

Title	自律分散型モバイルロボットによる衝突回避に関する研究
Author(s)	東原, 大記
Citation	
Issue Date	2007-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/3575
Rights	
Description	Supervisor:Defago Xavier, 情報科学研究科, 修士

修 士 論 文

自律分散型モバイルロボットによる衝突回避に関する研究

北陸先端科学技術大学院大学
情報科学研究科情報処理学専攻

東原大記

2007 年 3 月

修 士 論 文

自律分散型モバイルロボットによる衝突回避に関する研究

指導教官 Defago Xavier 助教授

審査委員主査 Defago Xavier 助教授
審査委員 片山卓也 教授
審査委員 丁洛榮 助教授

北陸先端科学技術大学院大学
情報科学研究科情報処理学専攻

410099 東原大記

提出年月: 2007 年 2 月

概 要

本稿では、複数の自律的モバイルロボット間における協調的な衝突回避システムに関して述べる。モバイルロボット間での衝突回避に関しては、センサーを用いた方法とネットワークコミュニケーションを用いた方法が考えられる。

センサーを用いた衝突回避の場合、衝突回避行動をタイムリーにおこなう事ができる。しかし、センサーは有効距離が短いため、モバイルロボット同士が接近するまでお互いの認識ができない。また、モバイルロボット間に障害物が存在する場合は、お互いの認識が衝突の直前となり衝突が回避できない場合があり、モバイルロボットが密集している場合には、モバイルロボット間に他のモバイルロボットがある場合には障害物となり認識できなくなる。

一方、ネットワークコミュニケーションを利用した場合は、ワイヤレスネットワークを用いるため通信遅延が発生する可能性があるため、センサーを用いた場合のようなタイムリーな処理は困難である。しかし、センサーを用いる場合と比べ広範囲で衝突回避処理が可能である。また、モバイルロボット間のモバイルロボット以外の障害物を意識する事無く衝突回避ができる。

本稿では、ネットワークコミュニケーションを用い、確実に衝突を回避可能なシステムの提案する。提案する衝突回避システムは、ネットワークコミュニケーションを用いモバイルロボットの移動経路を予約するシステムである。モバイルロボットが移動前に移動経路を予め予約しておく事により、他のモバイルロボットは予約されている移動経路が利用できない。予約が許可されるまで、モバイルロボットは移動を開始することができないため、モバイルロボット間で衝突が発生する危険は無い。また、移動経路を予約する方法は、集中管理をする方法と分散管理する方法が考えられる。本システムでは、移動経路を分散管理する方法を用いる。分散型の移動経路予約システムは、全てのモバイルロボットが同じ衝突回避システムのクローンを用いる。移動経路を分散管理する場合、分散されている全ての衝突回避システムへ入力する予約の順序を保つ事が重要となる。本稿では、全ての衝突回避システムへの入力の順序を保つために total-order broadcast を用いた。

また、提案したシステムをシミュレーションにより性能評価をおこなった。評価方法は、一定の大きさの仮想空間にモバイルロボットの台数と予約の経路の長さを変化させ、1秒間の平均の移動量と平均の移動速度をもとめた。

目次

第1章	はじめに	1
第2章	研究背景と関連研究	4
2.1	衝突回避とは	4
2.2	センサーを用いた衝突回避	5
2.3	ネットワークコミュニケーションを用いた衝突回避	5
第3章	システムモデル	7
3.1	モバイルロボット	7
3.2	予約	8
3.3	衝突の検出	9
3.3.1	デッドロックモデル	11
3.4	total-order broadcast	13
第4章	システム概要	15
4.1	システム概要	15
4.2	集中管理型	17
4.3	分散管理型	19
4.4	request	21
4.4.1	queue	22
4.4.2	グラフ	22
第5章	シミュレーション	26
5.1	シミュレーション目的	26
5.2	シミュレーション環境	26
5.3	シミュレーションモデル	27
5.4	シミュレーションシナリオ	29
5.5	シミュレーション結果と考察	29
5.5.1	移動モバイルロボット台数	30
5.5.2	移動モバイルロボット台数	31
5.5.3	平均移動速度	31

第 6 章	まとめと今後の予定	34
6.1	まとめ	34
6.2	今後の課題と予定	34
6.2.1	今後の課題	35

第1章 はじめに

近年、ロボット技術は急速な勢いで進化している。ロボット技術の進化に伴い、ロボットは複雑な作業が可能となったため、ロボットに求められるタスクも複雑化の傾向にある。しかし、複雑なタスクを単体のロボットで実行するためには、複雑で高機能なロボットが必要となり、金銭的なコストも大きくなる。また、複雑で高機能なロボットは、特定のタスクに特化している場合が多いため、異なるタスクを実行するためには多種のロボットを用意する必要がある。以上の事から、複雑化の傾向にあるロボットタスクは、複数のモバイルロボットで実行する事が望ましい。

単体のロボットでタスクを実行する場合には、リモコンなどを用いた遠隔操作も可能である。しかし、複数の台のモバイルロボットを遠隔操作する事は、モバイルロボット台数の増加に伴い複雑化する。よって、複数台のモバイルロボットを利用してタスクを実行する場合には、自律的なモバイルロボット群を利用する事が重要である。

複数台の自律的なモバイルロボット群を用いてタスクを実行する場合には、モバイルロボット間での協調 [6], [5] が重要となる。モバイルロボット群を用いた基本的な協調問題には、フォーメーションプランニングやリーダー追従などの問題がある。フォーメーションプランニング問題とは、複数のモバイルロボットが自律的に必要に応じてフォーメーションを組み、フォーメーションを維持する問題である。リーダー追従問題とは、モバイルロボット群内でリーダーを決め、リーダーの移動経路を追従するための問題である。前述の問題を実行するためにモバイルロボット間の協調を行うには、モバイルロボットの移動を適切に制御する必要がある。適切な移動の制御のためには、モバイルロボットは確実に衝突を回避する事が重要となる。[6] では、複数台のモバイルロボットがグループを作りタスクを実行する場合に、各モバイルロボットのグループ内での役割分担をどのように計画するかに関して述べている。ロボット群での協調を実現するためには、グループ内での各ロボットの役割を確実に計画できる事が重要な要素の一つである。[5] では、複数台のモバイルロボットが協調をとりながら物体の移動や持ち替えなどを行うためのアルゴリズムに関して述べている。本稿では、全てのモバイルロボットは非同期であるとしているが、将来的にはグループ内での同期処理が必要となる。

モバイルロボット群を用いる場合、移動経路設計やフォーメーションプランニングなどの様々なアプリケーションを実装する事が必要となる。前述のとおり、複数台のモバイルロボットが移動する場合には、モバイルロボット間での衝突を確実に回避する事が重要であるため、全てのアプリケーションで実装することになる。しかし、モバイルロボットどうしで異なる衝突回避システムを実装した場合には正しく衝突が回避できない場合も

ある。よって、全てのモバイルロボット制御アプリケーションは共通の衝突回避システム [1], [3], [4], [7], [8] を用いる事が重要である。[1] では、本稿で述べる衝突回避システムのプロトコルに関して述べている。本稿では、[1] に記述されているプロトコルのシミュレーションを実装し、性能評価を行った。[3] では、超音波を用いる事により、環境との衝突を回避する事に特化したシステムの構築を行っている。また、内輪差や移動速度なども考慮しており、モバイルロボット間での衝突回避に置いての衝突回避における問題と同種の問題を持っている。[4] では、ネットワークで接続された環境に設置された複数台のカメラ情報から、モバイルロボットの移動の誘導をするシステムである。カメラ画像の解析をする事によりモバイルロボットを誘導するため、多数のモバイルロボットが存在する場合には、複雑な計算が必要となる。また、高速なネットワークによる通信を保証する事が重要となり、ネットワークにかかる負荷の大きさが大きくなると、メッセージ遅延が発生し、正しくモバイルロボットを誘導できない可能性がある。[7] では、リアルタイム性の強い衝突回避をおこなうために、ソフトウェアのみではなくハードウェアに関しても特定している。リアルタイムな衝突回避をするためには、高速で安定したネットワーク網を構築する事が重要となる。[8] は、サッカーロボットを対象としたロボット間での衝突回避システムに関して述べている。サッカーロボットの場合、他チームのモバイルロボットとは通信する事は困難であるため、センサーを用いた衝突回避をしている。また、他チームのモバイルロボットの移動を考慮していないため、お互いに衝突する可能性がある。

本稿では、モバイルロボット間における衝突回避を確実にこなうことのできるシステムの提案をする。本衝突回避システムを用いる事により、他のモバイルロボット向けアプリケーションは他のモバイルロボットとの衝突を意識することなく実行する事ができる。提案するシステムは、ワイヤレスネットワークコミュニケーションを用いる事により移動経路の予約により他のモバイルロボットとの衝突を回避する。移動経路の予約を集中的に管理すると、サーバーのダウンなどの single point of failure に弱い。single point of failure の問題を解決するために、移動経路の予約を管理するサーバを状態機械アプローチを用いる事により全てのモバイルロボットに移動経路の予約を管理するサーバのレプリカを利用する。状態機械アプローチの保証を維持するためには、全てのレプリカへの入力の手順を統一する事が重要となる。全てのレプリカへの入力の手順を統一するために、我々は total-order broadcast [2] を用いた。Total-order broadcast は、多彩なアルゴリズムが提案されている。[2] では、主要な total-order broadcast に関する調査研究をしている。我々は、total-order broadcast の中でも、atomic broadcast を用いた。

また、提案したシステムをシミュレーションを用い性能評価をする。評価方法は、モバイルロボットの平均移動速度、一定時間に移動したモバイルロボットの台数、移動経路の交差数をモバイルロボットの密度と一度に予約する距離の観点からおこなう。

2 章では、一般的な衝突回避問題に関して述べる。まず、障害物との衝突回避とモバイルロボット間での衝突回避の違いに関して述べる。また、自律的なモバイルロボット間での衝突回避システム方法では、センサーとネットワークコミュニケーションを用いる方法が考えられる。上記の 2 種類の方法に関しても検討する。

3章では、本衝突回避システムのシステムモデルに関して述べる。まず、モバイルロボットのモデルに関して述べ、その後、移動経路の予約範囲のモデルとモバイルロボット間で発生する衝突のモデルに関して述べる。

4章では、本衝突回避システムのシステムデザインに関して述べる。まず、予約する移動経路の交換方法に関して述べる。交換された情報は適切に管理される必要があるため、情報の管理方法に関しても述べる。そして、システム全体の概要に関して述べる。

5章では、前章で述べたシステムデザインを実装したシミュレーションとシミュレーション結果に関して述べる。また、シミュレーション結果を基にした考察もおこなう。

6章では、まとめと今後の予定に関して述べる。

第2章 研究背景と関連研究

本章では、自律型モバイルロボット群におけるモバイルロボットの衝突回避に関して述べる。まず、一般的に衝突回避とはどのようなものかに関して述べる。次に、関連研究として、自律型モバイルロボット間での衝突回避は、センサーを用いた方法とネットワークコミュニケーションを用いた方法に関して述べる。センサーを用いた場合とネットワークコミュニケーションを用いた場合のそれぞれに関して述べる。

2.1 衝突回避とは

モバイルロボットが他の物体と衝突した場合、モバイルロボットの物理的な損傷につながる場合がある。モバイルロボットの物理的な損傷は、モバイルロボットのタスクの実行を困難とし、場合によっては部品の交換が必要な場合もある。交換する部品がない場合には、モバイルロボット群のタスクが実行できなくなる。以上のことから、衝突は確実に回避できる事が重要である。

