフトウェア上の設定変更のみで走行エリアや経路を変更できる。この特性により、AMRは従来AGVが導入困難であった動的な環境への適用が進んでおり、Society 5.0の中核を担う技術として期待されている [内閣 16]。

自律移動ロボットの屋外への適用

工場や倉庫といった屋内環境で培われたAMRの技術は、近年、より複雑かつ広範な屋外環境へとその適用範囲を拡大している。特に、労働力不足が深刻な物流や農業分野において、屋外対応型AMRの実用化が急速に進展している。

物流分野では、配送拠点からエンドユーザーへの最終区間であるラストワンマイルの配送コスト削減が最大の課題となっている。人の代わりに荷物の配送を行う自律移動ロボットは、従来のトラック配送と比較して、配送時間の短縮と環境負荷の低減を実現する有効

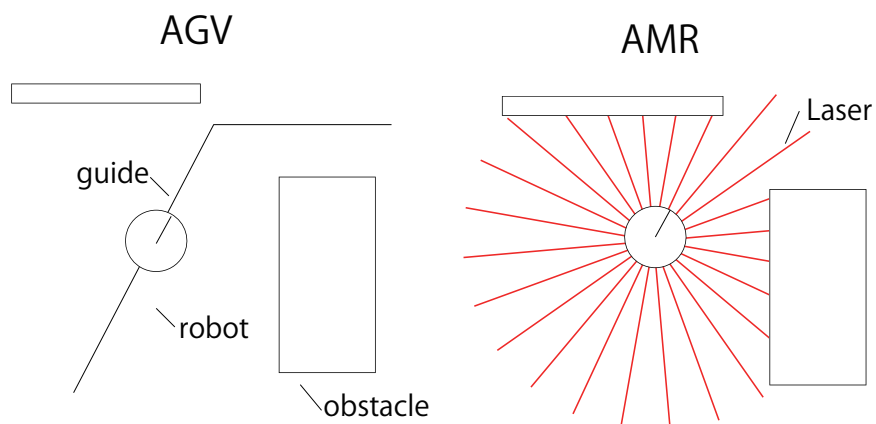


図 1.1 Comparison of AGV and AMR

な手段として位置付けられている [Alverhed 24]. 日本国内においても, 2023 年 4 月に施行された改正道路交通法により, 自律移動ロボットが遠隔操作型小型車として定義され, 届出制による歩道走行が可能となった [日本 22]. これにより, パナソニックや楽天といった企業が図 (1.2) に示すようなロボットを用いて, 住宅街での配送実証を行っており, 社会実装が進みつつある.



図 1.2 Rakuten's Autonomous Delivery Robot(Source: [楽天 25])

また, 農業分野では, 農林水産省がスマート農業を推進しており [日本 24], クボタやヤンマーなどの農機メーカーが有人監視下での自動走行 (レベル 2) および無人自動走行 (レベル 3) に対応したロボットトラクターを市場投入している [株式 17, 横山 19]. 同様に, 建設現場や鉱山といった過酷な環境においても, 資材搬送や巡回監視を行う AMR の

導入が進められている。

しかしながら，屋内環境と比較して，屋外環境はロボットにとってタスクの遂行が困難となる要素が多い．第一に，ロボットの走る路面のロボットへ与える影響がある．多くの移動ロボットは車輪型 [原 25] であり，平らで傾きのない屋内では，路面からロボットへ与える影響は小さい．一方，屋外では，路面に段差や凹凸があり，ロボットの姿勢の急激な変化や振動によるセンサノイズといった影響がある．第二に，天候や季節による路面状況の変化や，時間変化による照明条件の変化がある．雨，泥，雪，あるいは落ち葉などは，風景の見た目を大きく変化させる．また，直射日光による白飛びや逆光，夜間の低照度，朝日，夕日による色温度の変化といった光環境の変動は，視覚情報の大きな変化を伴う．

これらの環境要因からロボットがどのように周囲を認識し，どのように自身の位置を知り，どのように経路を引くかという課題が発生する．次項では，自律移動を実現するために現代の AMR がどのような技術要素によって構成されているか，どのような課題が存在するか，そのシステム概要について述べる．

1.1.2 自律移動ロボットの技術構成

センシング

センシングは，ロボットが周囲の環境から情報を得るための技術である．以下に，移動ロボットで用いられる代表的なセンサ技術を示す．

- **LiDAR**: レーザー光を照射し，反射光が戻ってくるまでの時間（Time of Flight）や位相差から距離を計測するセンサ．北陽に代表される 2 次元平面をスキャンする 2D LiDAR が主流であったが，図 1.3 右に示す Velodyne Lidar に代表される 3 次元点群を取得可能な 3D LiDAR が主流になりつつある．近年では，図 1.3 左に示す Livox など中国メーカーの特に廉価な LiDAR が多く流通している．
- **カメラ**: 可視光線を計測し，人の知覚する色の分布を捉えることができるセンサ．RGB 画像に加え，深度情報を取得できる RGB-D カメラやステレオカメラが利用される．Visual SLAM や物体認識との親和性が高い．@@@SLAM という言葉が初出です．略語の説明を．他の略語も注意を．SLAM 自体についても（後述）と書くか，その場で短く説明が必要です．@@@
- **オドメトリ（Odometry）**: 車輪の回転数（エンコーダ値）や IMU（慣性計測装置）のデータから，ロボットの相対的な移動量を推定すること [前山 97]．短期的には高精度だが，累積誤差が生じるため単独では長距離移動に適さない．
- **GNSS（Global Navigation Satellite System）**: アメリカの GPS（Global Positioning System）や日本の QZSS（Quasi-Zwynth Satellito Ststem, みちびき）といった衛星測位システムの総称．屋外環境において地球上での位置を取得する主要な手段となる [塚越 16]．基準局を用いて誤差の補正を行う RTK-GNSS（Real Time Kinematic-GNSS）により，数センチメートルの精度での測位が可能となり，

配送や農業を行うロボットに広く採用されている。

- **無線通信 (WiFi/5G):** アクセスポイントからの信号強度や電波の到達時間を用いて測位を行う。

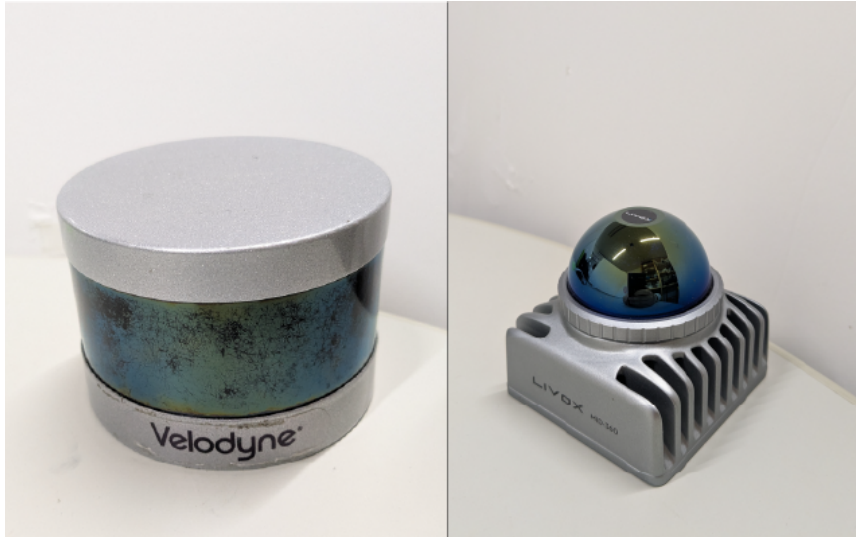


図 1.3 3D LiDAR (Right: Velodyne, Left: Livox)

