不正確さを許す環境マップと単眼ビジョンによる物体認識に基づく移動ロボットの行動制御の研究

工学研究科

筑波大学

2002年3月

友納 正裕

目 次

第 1 章

はじめに

1.1 本研究の背景

1.1.1 自律移動ロボットが行動する環境

近年，公共施設やオフィス，家庭などで活動するロボットへの期待が高まっている．その用途は，案内・運搬・清掃・介助など人間の作業代行，極限環境や災害復旧における作業支援，さらにイベントやアミューズメントまで多岐に渡る．これまで実用化されたロボットはほとんどが産業用であり，工場内で据え置き型で使われたのに対して，近年注目されているロボットは移動機能をもち，人間と活動環境を共有する点に特徴がある．これが本格的に実用化されれば，ロボットの応用範囲が飛躍的に広がり，21 世紀の重要な社会基盤になると期待される．

技術的な観点から見て，このような移動機能をもつロボットが従来の据え置き型の産業用ロボットに比べて本質的に異なるのは，対象とする世界が「開いている」ことである．工場の生産ラインのような環境は，ロボットが活動しやすいように「閉じた」世界として構成される．すなわち，考えるべき状態の数が限られており，外乱もほとんどなく，不確実性の小さい環境である．このような環境では，ロボットはプログラムや教示データなどで決められた手順にしたがって動作することで，所定の作業を遂行することができる [1]．産業用ロボットでは，このように環境をロボットに合わせることで成功をおさめてきた．ところが，人間と活動環境を共有する移動ロボットでは，環境をロボットに一方的に合わせるわけにはいかない．そこでは，多くの外乱が存在し，予期しない環境変化が起こる．しかも，ロボットが移動可能であるため，対象とする環境は据え置き型ロボットよりもずっと広く，考慮すべき状態数も膨大になる．これが自律型の移動ロボットの実現を困難にする大きな要因の 1 つである [37].

自律移動ロボットの実現には，このような「開いた」世界に対処することが本質的に重要であり，本研究の中心テーマもここにある.「開いた」世界を扱う際には様々な不確実性に

遭遇するが，その根底にあるのは環境情報の不確実性である．ロボットの自律行動は，基本的に，計画→動作→評価の繰り返しであるが，ロボット自身も含めた環境の状態を正しく知ることなしには，確実な計画も動作も実現できない．一方，ロボットがもつセンサや地図などから得られる一次情報には，様々な不確実性が含まれる．このため，このような一次情報から，確実性が高くロボットの行動制御に使える環境情報をいかに抽出するかが重要になる．

ロボットの行動に必要な環境情報は，ロボットが遂行すべき作業 (タスク) に依存する．このため，ロボットのタスクを具体的に想定して，そこで必要となる要素技術を探求するというタスク指向アプローチが重要である．本論文では，単眼カメラをもった車輪型の移動ロボットが屋内を走行して，指定された物体を取って来るというタスクを想定する．このタスクは，目標物体が置かれた場所に移動し，指定された物体を見つけてマニピュレータで把み，元の場所に戻る，という動作からなる．これらの動作はきわめて汎用的であるので，このタスクを実現するための要素技術は他の様々なタスクに応用できる．

これらの動作を実現するために最も基本となる環境情報は以下のものである．

- ロボットの位置姿勢
  ロボットが目的地まで走行したり，指定された場所で作業を行うには，自分が環境内のどこに居て，どこを向いているかを知る必要がある．ロボットが自分の位置姿勢を知ることを自己位置推定と呼ぶ．

- 物体の位置姿勢
  ロボットがタスクを実行するために必要な物体を環境内で見つける必要がある．たとえば，タスクで指定された物体を把むには，物体の 3 次元空間での位置姿勢を知る必要がある．また，ロボットの位置姿勢を知るために特定の物体を目印 (ランドマーク) に用いる場合は，その物体の位置姿勢を知る必要がある．

ロボットは，センサや地図などから得られる一次情報をもとに，これらの環境情報を計算する．ロボットが実環境でロバストに行動できるためには，これらの一次情報に含まれる不確実性に対処しなくてはならない．

なお，本論文では，これ以降，位置姿勢を単に姿勢と呼ぶ (1.4節参照)．

1.1.2 自律移動ロボットの環境認識における課題

自律移動ロボットが環境を認識する際に問題となるのは，認識のもとになる様々な一次情報に不確実性があることである．たとえば，移動ロボットの自己位置推定は，通常，オドメトリと外界センサのデータを融合して地図と照合することで行う．ところが，これらのそれぞれに不確実性が存在するため，ロボット姿勢も不確実になる．

一方，物体の位置姿勢は，地図が正確であれば直接的に求めることができる．そして，ロボット姿勢も正確ならば，ロボットは地図に書かれた物体姿勢に基づいて，物体にどのように近づいたり，どのように把めばよいかを計算することができる．しかし，地図やロボット姿勢が不確実だと，まず物体を見つけることから始めなくてはならない．

このように，環境認識における重要な課題として，一次情報の不確実性への対処とそれに波及して生じる物体認識がある．これらについて，以下に詳しく述べる．

(1) オドメトリの不確実性

移動ロボットの自己位置推定は，多くの場合，オドメトリデータをベースにする．オドメトリはロボットの内界センサの一種で，車輪の回転数からロボットの位置と方位を測定する．オドメトリは積分型のセンサなので，走行につれて誤差が累積され，データの不確実性が増大していく．

オドメトリの累積誤差を補正するためには，外界センサを用いるのが一般的である．すなわち，外界センサで環境内のランドマークを観測し，実環境とマップとでランドマークの位置を比較してロボット姿勢の推定値を修正する．しかし，外界センサにも不確実性がある．

(2) 外界センサの不確実性

移動ロボットの外界センサには，超音波センサ，レーザーレンジセンサ，カメラなど種々のものがあるが，それぞれのデータは誤差をもち，また，そのデータを解釈した結果にも誤りが含まれる．

本研究では，ロボットが自己位置を修正したり把持すべき物体を認識するために外界センサとしてカメラを用いる．カメラで撮影した画像には，レンズ収差による歪みやデジタル化による丸め誤差があり，さらに，画像処理や認識過程で多くの誤差や誤りが入り込む．

(3) マップの不確実性

ロボットがタスクを実行するためには，マップが必要である．マップは環境内に存在する物体の形状や配置を示した地図であり，ロボットの自己位置推定，目標物体の位置の予測，目的地までの経路計画などに用いられる．

マップにも種々の不確実性が存在する．本研究では，とくに，マップに記述された物体の位置姿勢の誤差を考える．これは，言い替えると，マップと実環境とで物体姿勢にずれがあることであり，次の 2 つの場合に生じる．

- 環境の変化
  環境が変化して物体の位置姿勢が変われば，マップと実環境にずれが生じる．前述の

目標タスクにおいても，たとえば，指定された物体の置き場所が変わったり，位置が多少ずれることはよく起こり得る．

- マップ作成の不完全性
  マップの作成には多大なコストがかかるので，完全なマップが得られるとは限らない．この場合も，マップと実環境にずれが生じる可能性がある．また，小さな物体や動きやすい物体はマップに正確に記述できない．

(4) 物体認識と個体同定

前述のように，ロボット姿勢やマップが不確実だと，まず物体を環境から見つけなくてはならない．これには，物体認識と個体同定がある．なお，本研究ではセンサとしてカメラを用いるため，画像から物体を認識する場合を考える．

物体認識では，ロボットが物体のモデルをもち，モデルの特徴と画像から得た特徴を照合して，画像の中でよく合致する部分を目標物体であると判断する方法が一般的である．物体認識には，画像に写っている物体が何かを知る問題と，目標物体が画像中のどこにあるかを知る問題がある．本研究では，ランドマークや操作対象物を見つけることを目的とするので，後者の問題を考える．

環境内に同じ種類の物体が複数個ある場合，上記の物体認識だけでは区別がつかない．たとえば，机やドアなどは同じ種類の物体が数多く存在するが，物体認識で用いる物体モデルだけに基づいて，その中から特定の物体 (個体と呼ぶ) を識別するのは困難である．しかし，移動ロボットが見つけるべきランドマークや操作対象物は，タスクで指定された個体であることがほとんどであり，物体モデル以外の情報も用いて個体同定を行う必要がある．

なお，物体が見つかると，移動ロボットがそれを扱うために，その 3 次元姿勢を推定する必要がある．

1.1.3 従来の研究

移動ロボットの行動制御において，オドメトリと外界センサの不確実性に対処する研究は多いが，マップの不確実性に対処しているものは少ない．また，移動ロボットの行動制御に物体認識を用いている例はほとんどない．以下に，これらの従来研究について概観する．

(1) 移動ロボットの自己位置推定

前述のように，移動ロボットが環境内を走行してタスクを行うには，ロボットの自己位置推定が不可欠である．ロボット姿勢はオドメトリを用いて推定されるが，オドメトリには累積誤差があるため，外界センサでランドマークを観測してロボット姿勢の推定値を修正するのが

一般的である．オドメトリと外界センサの不確実性に対処する方法として，これらのデータを確率的に融合してロボット姿勢を推定する手法が提案されている [6, 7, 18, 24, 45, 70, 80, 90]．そこでは，オドメトリ誤差と外界センサ誤差を確率分布で表現し，走行によってオドメトリ誤差が蓄積されてロボット姿勢の誤差が増える一方で，外界センサでランドマークを観測するとロボット姿勢の誤差が減るという現象をカルマンフィルタやマルコフモデルを用いて確率的に計算し，最も確からしいロボット姿勢を求める．

これらの方法の多くは，マップが正確であるという前提で実現されている．また，マップの不正確さを考慮する場合でも，その誤差は小さいとしてオドメトリ誤差に吸収して扱われることが多い．このような前提では，マップの誤差が大きい場合は，実環境を走行した移動量をマップに反映して得たロボット姿勢と，ランドマーク観測によって得たロボット姿勢がずれてデータ融合に失敗し，有意なロボット姿勢が得られなくなることが起こる．マップ誤差を明示的に導入した自己位置推定の研究もあるが [45]，マップ構成を大きく変更するため汎用性に乏しい．

(2) 移動ロボットによるマップ構築

移動ロボット自身にマップを構築させる研究も盛んに行われている．そこでは，未知環境をロボットが走行してマップを新しく構築したり，環境の変化を検出して既存のマップを修正することが行われる．ここで，本研究の目的はロボットが目標タスクを実行することであるので，未知環境のマップ構築は考えない．なぜなら，目標タスクを効率よく実行させるには，場所や操作対象の指示が必要であり，そのためには何らかのマップが必要だからである．したがって，ここでは，ロボットがマップを持って行動しながら，環境の変化を検出してマップを更新する状況を考える．このような状況では，当然ながら，マップの不確実性を扱うことが本質となる．

ロボット自身がマップを構築するには，ロボットが環境内を走行しながら，その周囲の環境の形状を外界センサで取得し (局所マップと言うこともある)，その時のロボット姿勢に基づいて局所マップを既存の全体マップに融合して，全体マップを更新していく方法が一般的である [12, 49, 50, 62, 72, 81, 92, 93, 95, 96] ここで，ロボットは走行しているので，ロボット姿勢にも不確実性がある．このため，ロボット姿勢の不確実性とマップの不確実性に同時に対処しなくてはならない．その方法として，外界センサのデータの中から環境内の安定した特徴だけを抽出し，その特徴を既存のマップと照合してロボットの自己位置推定を行って，そこで得たロボット姿勢をもとに局所マップを融合していくことが行われる．安定した特徴の抽出や局所マップの融合には，確率的な手法がとられることが多い．なお，センサデータから安定した特徴だけを抽出することは，障害物の検出や，安定した自己位置推定の実現のためにも行なわれている [23]．

こういった方式は，それ自体は有効であるが，得られたマップは，ロボットが通れる領域を示すだけであり，そこに何があるかを示すものではない．このようなマップ更新では，ドアが閉まったり椅子が動いたりして通行できなくなったことを反映することはできるが，位置が変化した目標物体を同定することは難しい．これは，外界センサのデータが低次レベルなため，それを解釈しても環境全体の形状しかわからないためである．多くの移動ロボットは外界センサとして，超音波センサやレーザレンジセンサを用いているが，これらのデータはある平面上でとらえた物体までの距離情報であり，これから特定の物体を認識することはきわめて困難である．外界センサとしてカメラを用いる方式でも，エッジや微分値などの低次レベルの特徴をそのまま使うものが多く，その場合は，やはり目標物体の認識は難しい．

(3) 物体認識と個体同定

コンピュータビジョンの分野では，多くの物体認識手法が提案されている．これには，レーザーレンジセンサやステレオカメラからの距離画像をもとに認識する方法 [4, 21, 71, 75, 77, 91]，物体の画像上の見え方で照合する方法 [56, 63, 66, 73]，幾何学的モデルと照合する方法 [5, 25, 26, 35, 39, 48, 54] など種々のものがある．これらの応用は様々であるが，ロボットへの応用としては，産業用のビンピッキング (ロボットアームが物体を拾いあげる作業) や，前述のようなロボットの自己位置推定が主である．実環境で高速かつロバストに物体の認識と姿勢推定ができ，しかも，移動ロボットに簡便に搭載できるシステムはほとんど実現されていない．

このように物体認識を実環境内で安定して行うのは難しいため，マップやモデルから目標物体の位置を予測することにより，認識の精度と効率を向上することが行われている [13, 19, 46, 29, 88]．これらの方法では，環境を CAD モデルで表し，ロボット姿勢とマップ (あるいは以前の認識仮説) から，たとえばドアや壁のエッジが写る画像上の位置を予測して，その付近の画像特徴を重点的にモデルと照合する．しかし，マップが不正確だとマップによる予測がはずれる可能性があるため，マップはできるだけ正確である必要がある．

また，物体そのものの認識ではないが，画像から得た特徴をマップと照合してロボットの自己位置推定を行う方法も研究されている．この場合，ロボットの周囲に見える顕著なエッジや色，あるいは，画像パターンの主成分などを特徴に用いる [3, 16, 67, 89]．これらの方法では，個々の物体よりも広い環境内の領域を照合用のモデルにして，その領域を認識していることになる．このため，マップが不正確だと，照合すべきモデル自体が歪むことになって，画像から得た特徴とうまく照合しない可能性が高くなる．したがって，これらの方法でも，マップは正確である必要がある．

移動ロボットにおける個体同定のためには，物体の姿勢が有力な情報となる．すなわち，同じ種類の物体でも，置かれた場所によって区別することができる．ロボットの自己位置推

定ができれば，それに伴って周囲の物体姿勢もわかるため，個体同定は自己位置推定の問題にほぼ帰着して考えることができる．実際，自己位置推定の中でとくにグローバル姿勢推定 (Global Localization) では，ロボット姿勢が確定する過程で個々のランドマークが徐々に同定されていく [6, 7, 24, 45, 70, 80]．ただし，マップが不正確な場合は，前述のように自己位置推定ができず，個体同定にも失敗する可能性が高くなる．

1.2 本研究の目的

本研究の目的は，自律移動ロボットが実環境においてロバストに行動するための行動制御システムを提案し，その有効性を検討することである．

自律移動ロボットが実環境でロバストに行動するためには，様々な不確実性に対処して環境を認識する必要がある．その中でも，ランドマークや操作対象物などの特定の物体を認識することは，様々なタスクを実行するために不可欠であるが，実環境においては，物体の位置が正確にはわからない状況で，この種の認識を行わなければならないことが起こる．この問題は，従来の研究では解決されておらず，移動ロボットの実用性を高める上で重要な課題である．本研究は，このような環境認識機能を中核とした行動制御システムの構築を目標とする．このような行動制御システムには以下の条件が必要であると考えられる．

- マップに物体姿勢の不正確さを表せること
- 不正確なマップの上でロボットが自己位置を推定できること
- 環境内の特定の物体の認識とその姿勢推定ができること
- 不正確なマップに基づいて目的地までのナビゲーションができること

本論文は，これらの条件を実現するための技術と，それらを統合して目標タスクを実行する行動制御システムの構成方法を体系的に論じていくものである．

1.3 本論文の構成

第 2 章以降は，次のような構成になっている．

第 2 章では，本研究で対象にするマップ不確実性の問題を明確にした後，目標タスクを定義して，それを実現するための要素技術を説明する．

第 3 章から第 6 章までは，各要素技術の実現方式を説明する．第 3 章では，物体の姿勢が不正確であることを許すマップの表現方法について述べる．第 4 章では，マップの不正確さを反映したロボットの自己位置推定法について述べる．第 5 章では，単眼ビジョンを用い

た物体認識手法を説明する．第 6 章では，これらの技術を統合して，不正確なマップから得た走行経路に基づいてナビゲーションを行う方法を説明する．

第 7 章では，これらの要素技術が互いに連係して相乗効果をもたらし，全体の処理効率や精度が向上する仕組みを説明する．

第 8 章では，これらの技術を実装したシステムを用いて，目標タスクを行わせた実験の結果について述べる．

第 9 章では，全体の考察と残された課題を述べる．第 10 章はまとめである．

また，付録 A では，実装したシステムのアーキテクチャを説明し，付録 B では，本研究の促進のために開発したツールを説明する．

1.4 本論文で用いる用語について

本論文で用いる用語で紛らわしいものについて，ここで整理しておく．

- 自律移動ロボットと移動ロボット
  「自律」とは，外部から誘導されることなく，ロボット自身の判断で動くことを意味する．また，電源，ハードウェア，ソフトウェアなど活動に必要なすべての資源をロボット自身が搭載して，かつ自律的に動けるロボットを「自立移動ロボット」と呼ぶ．本研究で用いるロボットは自立型であるが，本研究の内容自体は自律型ロボットでもあてはまるため，本論文では「自律」という用語を用いる．本論文では，単に「移動ロボット」という場合も，すべて自律移動ロボットを意味する．

- マップ，地図，物体モデル，環境モデル
  「マップ」は環境の形状や物体の配置を記述した地図である．「マップ」と「地図」は同じ意味である．「物体モデル」は，環境内に存在する個々の物体のモデルである．また，マップと物体モデルを合わせたものを「環境モデル」と呼ぶ．

- 実体と物体
  第 3章で述べるが，本論文では，マップを構成する要素を「実体 (Entity)」と呼ぶ．実体は空間領域と物体に分けられる．つまり，物体は実体に含まれる．マップの考察において，物体だけを対象とするときは，実体の代わりに物体と呼ぶことがある．

- 姿勢，位置姿勢，位置
  「姿勢」には，方位だけを意味する場合と，位置と方位をまとめて意味する場合があるが，本論文では，位置と方位を合わせて「姿勢」と呼ぶことにする．「位置姿勢」も，位置と方位の意味で用いる．「位置」は，厳密には方位を含まないが，本論文では，方位も含むけれども位置だけを意識すればよい場合に用いることがある．