モバイルロボットが衝突をする可能性がある物体は、他のモバイルロボットとモバイルロボット以外の障害物に分類可能である。モバイルロボット以外の障害物とは、壁などの様に非可動物体や可動物体であってもワイヤレスネットワークを用いた通信が出来ない物などを示す。モバイルロボットが移動する場合に、他のモバイルロボットや壁等の非可動な障害物、モバイルロボット以外の可動物体を同時に衝突回避することは、複雑な問題となる。モバイルロボットは、衝突の可能性のある物体を段階的に衝突回避することにより、問題を分割化することができる。

モバイルロボットは、非可動物体である障害物を予め情報を得ることができる場合には、経路計画時に衝突を回避することができる。予め情報を得る事ができない場合には移動時にタイムリーに衝突を回避する必要がある。経路計画とは、モバイルロボットがどのような経路を通るかを計画する事である。また、モバイルロボット以外の可動物体に関しては、予め位置情報や移動情報を入手する事ができない場合が多いため、移動時にタイムリーに衝突回避をする。

2.2 センサーを用いた衝突回避

センサーを用いた衝突回避とは、赤外線や超音波などを用いる事で、モバイルロボットの移動と同時に衝突回避を行う方法である。現在、モバイルロボット間の衝突回避に最も良く用いられている方法の一つである。

モバイルロボットにおいてセンサーとは、モバイルロボットが移動をする上で視覚的な情報を収集するための機能である。モバイルロボットが視覚的な情報を基に衝突を回避するためには、できるだけ新しい情報を利用する事が重要となる。モバイルロボットは空間を物理的に移動するため、古い情報を基に衝突の回避をおこなうと衝突回避の対象とするモバイルロボットは別の場所に移動してしまい、衝突する可能性がある。一方、常に他のモバイルロボットの情報を取得し、かつ確実に衝突回避をおこなうためには、複雑な処理を高速に行う事が重要となる。しかし、モバイルロボットがバッテリーを用いている事を考慮すると、バッテリーの電力は有限であるため、複雑な処理を常時高速でおこなうことは困難である。

前述のとおりセンサーを用いると複雑な処理を高速に行う事が重要であるにも関わらず、センサーが利用できる範囲は3メートル程度と短いため、お互いのモバイルロボットが接近するまで認識することができない。また、モバイルロボット間に障害物が存在する場合、他のモバイルロボットの発見が遅くなるため衝突する可能性がある。さらに、モバイルロボットがセンサーを用いて衝突回避を行う場合には、双方のモバイルロボットが自分と相手のモバイルロボット両方のセンサーの有効範囲や制動距離、加速度などを考慮した上で確実に停止できる移動速度を保つ事が重要である。

モバイルロボットの移動速度を重視した場合には、他のモバイルロボットとの衝突のリスクが高くなる。一方、確実に衝突を回避したい場合には、モバイルロボットの移動速度を確実に安全な速度で移動する事が重要となり、衝突のリスクとモバイルロボットの移動速度のトレードオフが重要となる。

センサーを用いた衝突回避システムは、タイムリーに処理が可能であるにも関わらず、センサーの特性を考慮するとモバイルロボットの移動速度を十分に発揮することができない。

2.3 ネットワークコミュニケーションを用いた衝突回避

センサーを用いた衝突回避を、ワイヤレスネットワークコミュニケーションを用いて実装する事も可能である。ワイヤレスネットワークコミュニケーションを用いる場合、モバイルロボットは視覚的な情報を入手する事はできないため、お互いの位置座標を交換する。

ワイヤレスネットワークを用いる事で、センサーを用いた場合と比べ広範囲で衝突回避が可能である。しかし、ワイヤレスネットワークの場合には、ケーブル接続のネットワークと比べ接続が不安定さや通信のバンド幅が狭い、通信トラフィックが混んでいるなどのワイヤレスネットワーク特有の問題がある。モバイルロボットの位置情報を交換し衝突回

避をしながら移動をする場合、前述の通りできるだけ新しい情報を利用する事が重要となる。

また、移動経路をあらかじめ予約し通知する事により、他のモバイルロボットは予約されている経路が解放されるまで移動できないため衝突が発生する事は無い。

センサーを用いた衝突回避は、タイムリーな処理が可能であるが衝突の回避能力は不確実である。一方、ネットワークコミュニケーションを用いた場合、タイムリーな処理には向かないが、確実に衝突が回避できる。前述のとおり、モバイルロボットの衝突は確実に回避する事が重要である。以上の事から本稿では、ワイヤレスネットワークを用いて移動経路をあらかじめ予約する事により、衝突を確実に回避する。

第3章 システムモデル

本章では，本稿で述べる衝突回避システムのシステムモデルに関して述べる

3.1 モバイルロボット

本節では，対象とするモバイルロボットはどのような物かに関して述べる．

- 自律的に移動

本システムの対象はモバイルロボットであるため，全てのモバイルロボットは自律的に移動可能であるとする．よって，移動のために他のモバイルロボットからの力を必要とするロボットは，本稿ではモバイルロボットとはしない．

- ポジショニングシステムが利用可能であり，必要に応じて随時にモバイルロボット自身が環境内の位置情報が取得可能である．

環境内の位置情報には，ローカル位置情報とグローバル位置情報がある．ローカル位置情報とは，位置情報を取得したいモバイルロボットが決定した特定点からの位置情報を示す．ローカル位置情報は，モバイルロボットごとに異なるため，初期の位置情報が異なるモバイルロボット同士では双方の位置情報を共有する事は困難である．

一方，グローバル位置情報とは，環境内にある定点からの絶対座標を示す．全てのモバイルロボットは，共通の位置情報を利用するため，他のモバイルロボットと容易に位置情報を共有することができる．以上の事から，本稿ではグローバル位置情報を利用する．

- ワイヤレスネットワークコミュニケーション

モバイルロボットを利用する環境内の全ての範囲で，モバイルロボットはワイヤレスネットワークに接続可能とする．また，ワイヤレスネットワークに接続可能な全ての範囲で安定した接続が保証されている物とする．以上の条件により，モバイルロボットは，ワイヤレスネットワークを介して移動経路の予約を他のモバイルロボットに通知する事が出来る．

- 移動経路の自動生成

本システムは、移動経路に関する入力に対して自律的に確実な衝突回避をおこなう。モバイルロボットを利用する環境には、他のモバイルロボット以外にモバイルロボットの移動を妨げる障害物がある。障害物には、壁等のようなネットワークを介して通信できない場合もあり、衝突回避が困難である。以上のことから、モバイルロボットは移動開始位置から目的地までの移動経路を設計するシステムが利用できるモバイルロボットを対象とする。

- 個体識別

モバイルロボットの識別方法には、外観の特徴から視覚的に識別する方法と、IDなどの非視覚的識別をする方法がある。視覚的情報を基に他のモバイルロボットの識別は、識別しようとするモバイルロボットと識別されるモバイルロボットが独立して行動でき、お互いに意識する必要が無い。しかし、利用可能距離が短くモバイルロボットの密集状態の場合には正確な識別ができない場合がある。

一方、非視覚的情報を基に他のモバイルロボットを識別する場合は、モバイルロボット間での情報交換が必要となる。しかし、視覚的情報を基に個体識別する場合と比べお互いの地理的空間を意識する事なく個体識別が可能となる。

以上のことから、本衝突回避システムにおいては、IDを用いる。

- モバイルロボットのタスクは非同期

前述のとおり、本衝突回避システムではワイヤレスネットワークを用いてメッセージを交換する事により衝突回避をする。ワイヤレスネットワークは、通信遅延などが大きいので、センサーを用いたタイムリーな衝突回避は困難である。複数のモバイルロボットが同期処理をとりながらタスクを実行している場合、衝突回避を行うためにはセンサーを用いた衝突回避と同等のタイムリーな処理が必要となる。以上のことから、モバイルロボットが実行するタスクは非同期であるとする。

3.2 予約

前述のとおり、本稿における衝突回避システムは、モバイルロボットの移動経路を予め予約しておく。本節では、モバイルロボットの移動経路が予約する範囲に関して述べる。

モバイルロボットの移動経路の予約は、平面上で行う物とする。モバイルロボットを利用する環境には凹凸がある可能性や段差を上下するなどの可能背が考えられる。しかし、地中へ潜れるや飛行可能なモバイルロボットは前提としていないため、凹凸や段差に関しては平面ととらえる事ができる。

図 3.1 は、モバイルロボットが予約する範囲の概要を示す。

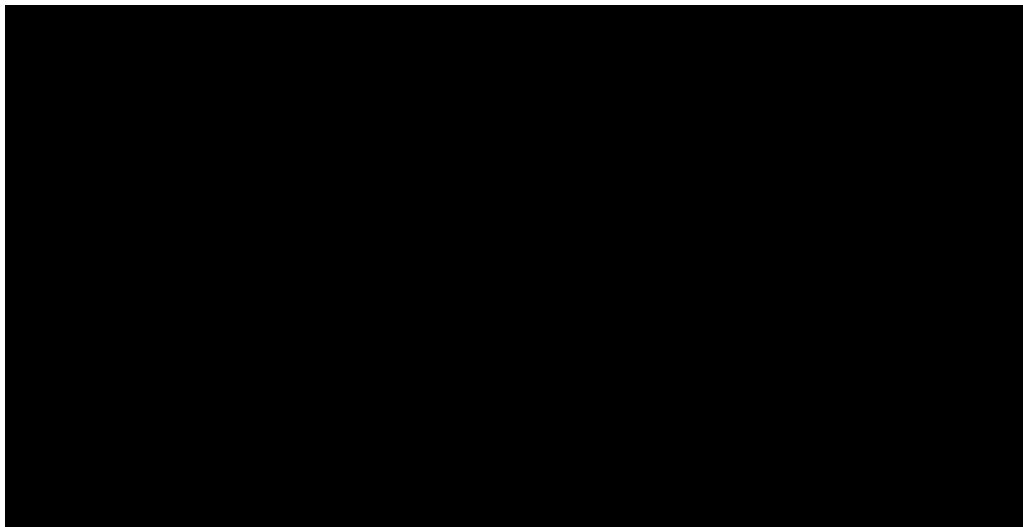


図 3.1: モバイルロボットの予約

移動経路を予約するモバイルロボットの外形を，モバイルロボットの中心から半径 r の円とする．モバイルロボットの予約する範囲を，モバイルロボットの中心とする円形にする事によりモバイルロボットの外観を考慮する事無く衝突回避が可能である．

モバイルロボットの移動経路の始点を $start$ とする． $start$ はモバイルロボットが移動経路を予約する始点であるため， $start$ 上にモバイルロボットの中心が存在する．予約する経路の始点から半径 r の範囲を $pre-zone$ とする．

end を予約する移動経路の目的地とする．モバイルロボットの移動の終着点の予約を $post-zone$ とする．

モバイルロボットの移動経路は， $start$ から end までとする．モバイルロボットの予約経路の $pre-zone$ と $post-zone$ の間の部分を $motion-zone$ とする．

実際にモバイルロボットが予約する範囲全体を $zone$ とし， $zone$ は $pre-zone$ と $post-zone$ ， $motion-zone$ を包括する部分を示す．

3.3 衝突の検出

本節では，モバイルロボットとモバイルロボットの衝突が発生する場合に関して述べる．

モバイルロボットの移動経路には，複数台のモバイルロボットが同時に衝突する可能性がある場合がある．しかし，複数台のモバイルロボットが関係する場合であっても，2台のモバイルロボット間での衝突の可能性の集合に分割可能である．