自己位置推定 (Localization)

自己位置推定は、周囲の環境を測定するセンサの情報からロボットがロボット自身の位置を推定する技術である。単に、自己位置推定といった場合は、各地点で環境から取得できる情報を記した地図を予めロボットが持っており、その地図上の自身の座標を推定することを指す。地図を持たない場合は、SLAM (Simultaneous Localization and Mapping) とよばれる。

■**SLAM** SLAM はロボットがセンサ情報と制御入力だけで地図を作る技術である [Thrun 05]。SLAM は、主に用いるセンサごとに LiDAR SLAM と Visual SLAM に大別される。それぞれの SLAM ごとに、代表的な実装例を以下に示す [友納 20]。

- **LiDAR SLAM:** 2D LiDAR を用いたものでは、パーティクルフィルタベースの GMapping[Gerkey 09] やグラフベースの Cartographer[Kohler 16] などがある。3D LiDAR では、スキャンマッチングによる Velodyne SLAM[?] や LOAM[?] などがある。近年の日本では、GLIM[Koide 24] がシェアを伸ばしている [原 25]。
- **Visual SLAM:** カメラ画像の特徴点を用いる ORB-SLAM[MA 15] や、直接法と呼ばれる密な点群を用いる LSD-SLAM[?] などがある。

■**地図あり自己位置推定 (Map Based Localization: MBL)** オドメトリの累積誤差を補正し、地図座標系上での絶対位置を特定するために、様々な手法が開発されてきた。1990

年代以降, Thrun らによって提唱された確率的ロボティクス (Probabilistic Robotics) [Thrun 05] は, センサのノイズや環境の不確実性を確率分布として扱うことで, ロバストな自己位置推定を可能にした. 代表的な手法として, (拡張) カルマンフィルタとパーティクルフィルタ (Particle Filter) がある. パーティクルフィルタの実装方法として広く使われているのが, モンテカルロ位置推定 (Monte Carlo Localization: MCL) [Fox 99] である. これは, ロボットの存在確率分布を多数の粒子 (パーティクル) で表現し, センサ観測と動作モデルに基づいて粒子の重みと位置を更新する手法である. MCL は, 開発された当時, ロバスト性において他の手法より優れており, 現在の AMR のデファクトスタンダードとなっている.

自己位置推定の成熟により, ロボットは環境内での自身の位置を推定できるようになった. また, SLAM により, 障害物の配置や通行可能な領域を表す環境の地図を作成することも可能となった.

経路計画 (Path Planning)

ロボットが実際にタスクを遂行するためには, 自己位置推定だけでなく, 現在地 (Start) から目的地 (Goal) まで, 障害物を回避しつつ, かつ効率的 (最短時間, 最小エネルギーなど) に到達するための行動を決定しなければならない. この行動を決定するための技術を経路計画という.

経路計画は, 自己位置推定の結果と環境地図, 目的地を入力とし, ロボットのアクチュエータ (モータ) への制御指令を出力とする, 自己位置推定がいかに正確なものであっても, 経路計画が不適切であれば, ロボットは遠回りをするか, 狭い通路で立ち往生するか, 最悪の場合は動的な障害物と衝突する危険性がある.

前章で述べたように, 社会実装が進むにつれて, ロボットが稼働する環境はより複雑化している. 静的な障害物だけでなく, 人や他のロボットといった動的な障害物, 路面に存在する微小な障害物によるノイズや, 環境の変化によるセンサ計測の不確実性などがその例として挙げられる. これらの要因を考慮し, 安全かつ条件に対して最適な経路をリアルタイムに導出することは, 自律移動ロボットの社会実装を進めるにあたって, 重要な研究課題である. 次節では, この経路計画に関する従来研究を概観し, 本研究で扱う価値反復法の位置付けを明らかにする.

1.2 従来研究

1.2.1 決定論的アプローチによる経路計画

探索手法を用いた経路計画器

経路計画には、離散数学におけるグラフ探索問題を応用して経路を求める手法がある。環境を、各要素を表すノードとそのノード同士の関係を表したエッジからなるグラフ構造としてモデル化することで、現在地（スタート）からゴールまでの経路を探すことをグラフ上を探索する問題に帰着できる。グラフ探索問題は、与えられたグラフ内に、スタートとゴールのノードが設定され、エッジをたどりノードを移動してゆき、最短の移動で、ゴールのノードにたどり着く 1 通りのノードとエッジの列を求める問題である。

グラフの構築方法には、環境の地図を格子状（グリッド）に分割し、各セルをノード、隣接セルへの移動をエッジとしてモデル化するグリッドベースの手法と、空間内にノードをランダムにサンプリングしてグラフを構築し、探索を行う手法がある。後述する Dijkstra 法 [E.W. 59] や A* アルゴリズム [Hart 68] は、主にグリッドベースの手法のグラフで探索を行う。

グリッドベースの手法では、多くの場合、セルに、障害物があり通行不可能、障害物がなく通行可能、不明の 3 種類を設定し、障害物がなく通行可能なセルからなるノードだけをたどる経路を算出することが求められる。また、エッジは、1 つのセルから隣接する 8 つのセルにエッジが繋がれており、繋ぐノード間の距離を重みとして持つ。軸方向の距離（重み）を 1 としたとき、斜めに移動するエッジは、その $\sqrt{2}$ 倍になる。

ただ、グリッドベースの手法は、格子状に区切った全てのセルをノードとするため、高次元の構成空間（Configuration Space）を持つロボットや、広大な環境においては、グラフに含まれるノードの数が多くなり、探索にかかる計算量が多くなる問題点がある。これをランダムにサンプリングしたノードで構築したグラフに置き換えることで、適切な量のノードがあれば十分に構成空間を覆ったグラフでかつグリッドベースのグラフに比べノードの数を少なくすることができる。代表的な手法に RRT がある。

■Dijkstra 法 Edsger W. Dijkstra によって考案された Dijkstra 法は、非負の重み付きグラフにおける単一始点最短経路問題を解くアルゴリズムである [E.W. 59]。各エッジの重み（＝距離）は、常に正であり、この重みを移動にかかるコストとして、スタートのノード n_s からゴールのノード n_g までこのコストが最小になるような、ノードの列（＝経路）を算出する。 n_s からあるノード n まで、累加したコストを $g(n)$ とするとき、 n_s から順に、移動可能な隣接するノードへの $g(n)$ （＝移動する距離）を計算し、計算したノードの中から $g(n)$ が最小のノードをコストが確定したノードとする。新しく確定したノードから移動可能な隣接するノードの $g(n)$ を再度計算し、最小のものを選り、と探索範囲を広げていく。常に $g(n)$ が最小となるノードから移動可能なノードからコストを計算することで、数学的に最短経路が保証される。しかし、探索が全方位に均等に広がるため、ゴー