移動ロボットは平面上を走るものとし，その姿勢を位置 (x,y) と方位 θ を合わせた (x, y, θ) で表す．マップに書かれた物体も平面上に置かれているとし，その姿勢を (x, y, θ) で表す．

一方，物体認識における姿勢推定では 3 次元空間での姿勢を求めるため，物体姿勢を 3 次元位置 (x, y, z) とオイラー角 (θ, ϕ, ψ) で表す．

- ロボットの自己位置
  「ロボットの姿勢」と同義である．「位置」と言っているが，積極的に方位も含む．慣例に習って，ロボットが自分の姿勢を推定することを「自己位置推定」と呼ぶ．

第 2 章

問題設定と基本戦略

第1章で述べたように，本研究は，実環境においてロバストに行動する自律移動ロボットの行動制御方式を目的とし，とくに，マップに不確実性がある中での環境認識を中心課題とする．ロボットの行動に必要な環境情報は遂行すべきタスクに依存するため，タスクを具体的に想定して，そこで必要となる要素技術を探求するというタスク指向アプローチをとる．

本章では，まず，本研究で対象とするマップの不確実性を明確にし，その条件下で目標タスクを定義する．次に，そのタスクを実行するためのロボット構成を説明する．そして，マップの不確実性によって生じる問題を分析し，その問題に対処するための要素技術を明らかにする．

2.1 マップの不確実性

本研究で対象とするマップの不確実性は，マップに記述された物体姿勢の不正確さである．本節では，マップに不正確さを許すことの利点と，本研究で仮定する不正確さの程度を述べる．

2.1.1 マップの不正確さを許すことの利点

1.1.2節で述べたように，マップの不正確さの要因には，環境の変化とマップ作成の不完全性がある．逆に，マップに不正確さを許容すると，それぞれに対して次のような利点が得られる．

- 環境変化に対するロバスト性の向上
  マップが不正確であることは，マップと実環境とでずれがあることである．したがって，不正確なマップで行動できることは，その不正確さの範囲内で起きた環境変化に対応できることと同等である．

- マップの作成工数の軽減
  マップが不正確でよければ，実環境を正確に計測する必要がなくなり，マップ作成工数が大幅に軽減される．

2.1.2 マップの不正確さの程度

マップが不正確であると言っても，無制限にマップの不正確さを許すのは現実的でない．実際，あまりに不正確な地図は，人間にも理解できないことがある．そこで，不正確さを許すことの利点を保存しながら，以下のような制限を加えることにする．

- トポロジーは正確である
  環境内に存在する物体でロボットが認識すべきものは，あらかじめマップに記述しておくこととする．存在しない物体がマップに記述されることはないものとする．また，通行可能な領域の空間的な接続関係は正しいとする．

- 誤差の程度は領域の広さにほぼ比例する
  実用的に見て，マップが一様に不正確であると仮定する必要はない．広い領域で精度よく位置を計測することは困難だが，狭い領域では目視でもかなりの精度でマップを作ることができる．そこで，局所的には正確であるが，大域的には正確でないマップを考える．ここで,「局所」の範囲は，1つの物体の数メートル程度の近傍内とする．また，マップに記述される個々の物体の形状は正確であるとする．

このような制限のもとでも，マップの作成工数は大幅に軽減される．また，環境変化に対しても，物体の位置がある範囲内で変化するような場合には対処できる．ただし，物体が消えてなくなるような場合は対処できない．

2.2 目標タスク

本研究では，ロボットの目標タスクとして，屋内において指定された物を見つけて取って来る作業を想定する．ユーザがロボットに「場所 XX にある物 YY を取ってきなさい」という指示を与えると，ロボットは，場所 XX まで移動して，その付近を探索して物 YY を見つけ，マニピュレータで物 YY を掴んで，ユーザの所まで戻ってくるというものである．ここで，物 YY はマニピュレータで簡単に掴める程度の大きさの日常生活品や工具などを想定する．また，場所 XX はある程度の広さの部屋で，その中の机やテーブルのような台 ZZ の上に物 YY が置かれている状況を考える．ここで，ロボットに与えるマップは不正確であり，物 YY が置かれている台 ZZ の位置も不正確にしか指示されないものとする．台 ZZ 以外にも，環境内の物体の姿勢は不正確にしかわからないものとする．

このようなタスクは，オフィス，工場，家庭などで頻繁に生じるものであり，配送や片付けなどへの直接的な応用の他，様々な複合作業の基本動作にもなる汎用性の高いタスクである．また，台 ZZ の場所が不正確にしかわからないことは，部屋のレイアウトが変わったり，台 ZZ 自体が位置の変化しやすい物である場合に生じる．環境内の他の物体についても同様である．

このタスクを実現するためには様々な機能が必要であるが，通常の移動ロボットのナビゲーションにはない特徴を以下に示す．

- 目標物体を見つける必要があること
  物 YY の位置も台 ZZ の位置も不正確にしか指定されないため，ロボットが実行時にそれらを見つける必要がある．前述のように，このためには，物体認識だけでなく，個体同定も必要である．

- 物体の姿勢推定をする必要があること
  物 YY を掴むためには，物 YY を認識し，さらに，その 3 次元の姿勢を推定する必要がある．また，物 YY にうまく接近するためには，物 YY だけでなく，台 ZZ の姿勢もきちんと知る必要がある．

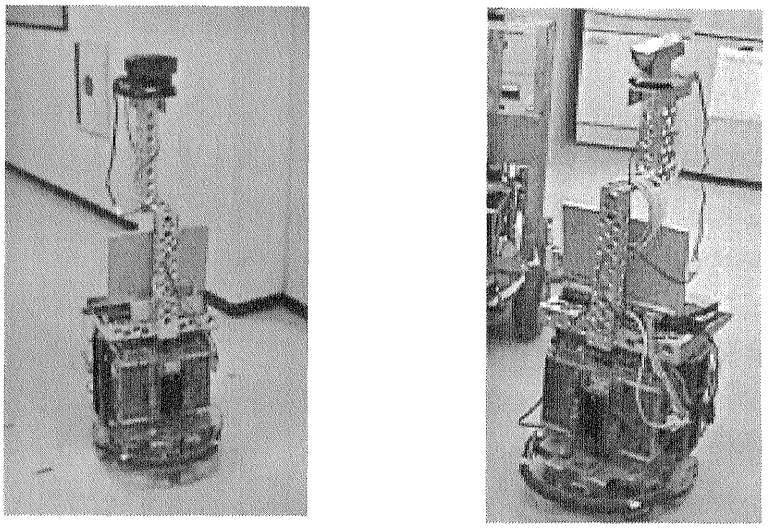
このタスクを完全に実現するには非常に多くの課題があるが，本研究では，環境認識に焦点をあてるために，マニピュレータの実装は考えない．ロボットは，物 YY は掴まずに，単に見つければよいことにする．ただし，将来マニピュレータで掴むことは想定するので，物 YY の姿勢推定は行う．

2.3 ロボットシステム

目標タスクでは，指定された物体を精度よく認識する必要があるため，ロボットがもつ外界センサとしてカメラを用いることにする．移動ロボットによく用いられる超音波センサは，物体の細かい形状や模様まで認識するには向かない．また，レーザーレンジセンサは 3 次元形状データを得るのに時間がかかり，装備も大きくなるという問題がある．ステレオビジョンも装置がやや複雑になる問題がある．このため，本研究では，簡便な外界センサとして単眼カメラを用いることにする．

これらを踏まえて，本研究で用いるロボットは，移動ロボットにパンチルトビデオカメラを載せた構成にする．図 2.1にそのロボットを示す．土台となる移動ロボットとして，筆者らの所属する研究室で開発した移動ロボットプラットフォーム「山彦」[94] を用い，その上にパンチルトカメラを載せる．また，物体認識，カメラ制御，および，全体の行動制御を行うためのノート PC を搭載する．ノート PC は，RS-232C を介して山彦およびカメラを制

御する．このロボットは2世代あり，表2.1に，それぞれの仕様を示す．初期の実験 (第3章～第5章) は1号機，その後の実験 (第6章～第8章) は2号機を用いて行われた．



(a) 1号機　　(b) 2号機

図 2.1: 研究で用いるロボット「山彦ポッキー」

表 2.1: ロボットの構成

世代	ロボット	CPU	メモリ	カメラ
1号機	山彦ポッキー	Pentium II 400MHz	96MB	Sony EVI-D30
2号機	山彦ポッキー	Pentium III 650MHz	128MB	Canon VC-C4

2.4　マップが不正確であることによって生じる問題

マップに不正確さを許すことで，目標タスクを実現する際に，以下のような問題が生じる．これらは，従来の移動ロボットで扱われたオドメトリと外界センサの不確実性によっても生じるが，マップが不正確な場合はより顕著に現われる．

- ロボットの自己位置推定が失敗する可能性がある
  マップが不正確だと，マップと実環境とでロボット姿勢の整合がとれなくなる．このため，実環境を走行した移動量をマップに反映して得たロボット姿勢と，外界センサで得たランドマークからの相対姿勢をマップに反映して得たロボット姿勢とが食い違い，両者の融合に失敗して有意なロボット姿勢が得られなくなる可能性がある．

- 目標物体を見つけにくくなる
  マップが不正確だと，目標物体が画像上に写る位置をマップとロボット姿勢から予測することができなくなり，目標物体の認識が困難になる．また，環境内に同じ種類の物体が複数あった場合に，位置情報をもとに個体同定をすることも難しくなる．

- 走行経路が不正確になる
  不正確なマップに基づいて求めた経路は，実環境と位置がずれているために，通行不能だったり，正しい目的地に到達しない可能性がある．

- ランドマークの撮影に失敗する可能性がある
  不正確なマップに基づいてランドマーク観測を計画すると，実環境ではランドマークの位置がずれていたために，予測した方向にランドマークが見えない可能性がある．

これらはすべて，ロボットが目標タスクを失敗する原因になりうる．これらの問題に対処することが，本研究の中心課題である．

2.5 基本戦略と要素技術

本研究では，マップの不正確さに対処するために，個々の物体を処理単位とする．これは，マップの構成要素を物体とすることと，環境認識の単位を物体とすることの 2 つの意味がある．物体を処理単位にする狙いは，マップの中で正確な部分と不正確な部分を分離することである．前述のように，本研究で対象とするマップの不確実性は，物体姿勢の誤差であり，マップに記述された物体の形状は正確である．したがって，個々の物体単位では正確な認識ができる．次に，用いるマップは局所的には誤差が小さいので，物体の近傍ごとにロボット姿勢の座標系を設定する．そして，この局所領域をつなぎ合わせて，ロボットの走行領域を広げていく．

この方針に基づいて，ロボットは次のようにナビゲーションを行う．ロボットはマップに記述されている物体を認識し，その近くではその物体を基準とした相対座標で自己位置を表す．用いるマップは局所的には誤差が小さいので，その物体の近傍では相対座標に基づいて安全に走行できる．走行するにつれてこの物体から離れると，次の物体を認識して，自己位置の基準をその物体に移す．これを繰り返して，目的地まで走行する．ここで，新しい物体に移る部分はマップ誤差が大きい可能性があるので，その切替え処理が重要な鍵となる．

この基本戦略を具体化するための要素技術を以下に示す．

- 不正確さを許すマップの表現
  前述のように，物体を構成単位としてマップを定義する．物体姿勢を大域的に定義す

ると誤差が大きくなりすぎるので，他の物体からの相対姿勢で定義する．さらに，物体姿勢に誤差を陽に与えることで不正確さを表す．

- 不正確さを許すマップの上でのロボットの自己位置推定
  マップと実環境とでロボット姿勢の整合がとれなくなる問題に対処するため，物体を基準とした相対姿勢でロボット姿勢を表し，ロボットが移動するにつれて基準とする物体を切替える．これにより，局所的にはつねに整合のとれたロボット姿勢が得られる．このロボット姿勢表現を，確率的な自己位置推定法であるマルコフ姿勢推定に組み込むことにより、不正確なマップの上で確率的に自己位置推定を行う．

- 単眼ビジョンによる物体認識
  単眼カメラで撮影した画像のエッジ情報と物体モデルを照合し，目標物体を認識する．その際，エッジの幾何学的制約を利用した種々のフィルタを用いて，処理効率と認識率を高める．また，マップから得られる物体姿勢の制約条件をフィルタとして用いて，さらに処理効率と認識率を高める．

- 不正確さを許すマップの上でのナビゲーション
  不正確なマップから求めた走行経路は実環境と整合がとれない可能性がある．そこで，ロボットは走行しながら環境を認識して経路の修正を行うようにする．このために，ロボット姿勢と同様に，経路も物体を基準とした相対位置で表す．ロボットは，物体からの相対姿勢に基づいて経路を走行する．移動につれて物体から離れると，ロボットは次の物体を認識し，その物体を基準とした自己位置を求める．得られた自己位置が走行経路とずれている場合は，ずれた分だけ走行経路を修正する．

次章以降では，これらの各要素技術について，詳しく論じていく．まず，第3章で不正確さを許すマップの表現方法を述べ，第4章でその上でのロボットの自己位置推定法を提案する．第5章で単眼ビジョンによる物体認識方式を提案し，第6章でナビゲーション方式について説明する．また，第7章では，これらの各機能が相互に連係して，性能を向上させる相乗効果について述べる．第8章では，これらの機能を統合したシステムで，目標タスクを実行する総合実験の結果を述べる．

第 3 章

不正確さを許すマップの表現

移動ロボットの屋内ナビゲーションでは環境マップに基づく方法が広く用いられている．だが，マップに基づく方法には，環境の変化に対処しにくい，精度の高いマップの作成に手間がかかる，などの問題がある．この問題に対処する方法ために，本章で TG マップ (Topological-Geometrical Map) と呼ぶ新しいマップの表現方法を提案する．

TG マップは，3 次元空間中に物体モデルを配置したものであり，トポロジカルマップと幾何マップの両方の特徴をもつ．TG マップでは，物体の姿勢記述は不正確でよく，しかも，絶対座標ではなく物体間の相対座標で定義できるため，マップ作成の工数が軽減される．物体姿勢の不確実性は，ロボットの走行時に物体認識と姿勢推定を行って補完する．このため，環境内の物体姿勢が多少変化していても対処できる．

3.1 従来研究と本研究のアプローチ

3.1.1 従来研究

移動ロボットに目的地を指示するためには，ロボットが何らかのマップをもっていなければならない．屋内ナビゲーションで用いられる主なマップに，幾何マップとトポロジカルマップがある．

幾何マップは，環境中に存在する物体の形状や姿勢を 2 次元平面 (通常は床を表す) にグリッドや線分で定量的に表現したものである．たとえば，平面をグリッドでセルに区分して物体が置かれている領域をセルの占有度で表現したり [20, 79, 93]，物体の境界を線分の連結で表現したりする [2, 12, 18, 96]．また，環境をモデル化した 3 次元空間内に物体モデルを配置して，3 次元マップを構成するものもある [29, 88]．これらのマップでは，位置情報をあらかじめ完全に与えておき，ロボットはその位置情報に基づいて目的地まで走行する．その際，ロボットはランドマークを用いて自己位置を修正し，つねにマップ上の自己位置と実環境内での自己位置の整合をとる．これらのマップには正確な位置情報が必要なため，人手で

作成するにしても，ロボットに自動作成させるにしても，マップの作成にはかなりの手間がかかる．しかも，マップの更新に手間がかかるため，位置が変化しやすい物体をマップに載せるのは難しい．しかし，幾何マップがあれば，自己姿勢の推定さえすれば，大きな環境認識の負荷なしに安全なナビゲーションが実現できる．また同時に，幾何マップはロボットの自己姿勢の推定にも大きな役割を果たす．

これに対して，トポロジカルマップは，ロボットが存在しうる場所を表すノードとその間の関係を表すアークからなるグラフ表現によって，定性的に環境を表す．アークにはロボットがなすべき動作が記述されており，ロボットはそれを逐次実行することでナビゲーションを行う [47, 59]．トポロジカルマップは，基本的には，物体の定量的な姿勢や形状の情報を持たないため，ロボットは時々刻々環境を認識して，これに基づいて進むべき方向を決めつつナビゲーションを行う．トポロジカルマップには物体の詳細な形状が定義されないので作成は容易であるが，ナビゲーション実行時にセンサおよび環境認識に大きな負荷がかかる．なお，マップに位置情報を全く持たないと非効率な経路を走行することが多いため，トポロジー情報と定量情報を統合したナビゲーションの研究がなされている [41]．また，マップ生成時の累積誤差やマップの不完全さを許容するために，ローカルマップをトポロジカルに結合して全体マップを構成したり [14, 74, 79]，物体の配置順序を保存した「定性座標」で記述された廊下のマップを用いてナビゲーションを行う研究もある [34]．

一方，ロボットによるマップ構築の分野では，マップの不正確さを扱うことが本質的である．ロボットによってマップを構築するには，ロボットが未知環境を走行しながらセンサで環境の局所形状 (ランドマーク) を検出し，マップ上に配置する方法がとられる [8, 12, 14, 30, 50, 81, 92, 93, 96]．ここで，ランドマークはロボット座標系で検出されるため，その位置をマップ座標系に変換したうえでマップ上に配置する．このとき，ランドマークの位置にロボット姿勢の誤差が入り込み，マップが不正確になる．そこで，ランドマークの位置を確率的に表し，ロボット姿勢の誤差とランドマーク位置の誤差が全体的に最小となるように計算する．また，広い領域に対処するために，局所的なマップをつなぎ合わせて全体マップを構築する方法も提案されている [9, 14, 74, 83]．

3.1.2 本研究のアプローチ

本章で述べる TG マップは，トポロジー情報と幾何情報を統合し，両者の長所を取り入れるアプローチをとる．すなわち，トポロジカルマップと同程度の容易さで作成できて，しかも，幾何マップと同程度の幾何学的な詳細さをもつことを目指す．また，マップの誤差を扱うため，マップ構築でよく行われるように，位置情報を確率で表現する．これらを実現するために，以下のアプローチをとる．

- マップの構成単位を物体とする
  従来の多くのマップでは，グリッドセルや線分などの抽象度の低い構成要素で，環境の形状を表現していた．これらは，従来の多くの移動ロボットのセンサデータ (超音波センサやレーザレンジセンサで得た距離情報やエッジ情報) と対応がとりやすいため，マップの構成要素として都合がよかった．

  一方，TG マップでは，机やドアなどの抽象度の高い物体をマップの構成要素とする．すると，物体をあらかじめ部品として定義しておけば，マップ構築は部品の配置だけですむので容易に行える．しかも，部品には幾何学的形状が定義されているので，マップは定量的な幾何情報をもつ．

- 物体の姿勢を相対的に与える
  物体をマップ上に配置する際には，絶対座標でなく，他の物体からの相対姿勢を与える．これにより，局所的な測量をもとにマップを作成できるため，マップ構築が容易になる．

- 物体間の姿勢を確率的に表す
  マップの不正確さを陽に表すために，物体間の相対姿勢を確率的に定義する．相対姿勢の正確さは，目視や歩数による測量程度の精度でよいとする．これにより，厳密な測量をしなくてもマップが作成できるので，マップ構築が容易になる．また，環境内の物体姿勢が変化しても，マップ誤差の範囲内であれば対応することができる．

TG マップは人手で作成することを前提とする．これには，2 つの意味がある．1 つは，上記のように，TG マップの作成が容易であるため，人手による作成が充分現実的であることである．もう 1 つは，後述する物体認識では，あらかじめ目標物体がわかっている必要があるため，全くの未知環境を前提にできないことである．このため，不正確でもよいから環境に関する事前知識を必要とする．

3.2 TG マップの概要

TG マップは以下のとおり，実環境に存在する幾何学的実体 (以下，単に実体) の空間的な包含関係と姿勢関係を記述したものである [85, 87].