図3.2は，2台のモバイルロボット間で発生する衝突可能性がある場合を示している．アルファベットの A-H は，モバイルロボットの ID とする．

- モバイルロボット A の $motion-zone$ とモバイルロボット B の $motion-zone$

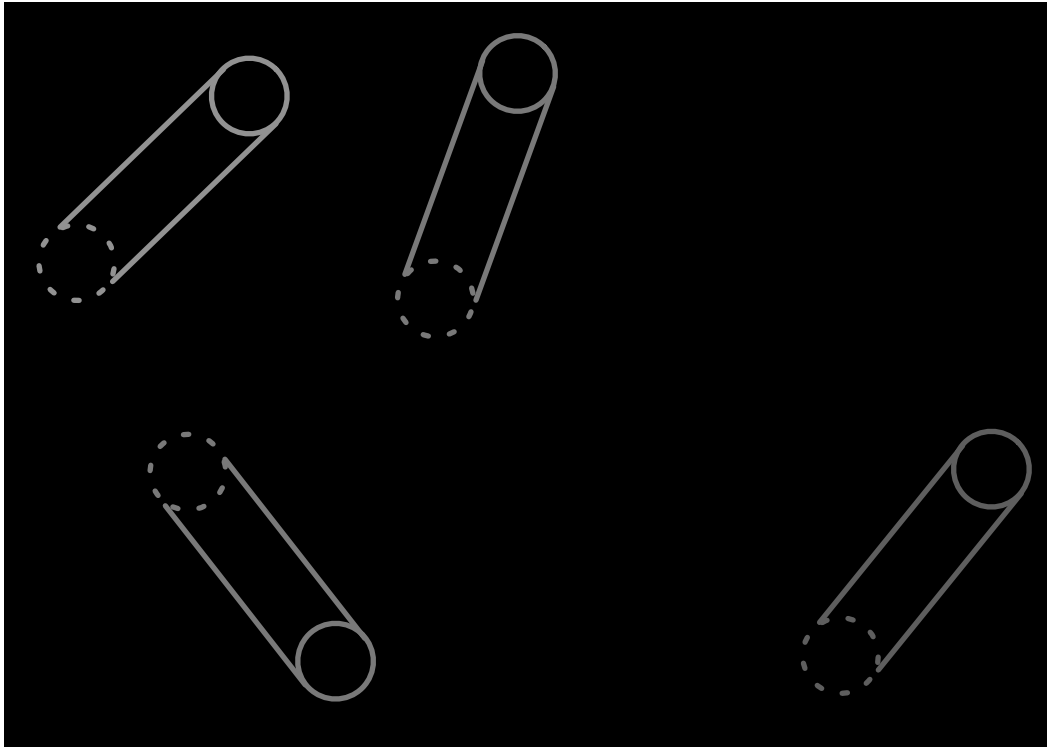


図 3.2: モバイルロボットどうして衝突の可能性がある場合

モバイルロボット A とモバイルロボット B が共に同じポイントを通過したい場合、モバイルロボット A が目的地（post-zone）へ到着し移動経路の予約を解放した後、モバイルロボット B が移動経路を移動可能になる。

- モバイルロボット C の motion-zone とモバイルロボット D の post-zone
モバイルロボット C の移動経路上にモバイルロボット D の移動の目的地が存在する場合、
- モバイルロボット E の post-zone とモバイルロボット F の post-zone が一つ以上の交点を持つ場合
モバイルロボット E の目的地とモバイルロボット F の目的地に共通点がある場合、モバイルロボット F は、モバイルロボット E が目的地（post-zone）まで移動し、次の移動経路を予約し解放するまで移動できない。
- モバイルロボット G の pre-zone とモバイルロボット H の post-zone
モバイルロボット G が目的地まで移動し移動経路の予約を解放するまで、モバイルロボット H は移動する事が出来ない。
また、モバイルロボット A の pre-zone とモバイルロボットの pre-zone において交点を持つ事はない。なぜなら、pre-zone どうして交点を持つ場合、すでに衝突が発生

している事になるためである。

以上が，モバイルロボットどうしでの衝突が発生する可能性のある場合である．しかし，前述の衝突の可能性以外にも衝突が発生する場合があるが，通常予約方法ではデッドロック [9] になる場合がある．[9] では，分散システムに置けるデッドロックに関して述べている．基本的なデッドロックに関しては，モバイルロボットの経路を予約する場合にも共通の事が言える．次小節では，デッドロックになるモデルに関して述べる．

3.3.1 デッドロックモデル

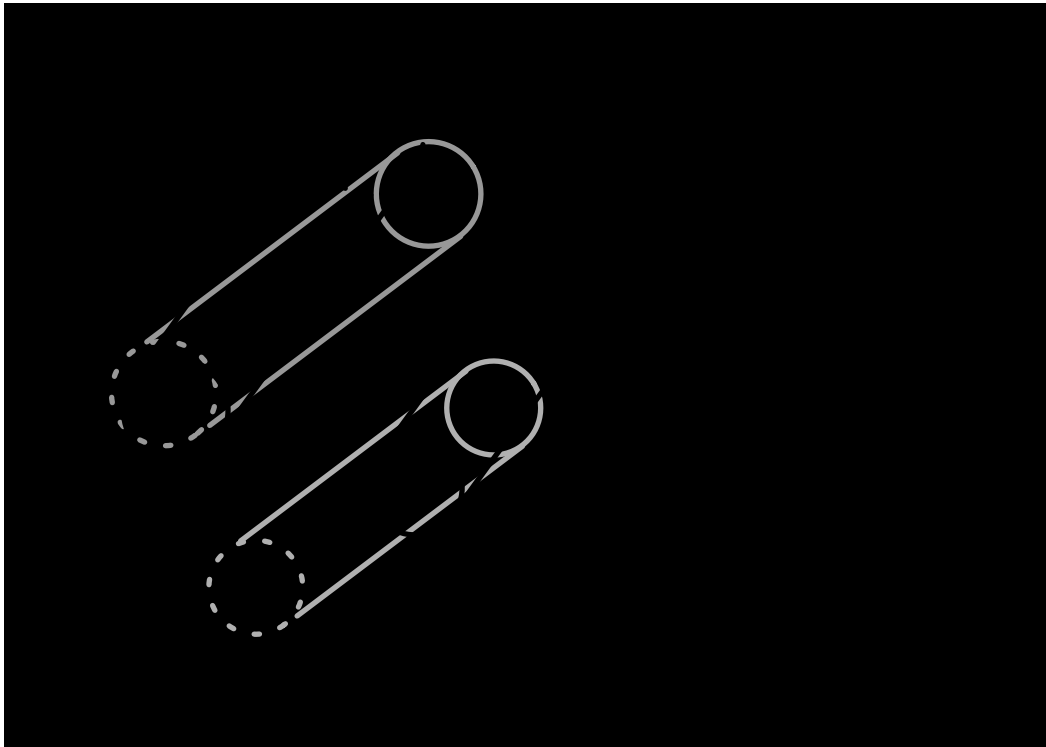


図 3.3: デッドロックが発生する場合

本小節では，複数台のモバイルロボットの移動経路予約で衝突が発生する可能性があるが，デッドロックになる場合に関して述べる．

図 3.3 は，2 台のモバイルロボット間でデッドロックが発生する場合である．アルファベット A-D は，モバイルロボットの ID を示している．

1. モバイルロボット A の pre-zone とモバイルロボットの post-zone が 1 つ以上の共通点を持ち，かつ，モバイルロボット A の post-zone とモバイルロボット B の pre-zone が 1 つ以上の共通点を持つ場合．

モバイルロボット A の目的地にモバイルロボット B が存在し、かつ、モバイルロボット B の目的地にモバイルロボット A が存在している場合、双方のモバイルロボットが、相手のモバイルロボットが移動し予約の解放を待ち続け、双方ともに移動できなくなる。

2. モバイルロボット C の pre-zone とモバイルロボット D の motion-zone が共通点を持ち、かつ、モバイルロボット C の motion-zone とモバイルロボット D の pre-zone が共通点を持つ場合。

モバイルロボット C の移動経路上にモバイルロボット D が存在し、かつ、モバイルロボット D の移動経路上にモバイルロボット C が存在する場合、本デッドロックは、1. の特別な場合である。

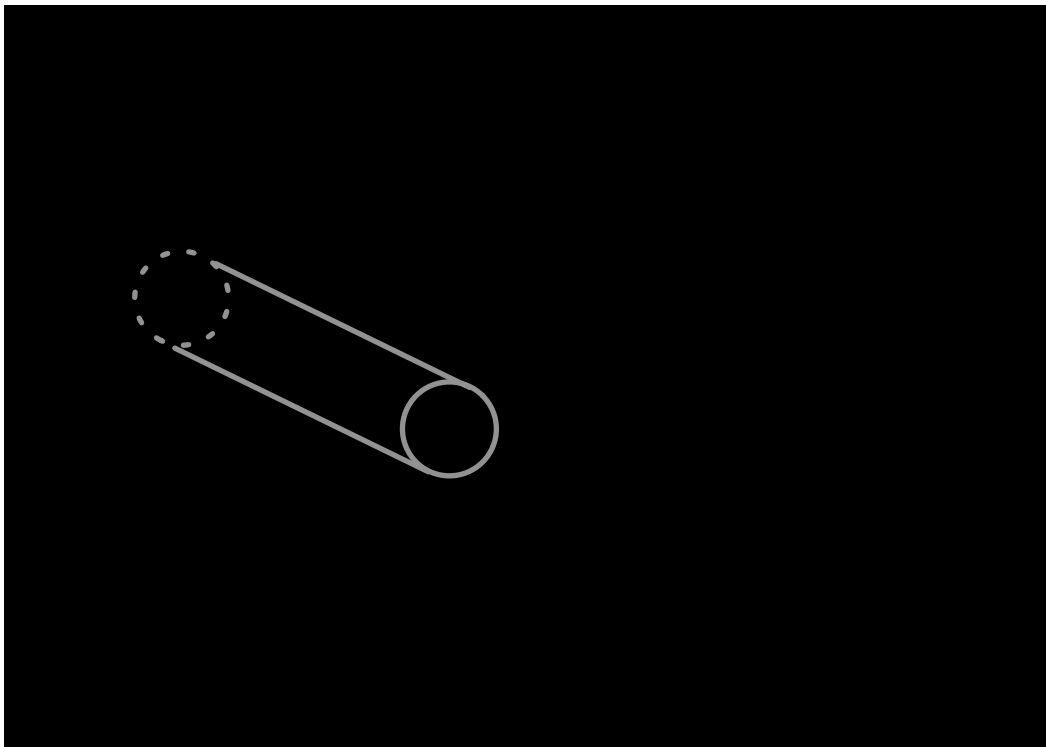


図 3.4: 環状のデッドロックが発生する場合

図??は、3 台以上のモバイルロボットが予約を環状に要求しデッドロックになるケースである。

まず、モバイルロボット A が移動経路の予約をした場合に、モバイルロボットの post-zone を予約したとする、モバイルロボット B は、モバイルロボット A が予約の要求をした後で、モバイルロボット A の zone と交差のある予約を要求する。最後に、モバイルロボット C がモバイルロボット B の zone と交差のある移動経路を予約する。

モバイルロボット A はモバイルロボット C が移動すれば移動可能となるが、モバイルロボット C はモバイルロボット B が移動するまで移動できない。また、モバイルロボット B もまた、モバイルロボット A が移動しないと移動できない。以上の場合にも、デッドロックが発生する可能性があるが、環状に予約する場合には予約の順序によってデッドロックが発生したり発生しなかったりする。

3.4 total-order broadcast

total-order broadcast は、分散システムにおける基本的なコミュニケーション方法である。total-order broadcast は、全てのメッセージを同じ順序で確実に届ける基本的な broadcast として定義されている。

total-order broadcast は、次のように定義される。

total-order broadcast は、以下の2つの基本的な操作から成り立つ。m は、ブロードキャストされるメッセージを示す。

- TO-broadcast(m)

モバイルロボット p が To-broadcast(m) を実行する場合、p TO-broadcast m と表記する。

- TO-deliver(m)

p TO-broadcast m を実行した場合、全てのモバイルロボットは To-deliver m を実行する。

本節で述べる正確なモバイルロボットとは、クラッシュなどの理由によりまちがった内容のメッセージを送信するモバイルロボットを示す。また、ネットワークトラフィックなどの問題によりメッセージの送受信に極端に大きい遅延のあるモバイルロボットなども正確なモバイルロボットとはしない。

- 停止性

正常のモバイルロボットがメッセージ m を TO-broadcast した場合、そのモバイルロボットは最終的に TO-deliver する。

- 一律合意

あるモバイルロボット（正常、故障には関わらない）がメッセージ m を受容した場合、正常な全てのモバイルロボットは m を TO-deliver する。

- 妥当性

メッセージ m を TO-Deliver(m) した場合、そのメッセージは TO-broadcast されたものである。また、どのメッセージも各モバイルロボットで最大1回しか TO-deliver しない。

- 全順序

モバイルロボット P_i と P_j がメッセージ m を TO-deliver しており, P_i が m の TO-deliver 以前に別のメッセージ m' を TO-deliver しているならば, P_j も m より前に m' を TO-deliver する.