ルの方角情報利用されず，探索範囲が膨大になる欠点がある．

■A*アルゴリズム A* (A-Star) アルゴリズムは，Dijkstra 法にヒューリスティック関数 $h(n)$ を導入することで探索を効率化した手法である．[Hart 68] Dijkstra 法の $g(n)$ に替わり用いる評価関数 $f(n)$ を以下のように定義する．

$$f(n) = g(n) + h(n) \quad (1.1)$$

ここで， $g(n)$ は， n_s からノード n までの実コスト， $h(n)$ はノード n から n_g までの推定コストである． $h(n)$ は，人の手によって設計され，移動ロボットの場合は， $h(n)$ として現在のセルからゴールセルまでのユークリッド距離やマンハッタン距離が用いられる． $h(n)$ が実際の最短コストを決して上回らない（許容的な，Admissible）場合，A*アルゴリズムは最適解を保証しつつ，Dijkstra 法よりも少ない計算量で解に到達できることが多い．現在でも最も広く使われている標準的なアルゴリズムである．

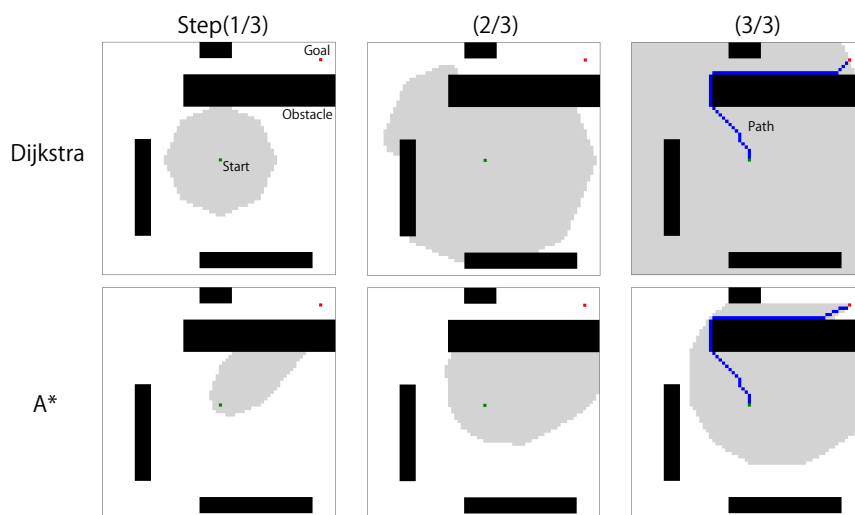


図 1.4 Comparison of search ranges between Dijkstra and A*

■RRT (Rapidly-exploring Random Tree) RRT は，グラフをツリー状として，ランダムにサンプリングされた点に向かってツリーを拡張していくことで，経路を探索する [LaValle 98]．計算量が特に少ないという利点がある一方で，RRT によって生成される経路は最適性が保証されず，ジグザグで遠回りな経路になりがちである．これを改良した RRT* (RRT-Star) [Karaman 11] は，無限回の探索で最適な経路が見つかる保証を有するが，計算量は，RRT に比べ多いものとなる．

これらの，スタートからゴールまでの経路を求めることを，大域経路計画 (Global Path Planning) と呼び，ロボットは算出した経路を追従することで，ゴールに向かうことができる．しかし，この経路は，予め地図に記された静的な障害物のみを回避する経路であり，動的な環境において現れる，経路上の障害物を回避することが求められる．動的な障害物の回避方法として，障害物をもう一度探索を行う方法がある．地図にない障害物によ

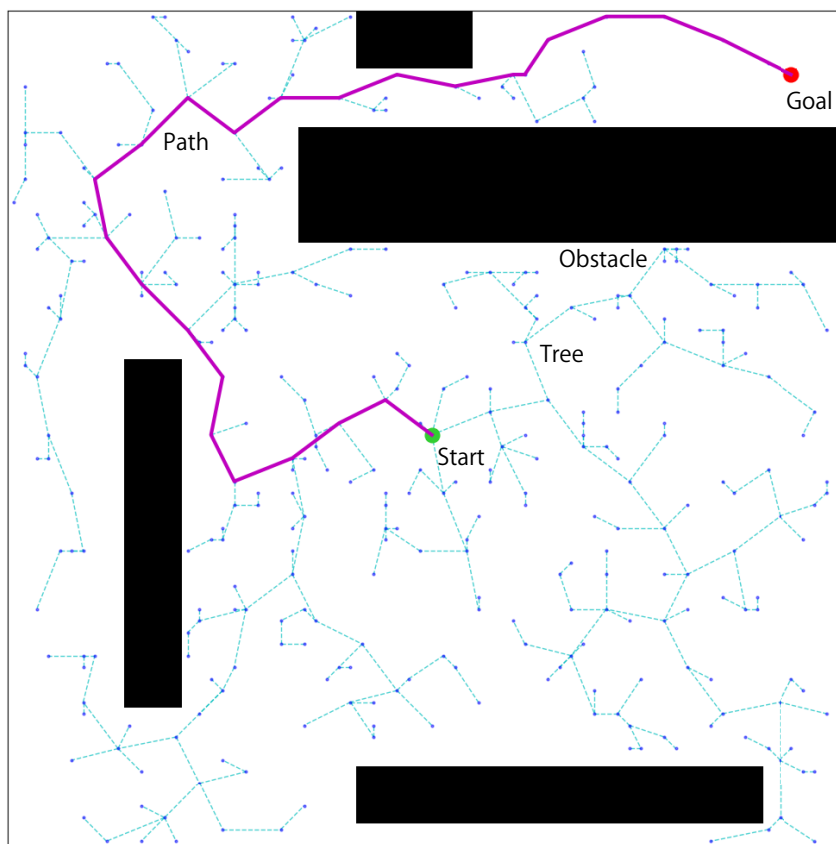


図 1.5 RRT search result

る影響を考慮し、幾度か探索を行う場合、再度探索を行うタイミングは、数秒～数分に 1 回か、または通路が塞がれるなどトポロジカルな構造の変更時などに設定される。

障害物を回避する他の方法として、動的な障害物を避ける短い経路を算出しながら大域経路を追従する局所経路計画 (Local Path Planning) がある。局所経路計画は、大域経路に追従しつつ、搭載されたセンサで検知した未知の障害物や動的障害物をリアルタイムに回避するための制御入力を生成する。更新頻度は高く (10Hz～100Hz)、ロボットの運動学的制約や動力的制約を考慮する。代表的な手法には、Dynamic Window Approach (DWA) や Model Predictive Control (MPC) などがある。

しかし、再度大域経路計画を行う方法や、局所経路計画を併用する方法を用いるときは、ロボットが行ったり来たりするチャタリングに陥ることがある。チャタリングに陥ると、ロボットは自力でのタスクの続行が困難になってしまうため、経路計画は、チャタリングを回避しなければならない。

ポテンシャル法

このチャタリングを回避する方法として、グラフ探索手法が、経路という点の集合を出力するのに対し、環境全体に値 (ポテンシャル) を対応付けたポテンシャル関数を定義し、

ポテンシャルの勾配にしたがって移動する手法が存在する。これらの手法は、大域経路計画と局所経路計画を同一の計算方法で計算でき、両者の矛盾（＝チャタリング）を回避できる利点がある。他にも、自己位置推定の結果のジャンプや振動に対応しやすいことも利点として挙げられる。

■人工ポテンシャル法 (Artificial Potential Fields) 人工ポテンシャル法は、ゴールからの引力と障害物からの斥力を合成したポテンシャル場を構築する手法である [Khatib 85]。電磁気学のクーロン法則を応用し、仮想的に、ロボットを弱い正の電荷を持つ粒子とし、障害物には強い正の電荷、ゴールには強い負の電荷を与える。すると、ロボットはポテンシャルの勾配に従って最もエネルギーが低い方向へ移動するだけで障害物を回避し、ゴールへ向かうことができる。これは、動的な障害物を追加しても計算負荷が非常に軽く、リアルタイムな障害物回避に適している。しかし、人工ポテンシャル法には、ゴール以外の窪みにハマって出られなくなる現象である局所解 (Local Minima) に陥ってしまうという問題がある。U 字型の障害物などに遭遇した場合、引力と斥力が釣り合ってしまう、ゴールに到達する前にポテンシャルの極小値で停止してしまう現象が発生する。