- 実体
  実体は，建物や廊下，部屋，机，ドアなど，実環境に存在する物体または空間領域である．実体は所定のクラスに属する．同じクラスの実体は，形状などの特徴は全く同じである．実体はローカル座標系をもち，その座標系で形状が定義される．実体の形

状定義は正確であるとする．実体には，一意な識別子が付与される．なお，本来，実体は3次元空間で定義されるが，本論文では簡単のため，2次元平面 (床面) への投影像で考える．

- 包含関係
  包含関係は，実体間の空間的な包含関係を表す．たとえば，建物は部屋や廊下を包含し，部屋は机やドアなどを包含する．包含関係は，空間構造を階層化し，人間によるマップ作成および理解やロボットによる姿勢推定および経路計画を効率化するのに有用である．ただし，本論文では包含関係は扱わない．

- 姿勢関係
  姿勢関係は，実体間の相対姿勢を表す．実体の姿勢はその実体のローカル座標系の姿勢であり，実体間の相対姿勢は，2つの実体のローカル座標系間の相対姿勢で定義される．姿勢関係は不正確でよい．たとえば，目視や歩数による測量程度の精度を許す．

TGマップは，実体をノードとし，包含関係および姿勢関係をアークとするグラフで表現する．筑波大学校舎の一部のフロアをTGマップで表現した例を図3.1に示す．図3.1(a) はグラフ表現であり，長方形は物体，角なし長方形は領域，アークは姿勢関係を表す．ただし，見やすさのため，包含関係は省略した．図3.1(b) はグラフ表現をもとに，システムによって作成された平面図である．図3.1(a) では省略しているが，壁もTGマップの記述に含むことができる．図3.1(b) には8枚の壁が描かれている．壁は占有範囲が広く，壁の長さ自体が不正確になる可能性が高いため，他の実体とはやや違う扱いをする必要がある．そのため，本稿では，ロボットの通行可能領域を規定する境界 (物理的には障害物) として壁を扱うにとどめる．

TGマップの作成は，実体を部品として表現しておき，その実体を床面に置いて実体間の相対姿勢を定義することで行う．あらかじめ実体を部品として用意しておけば，TGマップはトポロジカルマップと同程度の容易さで作成できる．しかも，実体が形状情報をもつため，幾何マップと同程度の幾何学的な詳細さをもつ．なお，従来のトポロジカルマップの多くは，ロボットが存在しうる点をノードとして，その点をアークでつないだものが多く，経路地図に近いものである．TGマップは，実体をノードとして，アークで実体間の位相関係を表す点に特徴がある．

TGマップでは，実体間の相対姿勢をもとに実体のマップ座標系での姿勢を決める．姿勢関係が直接定義されていない実体間の相対姿勢は，その間にある実体の相対姿勢を累算して求める．姿勢関係を累算すると誤差も累積されるため，高精度が要求されるマップでは，離れた位置にある実体間の相対姿勢を扱うときに矛盾が生じないよう注意が必要であるが，TGマップではあらかじめ誤差を許容しており矛盾は発生しない．

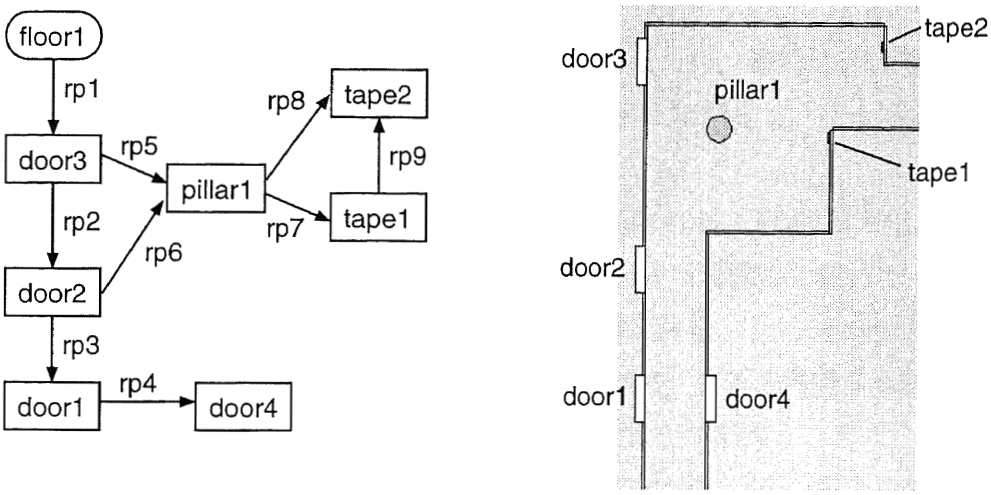


図 3.1: TG マップの例

3.3 TG マップの構成

ここでは，前節で紹介した TG マップの構成をより詳しく説明する．

3.3.1 実体

前述のように，実体は，建物や廊下，部屋，机，ドアなど，実環境に存在する物体または空間領域を指す．TG マップでは，個々の実体を 3 次元モデル化した実体モデルを 3 次元空間中に配置することで環境を表現する．ただし，本論文では 2 次元に限定する．なお，実体は実環境中の実在物，実体モデルは計算機内の対象であり別のものであるが，以下では，とくに断りがない限り，実体モデルを単に実体と呼ぶ．

実体は，型，識別子，形状，姿勢，一般属性をもつ．型は，実体を分類するために，類似した形状や属性をもつ実体を 1 つの集合にまとめたものである．実装上は，オブジェクト指向言語のクラスに対応させる．個々の実体は，クラスから生成されるインスタンスである．識別子は，個々の実体に付与する一意名である．形状は，実体のローカル座標系で定義され，たとえば，点，線，面からなる多面体で記述する．実体の姿勢は，ある座標系を基準にした実体のローカル座標系の姿勢として定義する．一般属性は，色や他の特徴など，ユーザが設定したい任意の属性である．形状，色，その他の特徴は，ロボットがその実体を認識するのに使われる．

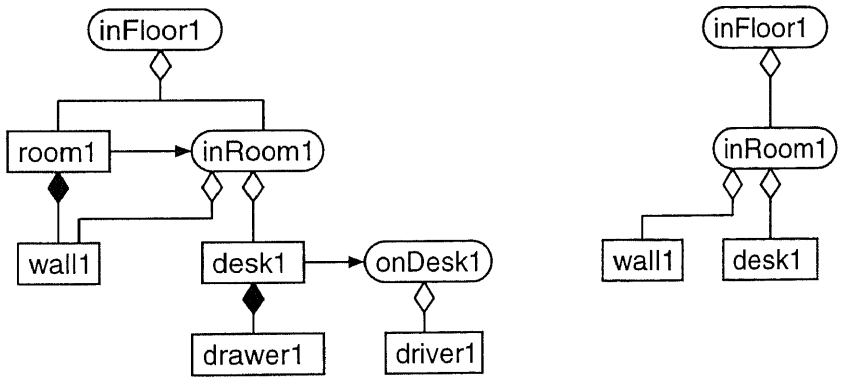
3.3.2 包含関係

包含関係は，実体間の空間的な包含関係を表す．たとえば，建物は部屋や廊下を包含し，部屋は机やドアなどを包含する．包含関係は，空間構造を階層化し，マップの作成および理解や経路計画を効率化するのに用いられる．

包含関係により，実体は領域と物体の 2 種類に分類される．領域は空間に対応する実体で，他の実体を包含できる．物体は形状すなわち占有空間を意識した実体で，他の実体を包含できない[1]．たとえば，部屋は領域であり，机は物体である．

領域か物体かは視点により異なる．たとえば，建物は，屋外を移動する場合はその外形を物体としてとらえた方がよいが，建物内を移動する場合は領域としてとらえた方がよい．また，机も移動ロボットにとっては物体としてとらえた方がよいが，机上の物体を操作するマニピュレータにとっては領域としてとらえた方がよいこともある．

このような現象を正確に表すためには，領域か物体かを明示する必要がある．直観的には，物体としての建物は「建物」，領域としての建物は「建物の中」，物体としての机は「机」，領域としての机は「机の上」や「机の右」のように表す．そして，物体「机」と領域「机の上」を適当な姿勢関係で関連づければ，空間を正確に表現したことになる．ただし，表現の正確さとマップ作成コストは一般に反比例するため，上記のように領域と物体を厳密に意識するとマップ作成が煩雑になる．そのため，マップの利用目的に応じた正確さで表現することが重要である．たとえば，屋内ナビゲーションでは，通常は,「建物」や「机の上」を無視して,「建物の中」と「机」だけを意識して表現したマップで十分である (図 3.2参照)．



(a) 完全なTGマップ　　(b) 簡易TGマップ

図 3.2: 物体と領域：白い菱形が包含関係，矢印は姿勢関係

[1]物体は包含関係とは別に部品階層をもつ．図 3.2の黒い菱形が部品関係を表す．ただし，本論文では扱わない．

3.3.3 姿勢関係

TG マップ上の姿勢関係は，実体間の相対的な姿勢関係を表す．環境の形状を正確に扱えるように，姿勢関係を 3 次元空間で 6 自由度のものとするのが適切であるが，移動ロボットのナビゲーションや姿勢推定を考える場合は，床面上の姿勢 $(x, y, \theta)^T$ に限定することができる．

いま，3 つの実体 e_0, e_1, e_2 が存在するとし，各々のローカル座標系を用いて，これらの姿勢関係を表す．e_i, e_j 間の相対姿勢を $q_{ij} = (x_{ij}, y_{ij}, \theta_{ij})^T$ と表すとすると，以下の式が成り立つ (図 3.3参照)．

$$q_{02} = M(\theta_{01})q_{12} + q_{01} \tag{3.1}$$

$$M(\theta_{01}) = \begin{pmatrix} cos\theta_{01} & -sin\theta_{01} & 0 \\ sin\theta_{01} & cos\theta_{01} & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.2}$$

一般に，姿勢関係を次々にたどって式（3.1）を適用することで，e_i から姿勢関係アークで到達可能な任意の他の実体 e_j の相対姿勢を求めることができる．また，たとえば e_0 を基準に式（3.1）を適用して，各実体の e_0 座標系での姿勢を求めることで，図 3.1(a) のようなグラフ表現から図 3.1(b) のような平面図を得ることができる．

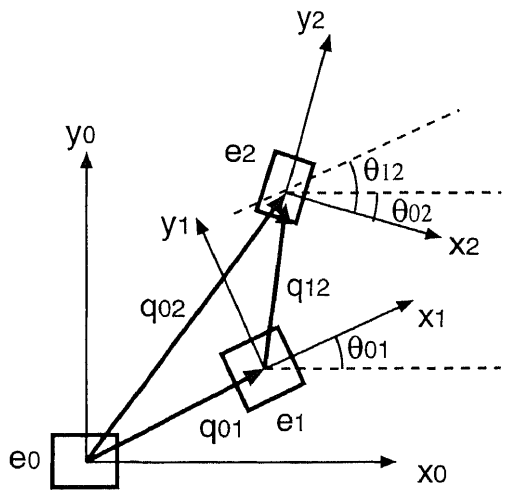


図 3.3: 実体間の姿勢関係

3.3.4 不正確さの表現

TG マップでは，実体間の相対姿勢は不正確でよいが，その不正確さの程度はマップに記述しておく．いま，実体 e_1 から見た実体 e_2 の相対姿勢 q_{12} を期待値 $\bar{q}_{12}$，共分散 Σ_{12} の正規

分布 $N(\bar{q}_{12}, \Sigma_{12})$ に従う確率変数と考え，この Σ_{12} を用いて不正確さの程度を表現する．ただし，マップの作成や理解をしやすくするために，不正確さの程度を何段階かに分類して定性的に表現してもよい．この場合，計算時には，各段階に適合する共分散を割り当てて再解釈する．

TG マップでは，すべての実体間に姿勢関係アークを定義する必要はない．実体 e_i と e_j の間に姿勢関係アークが直接定義されていない場合は，他の実体 e_k を経由して e_i から e_j への相対姿勢 q_{ij} を計算する．実体 e_i から e_j に到達する姿勢関係アークの経路が複数ある場合は，そのうち，q_{ij} の共分散が最小になる経路を採用する．これは，e_i から e_j まで多くの姿勢関係アークを経由して誤差が過剰に累積されるのを防ぐためである．各姿勢関係アークで適切な共分散を定義しておけば，これらの姿勢関係アークから合成される q_{ij} の共分散のうちの最小のもので q_{ij} の誤差を覆うことができるので，共分散が最小の経路が最も望ましい．ただし，e_i から e_j までの間にある各姿勢関係アークの共分散をもとに q_{ij} の共分散を解析的に計算するのは難しいので，各姿勢関係アークの共分散の大きさの和が最小になる経路で近似する．具体的には，アークのコストを $\det \Sigma_{ij}$ で定義し，ダイクストラ法を用いて，e_i から e_j への最小コスト経路を求めればよい．このような実体姿勢の不正確さの程度は，ロボットがランドマークを選択したり経路計画を行う際の評価基準の 1 つになる．

3.4 TG マップの作成

TG マップは概念的にはグラフ表現が基本となるが，実際に TG マップを作成する際に，グラフ表現をそのまま用いるのは効率が悪い．TG マップの作成において本質的に重要なのは相対姿勢アークの設定であるが，2 つの実体間の相対姿勢を数値で直接与えるのはかなり煩雑である．そこで，TG マップ作成に適したエディタを開発し，実体間の相対姿勢を容易に設定できるようにした．なお，実装言語としては Java を用いることを前提とする．TG マップエディタについては，付録 B.1を参照のこと．

TG マップは，あらかじめ定義しておいた実体をマップ上に配置していくことで作成する．以下にその手順を示す．図 3.4にその様子を示す．

- 実体クラスの定義

  実体クラスは実体のひな型であり，マップにはそのインスタンスが配置される．実体クラスは，Java 言語のクラスとしてプログラムで定義する．なお，実体クラスには，後述する物体認識に用いる 3 次元モデルがリンクされている．

  表 3.1に，現在用意されている実体クラスを示す．この表で，形状はマップに投影したときの 2 次元形状である．屋内の多くの実体の 2 次元形状は長方形で表現でき，そのサイズや色が異なるだけなので，プログラムでも容易に定義することができる．

- 実体の配置

  実体クラスのインスタンスを生成して，マウスを用いてマップ上に配置する．配置すると，実体の姿勢 (x, y, θ) は自動的に設定される．また，必要に応じて実体の大きさを変更する (現在，Wall の長さだけが変更可能)．

- 相対姿勢の設定

  2 つの実体間の相対姿勢を設定する．マウスで 2 つの実体を指定すると，その間の相対姿勢 (x, y, θ) が自動的に計算される．相対姿勢の誤差はユーザが入力する．

- ファイルへの保存

  作成した TG マップをファイルに保存する．

実体の配置，相対姿勢の設定，ファイルへの保存は，TG マップエディタで行うことができる．

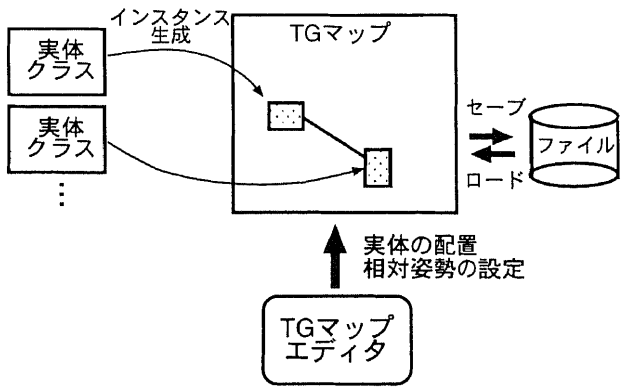


図 3.4: TG マップの作成手順

TG マップエディタで作成した TG マップの例を図 3.5に示す．この例は実験室を表しており，たくさん並んでいる長方形は机である．この机の列は，コピー/ペーストを使えば簡単に作成することができる．

表 3.1: 実体クラス

クラス名	実体	形状	サイズ	認識用モデル
DeskW	実験室の机 (白)	長方形	固定	あり
DeskG	実験室の机 (灰色)	長方形	固定	あり
DeskS	講義室の机	長方形	固定	あり
Cabinet	実験室のキャビネット	長方形	固定	あり
Refrig	実験室の冷蔵庫	長方形	固定	あり
Teabox	紅茶箱	長方形	固定	あり
DoorL1	校舎 L 棟のドア	長方形	固定	あり
DoorM1	校舎 M 棟のドア	長方形	固定	あり
DoorM2	校舎 M 棟のドア	長方形	固定	あり
DoorM3	校舎 M 棟のドア	長方形	固定	あり
Pillar	廊下の円柱	円	固定	なし
Wall	壁	長方形	長さ可変	なし