以上が total-order broadcast

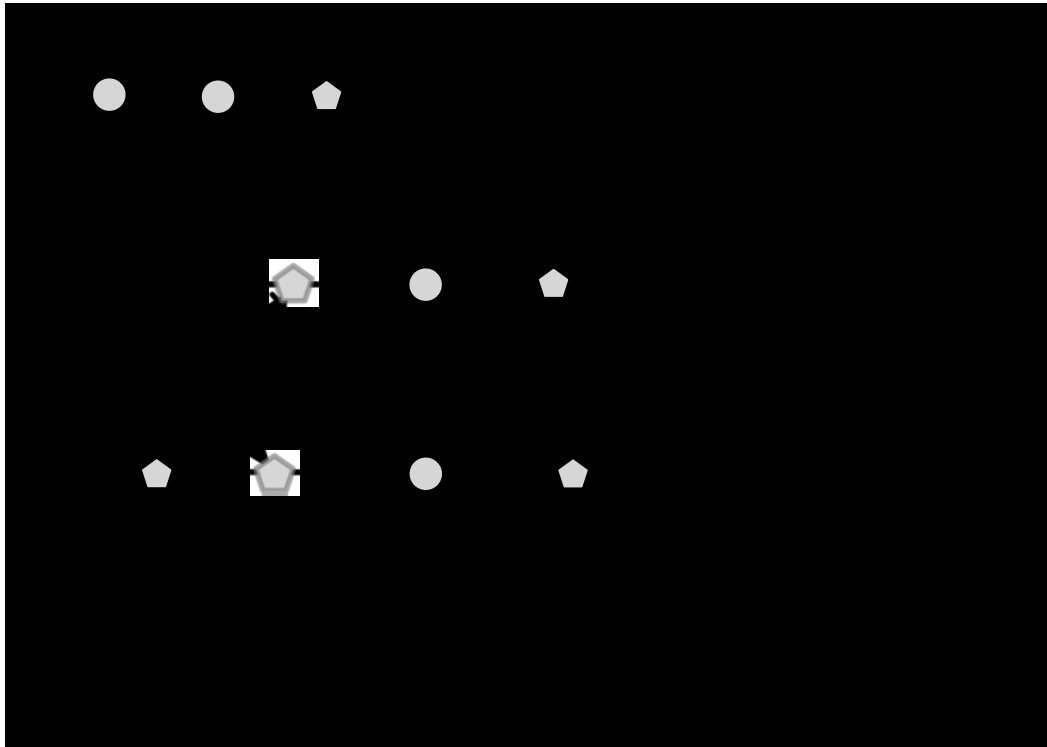


図 3.5: total-order broadcast の概要

図3.5は, total-order broadcast の概要図である. モバイルロボット A が TO-broadcast(m_1) を実行し, その後モバイルロボット C が TO-broadcast(m_2) を実行した場合についての図である.

モバイルロボット A がメッセージ m_1 を TO-broadcast し, メッセージ m_1 を TO-deliver する. しかし, モバイルロボット B と C は何らかの遅延によりモバイルロボット A からのメッセージを TO-deliver する事が遅れた. モバイルロボット C がメッセージ m_2 を TO-broadcast する. モバイルロボット A はメッセージ m_1 をすでに TO-deliver しているため, メッセージ m_2 を TO-deliver する. しかし, モバイルロボット B と C はメッセージ m_1 を TO-deliver していないため, メッセージ m_2 を受信しても TO-deliver しない. モバイルロボット B と C は, メッセージ m_1 を受信するとそれぞれのモバイルロボットがメッセージ m_1 を TO-deliver する. モバイルロボットが m_1 を TO-deliver すると, すでに受信しているメッセージ m_2 を TO-deliver する.

第4章 システム概要

本章では，ネットワークコミュニケーションを用いた衝突回避システムの概要に関して述べる．

前述のとおり，本衝突回避システムは，各々のモバイルロボットが移動したい経路を，ネットワークコミュニケーションを用いて交換し予約するシステムである．移動経路を予め予約する事により，他のモバイルロボットが同じ場所を予約不可能とする事により確実に衝突を回避して移動できる．

4.1 システム概要

本節では，モバイルロボット間における衝突回避システムの全体の動作に関して述べる．

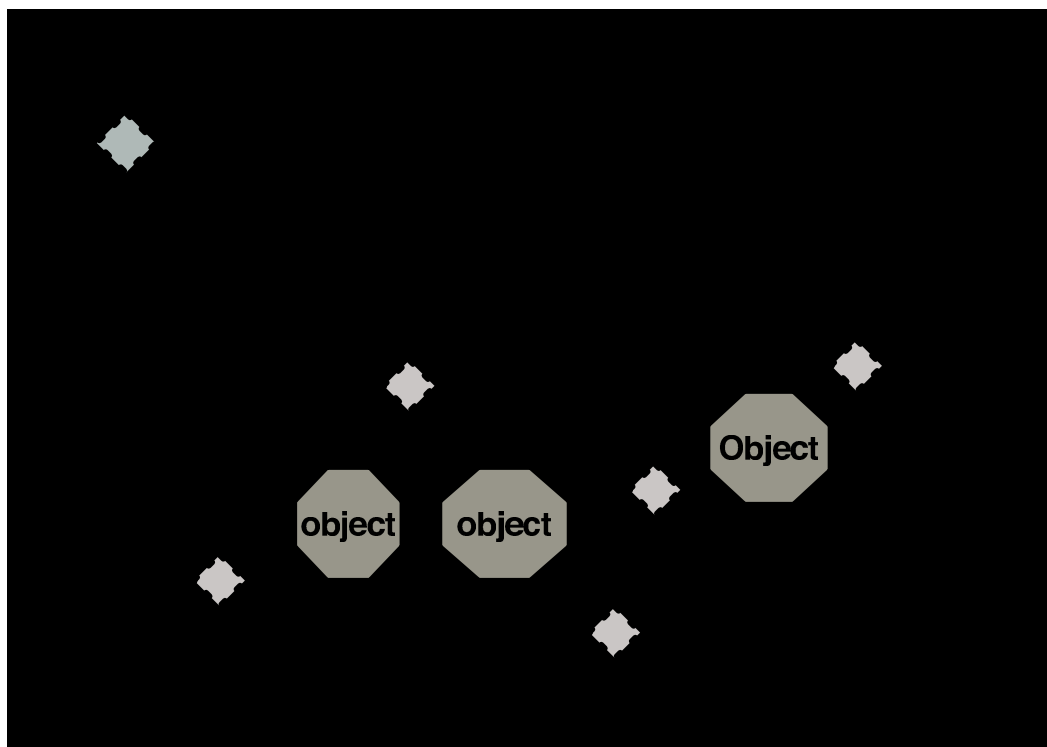


図 4.1: 移動経路予約システムの概要

図 4.1 は，3 台のモバイルロボットがある環境を移動する場合に関してである．各モバ

イルロボットは、移動開始前のモバイルロボットの位置 (X : X はモバイルロボット ID, A-C) から移動の目的地で (eX : X はモバイルロボットの ID, A-C) までを移動したいとする。Moving Path はモバイルロボットの移動経路, target point はモバイルロボットの移動経路の目的地を示す。object は、モバイルロボット以外の障害物を示している。モバイルロボットの障害物以外との衝突回避は、本システムの外部のシステムが移動経路の設計時に同時におこなう。また、各モバイルロボットが、各々独自に移動した場合、collision doubtful のポイントで衝突が発生する場合がある。本システムは、collision doubtful のポイントで発生する可能性のある衝突を回避する。

1 台のモバイルロボットが移動経路全てを 1 度に予約する事は、移動経路の予約を管理する事が容易となる。また、モバイルロボットの台数が増加した場合でも、移動経路の予約に必要な通信のコストも少なくなる。しかし、他のモバイルロボットは、予約しているモバイルロボットが移動を終了し予約を解放するまで移動する事ができず長い時間を待機する事になる。移動経路予約し移動しているモバイルロボットとの、衝突の可能性のあるモバイルロボットの台数が多くなるほど、モバイルロボットの待機時間は長くなる。モバイルロボットはタスクを実行するために移動をする場合が多いため、全ての任意の時間 t に対して移動中のモバイルロボットの台数は可能な限り多い事が重要である。

時間 t において、可能な限り多くの台数のモバイルロボットが移動可能にするためには、移動経路を切断し、切断した移動経路の断片を予約する方法がある。

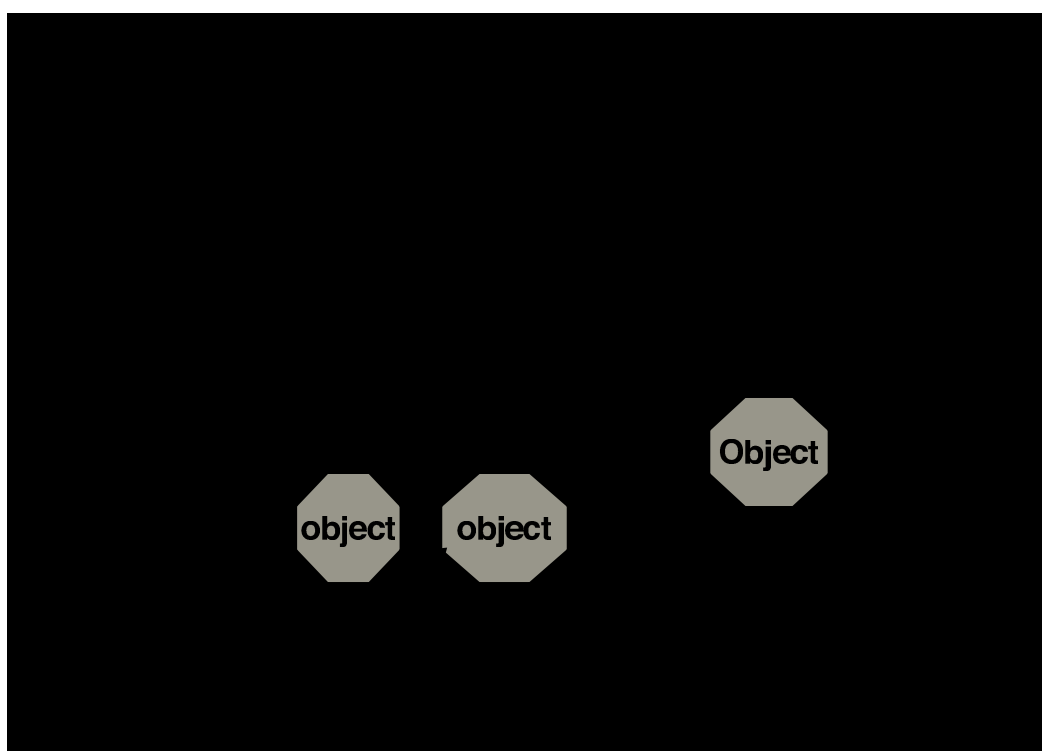


図 4.2: 移動経路予約システムの概要

図 4.2 は、図 4.1 の移動経路を切断して予約した場合を示す。

モバイルロボットが移動経路の始点から目的地まで移動する過程について述べる。

1. 移動経路の切断

モバイルロボットは、始点から目的地までの移動経路を切断し小さな経路に分割する。

2. 経路の予約

モバイルロボットが目的地まで移動するためには、モバイルロボットと隣接する最初の経路 (zone) を予約する必要がある。モバイルロボットは、zone を予約するために衝突回避システムに対して予約の要求を送信する。