■ナビゲーション関数 (Navigation Functions) 局所解の問題を解決するために、Koditschek と Rimon [Koditschek 90] は、ナビゲーション関数の概念を提唱した。これは、幾何学的な構成空間において、ゴールのみを唯一の大域的最低点 (Global Minimum) とし、その他すべてのポテンシャルの勾配が 0 となる点が周辺にポテンシャルの下る勾配を持つような不安定な鞍点 (Saddle Point) となるように設計された特殊なポテンシャル関数である。ナビゲーション関数が構築できれば、その勾配に従うだけで、ロボットはどのような初期位置からでも必ずゴールへ到達できることが数学的に保証される。

グリッドマップ上におけるナビゲーション関数の実装例として、Konolige の Gradient Method [Konolige 00] が挙げられる。Gradient Method では、 k 個の連続したセル p からなる集合（＝経路） $P_k = \{p_0, p_1, \dots, p_k\}$ に対して、ナビゲーション関数 $N(P_k) \in \mathbb{R}$ を以下の式で計算する。

$$N(P_k) = \min_{P_k} \left\{ \sum_{i=0}^k I(p_i) + \sum_{i=0}^{k-1} D(p_i, p_{i+1}) \right\} \quad (1.2)$$

ここで、 p_0 はゴールのセルであり、 $I(p_i)$ は、 p_i の路面の凹凸や、障害物との距離など、そのセルを通るとき、ロボットにかかる悪影響を数値化したコスト、 $D(p_i, p_{i+1})$ は、2 点 p_i, p_{i+1} のユークリッド距離である。

この N を Wavefront Algorithm [Arkin 90] を改良した LPN 用いて計算し、これをポテンシャル関数とする。こうして得られた場合は局所解を持たず、大域的に最適な経路情報を内包している。

1.2.2 確率的なアプローチによる経路計画

価値反復 (Value Iteration)

@@@[Bellman 1957]???@@@ これらの決定論的な大域経路計画に対し，本研究で取り扱う価値反復は，移動の不確実性や確率的な遷移を扱えるようにしたアルゴリズムである．この手法は，環境の不確実性を確率的に扱うことが可能であり，外乱に対してロバストな方策 (policy) を得ることができる．

マルコフ決定過程 (MDP) による定式化

価値反復では，環境をマルコフ決定過程 (MDP) としてモデル化する．MDP は，状態空間 \mathcal{S} ，行動の集合 \mathcal{A} ，状態遷移モデル \mathcal{P} ，報酬モデル \mathcal{R} の 4 つ組で定義される [中居 22, 上田 19]．また，MDP では，次の状態は，その 1 つ前の状態によって決まり，1 つ前以前の状態は，影響しないというマルコフ性と，完全に状態が観測可能であるという 2 つの仮定を置いている．決定論的なグラフ探索とは異なり，MDP では，「行動 a を行った結果，確率的に状態 s' へ遷移する」という状態遷移の不確実性を考慮して経路計画を行うことができる．

価値反復アルゴリズム

価値反復は，MDP を近似した有限マルコフ決定過程 (finite MDP) の枠組みの中で，方策 Π を計算するアルゴリズムである [上田 19]．finite MDP では， \mathcal{A}, \mathcal{S} が離散化され，有限の集合となる． Π は，式 (1.3) に示すように状態に対して，行動を対応付けた関数である．また， Π を計算するためには，式 (1.4) に示す状態に対して値を対応付けた状態価値関数 V を計算する必要がある．この V は，ポテンシャル関数やナビゲーション関数と同等なものである．

$$\Pi : \mathcal{S} \rightarrow \mathcal{A} \quad (1.3)$$

$$V : \mathcal{S} \rightarrow \mathbb{R} \quad (1.4)$$

最適な方策 π^* を導出するために，価値反復では以下のベルマン方程式 (Bellman Equation)

$$V^*(s) = \max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \mathcal{P}(s'|a, s) [\mathcal{R}(s, a, s') + V^*(s')] \quad (1.5)$$

を用いる．この式 (1.5) を全状態に対して適用し，反復計算することで， π^* と，環境内のあらゆる場所からゴールへ向かうための最適状態価値関数 V^* (最適なポテンシャル関数) が波及的に計算される．

価値反復は，ナビゲーション関数と同様に，局所解が発生しないという強力な数学的保証がある．また，確率的な状態遷移を用いて V を計算することで，実ロボットの移動誤差を，数値化したリスクとしてではなく，価値の期待値として計算し，行動を判断できる．

具体的には、ロボットが壁際を走行する際のリスクを考慮した経路計画が可能になることを示している [Thrun 05].

しかしながら、価値反復は全状態空間を離散化し、計算を行うため、広大な環境においては計算コストが甚大になるという課題が残されている。価値反復の離散化は、グリッドベースの手法よりも角度の次元の分 1 つ次元が大きいため、マップのサイズが大きくなった際の計算コストの増加が Dijkstra 法や A* アルゴリズムよりも大きい。

価値反復 ROS パッケージ

その一方、上田らは、将来的には、計算機の性能が向上し、この計算コストの問題が解決されるとして現在の移動ロボットで使われるミドルウェア (ROS) 上で実装した [Ueda 23, 上田 21, 上田 24]. 未知障害物の迂回による大域的な局所計画も行うことができた。

しかし、現代の計算機でも、計算には時間がかかる。また、自己位置推定の不確かさを考慮した経路計画 [上田 23] のように構成空間を拡張したり、純粋に地図が大きくなると、計算機の性能が向上しても計算に時間がかかることが考えられる。広大な環境を高い解像度でグリッド化すると、状態数は 6 億に達する。15[m] の経路計画に 30[s] 程度かかることを試算されている。ロボットの走り出しに時間がかかってしまう。

1.3 研究目的

以上から、価値反復 ROS パッケージにおいて、現代のコンピュータを用いて単純な経路を求める場合でも、走り出すまでに時間がかかる問題があることが分かる。ロボットが走り出すためには、価値反復によって生成される V の勾配が、ロボットの居る地点まで届く必要がある。

そこで、価値反復よりも計算量の少ないアルゴリズムで経路を算出し、その経路を用いることで勾配を生成すると、ロボットがより早いタイミングで走り始めることが期待できる。

よって、本研究の目的を、「価値反復 ROS パッケージを用いたナビゲーションにおいて、走り出しまでの時間を短縮すること」とする。走り始めるまでの時間が短くなることで、移動全体にかかる時間も短くすることができると考えられる。

V は、どのような初期値からも収束するため、 V の値を書き換えは、 V^* に影響は与えないが、計算中の V に対しては、影響を与えることができる。この影響により全体の移動時間が伸びることがないようにしなければならない。

1.4 論文の構成

- 1 章では、移動ロボット研究の背景、従来研究、本研究の目的を述べた。
- 2 章では、移動ロボットの経路計画問題について述べる。
- 3 章では、提案する手法について述べる。
- 4 章では、3 章の手法の実装について述べる。
- 5 章では、3 章のアルゴリズムについて、移動時間の短縮の効果を評価する。
- 6 章でまとめる。