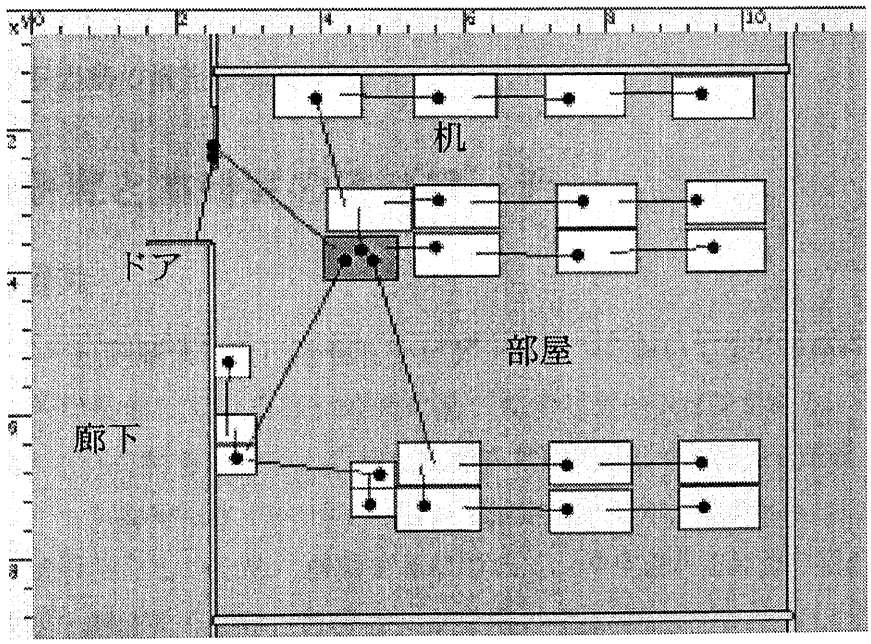


図 3.5: 作成された TG マップの例

第 4 章

不正確さを許すマップにおける自己位置推定

本章では，TG マップにおける移動ロボットの自己位置推定方式を提案する．TG マップを用いた移動ロボットの自己位置推定では，実体間の相対姿勢が不正確なためにマップが歪み，ロボット姿勢をグローバルな座標系で表すと一貫性がとれなくなるという問題が生ずる．この問題に対処するために，ロボット姿勢を局所領域ごとに異なる座標系で表し，ロボットが移動するにしたがって，基準とする局所領域を変えるというアプローチをとる．そして，実体間の相対姿勢の誤差分布を考慮するようにマルコフ姿勢推定を拡張することで，TG マップ上でのロボット姿勢の推定を実現する．

4.1 従来研究と本研究のアプローチ

4.1.1 従来研究

移動ロボットに目的地まで走行させるナビゲーションにおいて広く用いられている方法として，ロボットが推定している自己位置に基づくナビゲーションがある．この方法では，ロボットは環境マップを持ち，その上の自己位置を常に把握しながら，目的地まで到達する．多くの場合，ロボット姿勢はオドメトリを用いて推定されるが，オドメトリのような積分型の姿勢推定は累積誤差をもつため，移動するにつれて，推定される姿勢の誤差が大きくなっていく．この問題を解決するには，外界センサを用いて環境中のランドマークを観測し，実環境とマップとでランドマークの位置を比較することにより，ロボット姿勢の推定値を修正するのが一般的である．ここで用いるランドマークの種類はロボットが有する外界センサによって決定される．超音波やレーザレンジセンサを用いる場合は，壁面や壁の稜線が有力なランドマークとなる [2, 6, 18, 20, 80]．カメラを用いる場合は，環境中の顕著なエッジや色領域などの低次レベルの特徴から，ドアや柱などの高次レベルの物体まで，様々なものがランドマークとして用いられる [3, 13, 29, 34, 42, 67, 84, 88, 89]．

移動ロボットの姿勢推定においては，外界センサ，オドメトリ，マップなど，ロボットがもつ情報のどれもが不確実性をもつため，それらを融合して推定される姿勢については確率的に取り扱うのが有効である．移動ロボットの確率的姿勢推定方式は，大きく Position Tracking と Global Localization の 2 つに分けられる．前者では，一般にロボットの姿勢を正規分布で表し，初期姿勢を与えたうえで，オドメトリ値とセンサデータを用いた最尤推定により姿勢推定を行う [18, 90]．後者では，ロボット姿勢の確率分布の形を仮定せず，マルコフモデルとベイズ定理を用いて各時点での分布を推定する [6, 7, 45, 80]．Global Localization では，ロボットは初期姿勢を与えられなくても，移動するにつれて自己位置を確定していく．そのため，走行開始時だけでなく，走行中にスリップや環境の誤認識などによって自己位置を見失った場合でも，同じ枠組で自動復帰できるという特長がある．しかし，従来の多くの方法では，マップの誤差はオドメトリ誤差に吸収して扱われており，マップの不確実性に明示的に対処しているものは少ない．Koenig らは，トポロジカルマップに対してマップ誤差を明示的に導入した Global Localization を行っているが [45]，トポロジカルマップのノード構成を変更するため汎用性に乏しい．

4.1.2 本研究のアプローチ

TG マップを用いた自己位置推定では，実体間の相対姿勢が不正確なためにマップが歪み，ロボット姿勢をグローバルな座標系で表すと，マップと実環境とでロボット姿勢の整合がとれなくなるという問題が生じる．このため，実環境を走行した移動量をマップに反映して得たロボット姿勢と，外界センサで得たランドマークからの相対姿勢をマップに反映して得たロボット姿勢とが食い違い，両者の融合に失敗して有意なロボット姿勢が得られなくなることがある．この問題に対処するために，ロボット姿勢を局所領域ごとに異なる座標系で表し，ロボットが移動するにしたがって，基準とする局所領域を変えるというアプローチをとる．

本論文で提案する姿勢推定法は，Global Localization の実現法の 1 つであるマルコフ姿勢推定 [6, 7, 45, 80] に基づく．ただし，異なる局所領域間を移動したときに対処できるよう，ロボットが局所領域を変えるときに，TG マップに定義された局所領域間の相対姿勢を用いて，ロボット姿勢の座標変換を行う．TG マップには局所領域間の相対姿勢の誤差が指定されているので，この誤差を加えて座標変換を行うと，推定されるロボット姿勢の分布は新しい局所領域内で拡散される．この拡散により，マップが歪んでいても，センサデータと整合がとれるロボット姿勢を残すことが可能になる．

マップが不正確な場合，センサで観測されたランドマークがマップ上のどのランドマークに対応するかを一意に決められないことが多くなり，ランドマーク同定の問題が生じる．そのため，本方式では，Global Localization の枠組を採用し，ランドマーク観測によって得

られる推定姿勢を仮説として扱う．一般に，センサによってランドマークを観測すると，環境中にある同種のランドマークの個数だけの仮説が作られる．この仮説とオドメトリによって求められた推定姿勢からランドマークが同定され，より正しい姿勢が推定される．

なお，センサによる姿勢推定 (仮説の生成) は，センサデータと形状が一致する場所をマップから探し，その場所からのロボットの相対姿勢を求めることによって行う．従来の研究では，センサとして超音波やレーザレンジセンサを用いることが多かったが，粒度の小さい特徴を使って仮説生成を行うと，環境中に一致する箇所が多すぎて，仮説数が膨大になる．本方式では，ビジョンによって個々の物体を認識し，その結果をセンサデータとして用いるので，仮説数が少なくて済む．

4.2 確率的姿勢推定法

ナビゲーションにおいてロボットは環境中に存在する実体に対する相対的な自己姿勢を推定して走行する．このため，ロボット姿勢を近くにある実体とそれに対する相対姿勢の対で表現することとし，ロボットは常時これを確率的に推定することにする．基準となる実体として，ロボットが最近に外界センサでランドマークとして認識した実体を選ぶ．ただし，ロボットのセンサは実体のクラスしか認識できないので，環境中のそのクラスの実体の個数だけの仮説が成り立つことになる．そこで，各々の仮説の尤度を確率によって表現し，過去から現在までのセンサデータとオドメトリデータをマルコフ姿勢推定により融合して，ロボット姿勢の確率分布を推定する．

4.2.1 ロボット姿勢の表現

TG マップでは，実体の姿勢が不正確なため，ロボット姿勢をグローバル座標系で表現すると一貫性がとれなくなる．一方，ナビゲーションにおいては，ロボットの正確な姿勢は周辺の実体に対して相対的にわかればよい．そこで，ロボット姿勢を局所領域ごとに違う座標系で表し，ロボットが移動するにしたがって，基準とする局所領域を変えることにする．ここで，局所領域をロボットが最近認識した実体の近傍とし，その局所領域では，ロボット姿勢をその実体のローカル座標系で表す．

ロボットが実体 e のローカル座標系で姿勢 $r = (x, y, \theta)^T$ にあることを r_e と表す．この実体を**基準実体**と呼ぶ．ロボット姿勢は，基準実体とそこでの姿勢の対 $s =< e, r_e >$ で表す．一般に，この姿勢は e, r_e とも確率変数として扱う必要がある．なお，時点 t におけるロボット姿勢を $s^{(t)}$ と書く．

4.2.2 基準実体の切替え

ロボットが移動して基準実体を変えたとき，それに伴ってロボット姿勢も変える必要がある．これは，TG マップに記述されている実体間の相対姿勢を用いて，ロボット姿勢を前の基準実体の座標系から新しい基準実体の座標系に座標変換することで行う．いま，実体 e_1 の座標系で表したロボットの相対姿勢 $r_{e_1} = (x_1, y_1, \theta_1)^T$ と，実体 e_2 の座標系で表したロボットの相対姿勢 $r_{e_2} = (x_2, y_2, \theta_2)^T$ との間には，次の関係がなりたつ (図 4.1参照)．

$$r_{e_1} = M(\theta_{12}) r_{e_2} + q_{12} \tag{4.1}$$

$$r_{e_2} = M^{-1}(\theta_{12})(r_{e_1} - q_{12}) \tag{4.2}$$

r_{e_1}, r_{e_2}，および，実体間の相対姿勢 $q_{12} = (x_{12}, y_{12}, \theta_{12})$ はいずれも確率変数である．

ロボットが姿勢 $s_1 =< e_1, r_{e_1} >$ にあるときに基準実体を e_1 から e_2 に変えた場合に，ロボットが e_2 の座標系で相対姿勢 r_{e_2} にある確率 $P(r_{e_2}|s_1, e_2)$ は，q_{12} を確率変数として式 (4.2) から計算できる．すなわち，与えられた r_{e_1} と各 q_{12} に対して式 (4.2) から r_{e_2} を計算し，得られた r_{e_2} の値に対して，そのときの q_{12} の確率を付与すればよい．ここで，q_{12} は r_{e_1} に依存せず，その確率分布 $P(q_{12})$ は，3.3.4節で述べたように，TG マップに記述されている．

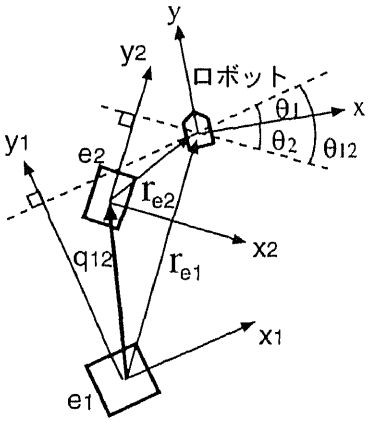


図 4.1: 基準実体の切替え

この座標変換によってロボットの推定姿勢の確率分布は拡散する．図 4.2にその例を示す．同図 (b) のマップでは，真のマップ (a) に比べて，実体 e_1 と実体 e_2 間の相対姿勢が不正確であり，y 方向の分散が大きいと定義されているとする．e_1 の座標系でのロボット姿勢 r_{e_1} は，e_2 の座標系に変換すると r_{e_2}(図の楕円) のように拡散する．

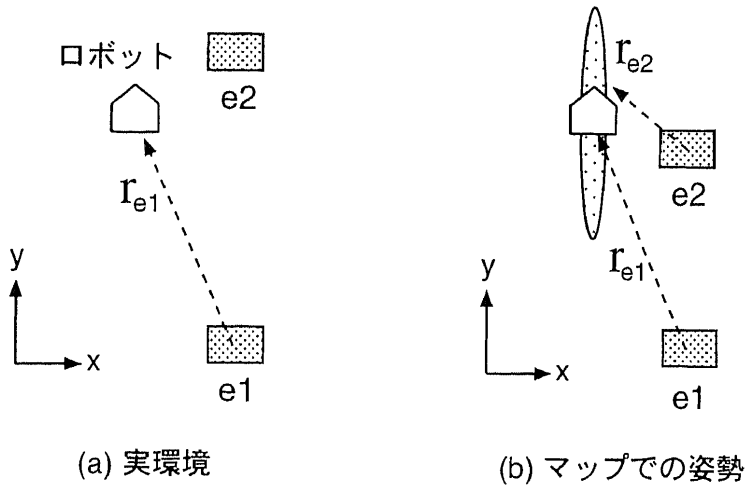


図 4.2: ロボット姿勢の拡散

4.2.3 環境の認識

ロボットは環境中の認識すべき実体のモデルをもち，ビジョンを用いて認識した実体のロボット座標系での姿勢をセンサデータとして得るとする．ただし，センサでは実体の種類 (クラス) を認識できるだけで，同じクラスに属する個体の識別はできない．すなわち，センサは認識した実体を TG マップ中の実体と同定することができない．このような同定されていない実体を**ラベルなし実体**と呼ぶ．ラベルなし実体 u がロボット座標系で姿勢 $q = (x, y, \theta)^T$ にあることを $z =< u, q >$ と表す．ロボットは，この z をセンサデータとして得る．時点 t のセンサデータを $z^{(t)}$ と書く．また，u のクラスに属する TG マップ中の実体の集合を $Ins(u)$ と表す．ロボットはセンサデータ $z =< u, q >$ を得ると，$Ins(u)$ に属する各実体 e_i がロボット座標系で相対姿勢 q にあるという仮説を生成する．この仮説を $h_i =< e_i, q >$ と書く．これらの仮説がもつ姿勢 q は，$Ins(u)$ に属する各実体 e_i に対してすべて同じである．時点 t の仮説を $h_i^{(t)}$ と書く．

この仮説を逆に考えると，各 $e_i \in Ins(u)$ の座標系でロボットが姿勢 q' にあるという仮説と見なすことができる．ここで，q' は q から逆算される値である．図 4.3に例を示す．ロボットが実環境で e_2 だけを認識したとする．マップ内には同じクラスの実体が 3 個 (e_2, e_3, e_4) 存在するため，それぞれに対してロボット姿勢の仮説が得られる．

このように，同じクラスに属する実体の識別ができないため，センサデータだけからではロボット姿勢の仮説が複数得られてしまう．そこで，過去から現在までのセンサデータとオドメトリデータをマルコフ姿勢推定により融合し，もっともらしい仮説を選択する．一般に，ロボット姿勢 $< e, r_e >$ の推定では，基準実体 e の同定とその座標系での姿勢 r_e の推定の両方が必要である．マップが不正確な場合は，実体を一意に識別できないことがあり，と

くに基準実体 e の同定の重要性が高くなる.

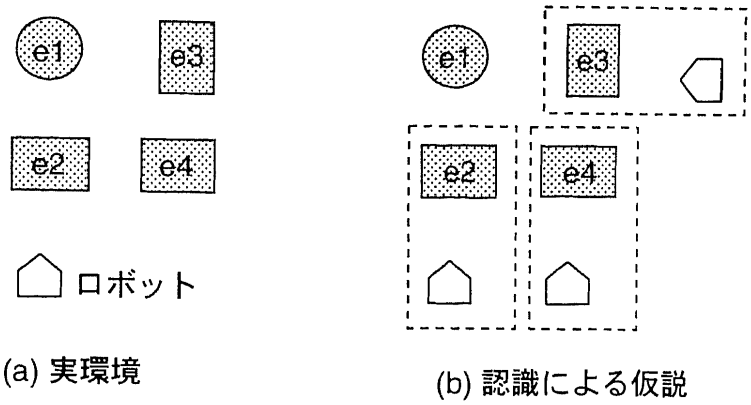


図 4.3: 環境の認識

4.2.4 従来のマルコフ姿勢推定

ロボットの姿勢推定は，環境認識によるセンサデータと走行によるオドメトリデータをマルコフ姿勢推定 [6, 7, 45, 80] を用いて融合することで行う．いま，ロボットに関する事象列を，$< z^{(1)}, m^{(1)}, \ldots, z^{(t)} >$ と表す．$z^{(t)}$ は，ロボットが時点 t において物体認識により得たセンサデータである．$m^{(t)}$ は，時点 t の環境認識と $t+1$ の環境認識の間に実行された走行事象である．以下では，簡単のため，$ZZ^{(t)} =< z^{(1)}, m^{(1)}, \ldots, m^{(t-1)}, z^{(t)} >$, $ZM^{(t)} =< z^{(1)}, m^{(1)}, \ldots, z^{(t)}, m^{(t)} >$ と書く．

移動ロボットの確率的姿勢推定は，事象列 $ZZ^{(t)}$ が与えられたときのロボット姿勢 $P(r^{(t)}|ZZ^{(t)})$ を推定する問題として定式化される．これは，マルコフ姿勢推定の枠組を用いて次のように計算できる．なお，従来の方法では，$r^{(t)}$ はグローバル座標系で表される．

$$\begin{aligned} P(r^{(t)}|ZZ^{(t)}) &= \frac{P(r^{(t)}, z^{(t)}, ZM^{(t-1)})}{P(z^{(t)}, ZM^{(t-1)})} \\ &= \frac{P(r^{(t)}, z^{(t)}|ZM^{(t-1)})}{P(z^{(t)}|ZM^{(t-1)})} \\ &= \alpha P(z^{(t)}|r^{(t)}, ZM^{(t-1)}) P(r^{(t)}|ZM^{(t-1)}) \\ &= \alpha P(z^{(t)}|r^{(t)}) P(r^{(t)}|ZM^{(t-1)}) \end{aligned} \tag{4.3}$$

ここで，$\alpha = P(z^{(t)}|ZM^{(t-1)})^{-1}$ であり，正規化係数として扱われる．式 (4.3) では，マルコフ性 $P(z^{(t)}|r^{(t)}, ZM^{(t-1)}) = P(z^{(t)}|r^{(t)})$ が仮定されている．これは，センサデータ $z^{(t)}$ が現在のロボット姿勢 $r^{(t)}$ だけに依存し，それ以前の事象には依存しないことを意味する．$P(z^{(t)}|r^{(t)})$ は，ロボットが姿勢 $r^{(t)}$ にあるとき，センサデータ $z^{(t)}$ を得る確率であり，マップとセンサ特性 (ビジョンによる姿勢推定の精度) から求められる．

一方，走行に関しては，以下の式を得る．

$$
\begin{aligned}
& P(r^{(t)}|ZM^{(t-1)}) \\
&= \int P(r^{(t)}|r^{(t-1)}, ZM^{(t-1)})P(r^{(t-1)}|ZM^{(t-1)})dr^{(t-1)} \\
&= \int P(r^{(t)}|r^{(t-1)}, m^{(t-1)})P(r^{(t-1)}|ZZ^{(t-1)})dr^{(t-1)} \qquad (4.4)
\end{aligned}
$$