予約の要求メッセージには、モバイルロボットの ID と移動経路の情報が含まれる。

3. 移動

衝突回避システムが予約の要求が許可されると、移動が許可された事を対象の ID を持つモバイルロボットに通知する。通知を受けたモバイルロボットは、pre-zone から post-zone まで motion-zone の移動を開始する。

4. 予約の解放

モバイルロボットが post-zone に到着すると、予約を解放する必要がある。モバイルロボットは、予約の解放メッセージを衝突回避システムへ送信し、2 の経路予約へ戻り処理を繰り返す。

モバイルロボットが1度に予約する移動経路の距離を短くする事により、衝突の可能性のある他のモバイルロボットの待機時間は短くなる。しかし、予約する移動距離を短く設定しすぎた場合、移動経路を予約するための通信頻度が多くなる。本システムはワイヤレスネットワークコミュニケーションを用いるため、通信頻度の増加は通信遅延などが発生しモバイルロボットの待機時間が長くなる。

次節以降では、衝突回避システムの予約の管理方法に関して述べる。

4.2 集中管理型

前節では、モバイルロボットがどのように移動経路の予約を行うかに関して述べた。本節では、衝突回避システムが移動経路の予約を集中的に管理する方法に関して述べる。集中管理型は、特定のサーバやモバイルロボットが全てのモバイルロボットの移動経路予約を集中的に管理する方法である。

図 4.3 は、モバイルロボットが zone の予約時の移動経路の集中管理の概要である。サーバ上で衝突回避システムが起動しており、モバイルロボット上では衝突回避システムが起動していない。衝突回避システムは、reserved と request から構成される。reserved とは、モバイルロボットを実際に利用している環境と同じ状態を持ち、モバイルロボットの

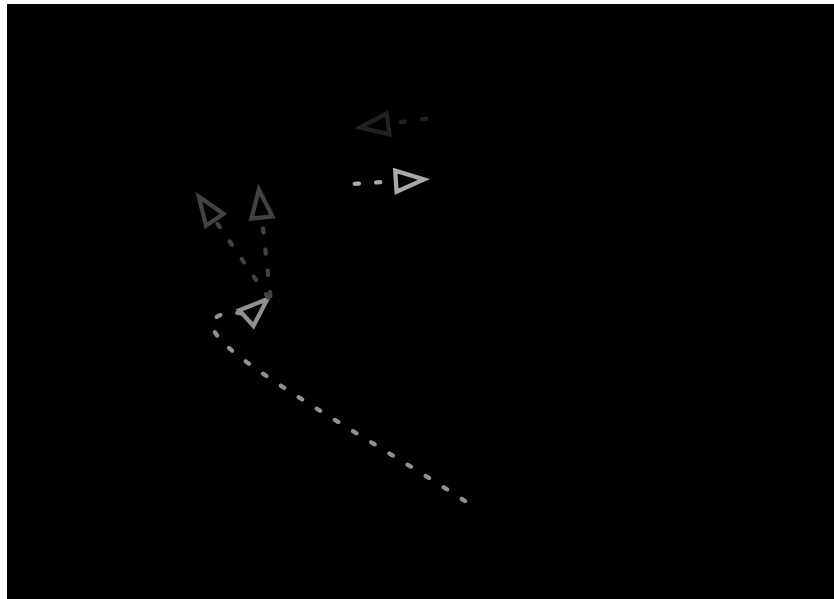


図 4.3: 集中管理型に置ける移動経路の予約

利用環境内全てのモバイルロボット情報がある。すなわち，reserved は，予約が受理され移動中のモバイルロボット情報と，待機中のモバイルロボットの待機位置の情報を持つ。request は，予約の要求をおこなったが他のモバイルロボットの移動経路と衝突の可能背があるため待機しているモバイルロボットの予約情報がある。

移動経路の予約を集中的に管理する場合のシステム全体の動作を以下に述べる。

1. 予約の要求

モバイルロボットは，移動したい zone の予約要求をおこなう。予約要求は，ワイヤレスネットワークを用い衝突回避システムサーバーへ送信する。

2. request との確認

予約の要求をモバイルロボットから受信したサーバは，待機中のモバイルロボットの zone と衝突の可能性が無いかを確認する。

待機中のモバイルロボットは，既に動いているモバイルロボットと衝突の可能性があるので移動していない。新しく要求された予約を優先的に処理した場合，既に要求されている待機時間が増加し最悪の場合には移動できない。よって，先に要求された予約を優先的に受理するために，待機中のモバイルロボットとの衝突の可能性を確認する。

新しく要求された予約が，待機中のモバイルロボットの予約と衝突の可能性があるのであれば，request に新しく要求された予約を入れ，サーバは次の要求を 1. へ戻り次の予約が要求されるのを待つ。

3. reserved との確認

request 内にある待機中であるモバイルロボットの予約全てと衝突の可能性が無い場合には、reserved 内の全てのモバイルロボットと衝突の可能性の有無を確認する。reserved 内には、移動中のモバイルロボットの zone と待機中のモバイルロボットの pre-zone があるが、モバイルロボットは待機中で静止しているモバイルロボットとも衝突の可能性があるため、両方の情報と確認する。

reserved 内の 1 台以上のモバイルロボットの予約と、新しく要求された予約が衝突の可能性のあるばあいには、新しく要求された予約を request へ入れる。衝突の可能性がない場合には、reserved 内の同じ ID のモバイルロボット情報と交換し、予約と同じ ID を持つモバイルロボットへ移動可能の通知を行う。

4. 予約の解放

モバイルロボットが pre-zone から post-zone まで移動が終了した場合、予約を解放する必要がある。モバイルロボットが post-zone へ到着すると、モバイルロボットは経路解放メッセージをサーバに対して送信する。サーバが解放メッセージを受信した場合、メッセージが持つ ID の予約から post-zone の情報を利用し、reservation におけるモバイルロボットの情報を待機状態にする。

以上が、モバイルロボットの移動経路予約の集中管理型のシステム動作である。

移動経路の集中的な管理は、モバイルロボットや予約の管理が容易にできる。しかし、モバイルロボットの台数が増加した場合などは、経路予約のための通信がサーバに集中し通信遅延が大きくなるため、移動できないモバイルロボットの台数が多くなる。また、サーバが何らかの理由でダウンした場合など single point of failure に弱く、crash recoveryなどを考慮する事が重要となり複雑となる。

4.3 分散管理型

モバイルロボットの移動経路予約を分散管理とは、集中管理型で述べた予約を管理するサーバのクローンを複数台のサーバで管理する方法である。

モバイルロボットの移動経路予約を集中的に管理する場合、特定のサーバが予約の管理をおこなうのでサーバが故障した場合には、複雑な crash recovery などを利用する事が重要となる。モバイルロボットの移動経路予約を分散的に管理する事により、特定のサーバを意識する事なく衝突回避が可能である。また、あるサーバが故障した場合でも、モバイルロボットの全体の移動に大きく影響する事がない。

図 4.4 は、モバイルロボットの移動経路の予約を分散的に管理する方法の概要である。衝突回避システムは、集中管理型で用いた衝突回避システムと同等のシステムを利用する。複数台のサーバ上で衝突回避システムを起動し分散的に予約を管理する方法が考えられるが、single point of failure の対処が必要となる。全てのモバイルロボットは、集中管

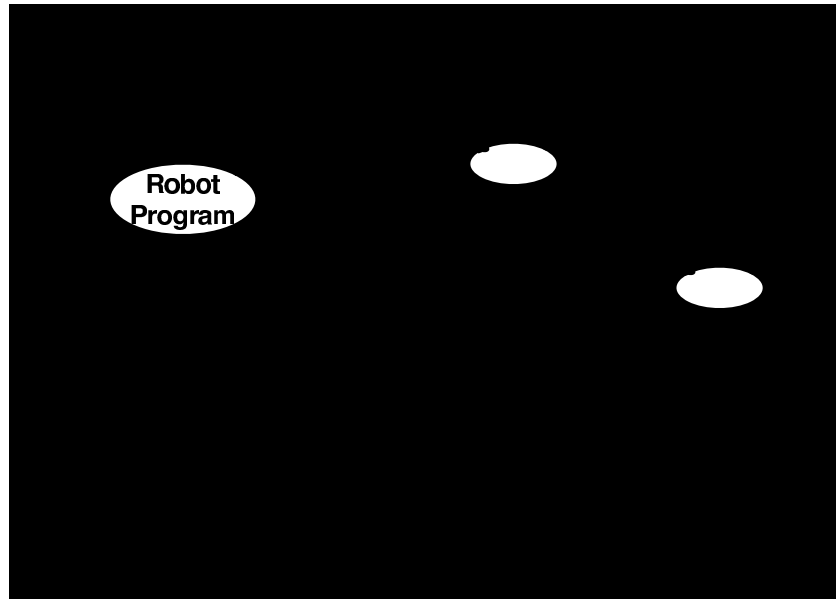


図 4.4: 移動経路の予約を分散管理する概要

理型で用いた衝突回避システムのクローンシステムを利用することにより，1 台のモバイルロボットに障害が発生した場合でも，他のモバイルロボットに影響する事はない。

分散管理型を用いた衝突回避システムの動作概要を以下に示す．衝突回避システム内での処理内容は，集中管理型で述べた処理と同じであるため省略する．

モバイルロボット A とモバイルロボット B が移動したい場合に関して述べる．

1. モバイルロボット A の予約の作成と送信

モバイルロボット A とモバイルロボット B は，移動の目的地と移動経路を作成する．作成された移動経路 zone をモバイルロボットは，グループコミュニケーションを用いて全ての衝突回避システムへ送信し，予約の要求をする．予約の zone を受信した衝突回避システムは，zone を受信した順に予約の確認をおこなう．図の場合，衝突回避システムは，モバイルロボット A の zone の要求を先に受信したため，モバイルロボット A の zone の予約を受理する．

2. モバイルロボット A の移動

衝突回避システムがモバイルロボット A の予約が受理されると，モバイルロボット A に対して移動の許可を通知する．衝突回避システムからの予約受理のメッセージを受け取ったモバイルロボット A は，zone を基に実査に移動を開始する．

3. モバイルロボット B の予約の作成と送信

衝突回避システムがモバイルロボット B の zone の予約を受信した場合，既にモバイルロボット A が移動しているためモバイルロボット B の要求を request へ投入する．モバイルロボット B は，予約が受理されるまで待機状態となる．

4. モバイルロボット A の移動終了

モバイルロボット A は、移動が終了すると移動終了の通知をグループコミュニケーションを用いて全ての衝突回避システムへ通知する。モバイルロボット A から移動終了の通知を受け取った衝突回避システムは、モバイルロボット A の予約を解放し、request にあるモバイルロボット B の予約を確認する。

5. モバイルロボット B の移動

衝突回避システムがモバイルロボット B の予約を受理すると、衝突回避システムは移動の許可メッセージをモバイルロボット B へ送信する。衝突回避システムから移動許可を受け取ったモバイルロボット B は、予約した zone の移動を開始する。

6. モバイルロボット B の移動終了

モバイルロボット B が post-zone まで移動を終えると、グループコミュニケーションを用いて衝突回避システムへ予約の解放メッセージを送信する。予約の解放メッセージを受けた衝突回避システムは、解放メッセージを基に reserved の予約を解放する。