第 2 章

移動ロボットの経路計画問題

2.1 問題設定

本研究では，移動ロボットを平面上で自律移動させる問題を扱う．これは，移動ロボットのプロセス間通信や，プラットフォームとしてよく用いられているミドルウェア ROS 2[Macenski 22] で広く用いられている Navigation2[LLC 18]（ROS 2 の標準ナビゲーションパッケージ）が扱う問題と同様の問題である．ロボットは，行動を始めるタイミングで目的地の座標を与えられ，障害物との衝突を避けながらできる限り短い時間で目的地まで到達しなければならない．ロボットが移動を行う空間を環境と言い，図 2.1 にその環境の例を示す．

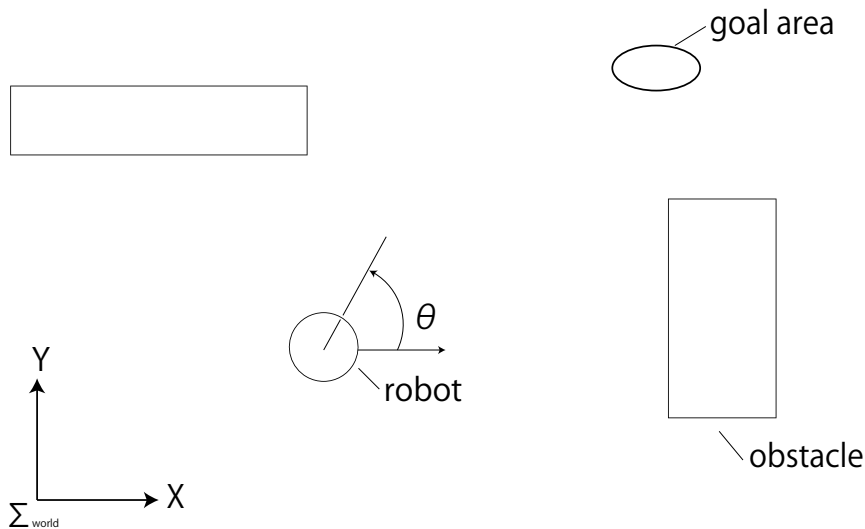


図 2.1 An enviroment for mobile robot navigation

環境には，世界座標系が 2 次元の直交座標系で設定されており，ロボットは，位置 (x, y) と， x 軸となす角 θ を向きとして持っている．これらをまとめてロボットの状態（位置と向き） $s = (x, y, \theta)$ と，3 つの変数で表す．

目的地（ゴール）は， XY 平面上の座標あるいは， $xy\theta$ 空間上の座標として与えられ，

図中の goal area のようにその座標を中心とした XY 平面上の領域や, $xy\theta$ 空間内の領域として扱われる. 実装によっては, 任意の領域を目的地とすることができる.

環境中には, 地図に記されておりその位置が既知である固定障害物と地図に記されない未知障害物が存在する, 未知障害物のうち, 特にロボットが移動する時間スケールで位置が変わる人や他のロボットなどの移動障害物と呼ぶ. 未知障害物は, 価値反復 ROS パッケージの既存の機能で対処可能である. しかし, 本研究では, 走り出しに關与する大域経路計画を扱うため, 局所経路計画である未知障害物の回避は陽には扱わない.

2.1.1 マルコフ決定過程 (Markov Decision Process: MDP) による定式化

この環境を MDP という枠組みで定式化する. MDP は, エージェント (ここでは, 移動ロボット) が環境と相互作用しながら学習・行動決定を行うための数理モデルであり, 以下の 4 つの要素の組 $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} \rangle$ で定義される. ここで, MDP は, マルコフ性を仮定しており, これは, 時刻 t から $t+1$ の間には, $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R} \rangle$ 以外のものは, 関与しないという性質である. 例えばロボットを長時間動かすなど, ロボットの動きが悪くなり \mathcal{P} が変化するが, このようなロボットをモデル化するには, マルコフ性を保つために, \mathcal{S} にロボットをメンテナンスしてから動かした時間などを追加する必要がある.

1. 状態空間 \mathcal{S} (**State Space**): ロボットが取り得るすべての状態の集合. 2.1 章で述べたように, ロボットは, 位置 (x, y) とロボットの方位 θ を持つ. これらを合わせた (x, y, θ) が一つの状態 $s \in \mathcal{S}$ に対応する.
2. 行動の集合 \mathcal{A} (**Action Space**): 各状態でロボットが選択可能な行動の集合. 価値反復 ROS パッケージでは, 速度 v と角速度 ω の組で表される. 例えば直進であれば, $(v, \omega) = (0.3[m/s], 0[rad/s]), (0.2, 0.1)$ @@@単位は前が数字の場合はカッコでかこまないそうです. あと, ブロック体です. $0.3[m/s]$ が正解@@@ 行動 $a \in \mathcal{A}$ となる.
3. 状態遷移モデル $\mathcal{P}(s'|a, s)$ (**Transition Probability**): 状態 s で行動 a を選択したときに, 次の時刻に状態 s' へ遷移する確率.

$$\mathcal{P}(s'|a, s) = \Pr\{S_{t+1} = s' | S_t = s, A_t = a\} \quad (2.1)$$

決定論的な環境 (A^* などが想定する世界) では, ある行動を行えば 100% 意図した隣接セルへ移動する. しかし実環境では, 移動はタイヤのスリップや慣性などといった要因による誤差が含む. MDP ではこの不確実性を確率分布としてモデル化する. 例えば, 「前進」を選択しても, 10% の確率で左右のセルに移動し, 5% で行き過ぎて奥のセルに移動する, といった表現が可能である.

4. 報酬モデル $\mathcal{R}(s, a, s')$ (**Reward Function**): 状態遷移に伴って得られる評価でスカラで表される. 経路計画問題においては, コストの最小化または負の報酬の最大化として定式化される. 例えば, ゴール状態に到達したときは, コストを 0 とし, 障害物に衝突したときに大きなコスト (ペナルティ) を与える. また, 移動に

かかる時間やエネルギーを表現するため、各ステップごとにわずかなコスト（ステップコスト）を与える。

ベルマン方程式と価値関数

MDP の目的は、各状態でどのような行動をとるべきかというルール、すなわち方策（Policy） $\pi: \mathcal{S} \rightarrow \mathcal{A}$ を見つけることである。最適な方策 π^* を計算するために、状態価値関数 $V: \mathcal{S} \rightarrow \mathcal{A}$ を導入する。状態価値関数は式 (2.2) のように定義される。これは、ある状態 s からスタートし、方策 π に従って行動し続けたときに得られる将来の報酬の総和（期待値）である。

$$V(s) = \left\langle R(s, a, s') + V(s') \right\rangle_{P(s'|a, s)} \quad (2.2)$$

最適な方策 π^* に従ったときの価値関数を最適状態価値関数 $V^*(s)$ と呼ぶ。 $V^*(s)$ は、式 (2.3) に示すベルマン方程式（Bellman Equation）を満たす。ベルマン方程式は再帰的な構造をしており、ある状態の価値は、そこで最適な行動をとった際に期待される即時報酬と、遷移先の状態の価値の和によって決まることを意味している。

$$V^*(s) = \max_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} P(s'|a, s) [R(s, a, s') + V^*(s')] \quad (2.3)$$