ここで，$r^{(t-1)}$ は時点 $t-1$ でのロボット姿勢なので，$P(r^{(t-1)}|ZM^{(t-1)}) = P(r^{(t-1)}|ZZ^{(t-1)})$ である．また，$P(r^{(t)}|r^{(t-1)}, ZM^{(t-1)}) = P(r^{(t)}|r^{(t-1)}, m^{(t-1)})$ が仮定されている．これは，$r^{(t-1)}$ から走行 $m^{(t-1)}$ を行った結果は，それ以前の事象に依存しないことを意味する．

従来のマルコフ姿勢推定では，式（4.3）と式（4.4）を結合した漸化式を用いて，各時点でのロボット姿勢を次々と求める．

4.2.5 TG マップにおける姿勢推定

TG マップにおけるマルコフ姿勢推定の問題点は，ロボット姿勢が基準実体からの相対姿勢で表されるため，基準実体の切替えが起こると，式（4.3）と式（4.4）を単純に結合できないことである．いま，ロボット姿勢の基準実体を，その時点で認識された実体に設定し，1 つの走行事象中は変えないとする．すると，式（4.3）の $r^{(t)}$ を表す基準実体は，時点 t でのセンサデータを $z^{(t)} =< u_2, q_2 >$ として，$Ins(u_2)$ に属する実体のどれかになる．一方，式（4.4）では，時点 $t-1$ での姿勢からの走行なので，時点 $t-1$ でのセンサデータを $z^{(t-1)} =< u_1, q_1 >$ とすると，基準実体は $Ins(u_1)$ に属する実体のどれかになる．ここで，式（4.3）と式（4.4）の基準実体が異なる場合は，このままでは両式を結合できない．

そこで，両式を結合するために，時点 $t-1$ と時点 t のロボット姿勢の基準実体が同じになるように，基準実体の切替え操作を式の中に導入する．まず，式（4.3）のかわりに，各 $e_2 \in Ins(u_2)$ を基準実体としたロボット姿勢の確率 $P(s_2^{(t)}|ZZ^{(t)})$ を計算する．ただし，$s_2^{(t)} =< e_2, r_{e_2}^{(t)} >$ である．

$$
\begin{aligned}
& P(s_2^{(t)}|ZZ^{(t)}) = P(s_2^{(t)}|z^{(t)}, ZM^{(t-1)}) \\
&= P(< e_2, r_{e_2}^{(t)} > | < u_2, q_2 >, ZM^{(t-1)}) \\
&= \frac{P(e_2, r_{e_2}^{(t)}, q_2|u_2, ZM^{(t-1)})}{P(q_2|u_2, ZM^{(t-1)})} \\
&= \alpha P(r_{e_2}^{(t)}, q_2|e_2, u_2, ZM^{(t-1)})P(e_2|u_2, ZM^{(t-1)}) \\
&= \alpha P(r_{e_2}^{(t)}, q_2|e_2, ZM^{(t-1)})P(e_2|u_2) \\
&= \alpha P(q_2|r_{e_2}^{(t)}, e_2, ZM^{(t-1)})P(r_{e_2}^{(t)}|e_2, ZM^{(t-1)})\frac{1}{N_{u_2}} \\
&= \alpha' P(e_2, q_2|e_2, r_{e_2}^{(t)}, ZM^{(t-1)})P(r_{e_2}^{(t)}|e_2, ZM^{(t-1)}) \\
&= \alpha' P(h_2^{(t)}|s_2^{(t)}, ZM^{(t-1)})P(r_{e_2}^{(t)}|e_2, ZM^{(t-1)})
\end{aligned}
$$

$$= \alpha' P(h_2^{(t)}|s_2^{(t)}) P(r_{e_2}^{(t)}|e_2, ZM^{(t-1)}) \tag{4.5}$$

ここで，$h_2^{(t)} =< e_2, q_2 >$，$\alpha' = 1/P(q_2|u_2, ZM^{(t-1)})N_{u_2}$ である．また，式 (4.5) において，$h_2^{(t)}$ と $s_2^{(t)}$ は，常に同じ実体 (ここでは e_2) を対象にするものとする．

式の導出において，4 行目 $P(r_{e_2}^{(t)}, q_2|e_2, u_2, ZM^{(t-1)})$ の条件部の u_2 は e_2 があるので冗長であり，$P(r_{e_2}^{(t)}, q_2|e_2, ZM^{(t-1)})$ と省略した．また，4〜5 行目で，$P(e_2|u_2, ZM^{(t-1)}) = P(e_2|u_2)$ としているが，これは，$Ins(u_2)$ から e_2 を選ぶ確率がそれまでのセンサデータや走行とは独立であると仮定したことによる．$P(e_2|u_2, ZM^{(t-1)})$ の条件部にある u_2 は，u_2 の姿勢に関する情報を何ももたないので，この仮定は成り立つ．さらに，この確率は一様分布であると仮定し，6 行目で $P(e_2|u_2) = 1/N_{u_2}$ としている．N_{u_2} は，$Ins(u_2)$ の要素数である．さらに，8〜9 行目で，$P(h_2^{(t)}|s_2^{(t)}, ZM^{(t-1)}) = P(h_2^{(t)}|s_2^{(t)})$ としているが，これは，前節で述べたマルコフ性の仮定に相当する．

式 (4.5) の $P(r_{e_2}^{(t)}|e_2, ZM^{(t-1)})$ は，時点 $t-1$ でロボットの姿勢が $e_1 \in Ins(u_1)$ を基準実体にして表現されている時，そのまま時点 t まで走行して，その後基準実体を e_2 に変換したと考えると，以下のように変形できる．

$$\begin{aligned}
&P(r_{e_2}^{(t)}|e_2, ZM^{(t-1)}) \\
&= \sum_{e_1 \in Ins(u_1)} \int P(r_{e_2}^{(t)}, s_1^{(t)}|e_2, ZM^{(t-1)}) dr_{e_1}^{(t)} \\
&= \sum_{e_1 \in Ins(u_1)} \int P(r_{e_2}^{(t)}|s_1^{(t)}, e_2, ZM^{(t-1)}) \times \\
&\qquad\qquad P(s_1^{(t)}|e_2, ZM^{(t-1)}) dr_{e_1}^{(t)} \\
&= \sum_{e_1 \in Ins(u_1)} \int P(r_{e_2}^{(t)}|s_1^{(t)}, e_2) P(s_1^{(t)}|ZM^{(t-1)}) dr_{e_1}^{(t)}
\end{aligned} \tag{4.6}$$

ただし，$s_1^{(t)} =< e_1, r_{e_1}^{(t)} >$ である．式の導出において，3〜4 行目で，$P(r_{e_2}^{(t)}|s_1^{(t)}, e_2, ZM^{(t-1)}) = P(r_{e_2}^{(t)}|s_1^{(t)}, e_2)$ を仮定している．$P(r_{e_2}^{(t)}|s_1^{(t)}, e_2, ZM^{(t-1)})$ は，時点 $t-1$ で基準実体が $e_1 \in Ins(u_1)$ であり，かつ，そのまま時点 t まで走行したときのロボット姿勢が $s_1^{(t)}$ であるとき，基準実体を $e_2 \in Ins(u_2)$ に変換した場合に，ロボットが e_2 のローカル座標系で $r_{e_2}^{(t)}$ の姿勢にある確率である．上記仮定は，この確率が $ZM^{(t-1)}$ とは独立に，マップだけで決まることを意味する．すると，$P(r_{e_2}^{(t)}|s_1^{(t)}, e_2)$ は時点 t に独立になるので，$P(r_{e_2}|s_1, e_2)$ と書ける．

式 (4.6) の $P(s_1^{(t)}|ZM^{(t-1)})$ は，基準実体が e_1 に固定されているので，式 (4.4) と同様に，以下のようになる．

$$\begin{aligned}
&P(s_1^{(t)}|ZM^{(t-1)}) \\
&= \int P(s_1^{(t)}|s_1^{(t-1)}, ZM^{(t-1)}) P(s_1^{(t-1)}|ZM^{(t-1)}) dr_{e_1}^{(t-1)} \\
&= \int P(s_1^{(t)}|s_1^{(t-1)}, m^{(t-1)}) P(s_1^{(t-1)}|ZZ^{(t-1)}) dr_{e_1}^{(t-1)}
\end{aligned} \tag{4.7}$$

式（4.7）の $P(s_1^{(t)}|s_1^{(t-1)}, m^{(t-1)})$ は，$s_1^{(t-1)}$ から走行 $m^{(t-1)}$ を行ったときに，$s_1^{(t)}$ に到達する確率である．各 $e_1 \in Ins(u_1)$ の近傍の床面の状態が同じで障害物もないと仮定すれば，この確率は e_1 に依存しない．そこで，これを $P(r^{(t)}|r^{(t-1)}, m^{(t-1)})$ と書く．この分布は，後述のように，走行事象 $m^{(t-1)}$ に伴うオドメトリの測定結果だけから計算できる．

このように，基準実体が切り替わるときは，式（4.5），(4.6），(4.7）を結合した漸化式を用いてロボット姿勢を求める．基準実体の切替えがない区間は，その基準実体の座標系で従来のマルコフ姿勢推定を行えばよい．

4.2.6 モデル分布

以上の姿勢推定を行うのに必要な確率分布は，$P(h_i^{(t)}|s_i^{(t)})$，$P(r^{(t)}|r^{(t-1)}, m^{(t-1)})$，$P(r_{e_2}|s_1, e_2)$ である．これらは，モデル分布としてあらかじめ与えておく．

(1) $P(h_i^{(t)}|s_i^{(t)})$

$P(h_i^{(t)}|s_i^{(t)})$ は，ロボットが姿勢 $s_i^{(t)} =< e_i, r_{e_i}^{(t)} >$ のときに，センサデータから仮説 $h_i^{(t)} =< e_i, q_i >$ を得る確率，すなわち，実体 e_i がロボット座標系で姿勢 q_i にある確率である．これはセンサ誤差と物体認識誤差によって生じる確率である．$h_i^{(t)}$ はセンサデータから得られるので，実際には，この確率はロボットの姿勢に対する尤度となる．いま，物体認識について誤認識がないと仮定すれば，この確率は，$r_{e_i}^{(t)}$ から逆算したロボット座標系での実体 e_i の姿勢と q_i との差に対して正規分布 $N(0, \Sigma_1)$ を適用して求めることができる．ここで，Σ_1 の値はセンサおよび物体認識の精度に応じて設定する．

物体形状によっては，姿勢 q_i が特定できない場合がある．たとえば，円柱の姿勢の方位成分は認識では得ることができない．この場合，$P(h_i^{(t)}|s_i^{(t)})$ は，円柱 e_i から等距離にあり，かつ，ロボットから見える円柱の方向が同じである姿勢 $r_{e_i}^{(t)}$ に対して同じ値になる．言い替えると，このロボット姿勢の尤度は，円柱の周囲をとりまくドーナツ状に分布する．

(2) $P(r^{(t)}|r^{(t-1)}, m^{(t-1)})$

$P(r^{(t)}|r^{(t-1)}, m^{(t-1)})$ は，ロボットが $r^{(t-1)}$ にいるときに走行事象 $m^{(t-1)}$ によって $r^{(t)}$ に至る確率である．ここで，$r^{(t)}$ の期待値はオドメトリによって測定される．また，その誤差は正規分布 $N(0, \Sigma_2)$ に従うと仮定する．Σ_2 は，ロボットのオドメトリの精度とその間の走行距離によって与える．

(3) $P(r_{e_2}|s_1, e_2)$

$P(r_{e_2}|s_1, e_2)$ は，3.3.4 節で述べた $P(q_{12})$ の確率分布 $N(\bar{q}_{12}, \Sigma_{12})$ をもとに，4.2.2 節で述べたようにして求める．

4.2.7 例

図 4.4に TG マップ上でのマルコフ姿勢推定の様子を模式的に示す．同図 (a) は実環境の正確なマップであり，ロボットは A 点から B 点まで移動するものとする．図で，q_1, q_1' は，それぞれ，e_1 から見た A 点，B 点の相対位置ベクトルであり，q_2 は e_2 から見た B 点の相対位置ベクトルである．e_1, e_2 は同じクラスの実体であり，ロボットからは区別がつかないとする．ロボットには，同図 (b)〜(f) のように，e_1 と e_2 間の距離が短い不正確なマップが与えられたとし，マップでは，e_1 と e_2 の相対姿勢は，両者をつなぐ直線方向の分散が大きいように定義されているとする．

ロボットは A 点で e_1 を観測し，次に B 点まで移動して e_2 を観測する．このとき，まず，ロボットは A 点で e_1 を観測したことにより，同図 (b) に示すような仮説を生成する．ロボットは e_1, e_2 の区別がつかないため，両者に対して相対姿勢 q_1 を中心とした姿勢分布が生成される．点線の矩形は，それぞれ，e_1, e_2 を基準実体としたローカル座標系を表す．2 つの仮説はそれぞれのローカル座標系で定義される．同図 (c) はロボットが B 点まで走行したとき，オドメトリ誤差によって拡散したロボットの推定姿勢である．これは，式 (4.7) に基づいている．同図 (d) は，同図 (c) の $r_{e_1}^{(t)}$ を基準実体を e_2 に変えて表わしたものである．これは，式 (4.6) に基づいている．このときの推定姿勢は基準実体間の姿勢の不正確さによって大きく拡散している．見やすさのため図では省略しているが，$r_{e_2}^{(t)}$ から $r_{e_1}^{(t)}$ への座標変換も存在する．これについては，同図 (c) の上の円が e_1 のローカル座標系に変換されて楕円形に拡散する．同図 (e) は，B 点においてロボットが実体 e_2 を観測して生成した仮説である．同図 (f) は，(d) と (e) のロボット姿勢を融合した結果である．2 つあった仮説が 1 つに収束し，ロボットは，e_2 を基準実体として，同図 (a) とほぼ同じ相対姿勢にいることがわかる．

本方式の特徴は，図 4.4(d) のステップにある．もしこれがなく，同図 (f) の融合が (c) と (e) のロボット姿勢で行われるとすると，(c) と (e) のロボット姿勢が空間的に重なる部分がきわめて小さいため，得られる確率値が小さくなり，姿勢推定は不安定になる．また，たとえば，e_2 の先に同じクラスの実体 e_3 があった場合，e_3 を観測したように間違える可能性も生じる．本手法では，基準実体間の姿勢関係が不正確であることが陽に与えられているので，この場合でも，e_2 を観測したという仮説と e_3 を観測したという仮説の両方が生き残る．

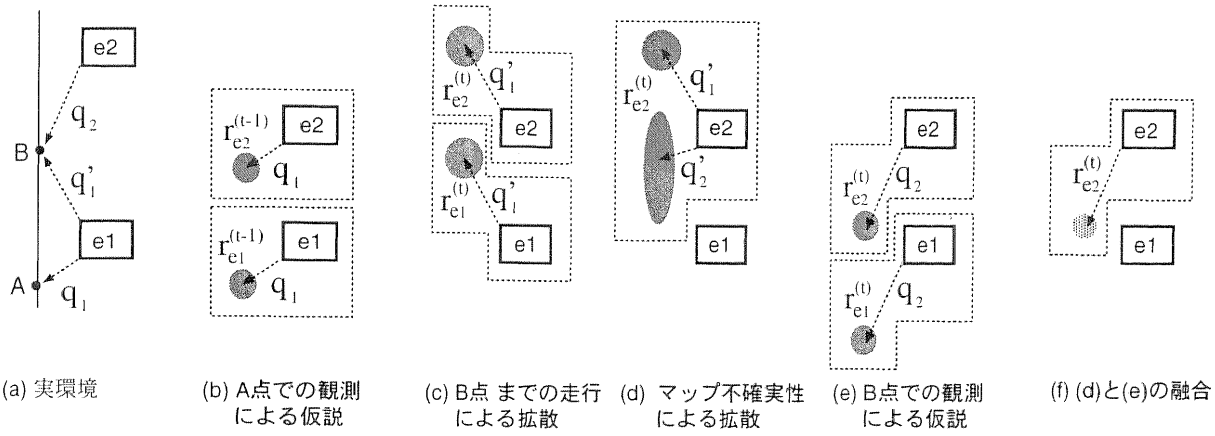


図 4.4: TG マップにおけるマルコフ姿勢推定の動作例

4.3 モンテカルロ法による姿勢推定アルゴリズム

ロボット姿勢の確率分布を効率よく近似計算する方法として，モンテカルロ法が提案されている [24, 40]．モンテカルロ法による姿勢推定では，ロボットの推定姿勢の確率分布をサンプルの集合で表現する．ロボットが外界センサでランドマークを認識すると，そのセンサデータを用いて，各サンプルの重み (確率) を更新し，その重みにしたがってリサンプリングを行う．ロボットが走行すると，オドメトリの誤差分布にしたがってサンプルを拡散させる．この 2 つのステップを交互に繰り返すことで，ロボットの推定姿勢の確率分布が絞り込まれる．ロボットの初期姿勢が未知の場合は，初期姿勢の分布をマップ全域に対する一様分布と仮定すればよい．

本方式でもモンテカルロ法を用いるが，本方式では，ロボット姿勢が基準実体とそこからの相対姿勢の対で表されるため，ロボットが基準実体を変えたときにロボット姿勢の座標変換を行う処理を加える．この座標変換は，1 つ前の時点の仮説の基準実体と現時点の仮説の基準実体のすべての組み合わせについて計算する必要がある．このため，ロボットが実体を認識して仮説を作った時に，いったん仮説数に比例してサンプル数を増加させ，リサンプリングによって再び元のサンプル数に戻すようにする．

モンテカルロ法により，式 (4.5)，(4.6)，(4.7) を計算するアルゴリズムを以下に示す．ステップ (1)，(2) が式 (4.5)，(4.6) に対応し，ステップ (3) が式 (4.7) に対応する．ナビゲーションの間，ステップ (1)〜(3) が繰り返される．

いま，サンプル数を N とし，時点 t におけるロボット姿勢のサンプルを $s^{(t)}[j] =< e^{(t)}[j], r^{(t)}[j] >$ $(1 \leq j \leq N)$ と表す．$e^{(t)}[j]$ はそのサンプルに対する基準実体，$r^{(t)}[j]$ は $e^{(t)}[j]$ の座標系でのロボットの相対姿勢である．時点 t において，ロボットが実体を認識して仮説を生成すると，各仮説ごとに N 個のサンプルを新しく生成する．このサンプルを

$ss^{(t)}[k] =< e^{(t)}[k], r^{(t)}[k], w^{(t)}[k] >$ $(1 \leq k \leq MN)$ と表す．M は仮説数である．$e^{(t)}[k]$ はそのサンプルに対する基準実体，$r^{(t)}[k]$ は $e^{(t)}[k]$ の座標系でのロボットの相対姿勢である．また，$w^{(t)}[k]$ はロボット姿勢の尤度から与えられるサンプルの重みである．