以上が分散管理型を用いた衝突回避システムの動作である。

モバイルロボットの移動経路の予約を分散管理する場合、全てのモバイルロボットの衝突回避システムが全く同じ順序で予約と解放のメッセージを受け取る事が重要である。各モバイルロボットの衝突回避システムが、異なる順序でメッセージを受信した場合、衝突回避システムごとに異なるモバイルロボットが移動可能となる。衝突回避システムごとに異なるモバイルロボットが移動可能となった場合、衝突の可能性がある複数台のモバイルロボットが同時に移動を始め、衝突する可能性がある。

全ての衝突回避システムへ全く同じ順序でメッセージを渡す方法として、total-order broadcast がある。total-order broadcast に関しては、前述のとおりである。total-order broadcast を用いる事により、全てのモバイルロボット上で稼働している衝突回避システムが全く同じ順序で予約の要求を受ける事ができる。また、1 台のモバイルロボット上の衝突回避システムが予約の要求 A を受けたなら、全てのモバイルロボット上の衝突回避システムが予約の要求 A を受信することができる。

4.4 request

本節では、衝突回避システムの request において、zone をどのように管理するかに関して述べる。

前節で述べたとおり、予約を分散管理型の衝突回避システムを用いる場合、全ての衝突回避システムへ同じ順序で入力する事が重要となる。特に、request は zone の予約を要求しているが、他のモバイルロボットと衝突の可能性があるため待機中のモバイルロボット

が収納されている。全ての衝突回避システムが全く同じ入力を受けたとしても、待機中のモバイルロボットを正しく管理しないと衝突回避システムごとに異なるモバイルロボットに移動の許可を与える場合がある。

4.4.1 queue

入力の順序を確実に管理する方法に、queue を用いる方法がある。
queue を用いた request の管理方法を述べる。

1. total-order broadcast から受けた新しい zone の予約は、queue の最後尾に格納する。
2. queue の最後尾に格納された zone は、queue のひとつ前の要素と衝突の可能性があるかを確認する。

衝突の可能性がない場合には、1つ前の要素と入れ替え、衝突の可能性があるまで繰り返す。衝突の可能性がある場合には、処理を停止し、1. から新しい予約の衝突の可能性を確認する。

3. 全ての queue のと衝突の可能性がない場合には、reserved の全ての要素と衝突の可能性を確認する。

全ての reserved の要素と衝突の可能性がない場合には、zone の予約を許可する。reserved の要素と衝突の可能性がある場合には、request の先頭に予約の zone を入れる。

以上を繰り返す事により queue を用いた予約の管理をする。

queue を用いて衝突回避を行う場合、zone の予約の入力順序を容易に管理することができる。

しかし、システムモデルで述べたデッドロックを検出する事は、困難である。

4.4.2 グラフ

デッドロックを予防するために、我々は request の管理にグラフを用いた。グラフを用いた予約の管理方法の概要を以下に示す。

モバイルロボットの予約には、次の2種類が考えられる。

- 移動のための予約
- 待機中のモバイルロボットの予約

移動のための予約に関しては前述のとおりである。受理された予約を移動し終え予約を解放した後も、モバイルロボットは post-zone に存在する。モバイルロボットは、移動中

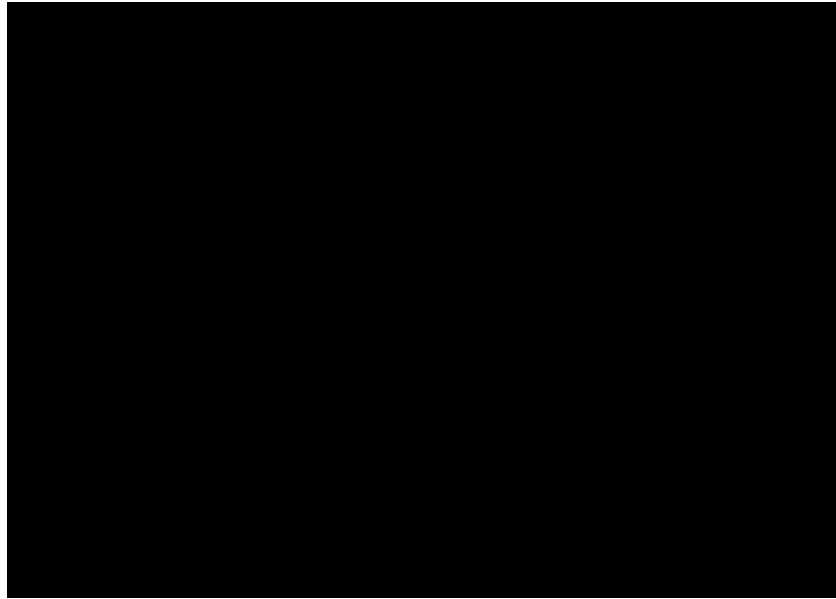


図 4.5: グラフの概要

や移動経路の予約をしているモバイルロボットの他にも待機中のモバイルロボットも考慮する事が重要である。

図 4.5 は、グラフを用いた場合の予約受理システムの概要である。グラフを用いた予約の管理は、次の 3 層で構成される。

- wait

要求した予約が他のモバイルロボットの移動経路と交差し衝突の可能性がある場合、要求された予約は衝突の可能性のあるモバイルロボットの経路予約解放を wait で待つ。

- move

受理されたモバイルロボットの予約は、移動可能状態になり move に入る。

- idel

移動を終了して、予約を解放したモバイルロボットのリストである。

モバイルロボット環境は、実際にモバイルロボットを利用する環境を示す。環境には、モバイルロボット 1-6 がある。予約受理システムは、衝突回避システムの予約の受理をする部分をしめす。モバイルロボット中央の数字は、モバイルロボットの固有 ID をしめす。

モバイルロボット 1 とモバイルロボット 5 は停止中のモバイルロボットをしめす。モバイルロボットの 1 と 5 は、予約受理システムの idel 中の i1 と i5 にマッピングする。モバイルロボット 2 と 3 は移動中のモバイルロボットをしめす。移動中のモバイルロボット 2 と 3 は、予約受理システムの move 内のモバイルロボット m2 と m3 にマッピングされる。

モバイルロボット 4 と 6 は、移動のための予約を要求したが、他のモバイルロボットの移動経路や停止中のモバイルロボットと衝突の可能性があるため、待機状態である。予約受理システムの wait 内 w4 と w6 にマッピングされる。また、予約の要求をおこなったが受理されていないモバイルロボットに関しては、モバイルロボットの実体の位置の予約が必要であるため、idle ないの i4 と i6 にマッピングされる。予約受理システムの w4 は、移動中を示す m2 の移動経路と衝突の可能性があるため m2 と相互的にリンクする。また、w6 は停止中のモバイルロボット i5 と衝突の可能性があるため i5 と相互リンクする。

新しく予約を要求する場合、予約の受理システムはまず wait 内の全てのモバイルロボットの予約と衝突の可能性を確認する。衝突の可能性がある場合には、衝突の可能性のあるモバイルロボットの予約と相互リンクする。wait 内の衝突の可能性の有無にかかわらず、新しい予約の要求は move 内の全てのモバイルロボットとも衝突の可能性に関して確認をする。衝突の可能性がある場合には wait の場合と同様に対象の予約と相互リンクする。また、idle 中の予約の全てとも衝突の可能性の有無に関して確認をし、同様の作業を行う。

モバイルロボットが移動を終了し、予約経路の解放を予約の受理システムが受け取ると、move 内の予約を消去し idle 内へ post-zone の予約を入れる。同時に、予約を解放した事を予約中に待機していた予約へ対して通知し、リンクを削除する。

以上の作業を繰り返す事により、グラフを用いて衝突回避を行う。

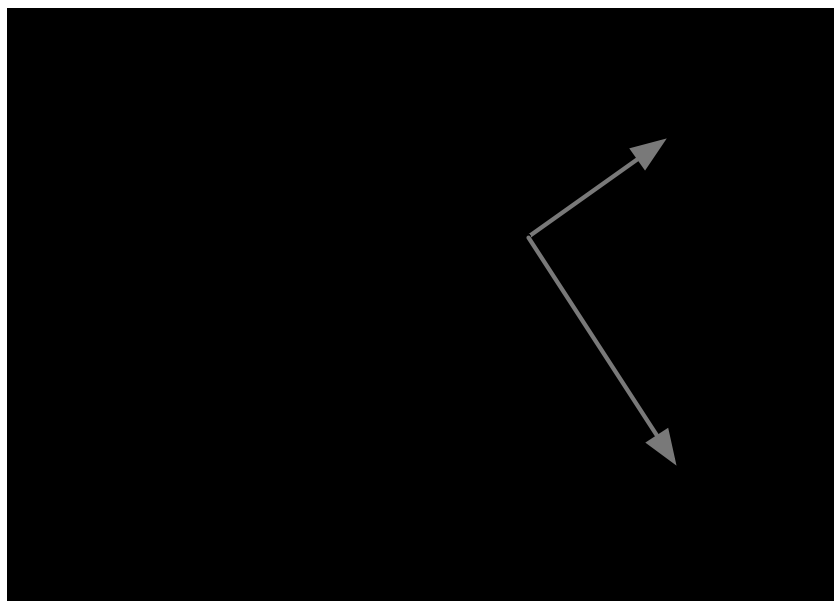


図 4.6: デッドロックの予防

また、グラフを用いる事により予約のデッドロックを予防することができる。図 4.6 は、グラフを用いた場合にループのデッドロックが発生した場合に関してである。モバイルロボット A, B, C があり、それぞれがループ状に予約した場合を示している。デッドロックが発生する場合は、idle 中のモバイルロボットの予約を先頭として、同じ線上に同じ ID を持ったモバイルロボットが wait で待機した場合にデッドロックが発生する。以上の状

態になる事を検出すると、wait の一番最初の予約を取りやめ改めて予約する事によりループ状のデッドロックの場合には順序が変わるために予防することができる。しかし、デッドロックのモデルの中には、予約が重複している場合もある。よって、再予約する場合には、カウンターを利用し、再予約数が善モバイルロボットの台数よりも大きくなった場合にはループのデッドロックではなく重複である事がわかるので、移動経路を再設計する。以上の方法により、デッドロックを予め予防することができる。

第5章 シミュレーション

本章では、前章で述べたシステムをシミュレーションを用いて性能評価を行う。性能評価は、環境内にあるモバイルロボットの密度と予約する zone の長さの関係に関して調査する。

5.1 シミュレーション目的

本稿で述べる衝突回避システムの性能評価をおこなう事が重要である。本衝突回避システムの性能評価のために、実際のモバイルロボットを利用して評価をおこなう方法が考えられる。しかし、個々のモバイルロボットの費用は高額であるため、性能評価をおこなうために十分な台数のモバイルロボットを用意する事は困難である。また、実際のモバイルロボットを用いて性能評価を行う場合、各モバイルロボットの実際の移動制御のためのプログラムを書く必要があり、その他にも障害物との衝突回避システムなどの実装も必要となる。以上の事から、本稿ではシミュレーションを用いて衝突回避システムの性能評価をおこなう。