2.2 価値反復アルゴリズム

ロボットの位置・向きの $xy\theta$ 空間を格子状に離散化して作った離散状態 $s \in \mathcal{S}$ に対し、そこから目的地までのコストを計算した $V(s)$ が計算される。また、コストというのは、時間と、時間換算で与えるペナルティーを足した値である。ペナルティーは悪路に相当する s など、ロボットが入るとなんらかの問題のある状態に与えられる。

V は、価値反復の計算が進むと正解の値 V^* に近づいていく。図 2.2 は、価値反復 ROS パッケージの V の計算の過程を任意の θ で固定した XY 平面を描画したもので、白から黒くなるほど V の値が大きいことを示している。薄い緑は、値がとても大きい状態、濃い緑は障害物である。このとき、 V の値は目的地の周りから収束していき、目的地から遠いところが最後に収束する。この計算は、「幅優先探索」のコスト計算に相当する処理となる。

V が収束すると、ロボットは V の値が小さくなるような行動を選択し続けることで、最適な経路を選択して移動できるようになる。また、 V が完全に収束しなくても、ロボットのいる s から目的地に向かって値が単調減少していると、ロボットは目的地に向かうことができる。このような状態を、本稿では「経路が見つかる」と表現する。

$V(s)$ には初期値として、 s が目的地の領域に含まれる場合に 0、そうでないときは無限大（実装上は目的地まで 10 万秒など、非常に大きな値）が与えられる。価値反復は終端状態以外の $V(s)$ の値を、繰り返し計算で実際のコストまで値を小さくしていく。

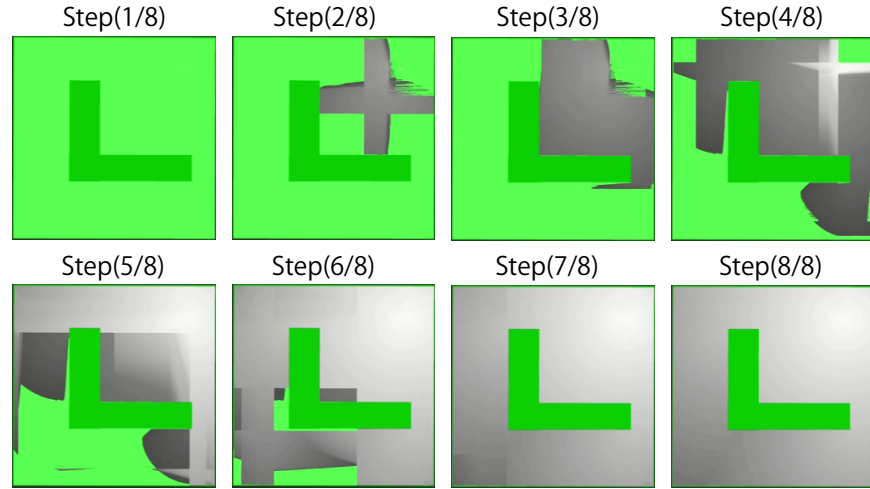


図 2.2 Convergence of the value function

完全に収束した V は最適状態価値関数 V^* と呼ばれ、 V^* からは、最適な行動を求めることができる。ただし、ロボットが移動を開始するためには、 V^* を厳密に計算する必要はない。現在地の周辺において、目的地に向かうための適切な勾配が V にできると、ロボットは目的地に向かう（ V のコストを減らす）ように行動をとることができる。

しかしながら、この勾配の伝播は A^* などの探索手法が経路を探すよりも遅い。価値反復は、探索の手法として見ると幅優先探索になっており、 V の勾配は目的地の周辺からできはじめ、それがより遠い地点に伝播していく。またこの伝播の計算は、ロボットが通らないであろう経路でも行われる。そのため、目的地までの経路が存在すれば、それが複雑に入り組んでいても必ず見つけることはできるが、ロボットは現在地に勾配が到達するまで、長く待たされることになる。

2.2.1 価値反復アルゴリズムによる経路計画

ベルマン最適方程式 (??) を用いて、反復計算により $V^*(s)$ を求める手法が価値反復 (Value Iteration) である。詳細な計算とアルゴリズムの手順は以下の通りである。

1. 初期化: 目的地に含まれる状態 s_{goal} の $V(s_{goal})$ は、0、その他の s の $V(s)$ は任意の値（通常は無限大）に初期化する。
2. 価値更新: 以下の更新式を、すべての状態 s に対して適用する。

$$\pi(s) \leftarrow \operatorname{argmin}_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \mathcal{P}(s'|a, s) [\mathcal{R}(s, a, s') + V(s')] \quad (2.4)$$

$$V_{k+1}(s) \leftarrow \min_{a \in \mathcal{A}} \sum_{s' \in \mathcal{S}} \mathcal{P}(s'|a, s) [\mathcal{R}(s, a, s') + V_k(s')] \quad (2.5)$$

ここで k は反復回数を表す。

3. 収束判定: 全状態における価値関数の更新量 $|V_{k+1}(s) - V_k(s)|$ の最大値が, 事前に定めた閾値 ϵ 未満になれば停止する.

また, 一回の反復計算は, 式 (??) と式 (??) によって行われる. この2つの式のシグマ以降は同様の式で, すべての行動 a の中から V を最小にする a を選ぶというものである. それぞれの行動による V は, 移動先の状態 s' の V と移動にかかる報酬 \mathcal{R} の和に s' に移動する確率をかけたものである.

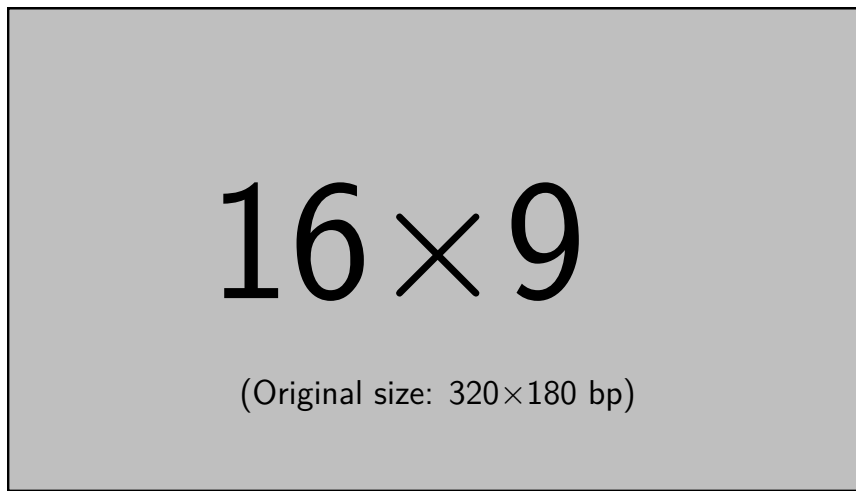


図 2.3 Calculation of the state value function

この計算により, 環境内のあらゆる場所からゴールへ向かうための最適な「勾配」が得られる. これはポテンシャル場に似ているが, ポテンシャル法が抱える「局所解 (Local Minima)」の問題 (ゴール以外の窪みにハマって出られなくなる現象) が発生しないという強力な数学的保証がある. なぜなら, ベルマン方程式による更新は, 大域的な最適性を伝播させる処理だからである.

価値反復を動的な環境で用いるときには, ロボットの周辺で, 障害物の情報を追加して計算することで障害物回避が可能になる. この機能は, 大域経路計画と独立した機能であり, 走り出しまでの時間の短縮のためにいじる機能としない.