(1) 認識によるサンプル更新

外界センサによるランドマーク認識でセンサデータ $z^{(t)} =< u^{(t)}, q^{(t)} >$ が得られたとする．各 $e_i^{(t)} \in Ins(u^{(t)})$ に対して仮説 $h_i^{(t)} =< e_i^{(t)}, q^{(t)} >$ が生成されたときのサンプル $ss^{(t)}[k]$ を以下のように生成する．ただし，$M^{(t)}$ は $Ins(u^{(t)})$ の要素数である．

$$
\begin{aligned}
&for \;\; i = 1 \;\; to \;\; M^{(t)} \\
&\quad for \;\; j = 1 \;\; to \;\; N \\
&\qquad k = N(i-1) + j \\
&\qquad ss^{(t)}[k] =< e_i^{(t)}, r, w > \\
&\text{ただし，} \\
&\qquad r = trans(r^{(t-1)}[j], e^{(t-1)}[j], e_i^{(t)}) \\
&\qquad w = \beta^{(t)} \cdot P(h_i^{(t)} | < e_i^{(t)}, r >)
\end{aligned}
$$

ここで，$trans(r^{(t-1)}[j], e^{(t-1)}[j], e_i^{(t)})$ は，$e^{(t-1)}[j]$ の座標系でのロボットの相対姿勢 $r^{(t-1)}[j]$ を $e_i^{(t)}$ の座標系に変換する関数である．この関数により，4.2.2節で述べたように，ロボットの姿勢は $P(r_{e_i^{(t)}} | < e^{(t-1)}[j], r_{e^{(t-1)}[j]} >, e_i^{(t)})$ の分布に従って拡散する．また，$\beta^{(t)}$ は $w^{(t)}[k]$ の合計を 1 とするための正規化係数である．

(2) リサンプリング

$ss^{(t)}[k]$ の重み $w^{(t)}[k]$ に比例する確率で，$M^{(t)}N$ 個の $ss^{(t)}[k]$ の中から，N 個のサンプルを取り出す．そして，取り出したサンプルを $s^{(t)}[j] =< e^{(t)}[j], r^{(t)}[j] >$ $(1 \leq j \leq N)$ とする．新しいサンプルの $e^{(t)}[j]$ と $r^{(t)}[j]$ は，$ss^{(t)}[k]$ の $e^{(t)}[k]$ と $r^{(t)}[k]$ を k から j にナンバリングしなおしたものである．

(3) 走行

走行事象 $m^{(t-1)}$ が起こると，各サンプル $s^{(t-1)}[j]$ を以下のように更新する．

$$
\begin{aligned}
&for \;\; j = 1 \;\; to \;\; N \\
&\quad s^{(t)}[j] =< e^{(t-1)}[j], r^{(t-1)}[j] + dr[j] >
\end{aligned}
$$

ただし，$dr[j]$ は確率 $P(r^{(t)} | r^{(t-1)}, m^{(t-1)})$ にしたがって発生された移動量である．

なお，初期サンプルは，時点 0 におけるセンサデータによる仮説 $h_i^{(0)} =< e_i^{(0)}, q^{(0)} >$ をもとに生成する．その処理は，$ss^{(0)}[k]$ を以下のように計算する以外は，ステップ (1) と同じである．

$$ss^{(0)}[k] =< e_i^{(0)}, r, w >$$
ただし，
$$r = pick(e_i^{(0)}, q^{(0)})$$
$$w = \beta^{(0)} P(h_i^{(0)} | < e_i^{(0)}, r >)$$

ここで，$pick(e_i^{(0)}, q^{(0)})$ は，ロボットから見て実体 $e_i^{(0)}$ が $q^{(0)}$ の姿勢に見える $e_i^{(0)}$ 座標系での点の近傍領域からランダムにロボット姿勢を返す関数である．

4.4 シミュレーション

上に述べたアルゴリズムの動作を確認するため，シミュレーションを行った．図 4.5(a) は，ロボットが走るべき実環境である．ロボットは A 点から D 点まで走り，A 点で e_1，B 点で e_2，C 点で e_3，D 点で e_4 を認識するが，e_1〜e_4 は同じクラスの実体であり，ロボットは区別できないものとした．ここで，q_1 は e_1 から A 点への相対位置ベクトルを表す．q_2〜q_6 についても，各々，図 4.5(a) に示すとおり，e_1〜e_4 の各実体から見た B〜D 点の相対位置である．図 4.5(b)〜(f) はロボットに与えられた不正確なマップであり，図 4.5(a) に比べて e_2 と e_3，e_3 と e_4 の間隔が約半分に短く定義されている．ロボットは初期姿勢についての情報はもたず，このマップを用いて自己姿勢を推定しながら，図 4.5(a) の A 点から D 点まで走行するものとしてシミュレーションを行った．

図 4.5(b)〜(f) に，シミュレーション結果を示す．これらの図で，マップ上の細かな点は，モンテカルロ法によるロボット姿勢のサンプルを表す．このサンプルの分布がロボット姿勢の分布を表し，密集している領域ほどロボットのいる確率が高い．サンプル数は 1000 である．図では点の個数が変化しているように見えるが，これはリサンプリングによって複数のサンプルが同じ点に重なることがあるためである．また，点線で囲まれた領域は，各仮説の基準実体のローカル座標系を表す．

図 4.5(b) は，A 点で e_1 を認識したことにより推定されたロボット姿勢の初期分布である．ロボットは，e_1〜e_4 の区別がつかないので，ここでは e_1〜e_4 それぞれの近傍に仮説が存在している．図 4.5(c) は，B 点まで走行した直後のロボット姿勢分布である．初期分布が走行分だけ移動している．図 4.5(d) は，B 点で e_2 を認識した後のロボット姿勢分布である．e_1 と e_2 を連続して観測したことにより，初期分布の 4 つの仮説が 1 つに絞り込まれている．これは，4 つの仮説のうちの 1 つが他に対して大幅に優勢な確率をもつようになり，リサンプリングの結果，ほぼその仮説のサンプルだけが残ったことを意味する．

図 4.5(e) はC点で e_3 を認識した後のロボット姿勢分布である．前述のように，ロボット姿勢の推定には基準実体の同定が重要であるが，マップが不正確な場合は，一意に同定できないことが起こる．図 4.5(e) はその例になっている．図 4.5(e) では，e_2〜e_4 の間隔が実環境より短いため，認識した実体が e_3 か e_4 か決められず，2つの仮説が生成されている．C点からかなり離れた場所でも仮説が生きているのは，e_2〜e_4 の相対姿勢の誤差が大きく，基準実体を切替える際の座標変換によりロボット姿勢が拡散したためである．この2つの仮説は，図 4.5(f) に示すように，D点で e_4 を認識した後，e_3 と e_4 の観測を融合することにより，1つに絞り込まれた．

このように個体が一意に識別できない現象は，不正確なマップにおいて同じクラスに属する複数の実体が比較的近くに集まっている場合によく起こりうるものであり，これに対処することが，本方式が Global Localization の枠組を採用した理由の1つである．この現象は，言い方を変えると，図 4.5(e) の e_3 と e_4 の中間地点に図 4.5(a) の e_3 を見つけたことに相当する．ロボットは図 4.5(e) のマップしかもっておらず，物体認識では両者の区別はつかないので，これらの情報だけからでは，見つけた実体が e_3 か e_4 かを決めることはできない．Position Tracking のように1つの仮説を追跡する方法では，いったんどちらかの仮説を選ぶ必要があるが，どちらかをランダムに選んだ場合は，後で失敗した場合にバックトラックなどの別の手続きが必要になる．あるいは，この時点で e_3 と e_4 の確実な方を選ぶために，たとえば，点Cでは e_3 が e_4 より手前にあるので先に見えるはずだ，というようなヒューリスティクスを利用することが考えられる．しかし，一般には，e_3 と e_4 は任意の配置が可能であり，一般の2次元配置の制約を解くようなヒューリスティクスが必要になる．一方，本方式では，Global Localization の枠組をもっているため，つねに複数仮説を保持することが可能であり，バックトラックの必要はない．しかも，基準実体を切替えてデータ融合を行う過程で，実体間の2次元配置の制約を自然に取り入れているため，上記のようなヒューリスティクスを用いなくても，図 4.5(f) に示すように，後続のセンサデータを融合していくことで仮説を絞り込むことができる．

4.5 基礎実験による本方式の有効性の検証

4.5.1 実験条件

筑波大学校舎の廊下において，実環境データを用いた実験を行った．図 4.7(a) に正確なマップを示す．図 4.7(b) に目視で作成した TG マップを示す．TG マップの誤差は，30[cm]〜3[m] である．たとえば，正確なマップと比べて，door2 と door3 の間隔，および，door2 と pillar1 の間隔は約 3[m] 短く，door1 と door2 の間隔は 1[m] 長い．また，pillar1 と tape1，tape1 と tape2 の間隔も 1〜2[m] 短い．図 4.6に実験環境と認識対象を示す．

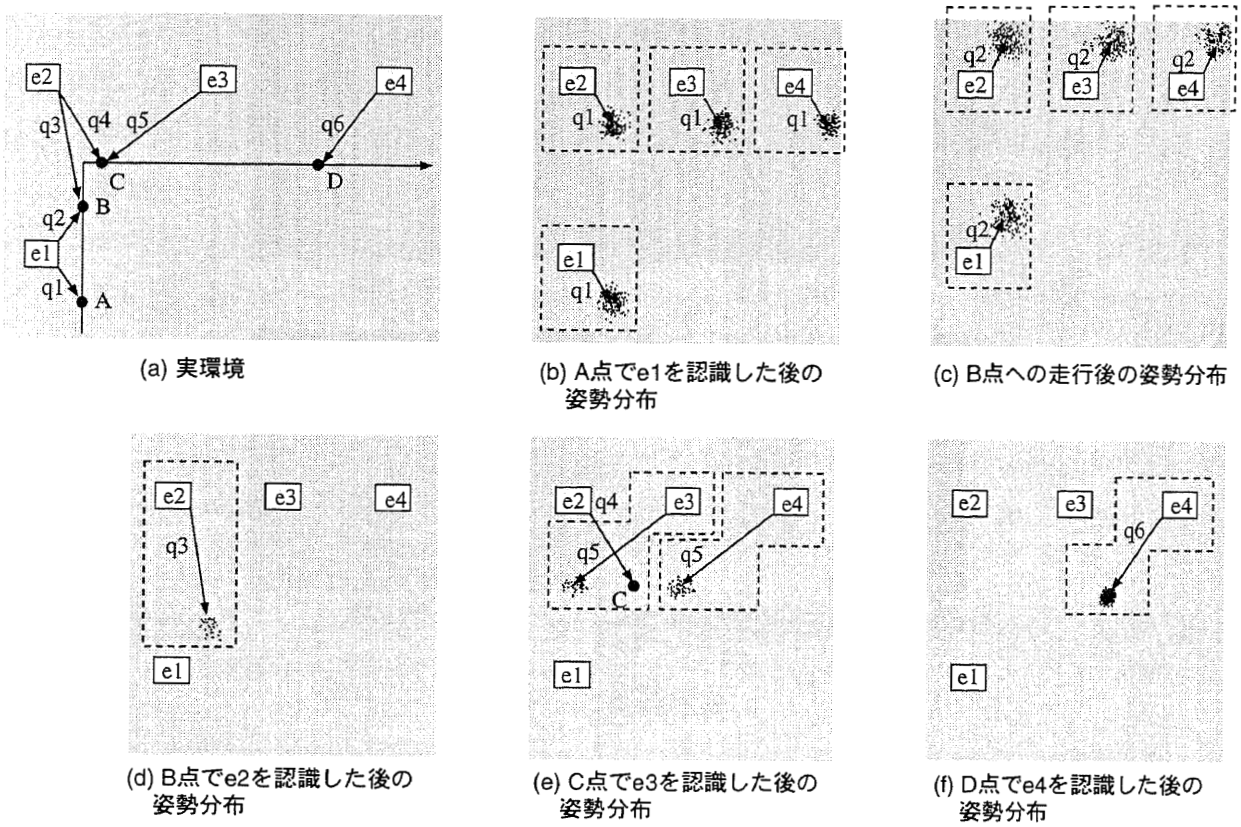


図 4.5: シミュレーション結果

ロボットは図 4.7(a) の A 点から E 点までを走行し，A～E の各点で画像を撮影し，ドア (door1～door4)，円柱 (pillar1)，および，壁に張り付けたビニールテープ (tape1,tape2) をランドマークとして認識するものとした．なお，基礎実験であるため，走行時にはロボットは各点で撮った画像とその間のオドメトリデータを記憶し，走行後オフラインで，これらの画像からドア，円柱，ビニールテープの認識と姿勢推定を行い，その結果をもとに各点でのロボット姿勢の推定を行った．

ドア，円柱，ビニールテープの認識および姿勢推定は，次節の物体認識手法は用いず，画像のエッジと色領域を認識特徴として用いる簡易手法で行った [87]．4.2.6節で述べた Σ_1 は，$\sigma_x = 10[cm]$, $\sigma_y = 10[cm]$ と設定した．σ_θ は方位が定まるドアについては 3[度] と設定した．多くの場合，σ_x, σ_y はマップ誤差に比べると小さいので，センサデータによってロボット姿勢の精度は向上する．[87] の方法では，ドアの姿勢は，位置と方位の両方が認識により確定できるが，円柱とビニールテープの方位は一意に定まらない．したがって，円柱やビニールテープの認識によるロボット姿勢の仮説の尤度は，円柱やビニールテープを取り巻くドーナツ状に分布する．このような実体では，局所的なロボット姿勢さえもセンサデータだけで決めることはできず，ロボットがどこから来たかという過去のデータとの融合が本質的な役割

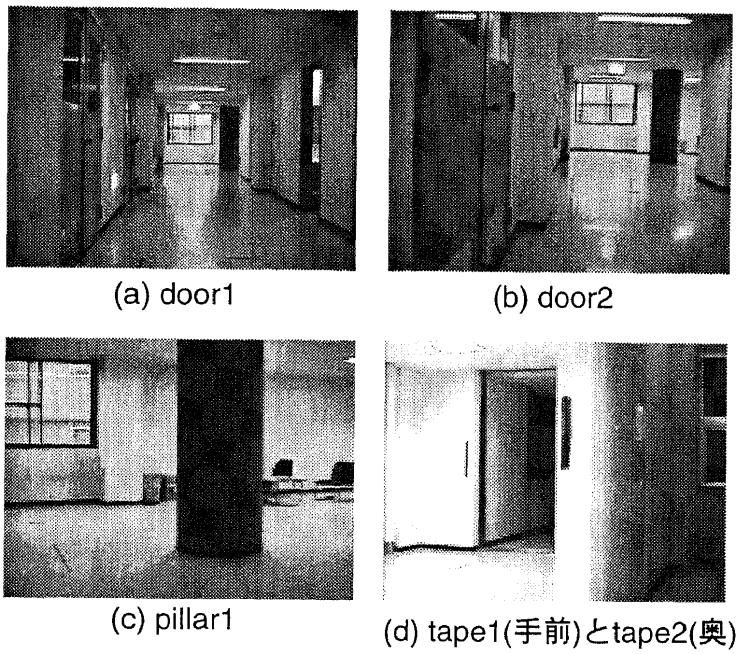


図 4.6: 認識対象

を果たす.

画像はパンチルトカメラ Sony EVI-D30 をロボットに載せた状態で撮影し，自己位置推定の計算はノート PC (CPU : PentiumII 400MHz，メモリ 96MB) で行った．モンテカルロ法で用いたサンプル数は 1000 個である．データ融合の過程で，一時的に，サンプル数は実体の組み合わせに比例した分だけ増える．たとえば，この実験では同じドアが 4 枚あるため，サンプル数は一時的に 4000 個に増える．

4.5.2 実験結果

図 4.7(b)〜(l) に実験結果を示す．シミュレーションと同様，マップ上の細かな点は，ロボット姿勢のサンプルを表し，点線で囲まれた領域は各仮説の基準実体のローカル座標系を表す．図 4.7(b) は，A 点における初期推定姿勢を表している．A 点では，ロボットは door1 を認識したが，この段階では，他のドアと区別がつかないため，door1 から A 点への相対姿勢を各ドアに適用して複数の初期姿勢仮説が生成されている．次に，B 点では，ロボットは door2 を認識した．まず，B 点まで走行した時点では，図 4.7(c) に示すように複数の仮説をもっている．B 点での認識による姿勢推定で，図 4.7(d) に示すように，door1 と door2 の認識結果が融合されて，正しい仮説が選ばれた.