5.2 シミュレーション環境

本節では、本稿で述べる衝突回避システムのシミュレーションをおこなう環境に関して述べる。

- プログラム言語

シミュレーションは、java ver.1.5 を用いた。

モバイルロボットのシステムは、モバイルロボットごとに異なる。java はマルチプラットフォームで利用できるため、多種のモバイルロボットが混在した環境でも共通して利用できる。また、シミュレーションを java で書く事により、実際の衝突回避システムを実装する場合にも転用可能である。

java ver.1.5 は次のサイトで公開されている。

<http://java.sun.com/j2se/1.5.0/ja/download.html>

- シミュレーター

シミュレーターは、desmo-j ver.2.1.1 を用いた。

desmo-j[10] は、Department of Computer Science, University of Hamburg で開発された離散事象型のシミュレータである。現在、2006 年 4 月にバージョン 2.1.1 がリリースされている。また、1999 年 10 月にバージョン 1.0 がリリースされた。

desmo-j は、java で書かれたシミュレーションモデルのプログラムを実装するためのオブジェクト指向フレームワークである。離散的で論理的なイベントシミュレーションをサポートする。

desmo-j は以下のサイトから入手できる。

<http://asi-www.informatik.uni-hamburg.de/desmoj/>

5.3 シミュレーションモデル

本節では、前述のシステムモデルを基に作成したシミュレーションのモデルについて述べる。

- 仮想空間

モバイルロボットを利用する環境は、1 辺が 100 メートルの正方形であると想定する。空間と空間外の境界には壁が存在しモバイルロボットは境界を越えて zone を予約する事は出来ない。また、モバイルロボットは凹凸などではなく平面上を移動するものとする。

空間内には、モバイルロボットのみが存在しモバイルロボット以外の障害物は存在しないものとする。

空間内の全ての場所でワイヤレスネットワークが安定して利用可能とする。空間内では GPS などのインフラストラクチャが確実に利用可能であり、モバイルロボットは任意の場所で随時に空間内の絶対座標を取得することができる。

- モバイルロボット

図 5.1 は、具体的なモバイルロボットモデルの 1 例である。写真は、Acroname 社の GARCIA である。図上段は GARCIA の上部からの写真で、下段は GARCIA の横からの写真で、詳しい情報は次のアドレスから入手可能である。

<http://www.acroname.com/garcia/garcia.html>

GARCIA は、全長が約 27.6 センチメートルのモバイルロボットで、車輪の回転により前後や回転移動をする。

モバイルロボットは、車輪などを用いる事により自律的に移動可能である。

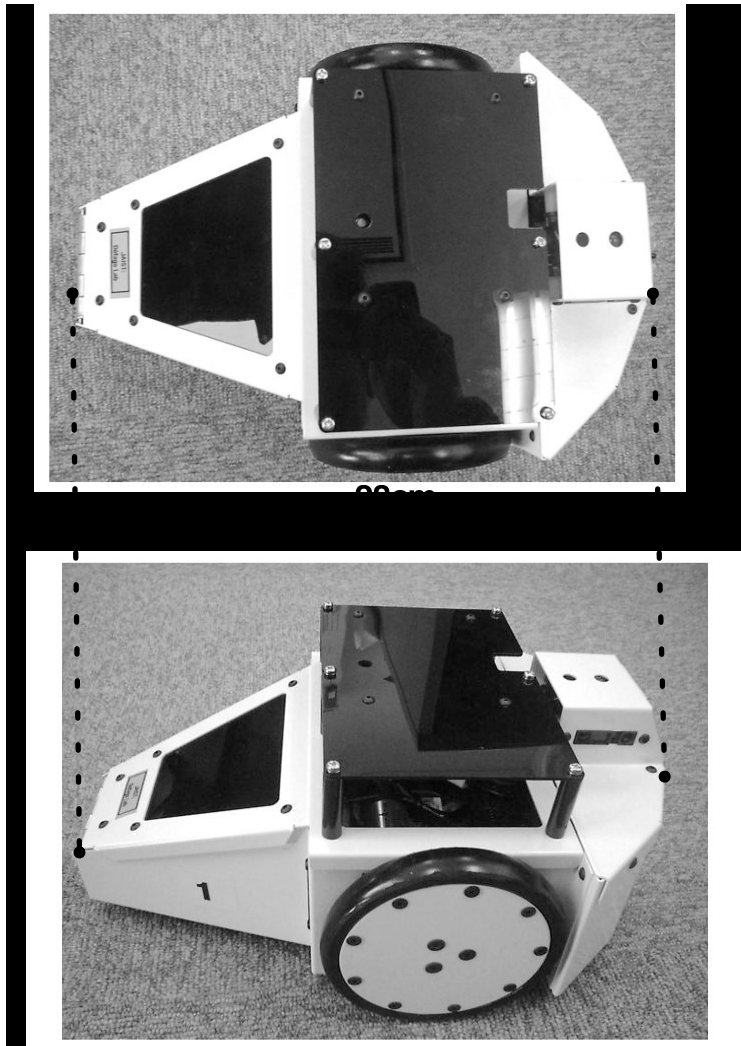


図 5.1: モバイルロボットモデル

モバイルロボットは、半径15センチメートルの円とする。移動速度は1メートル/秒で移動し、加速や減速のために移動速度は変化しない物とする。

モバイルロボットは、ワイヤレスネットワークとポジショニングシステムが任意のタイミングで利用可能である。

モバイルロボットはモーションプランニングシステムを利用し、直線的な移動経路を設計可能である。全てのモバイルロボットは、非同期のタスクを実行しているため、全てのモバイルロボットは他のモバイルロボットのタスクを意識する事無く移動可能である。

5.4 シミュレーションシナリオ

本節では、シミュレーションをどのように行うかに関して述べる。

シミュレーションは、シミュレーションモデルで述べた仮想空間にモバイルロボットを2台から30台まで2台ずつ増加させる。モバイルロボットの初期位置はランダムに設置する。

1度に予約する移動経路の長さは、5メートルから5メートルずつ増加させる。

1度のシミュレーションは、250000秒(約3日間)分をおこなう。シミュレーション中にモバイルロボットが仮想空間への出入りは無いものとする。

全てのモバイルロボットは、初期位置から同時に移動を開始する。モバイルロボットの動作は次のとおりである。

1. 初期化

各モバイルロボットは、ポジショニングシステムを用いて各々の初期位置を確認する。また、同時に各々の固有のIDの初期化をおこない、全てのモバイルロボット間で情報の共有を可能にする。

2. 目的地の作成

モバイルロボットは、移動の目的地を作成し、移動経路を作成する。

3. 予約

モバイルロボットは作成した移動経路を total-order broadcast へ送信し、衝突回避システムから移動の許可を受けるまで待機する。

4. 移動

衝突回避システムから移動の許可を受けたモバイルロボットは、予約の経路を移動する。移動を終了すると、ポジショニングシステムを用いて位置の確認をおこなう。次の目的地を決定するため2.から繰り返す。

以上がモバイルロボットの動作である。

5.5 シミュレーション結果と考察

本節では、前節で述べたシミュレーションシナリオのシミュレーション結果と考察に関して述べる。

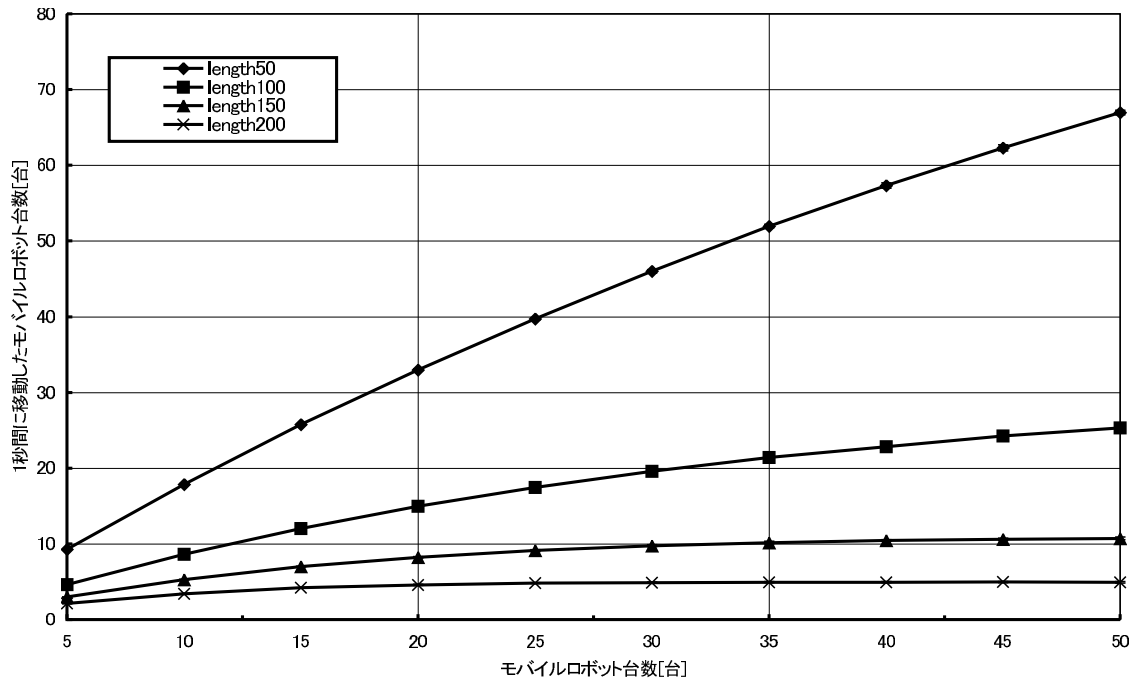


図 5.2: 1 秒間に移動したモバイルロボット台数

5.5.1 移動モバイルロボット台数

図5.2は、前述のシミュレーションモデルとシミュレーションシナリオを基にシミュレーションをおこない、1秒間に移動したモバイルロボットの平均台数を示す。縦軸に1秒間に移動したモバイルロボットの平均台数、横軸に環境内に存在するモバイルロボットの台数を示している。各プロットには、95パーセントの信頼区間を示したが、全てのプロットで誤差範囲は0.1台以下と安定している事がわかる。

予約距離が0.5メートルの場合には、他の予約距離と比べ仮想空間内のモバイルロボット数が増加するとともに大幅に移動したモバイルロボット数も増加している。1度の予約距離を長くするにつれて仮想空間内のモバイルロボット台数の増加にともなう移動台数の増加の傾きは小さくなり、予約距離が2メートルの場合には移動したモバイルロボット台数の傾きは20台を超えてほとんど増加していない。

モバイルロボットの移動速度は、1秒間に1メートルである。

表5.1は、モバイルロボットが1秒間に最大で移動できる回数である。予約経路の長さが0.5メートルの場合には、予約経路が1メートルの場合と比べ2倍の台数のモバイルロボットが移動可能である。0.5メートルと1メートルの予約経路のグラフの傾きに大きな

表 5.1: モバイルロボットが1秒間に移動可能な台数

予約距離 メートル	1秒間の移動回数 回
0.5	2
1.0	1
1.5	0.66
2.0	0.5

差があるのは、1秒間に移動可能なモバイルロボットの台数の違いが原因であると考えられる。しかし、予約経路の長さが1.5メートルの場合と2.0メートルの場合にはグラフの傾きの大きな違いが見られない。また、予約の経路が2.0メートルの場合にはモバイルロボット台数が20台、予約経路の長さが1.5メートルでモバイルロボット台数が35台以上の場合には、1秒間の移動台数に大きな違いがみられない。以上の事から、予約経路の長さが長くなると、1度の予約経路に対して多くのモバイルロボットが待機中となるため、移動状態のモバイルロボット台数が少なくなる事が原因であると考えられる。

5.5.2 移動モバイルロボット台数

図5.3は、衝突が発生しないゴーストモバイルロボットがあると仮定し、衝突回避システムを利用しない場合の総移動台数を100パーセントとし、本衝突回避システムを用いた場合にどの程度の割合で台数のモバイルロボットが移動できたかを示している。横軸に仮想空間のモバイルロボット台数をしめし、縦軸に1秒間の移動確率をしめしている。図5.2のグラフと比べると、各予約距離の長さの間隔に大きな違いが無い。衝突の可能性がないモバイルロボットと比べた場合、1台のモバイルロボットが移動できる移動距離と速度は固定であるため、どの長さでも同じように台数増加に伴い移動できるモバイルロボット台数も減少したと考えられる。