第 3 章

提案手法

価値反復 ROS パッケージにグラフ探索手法を組み合わせ、グラフ探索手法で算出した経路から V に勾配を作ることで、走り出しまでの時間を短縮する手法を提案する。3.1 節は、文献 [中村 24] で発表したものであり、3.2 節は、文献 [中村 25] で発表したものである。

3.1 価値反復と A*を組み合わせた大域計画

A*と価値反復を並列に計算し、A*の計算結果を価値反復に適用する。既存の価値反復と並行で A*探索を実行し、図??のように、A*の見つけた経路沿いに V の値を書き換えるというものである。図のようにロボットが行動を開始する状態から目的地まで、 V の値を少しずつ減らしていくように書き換えることで、価値反復が経路を見つける前に、ロボットを目的地に向かわせるようにする。

V の書き換えは、A*の見つけた経路沿いの各 s に対して、

$$V(s) \leftarrow Kf(s) \quad (3.1)$$

という代入の式を用いて行う。 K は定数であり、 f は s から目的地までの経路上での道のりである。

この方法では、ロボットと目的地の間の環境が迷路のようになっていなければ、A*で見つかる経路がほぼ最適な経路となり、価値反復のみの場合よりもロボットが速やかに目的地に向かえるようになる。一方、迷路のような環境だと、たとえば A*で見つけた経路が遠回りで、そのあとで価値反復がよりよい経路を見つけると、目的地までの時間が増えてしまう可能性がある。

また、A*がほぼ最適な経路を見つけられる場合、価値反復は大域計画に対しては必須ではなくなる。しかし、A*の見つけた経路の最適化や、未知の障害物が現れた場合の V の修正や迂回先の V の計算に必要となる。

ここで用いる A*は、XY 平面を探索するものであり、価値反復が探索する $XY\theta$ 空間に比べ、1 つ次元が低い。これにより、特に地図が大きくなったとき、次元の違いから探

索にかかる時間が価値反復の勾配の伝播に比べ小さくなることが期待される。その一方で、算出する経路が $XY\theta$ において最適ではないものとなり、 V の勾配が思わぬ方向に傾くような悪影響が考えられる。

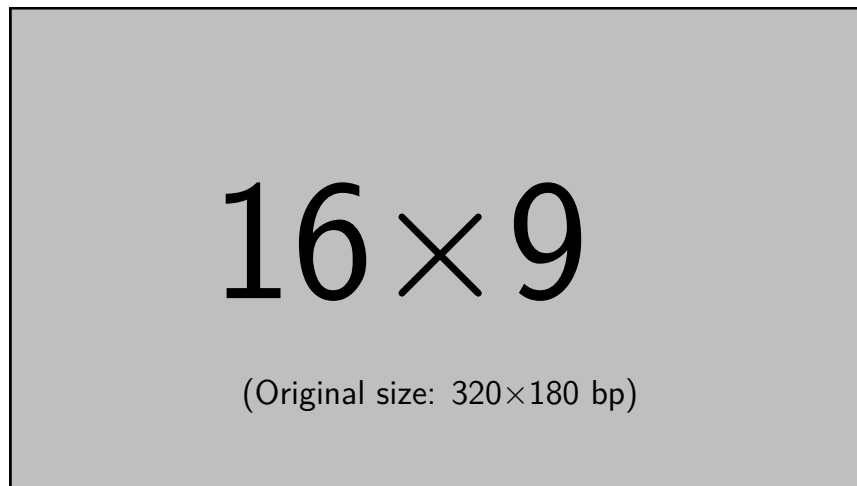


図 3.1 propose method with 2D A*

3.2 3D A*の適用

そこで、価値反復と同じ探索空間を持つ A* (3D A*) を用いて経路を算出し、用いる手法を提案する。グリッドに加え、角度の方向にも離散化し、探索空間を拡張する。

3D A*の算出する経路は、用いるヒューリスティック関数 H が許容的にあるとき、最適性が保証される。これにより、価値反復のコストと大きな差がない暫定的なコストを計算できることが期待できる。A*は、決定論的な状態遷移を用いて探索を行うのに対し、価値反復は、確率的な状態遷移を行うため、算出される経路は、必ず同じものとはならないが、平面上を探索する 2D A*に比べると価値反復の経路に近くなると考えられる。

その一方で、当然ながら 2D A*に比べて計算量は多くなる。そのため、走り出しまでの時間の短縮の効果は短くなると考えられる。

2D A*の算出した経路を V に適用による移動時間の増加量と 3D A*で増加した計算時間の増加量のどちらが多いのか、非自明であり、検証する必要がある。

第 4 章

実装

4.1 A*の実装と価値反復への適用

4.1.1 A*の実装

2D A*の実装

価値反復 ROS パッケージに, A*探索を行うスレッドを 1 つ追加した. ROS 2 のノードの形式で実装し, 実装には, `ike_nav[?]` パッケージ内の `ike_nav_planner` を A* 探索のプログラムとして流用した. `ike_nav_planner` はマップと現在地と目的地に対して A* で計算された経路を出力する.

3D A*の実装

そこで, $XY\theta$ 空間で探索を行う A*を実装した. A*の具体的な実装については, `ike_nav[?]` パッケージ内の A*探索ノード (`ike_nav_planner`) を 3 次元での探索に拡張し, 価値反復 ROS パッケージに移植して利用した. 3D A*は, 価値反復と同様に, グリッドに加えて θ 方向にも離散化し, グラフを構築する. XY 方向の遷移コストと θ 方向の遷移コストは単位が一致するよう距離から時間に変更した.

ヒューリスティック関数 H については, $[?]$ での平面上のユークリッド距離を計算するものから,

$$H(s) \triangleq W_1 \sqrt{(x_c - x_g)^2 + (y_c - y_g)^2} + W_2 |\theta_c - \theta_g| \quad (4.1)$$

に変更する. ここで, (x_c, y_c, θ_c) は各離散状態の中心の座標, (x_g, y_g) は目的地の中心地点の xy 座標, $\theta_c - \theta_g$ は, (x_c, y_c, θ_c) から見た (x_g, y_g) の方角である. W_1, W_2 は定数である. W_1 と W_2 の比は, ロボットが一定時間あたりに移動できる距離と方向転換できる角度の比で決まる.

4.1.2 価値反復と A*の併用

価値反復 ROS パッケージ内に A*で算出した経路上で、式 (??) に従って、 V の値を書き換えるコードを追加した。ただし、2D A*は、出力される経路は XY 平面上のもので、 θ 方向の情報がない。そのため、 XY 平面上で同じ位置にある離散状態にはすべて同じ値を式 (3.1) で代入し、あとは価値反復で θ 方向の V の値を計算させることとした。この実装には改良の余地があり、 $XY\theta$ 空間内での A*探索を用いるほうが、より時間短縮の効果が得られるものと考えられる。

第 5 章

実験

5.1 シミュレータ実験

提案手法の走り出しまでの時間を短縮する効果の有無を検証するために、シミュレータを用いて実験を行う。シミュレータ環境では、動的な障害物が存在せず、障害物回避による移動時間の増加を考慮する必要がない理想的な環境で効果を検証できる。

5.1.1 実験条件

シミュレータ実験では、千葉工業大学津田沼キャンパスで得られた図??の地図を使用する。紙面横方向が 300m、縦方向が 200m の 6ha の環境の地図で、形式は、ROS のナビゲーションスタックで用いられる占有格子地図である。白色が障害物のない画素、黒色が障害物のある画素、灰色が不明な画素を表す。この地図のパラメータは表??の通りである。