図 4.7(e)〜(h) は，C 点での姿勢推定における基準実体の切替えと座標変換によるロボット姿勢の拡散の様子を詳しく示している．図 4.7(e) は，C 点まで走行したときの door2 を基

準実体にしたロボット姿勢を表す．マップが不正確なため，このロボット姿勢は pillar1 から見ると少しずれている．C 点でロボットが pillar1 を認識すると，pillar1 に対するロボット姿勢の仮説が図 4.7(f) に示すようにドーナツ状に得られる．このままでは，この仮説と door2 を基準実体にしたロボット姿勢は重ならず，両者を融合しても有意な姿勢は得られない．そこで，基準実体を door2 から pillar1 に切替え，ロボット姿勢を pillar1 の座標系で表す．すると，door2 と pillar1 間の相対姿勢の誤差分布を考慮した座標変換によって door2 を基準にしたロボット姿勢の分布が拡散され，図 4.7(g) に示すように，pillar1 に対する仮説とそれまでのロボット姿勢とが重なる領域ができる．そして，データ融合によってこの領域が取り出され，図 4.7(h) に示すように，pillar1 を基準実体とした正しいロボット姿勢が得られる．図 4.7(i) と (j)，図 4.7(k) と (l) でも同様に，マップが不正確なため，tape1 および tape2 から見てロボット姿勢がずれていたが，基準実体の切替えにより正しい位置に修正されたことがわかる．

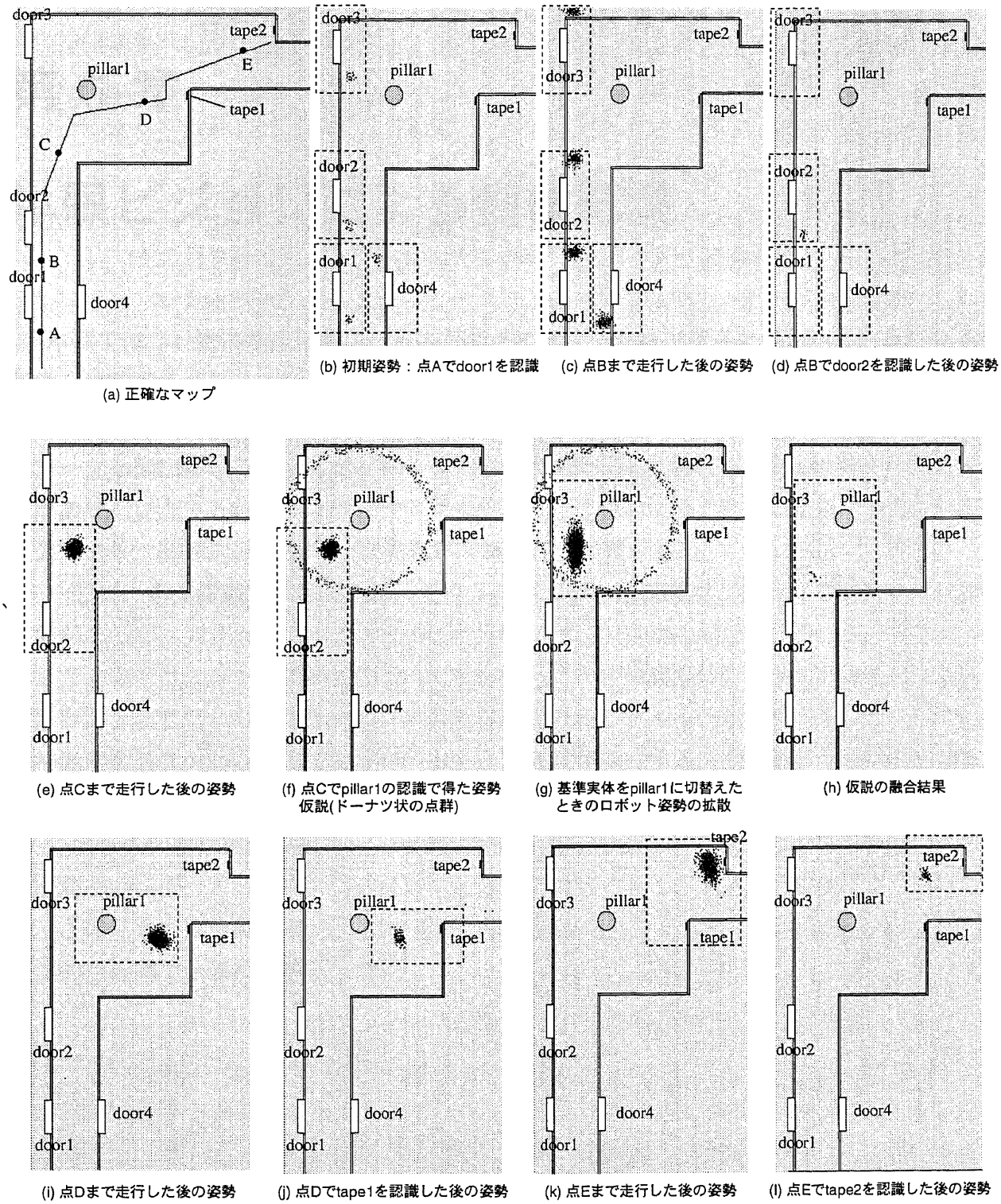

door3
tape2
pillar1
E
D
tape1
C
door2
B
door1
door4
A
(a) 正確なマップ
(b) 初期姿勢：点Aでdoor1を認識
(c) 点Bまで走行した後の姿勢
(d) 点Bでdoor2を認識した後の姿勢
(e) 点Cまで走行した後の姿勢
(f) 点Cでpillar1の認識で得た姿勢
仮説(ドーナツ状の点群)
(g) 基準実体をpillar1に切替えた
ときのロボット姿勢の拡散
(h) 仮説の融合結果
(i) 点Dまで走行した後の姿勢
(j) 点Dでtape1を認識した後の姿勢
(k) 点Eまで走行した後の姿勢
(l) 点Eでtape2を認識した後の姿勢

図 4.7: 実験結果

第 5 章

単眼ビジョンによる物体認識

本章では，1 枚の濃淡画像から透視投影のもとで多面体モデルに基づいて 3 次元物体の認識と姿勢推定を行う方法を提案する．本方式は，画像から目標物体を見つけて，カメラに対するその 3 次元姿勢を求めるものである．目標タスクでは，ナビゲーションのためのランドマークを検出したり，マニピュレータで物体を把持するために環境中の物体を認識する必要がある．ロボットがこのようなタスクを確実に行うためには，物体の姿勢を知ることが不可欠であり，認識だけでなく姿勢推定も同時に行う必要がある．

一般に，物体認識では照明条件，オクルージョン，混雑環境 (clutter) の問題に対処する必要があるが，ロボットが活動する環境では，これらの問題は顕著に現われる．屋内でも，カメラの向きによって照明条件が大きく変わり，ノイズや認識特徴の欠如の原因となる．また，カメラの視野からはみ出たり，他の物体に隠れてオクルージョンが発生したりして，認識特徴が大量に失われることも多い．さらに，人間がふつうに活動する環境を対象にするため，多くの物体が混在し，認識特徴が過剰に発生して計算量が増え，誤認識も多くなる．

これらの問題に対処するため，提案方式は，認識特徴として線分の組を用い，物体の姿勢空間を回転成分と並進成分に分解して段階的に照合することで認識と姿勢推定を同時に行う．この姿勢空間分解と線分に関する種々の特徴フィルタにより，計算量を大幅に削減し，現実的な時間での処理を実現する．

5.1 従来研究と本研究のアプローチ

5.1.1 単眼画像からの物体認識と姿勢推定

ロボットビジョンの分野では，物体の 3 次元姿勢を推定する有効な方法として，レーザーレンジセンサやステレオ視から得た 3 次元データを用いる方法が提案されている [4, 21, 71, 75, 77, 91]．これらは，3 次元データを利用するため，高精度な推定が期待でき，ビンピッキングなどにとくに有効である．しかし，自律移動ロボットに用いる場合は，レーザーレンジ

センサでは物体の詳細形状のデータ取得に時間がかかるという問題があり，ステレオ視では距離の離れた物体に対して精度が悪くなるという問題がある．また，両者とも，単眼カメラに比べると装置が大がかりになる．これらのことから，本研究では，単眼カメラによる1枚の濃淡画像から3次元物体の認識と姿勢推定を行うことを目標としている．

1枚の濃淡画像からの3次元物体認識は，どのような特徴を使うか，どのような照合アルゴリズムを用いるかによって，様々な方式が提案されている [76]．近年の有力な方法として，物体の見え (appearance) に基づく方法と幾何モデルを用いる方法がある．見えに基づく方法では，入力画像に含まれる3次元物体の投影像の2次元特徴を，あらかじめ用意したテンプレートと照合する [56, 63, 66, 73]．このアプローチは，面倒な幾何モデル構築の必要がないという利点をもち，物体の認識には有効である．しかし，一般に，モデル特徴も2次元であるため，物体の姿勢推定を正確に行うのは容易ではない．一方，幾何モデルを用いる方法は，物体の3次元構造の情報を特徴に含めることができるため，正確な姿勢推定が可能である．このため，本方式では幾何モデルアプローチを採用することにした．

3次元空間での姿勢推定は，ピンホールカメラモデルにおける透視投影の関係を用いて，物体の回転成分と並進成分の6自由度の方程式を解く問題に帰着する．従来より，点や直線の組を特徴として用いて姿勢推定を行う方法は多数提案されている [11, 15, 22, 27, 28, 33, 32, 51, 52, 55, 69]．しかし，その多くは画像特徴とモデル特徴の対応がついている場合の方法であり，特徴の対応がついていない場合，すなわち，物体の認識と姿勢推定を同時に行う場合は問題がひとまわり難しい．この場合は，特徴空間と姿勢空間の両方を探索して特徴の対応づけと姿勢推定の両方の問題を同時に解く必要があり，計算量が膨大になる．このため，そこで用いる特徴には，画像から安定に抽出できるというロバスト性の他に，計算量が少ない照合アルゴリズムが実現できるという性質が必要である [26]．

5.1.2 幾何モデルによるアプローチ

幾何モデルアプローチにおいても，これまで様々な方法が提案されている [5, 25, 26, 35, 39, 48, 54, 64]．幾何モデルとして，一般化円筒 [5] や楕円 [64] など特定の図形を用いた方法もあるが，本方式では，ロボットが動く環境中の様々な物体を扱うため，任意形状の多面体 (ワイヤーフレーム) をモデルとして用いることにした．そして，多面体の辺と頂点を特徴として，モデルと画像の照合を行う．なお，多面体の面を細かくすれば曲面を近似できるが，本方式ではさしあたり曲面は対象としない．

多面体の辺や頂点を特徴として用いることで，透視投影での姿勢推定問題に容易に結びつけることができる．透視投影における姿勢推定は，モデル特徴と画像特徴の組から，カメラ座標系に対する物体座標系の姿勢を決める3個の回転成分と3個の並進成分を求める問題である．特徴として，直線や点がよく用いられ，多面体モデルと親和性がよい．姿勢推定問

題についても，これまで多くの方式が提案されており，数値解法と解析的方法に大別される．

数値解法は，画像特徴とモデル特徴の対応を複数個とり，誤差関数が最小になる物体姿勢を非線形最適化法や最小自乗法などを適用して求める [27, 52, 55, 69]．用いる対応の数を多くすれば，画像特徴に誤差があっても精度よく姿勢推定ができる．この方法は，画像特徴とモデル特徴の対応が既知であれば有効であるが，その対応が未知の場合，つまり，認識と姿勢推定を同時に行う場合には難点がある．認識と姿勢推定を同時に行うには，姿勢空間の他に，特徴の対応空間も探索する必要がある．その際，対応空間の各点 (モデル特徴と画像特徴の対) に対して姿勢値を計算し，投票や仮説検証などの方法で候補を求めることになる．しかし，このための姿勢値計算を数値解法で行うと，計算時間がかかりすぎる．

対応空間の探索と姿勢空間の探索を同時に行わず，逐次的に行う方法も考えられる．たとえば，まず物体の見えによって特徴対応を求め，その特徴対応を用いて数値解法による姿勢推定を行う方法が考えられる．この方法は実用性が高いと考えられるが，特徴点の誤差や誤対応が姿勢推定に直接影響する．そのため，特徴対応の精度が高いこと，また，姿勢推定でのロバスト性が高いことが要求される．これに対して，対応空間と姿勢空間の同時探索では，モデルの 3 次元の幾何制約を利用して精度の高い特徴対応が実現できるという利点がある．

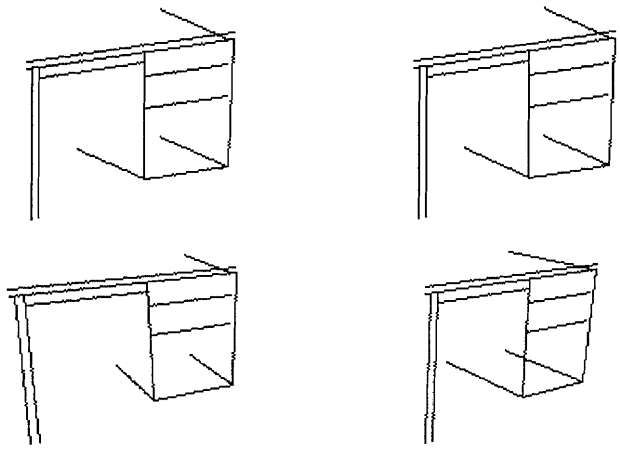
一方，解析的方法は，これらのパラメータの closed-form solution を求めるものである [11, 15, 22, 28, 32, 51]．もっとも直接的な 3 点法では，モデル特徴点と画像特徴点の対を 3 個用いて，並進パラメータと回転パラメータを計算する [22, 28]．また，3 本の直線を用いる方法 [15]，6 本の直線を用いる方法 [11]，4 点を用いる方法 [32] も提案されている．これらの方法では 4 次方程式または 2 次連立方程式を解く必要があるため，計算が複雑になり，また，解の安定性が問題になる．

5.1.3 弱透視投影によるアプローチ

透視投影ではなく弱透視投影を仮定すると，解析的方法でかなり簡単な解を得ることができる [33, 35]．これを利用して認識と姿勢推定を同時に行う方法として，Alignment 法がある [3, 35]．Alignment 法では，モデル特徴点数を M，画像特徴点数を N として，検証まで含めて $O(M^3N^2logN)$ の計算量となる．また，弱透視投影のもとで成り立つアフィン不変量を利用した方法も提案されている [39, 48]．

しかし，弱透視投影では，カメラからの距離に対して奥行きの長さが無視できない物体に対しては歪みが大きくなり，自律移動ロボットへの応用には限界がある．自律ロボットが対象とする物体は，ドアや机，廊下の壁など奥行きのあるものが多く，透視投影によるパース効果が無視できない．図 5.1にその例を示す．これは，机が床面上に置かれていると想定して，カメラが机に対して横方向 (y 方向) に 150cm 移動した場合の弱透視投影 (上段) と透視投影 (下段) による机の見え方の違いを模擬したものである．カメラに対して机は垂直軸

回りに 30 度回転した姿勢にあり，カメラは下向きに 15 度傾けている．カメラと机の奥行き方向の距離は 300cm，カメラの床面からの高さは 135cm に固定している．このような位置関係は自律移動ロボットではよく起こり得るが，弱透視投影では図 5.1のように線分の長さや傾きがかなり違ってしまうため，認識や姿勢推定が正しくできないことが多い．



(a) y=-40cm　　(b) y=110cm

図 5.1: 弱透視投影 (上段) と透視投影 (下段) での物体の見え方の違い

5.1.4 本研究のアプローチ

本研究では，透視投影のもとで 3 次元物体の認識と姿勢推定を同時に行うことを目標とする．これを実現するための鍵となるのが，姿勢空間の分解と幾何制約を用いたフィルタリングである．そこでは，特徴の対応空間と姿勢空間を同時に探索する際の計算量の増大に対処するために，(1) 姿勢空間を分解して回転成分と並進成分を段階的に探索し，(2) 特徴フィルタにより対応空間を段階的に縮小する，という方策をとる．姿勢空間の分解は，画像平面に投影された線分の傾きがある条件のもとで並進不変となることを利用して行う．回転成分の探索では傾きが一致しうるモデル線分と画像線分を絞り込むが，これが強力な特徴フィルタとなり，並進成分探索での投票の効率と精度を向上させる．これはモデルと画像にまたがるフィルタであるが，画像側でも種々の特徴フィルタを通して対応空間を縮小する．

なお，本方式では，回転成分を離散化して全探索するため，計算量にかかる係数がかなり大きな数となる．しかし，自律移動ロボットが活動する屋内環境では，対象物体の姿勢範囲は限定されていることが多く，これを利用すると探索空間は小さくてすむ．したがって，このような制約がはたらく環境では，計算時間は十分現実的なものになる．

5.2 物体認識方式の概要

5.2.1 問題の定義

本方式は，ピンホールカメラによる透視投影モデルに基づいて，2 次元画像に写っている物体の認識とその姿勢を推定するものである．カメラの内部パラメータは既知であるとする．カメラ座標系の原点がレンズ中心であり，z 軸が光軸であり，xy 平面が画像平面に平行であるとする．カメラ座標系の点を $\mathbf{P} = (X, Y, Z)^T$ と表し，その画像平面への投影点を $\mathbf{P}^s = (x, y, f)^T$ と表す．f は焦点距離である．

物体は剛体であるとし，その形状は物体のローカル座標系で定義されるとする．物体座標系の点を $\mathbf{P}^m$ と表す．いま，カメラ座標系に対する物体座標系の姿勢を 3×3 回転行列 R と 3 次元並進ベクトル $\mathbf{T} = (t_x, t_y, t_z)$ で表すと，物体上の点 $\mathbf{P}^m$ はカメラ座標系の点 $\mathbf{P}$ に次のように変換される．

$$\mathbf{P} = R\mathbf{P}^m + \mathbf{T}. \tag{5.1}$$

回転行列 R はオイラー角 $E =< \theta, \phi, \psi >$ で決まるとする．次に，透視投影により，$\mathbf{P}$ は画像平面上の点 $\mathbf{P}^s$ に次のように投影される．

$$\mathbf{P}^s = (f\frac{X}{Z}, f\frac{Y}{Z}, f)^T. \tag{5.2}$$

本方式では，基本的には，モデルと入力画像の間で特徴点を照合することで物体認識を行う．すなわち，モデル特徴点の投影像と画像特徴点の位置が一致する物体姿勢とその時のモデル特徴点と画像特徴点の対応を求める．いま，モデル特徴点の集合を PP，画像特徴点の集合を QQ とし，PP から QQ への写像を $g \subset PP \times QQ$ とする[1]．次式の S が求める解である．

$$S = \{(E, \mathbf{T}, g) \mid \sum_{(\mathbf{P},\mathbf{q})\in g} \|\mathbf{P}^s - \mathbf{q}\| = 0\}. \tag{5.3}$$

ただし，P および P^s はモデル特徴点とその投影像，q は画像特徴点である．ここで，$\mathbf{P}^s$ は $\mathbf{P}$ の関数であり，その値はオイラー角 E と並進ベクトル $\mathbf{T}$ の 6 個のパラメータに依存する．

上記特徴点として，線分の交点を用いる．モデルは多面体なので，モデルエッジはすべて直線分である．モデル上の線分の交点は，縮退した場合を除けば，画像上でも線分の交点となる．入力画像では，エッジ抽出の後，各エッジはすべて直線分に分割されるものとする．

なお，特徴対応空間と姿勢空間の同時探索による計算量の増大に対処するために，線分の交点だけでなく，線分の傾きも利用する．後で述べるように，線分の傾きは姿勢空間を段階的に探索する際に重要な役割を演じる．

[1] m は部分写像である．対応する画像エッジをもたないモデルエッジがあってもよい．

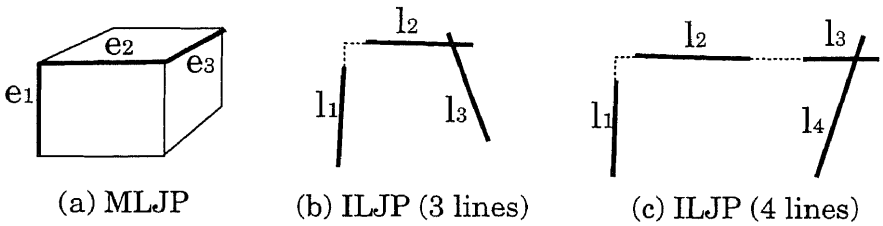
5.2.2 線分交差対を単位とした認識

上記の線分の傾きと交点をまとめて扱うために，**線分交差対**と呼ぶグループを考え，これを単位に認識と姿勢推定を行う．**線分交差**は2本の交差する線分の組であり，線分交差対は，2つの線分交差の組である．一般には線分交差対は4本の線分をもつことになるが，本方式では，本質的に3本の線分からなる線分交差対だけを扱う．