5.5.3 平均移動速度

図5.4にモバイルロボットの平均移動時間を示す。

横軸は環境内にあるモバイルロボットの台数を示す。縦軸は、モバイルロボットが1回に予約を移動するために要した平均移動時間を示している。縦軸にモバイルロボットが1回に予約する長さの平均移動速度を示す。移動経路の予約距離は、0.5メートル、1.0メートル、1.5メートル、2.0メートルである。また、各プロットには95パーセントの信頼区間を示す。

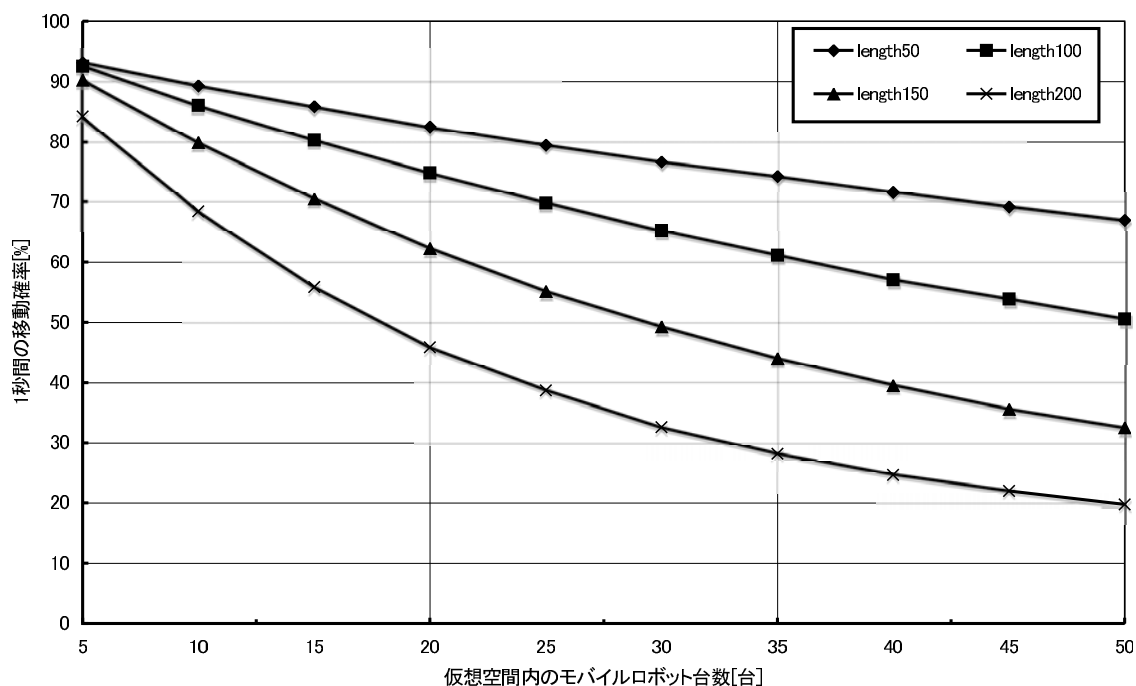


図 5.3: 1 秒間に移動したモバイルロボットの割合

移動経路の予約距離が2.0メートルでモバイルロボットの台数が30台以上の場合には、信頼区間が0.1以上となっているが、他の場所では信頼区間が0.1以下となっている。移動経路の予約距離が1.5メートル以上の場合には、環境内のモバイルロボットの台数が増加するごとに平均の移動時間も増加している事がわかる。また、移動経路の予約距離が1.0メートルの場合にも、モバイルロボットの台数の増加による平均移動時間の増加も若干であるが増加している事がわかる。しかし、移動経路の予約距離が0.5メートルの場合にはモバイルロボットの台数増加による平均移動時間はない。

また、グラフ全体としては、1秒間に移動できた移動台数の場合と全体として逆のグラフになっている事がわかる。しかし、予約距離が2.0メートルの場合には、95パーセントの信頼区間が他の予約距離の場合と比べ大きい事がわかる。以上の事から、予約経路が長くなると、1度の予約距離が長く、予約の解放にも時間がかかる。また、1つの予約に対して複数台のモバイルロボットの予約が待機状態になる事がわかる。

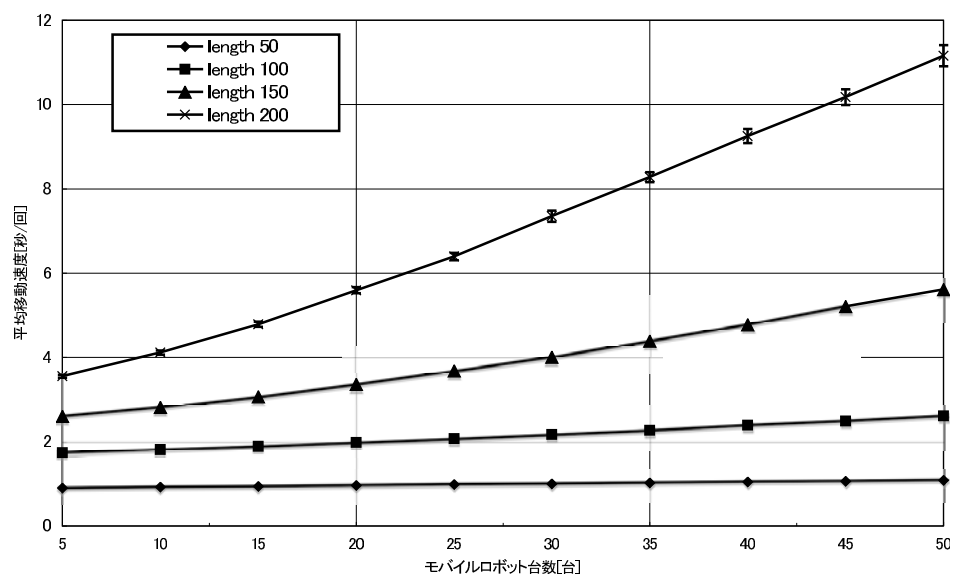


図 5.4: モバイルロボットの平均移動時間

第6章 まとめと今後の予定

本章では、本稿におけるまとめと今後の課題に関してのべる。

6.1 まとめ

本稿では、モバイルロボット間における衝突回避システムの提案をおこった。モバイルロボット間の衝突可否は、衝突の可能性がある双方が移動物体であり、モバイルロボットの利用環境内にあるモバイルロボットのシステムが共通とは限らない事が問題である。衝突回避システムは、任意のアプリケーションから入力でき確実に衝突回避ができる事が重要である。本稿では、衝突回避のために移動経路の予約を行った。移動経路を予約する事により、排他的に移動経路を占有することができる。予約のためには、移動経路の始点と終点と経路の情報を入力するだけで衝突の回避ができる。単独のサーバで衝突回避システムを実行する方法もあるが、サーバがダウンした場合に single point of failure に弱いという問題点がある。我々は、サーバを能動的多重化しサーバのレプリカを全てのモバイルロボット上で実行する事により、サーバの single point of failure に対応した。サーバを能動的多重化する場合には、サーバへ入力するデータの順序が重要となる、全てのサーバへ同じ順序のデータを確実にブロードキャストするために、total-order broadcast を用いた。

また、本稿では提案したシステムをシミュレーションにより性能評価を行った。シミュレーションの結果、予約の経路を長く設定する事により、他のモバイルロボットの移動経路との交差数が増加するため1秒間にいどうできるモバイルロボット台数が減少する。本シミュレーションでは、予約の距離を短くすることで、全体のモバイルロボットが平均的に多く移動できる事がわかった。しかし、本シミュレーションでは、通信のためのオーバーヘッドを静的に設定しているため大きな影響は無かったが、実際にはさらに多くのオーバーヘッドが通信に発生する事も考えられる。通信のオーバーヘッドを考慮する事も今後の課題であると考えられる。

6.2 今後の課題と予定

本節では、本稿で述べた衝突回避システムの今後の課題と博士後期課程において取り組みたい研究内容に関して述べる。

6.2.1 今後の課題

今後の課題としては、次の点が上げられる。

- シミュレーションの改善
- シミュレーションを基に衝突回避システムの実装と公開
- 予約のリレー

本シミュレーションでは、zone はモバイルロボットの移動経路のみであった。実際のモバイルロボットの利用環境には3種類の誤差が考えられる。誤差とは、座標を取得するロケーションシステムの誤差とモバイルロボットが移動する時の移動距離の誤差、モバイルロボットが移動する角度の誤差を示す。zone は、誤差を考慮する事が重要である。

本シミュレーションで述べた zone は移動経路を予約するのみであった。しかし、モバイルロボットの移動経路は曲線である事も考えられるため、zone の経路は曲線にも対応する事が重要である。

本稿で述べて衝突回避システムは、モバイルロボットが移動するごとにモバイルロボットの移動経路の予約をおこなっていた。よって、モバイルロボットは移動経路の予約が受理されると移動を開始し、移動が終了すると次の予約ができる。しかし、モバイルロボットが目的地に到着した後に移動経路を予約する場合、モバイルロボットは各予約ごとに停止する必要がある。モバイルロボットが予約した経路を移動中に次の予約の要求をする事により、モバイルロボットは予約の目的地に達した場合でも停止する事なく次の予約を移動可能になる。移動中に次の予約の要求をする場合、モバイルロボットの移動速度を考慮する事が重要である。モバイルロボットの移動速度が遅すぎる場合には、次の予約が受理された後で他のモバイルロボットが予約を要求しても先に予約したモバイルロボットの予約が受理されることを待ち続ける事になる。また、モバイルロボットの移動速度が速すぎる場合には、モバイルロボットが予約を移動し終えても次に要求した予約がワイヤレスネットワークの遅延などで受理されず停止する必要がある。移動中に次の予約を要求したい場合には、モバイルロボットの移動速度とワイヤレスネットワークの状態などによる通信遅延のトレードオフを考慮する事が重要である。

以上が本衝突回避システムシミュレーションの今後の課題である。

参考文献

- [1] Rami Yared, Fail-safe Mobility Management and Collision Prevention Platform for Cooperative Mobile Robots with Asynchronous Communication, Doctoral Dissertation, JAIST ,Septmber, 2006.
- [2] Xavier Defago, Andre Schiper and Peter Urban, Total Order Broadcast and Multicast Algorithms : Taxonomy and Survey, ACM, Computing Survey, Vol. 36, No.4, December 2004, pp.372-421.
- [3] 藤原 茂喜, 山下 秀樹, 前田 裕史, 岡野 正紀, 特集 生産技術 : 衝突回避機能付き全方向移動型パワーアシストカート, 松下電工技報, Vol. 53, No.2.
- [4] 十河 卓司, 木元 克美, 石黒浩, 石田亨, 分散視覚システムによる移動ロボットの誘導, 日本ロボット学会誌, vol. 17, No 7, pp. 0-7, 1999.
- [5] 宮田 なつき, 群ロボットによる異種作業割り付け型協調搬送システム, 博士論文, 平成11年度.
- [6] 柏村 洋平, 上野 敦志, 辰巳 昭治, マルチロボットシステムにおける自発的な協調方法を用いた動的役割割当, The 20th Annual Conference of the Japanese Society for Artificial Intel ligence, 2006 .
- [7] Sandeep K. Lodha, Shashank Gupta, M. Balakrishnan, Subhashis Banerjee, Real Time Collision Detection and Avoidance: A Case Study For Design Space Exploration in HW-SW Codesign, VLSI Design, 1998. Proceedings., 1998.
- [8] Polesel, R. Rosati, R. Speranzon, A. Ferrari, C. Pagello, E, Using collision avoidance algorithms for designing multi-robotemergent behaviors, Proceedings of the 2000 IEEE / RSJ international Conference 013 intelligent Robot and Systems. Chicago, 1403—1409, 2000.
- [9] Leslie Lamport, Implementation of Reliable Distributed Multiprocess Systems Computer Networks: The International Journal of Distributed Informatique, Vol. 2, No. 2. (May 1978), pp. 95-114.
- [10] DESMO-J, <http://www.desmoj.de/>.