価値反復 ROS パッケージでは、地図のグリッドに加え、 θ 方向にも離散化を行い \mathbb{S} を構成する。 S の諸元を表??に示す。要素数は 1 億に達し、移動可能なエリアだけでも 1 千万に達する。

実験で使用するシミュレータは、図??から作成したシミュレータを使用する。シミュレータには、ROS の標準的なシミュレータである Gazebo を使用し、シミュレータ内で用いる静的な障害物の作成には、占有格子地図から作成するプログラム [?] を使用した。

シミュレータ内で走行させるロボットは、図?? の差動二輪型のロボットである。差動二輪型のロボットは、進行方向に対して前後に進むことができるが、横に進むことができない。そのため、当然ながらゴールの方向へ進むにはその方向を向く必要があり、位置と方向の 3 次元で経路を探索することで効率化が計れる。

実験に用いた計算機は、CPU として Ryzen 9 7945HX3D (16 コア 32 スレッド)、DRAM として DDR5-4800 32GB が 2 枚 (64GB) 搭載されたものである。価値反復 ROS パッケージは、CPU ですべて計算するため、コアの数が重要であり、このコンピュータは、現代の家庭用コンピュータの中でも、コアの数の多い CPU である。DRAM は、地図の大きさに応じて決定した。

表 5.1 Configurations of the Map

map size	294.6[m]×199.95[m]
cell resolution	0.15[m]×0.15[m]
number of cells	2,615,346
number of free cells	165,076
the area of the free cells	3714.98[m ²]

価値反復 ROS パッケージはマルチスレッド化されており、環境の全域で V を計算する大域計画器と、ロボットの周囲で V を計算する局所計画器に任意のスレッドの数を割り当てることができる。本稿の実験では、大域計画器に 8、局所計画器に 6 のスレッドを割り当てた。A*は別のプロセスで動作する。CPU から見ると、A*には 1 つのスレッドを割くこととなる。また、シミュレーションには Gazebo を使用するため、これにも計算機のリソースを使用している。

実験では、図??3 つの目的地で比較を行った。Goal1 は...

5.1.2 実験結果

要追加

5.2 実機実験

第 6 章

結論

よくなりました.

参考文献

- [Alverhed 24] E. Alverhed, S. Hellgren, H. Isaksson and L. Olsson, H. Palmqvist, and J. Flodén. Autonomous last-mile delivery robots: A literature review. *European Transport Research Review*, Vol. 16, No. 4, 2024.
- [Arkin 90] Ronald C. Arkin. Integrating Behavioral Perceptual World Knowledge in Reactive Navigation. No. 6, pp. 105–122, 1990.
- [E.W. 59] Dijkstra E.W. A note on two problems in connexion with graphs. *Numerische Mathematik*, Vol. 1, pp. 269–271, 1959.
- [Fox 99] Dieter Fox, Wolfram Burgard, Frank Dellaert, and Sebastian Thrun. Monte Carlo Localization: Efficient Position Estimation for Mobile Robots. In *Proc. of AAAI-99*, pp. 343–349, 1999.
- [Fragapane 21] Giuseppe Fragapane, René de Koster, Fabio Sgarbossa, and Jan Ola Strandhagen. Planning and control of autonomous mobile robots for intralogistics: Literature review and research agenda. *European Journal of Operational Research*, Vol. 294, No. 2, pp. 405–426, 2021.
- [Gerkey 09] Brian Gerkey. gmapping, 2009. Accessed on 27.11.2025.
- [Hart 68] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, Vol. 4, No. 2, pp. 100–107, 1968.
- [Karaman 11] Sertac Karaman and Emilio Frazzoli. Incremental sampling-based algorithms for optimal motion planning. In *Robotics: Science and Systems VI*. The MIT Press, 08 2011.
- [Khatib 85] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, Vol. 2, pp. 500–505, 1985.
- [Koditschek 90] Daniel E Koditschek and Elon Rimon. Robot navigation functions on manifolds with boundary. *Advances in Applied Mathematics*, Vol. 11, No. 4, pp. 412–442, 1990.
- [Kohler 16] Damon Kohler, Wolfgang Hess, and Holger Rapp. Cartographer,

2016. Accessed on 27.11.2025.
- [Koide 24] Kenji Koide, Shuji Oishi, Masashi Yokozuka, and Atsuhiko Banno. GLIM: 3D Range-Inertial Localization and Mapping with GPU-Accelerated Scan Matching Factors. *Robotics and Autonomous Systems*, Vol. 179, p. 104750, 2024.
- [Konolige 00] K. Konolige. A gradient method for realtime robot control. In *Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000) (Cat. No.00CH37113)*, Vol. 1, pp. 639–646 vol.1, 2000.
- [LaValle 98] Steven M. LaValle. Rapidly-exploring random trees : a new tool for path planning. *The annual research report*, 1998.
- [LLC 18] Open Navigation LLC. Navigation2, 2018. Accessed on 1.12.2025.
- [MA 15] Raúl Mur-Artal, J. M. M. Montiel, and Juan D. Tardós. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Transactions on Robotics*, Vol. 31, No. 5, pp. 1147–1163, 2015.
- [Macenski 22] Steven Macenski, Tully Foote, Brian Gerkey, Chris Lalancette, and William Woodall. Robot operating system 2: Design, architecture, and uses in the wild. *Science Robotics*, Vol. 7, No. 66, p. eabm6074, 2022.
- [Thrun 05] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic ROBOTICS*. MIT Press, 2005. (邦訳) 上田訳: 確率ロボティクス, 毎日コミュニケーションズ (2007) .
- [Ueda 23] Ryuichi Ueda, Leon Tonouchi, Tatsuhiko Ikebe, and Yasuo Hayashibara. Implementation of brute-force value iteration for mobile robot path planning and obstacle bypassing. *Journal of Robotics and Mechatronics*, Vol. 35, No. 6, pp. 1489–1502, 2023.
- [横山 19] 横山和寿. ヤンマーテクニカルレビュー 自動運転農機「ROBOT TRACTOR」の紹介. https://www.yanmar.com/jp/about/technology/technical_review/2019/0403_1.html, 2019. (2025-11-27 参照) .
- [楽天 25] 楽天グループ株式会社. 楽天、商品配送サービス「楽天無人配送」において、新たなロボットの導入、対象店舗や地域の拡大などサービスを拡充, 2025. https://corp.rakuten.co.jp/news/press/2025/0226_01.html (2025-12-11 参照).
- [株式 17] 株式会社クボタ. 自動運転農機「アグリロボトラクタ」を市場”初”投入 . <https://www.kubota.co.jp/news/2017/17-23j.html>, 2017. (2025-11-27 参照) .
- [原 25] 原祥亮, 萬礼応, 富沢哲雄, 伊達央, 大川一也, 大矢晃久. つくばチャレンジ 2024 全チームの技術動向調査. ロボティクス・メカトロニクス講演

- 252CSp%255B31%255D%252CSp%\255B39%255D%252CSp%255B39%
255D%252CSp%255B44%255D%252CSp%255B44%255D%252CSp%
255B66%\255D%252CSp%255B66%255D (2025-11-27 参照).
- [日本 24] 日本国政府. 農業の生産性の向上のためのスマート農業技術の活用の
促進に関する法律. 令和六年法律第六十三号, 2024. [https://elaws.
e-gov.go.jp/law/506AC0000000063](https://elaws.e-gov.go.jp/law/506AC0000000063) (2025-11-26 参照).
- [友納 20] 友納正裕, 原祥堯. SLAM の現状と今度の展望. Vol. 64, No. 2, pp.
45–50, 2020.