モデル側の線分交差対は，3本の連接するモデル線分とそれらの間の2つの交点からなる．図5.2(a)に例を示す．モデルは任意の3次元ワイヤーフレームで構成される．図5.2(a)では，線分e1と線分e2が1つの線分交差を，線分e2と線分e3が1つの線分交差をなし，合わせて，線分交差対を構成している．4本ではなく3本の線分に限定しているのは以下の理由による．すなわち，後述するように，線分交差対のもつ2つの交点は並進成分の計算に用いられるが，この2つの交点が同じ線分上にあるという制約を与えると，交点対応の候補数が大幅に減り，並進成分の投票の精度を上げることができるからである．

画像側の線分交差対は，3本あるいは4本の画像線分とそれらの間の2つの交点からなる． 図5.2(b)および(c)に例を示す．線分が直接交わらない場合でも，線分の外挿直線の交点が線分の端点から近ければ，線分交差対をなすと見なす(5.3.1節参照)．4本の線分からなる場合は，図5.2(c)にあるように，そのうちの2本は同一直線上にあるという制約をつける．この制約は，2つの交点が同じ線分上にあるという条件を満たしながら，かつ，オクルージョンなどによって画像線分が分断された場合にも対処できるためのものである．

以後，モデル線分交差対をMLJP (Model Line-Segment Junction Pair)，画像線分交差対をILJP (Image Line-Segment Junction Pair)と略す．



(a) MLJP　(b) ILJP (3 lines)　(c) ILJP (4 lines)

図 5.2: 線分交差対

5.2.3 認識アルゴリズムの流れ

本方式は，線分交差対を用いて姿勢空間を回転成分と並進成分に分解しながら，段階的に照合を行う．処理のおおまかな流れを図5.3に示す．

(1) 画像特徴フィルタによるILJPの絞り込み
画像レベルで幾何制約による特徴フィルタをかけて，モデルと対応する可能性の小さいILJPを取り除く．

(2) 回転成分の探索
本方式では，回転成分を離散化して全探索する．各回転成分値において，MLJPとILJPの各組合せに対して，その各線分の傾きが一致しうるかどうかを調べる．傾きが一致する個数が所定の閾値を越えれば，そのときのMLJPとILJPを用いて，並進成分の探索を行う．

(3) 並進成分の探索
回転成分の探索で得られたMLJPとILJPの各対応について，線分交差対の2つの交点から並進成分を計算する．得られた並進成分値に対して投票を行う．得票数が所定の閾値を超えれば，そのときの並進成分値と回転成分値を用いて，線分対応の検証を行う．

(4) 線分対応の検証によるスコアリング
得られた回転成分値と並進成分値に対するモデルの投影像を計算し，投影されたモデル線分が画像線分と実際に一致しているかを調べる．ここでの一致線分数と並進成分探索での得票数の和をスコアとし，スコアが所定の閾値を超えたものを最終的な解候補とする．

次章で，これら各処理について説明する．

5.3 認識アルゴリズム

5.3.1 画像線分交差対のフィルタリング

5.3.5節で述べるように，入力画像から単純にILJPを作ると，その個数は$O(N^4)$となり，計算時間に大きな影響を与える．一般に，特徴を絞り込むには幾何制約を用いた特徴フィルタリングが有効である[25, 26, 54]．本方式でも，種々の特徴フィルタを通すことでILJPを削減する．主なフィルタを図5.4に示す．いずれも基本的なものであり，処理も軽い．

- 線分長によるフィルタ
どちから一方の線分が所定の閾値より短いと，その線分交差は作らない．これは，エッジ抽出で生じたノイズエッジを除くのに有効である．
- 角度によるフィルタ
2つの線分のなす角が所定の閾値より小さいと，その線分交差は作らない．平行に近

```
-----------------------------------------------
 画像特徴フィルタにより ILJP を絞り込む;
 for (i=1; i<=回転成分離散値の個数; i++) {
   E = 回転成分の i 番目の離散値;
   C = E で傾きが一致しうる MLJP と ILJP の対の集合;
   if (C に含まれる MLJP 数 >= 閾値 1) {
     C の各要素と E から並進成分を計算して投票する;
     L = 得票数が閾値 2 を超えた並進成分の集合;
     for (j=1; j<=L の要素数; j++) {
       v1 = L の j 番目の要素の得票数;
       T = L の j 番目の要素の並進値;
       v2 = (E,T) を姿勢値としたときに，一致する
            画像線分をもつモデル線分の本数;
       if (v1+v2 >= 閾値 3)
         (E,T) を解候補に加える;  } } }
-----------------------------------------------
```

図 5.3: 認識アルゴリズムの流れ

い線分の交点はその位置誤差が大きくなるが，そのような線分交差を除くためにこのフィルタを用いる．

- 近接性によるフィルタ
  線分の交点と各線分の端点の距離が所定の閾値より離れている場合，その線分交差は作らない．画像上で交点と端点が大きく離れた線分は，モデル上で交差している可能性は小さいという仮定に基づいている．
- 同一直線性によるフィルタ
  2 つの線分交差から線分交差対を作る際に，一方の線分交差のどちらかの線分が他方の線分交差のどちらかの線分と同一直線にない場合，その線分交差対は作らない．5.2.2 節で述べたように，このフィルタは，線分の分断に対処しながら ILJP の候補数を減らすのに非常に有効である．

実験では，これらのフィルタにより ILJP の個数が実質的に $O(N)$ に近くなるという結果が得られた．

5.3.2 回転成分の探索

回転成分の探索においては，まず回転成分を離散化し，各離散値に対して，モデルと画像とで線分交差対の各線分の傾きが一致しうるかどうかを調べる．離散化の細かさは求める

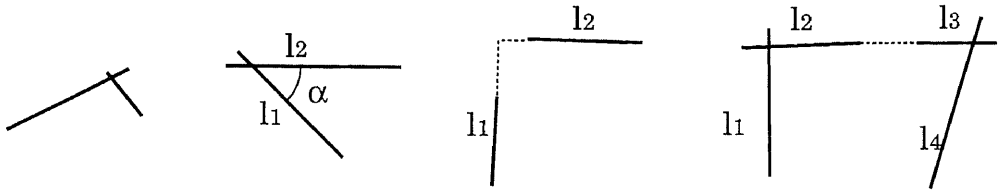


図 5.4: 特徴フィルタ

姿勢の精度に応じて決める．ここでのポイントは，次の命題を利用して，並進成分と独立に線分の傾きの照合を行うことである．

命題 1 モデル線分 e 上の点 $\mathbf{P}$ の画像平面上への投影点を $\mathbf{P}^s$ とする．モデルの回転成分を固定したとき，$\mathbf{P}^s$ が画像平面上で動かないモデルの並進に対して，e の画像平面上への投影像 e^s の傾きは不変である．

証明. 図 5.5からわかるように，$\mathbf{P}$，$\mathbf{P}^s$，および，レンズ中心 O は直線 k 上に載る．また，e，e^s，および，レンズ中心 O は平面 A 上に載る．ここで，$\mathbf{P}^s$ が画像平面上で動かないためには，$\mathbf{P}$ は直線 k 上に載っていなければならず，$\mathbf{P}$ の並進自由度は直線 k 上の移動しかない．ところが，回転成分を固定しているので，$\mathbf{P}$ が直線 k 上を移動する限り，e は平面 A の上に載っている．したがって，並進を行っても，e^s は必ず平面 A と画像平面の交線となり，その傾きは不変である． □

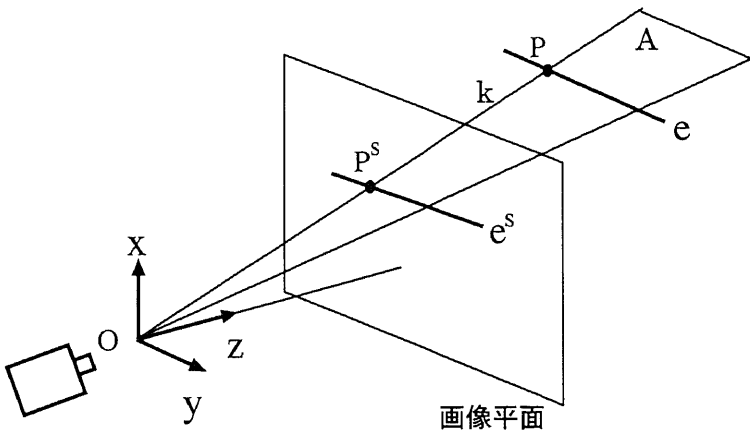


図 5.5: 投影線分の傾き

命題 1 により，投影点 $\mathbf{P}^s$ を画像線分 l 上の点 $\mathbf{q}$ に固定すれば，投影線分 e^s と画像線分 l の傾きが一致する回転成分を並進成分と独立に探索することができる．ここで，モデル線

分 e 上の任意の点は平面 A 上にあるので，どの点を P として用いても投影線分 e^s の傾きは変わらない．一方，画像線分 l 上の対応点 $\mathbf{q}$ をどう選ぶかで，投影線分 e^s の傾きは変わる．しかし，投影線分 e^s と画像線分 l が同一直線上にあるときは，画像線分 l も平面 A 上にあるため，$\mathbf{q}$ を画像線分 l 上のどこに選んでも，投影線分 e^s の傾きは変化しない．

上記のように，モデルの投影線分と画像線分とで 1 点が一致し，しかも，傾きが一致することは，投影線分と画像線分が同一直線上にあることを意味する．一方，投影線分と画像線分が同一直線上にあることは，モデル投影像と画像が一致することの必要条件である．したがって，上記の方法で投影線分と画像線分の同一直線性を調べることで，モデル線分と画像線分の対応候補を絞り込むことができる．

具体的には，同一直線性のチェックは以下のように行う．上記の議論により，一致させる点は任意でよいので，モデル線分，画像線分とも端点を一致点として用いる．いま，モデル線分 e の端点を $\mathbf{P}_1 = (X_1, Y_1, Z_1)^T$ および $\mathbf{P}_2 = (X_2, Y_2, Z_2)^T$ とし，画像線分の端点の 1 つを $\mathbf{q} = (u, v, f)^T$ とする．$\mathbf{P}_1^s$ と $\mathbf{q}$ を一致させるとすると，投影線分 e^s の傾き角 $\alpha(e^s)$ は以下のように計算できる．

$$\begin{aligned} \alpha(e^s) = \quad & atan2 \, (\quad f(Y_2 - Y_1) - v(Z_2 - Z_1), \\ & \qquad f(X_2 - X_1) - u(Z_2 - Z_1)). \end{aligned} \tag{5.4}$$

ただし，$atan2(y, x)$ は，$cos\alpha = x, sin\alpha = y$ を満たす角度 α を返す関数である．ここで，式 (5.1) より，

$$\mathbf{P}_2 - \mathbf{P}_1 \quad = \quad R(\mathbf{P}_2^m - \mathbf{P}_1^m)$$

であるから，$\mathbf{P}_2 - \mathbf{P}_1 = (X_2 - X_1,\ Y_2 - Y_1,\ Z_2 - Z_1)^T$ は並進成分 T によらず，式 (5.4) は T を用いずに計算できる (これは，命題 1 の別の証明になっている)．式 (5.4) より $\mathbf{P}_1^s$ と q を一致させたときの投影線分の傾きを求めて，画像線分の傾きと比較すれば，モデルの投影線分と画像線分の同一直線性を調べることができる．

次に，線分単位の同一直線制約を線分交差対に拡張する．回転成分の各離散値に対して，以下の処理を行う．

(1) 上記の方法で，同一直線になりうるモデル線分と画像線分の対応を求める．この際，誤差に対処するため，傾きの比較にはある程度のマージンをとる．

(2) 同一直線制約を満たす MLJP と ILJP の対応を求める．ここで，MLJP を構成するすべての線分が ILJP を構成するどれかの線分と同一直線になり得れば，両交差対は対応すると定義する．このチェックは，(1) で求めたモデル線分と画像線分の対応を参照して行う．

(3) 1つでも対応するILJPをもつMLJPの個数を現在の回転角のスコアとする．そして，スコアが所定の閾値を越えた場合，この回転角を解の候補とし，そのときのMLJPとILJPの組を保存する．

この同一直線制約は3次元の幾何制約に基づいた強力な特徴フィルタであり，これによって回転成分空間と対応空間の両方が小さくなる．5.4節で述べる実際の屋内環境の画像を対象とした実験では，回転成分空間の候補は約1/5に減った．また，モデル線分1本当たりに対応する画像線分の本数は約1/10に，MLJP1個当たりに対応するILJP候補の個数は約1/100に削減された．

5.3.3 並進成分の探索

MLJPの交点とILJPの交点の対応を特徴点対応として用いて，並進成分を探索する．回転成分を固定すると，並進成分は2組の正しい特徴点対応があれば，線形計算で求めることができる．幾何学的には，図5.6のようになる．まず，正しい特徴点対応が1組わかっている場合を考える．モデル上の点$\mathbf{P}$の投影点$\mathbf{P}^s$と画像エッジ上の点$\mathbf{q}$が一致するとすると，前述のように，$\mathbf{P}$の移動は直線k上に制約される．$\mathbf{P}$はモデル上の点であるので，$\mathbf{P}$が直線k上にあるとき，モデル座標系は直線kを$R\mathbf{P}^m$だけ平行にシフトした直線k'上にある．$q=(x,y,f)^T$とすると，kの方向ベクトルは$\vec{k}=(x,y,f)^T$の定数倍となるので，直線k'は次式で与えられる．

$$\mathbf{P}' = \vec{k}s - R\mathbf{P}^m. \tag{5.5}$$

ここで，$\mathbf{P}'$はk'上の点であり，sはスカラーのパラメータである．

次に，正しい特徴点対応が2組得られた場合を考える．モデル上の2点$\mathbf{P}_1, \mathbf{P}_2$に対して，それぞれ画像上の点$\mathbf{q}_1, \mathbf{q}_2$が対応するとする．$\mathbf{P}_1$と$\mathbf{q}_1$の対応に対してモデル座標系が載る直線を$k_1'$，$\mathbf{P}_2$と$\mathbf{q}_2$の対応に対してモデル座標系が載る直線を$k_2'$とすると，並進成分は$k_1'$と$k_2'$の交点として求めることができる．$\mathbf{q}_1=(u_1,v_1,f)^T, \mathbf{q}_2=(u_2,v_2,f)^T$とおくと，その解は以下のとおりである．

$$T = FR(\mathbf{P}_2^m - \mathbf{P}_1^m) - R\mathbf{P}_1^m, \tag{5.6}$$

$$F = \frac{1}{u_2-u_1}\begin{pmatrix} u_1 & 0 & -\frac{u_1u_2}{f} \\ v_1 & 0 & -\frac{v_1u_2}{f} \\ f & 0 & -u_2 \end{pmatrix}. \tag{5.7}$$

特徴点の対応が既知の場合は，上式によって解を求めずに，多くの対応点の組から最小自乗法で最適な解を求めることが行われる[31, 44]．しかし，本方式で対象とする問題では特徴点の対応が未知であるため，並進成分空間での投票を用いて解を求める．すなわち，前節の回転成分探索で求めたMLJPとILJPの対応の全候補に対して上記解を計算し，投票

を行って得票数の多い並進成分を解候補とする．この投票においては，同じ姿勢においてMLJP に対応する ILJP 数が複数個ある場合でも，1 票しか投票しないことにする．これにより，1 つのビン内の得票数は，そのビンが示す姿勢において少なくとも 1 個の ILJP と対応する MLJP の個数になる．

なお，投票を並進成分全体に対して行うと 3 次元の投票になり，精度が悪くなる．そのため，実装においては，1 つの並進成分について投票を行って候補を絞った後，残りの 2 成分はクラスタリングにより求める．そして，対応する ILJP を少なくとも 1 個もつ MLJP の個数をそのクラスタ (すなわち姿勢) のスコアとし，スコアの高いクラスタを解の候補とする．

ここで重要なことの 1 つは，2 組の特徴点対応で 1 回の投票ができることである．これにより，特徴点の組み合わせが少なくてすみ，計算効率がよくなるとともに，誤投票の割合も少なくなる．もう 1 つは，回転成分の探索ですでに MLJP と ILJP の対応候補が絞り込まれているため，やはり誤投票が少なくなり，得票のピークが出やすいことである．

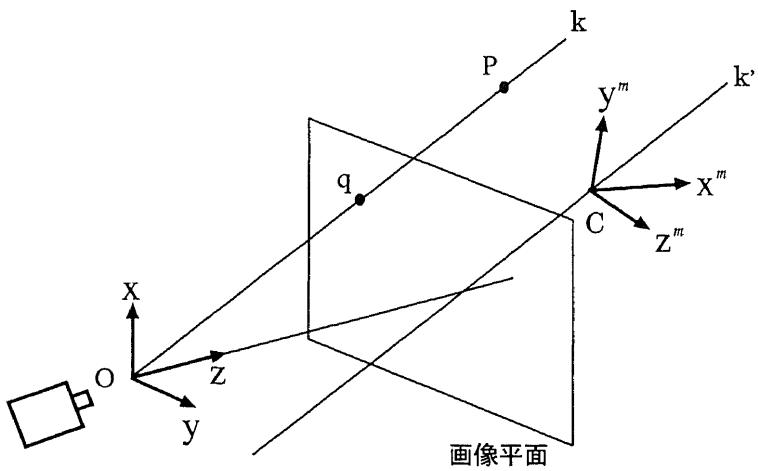


図 5.6: モデル座標系がなす軌跡 k'

5.3.4 線分対応の検証によるスコアリング

検証段階では，得られた各候補に対して，その姿勢値からモデル投影像を計算し，投影線分が画像線分と実際に一致するかを調べる．これは，両線分が同一直線上にあり，かつ，重なり合う部分をもつかどうかによって判定する．そして，少なくとも 1 本の画像線分と対応するモデル線分の本数をここでのスコアとする．

最終的なスコアは，ここでのスコア (一致するモデル線分数) と並進成分探索でのスコア (一致する MLJP の個数) の和とする．このスコアが所定の閾値を超えたものを最終的な解候補とする．

5.3.5 計算量

モデル線分数を M，画像線分数を N とすると，候補生成 (回転成分の探索と並進成分の探索) の計算量は $O(MN^4)$ となる．MLJP を 1 本の線分を共有する 2 つの線分交差とみなせば，MLJP はその共有線分に隣接する線分から構成されると考えることができる．すると，MLJP 数は各共有線分ごとに少数しかなく，その総数はほぼモデルの線分数に比例すると考えてよい．ILJP は，基本的には，任意の 4 本の線分の組み合わせになるため N^4 に比例する．

線分対応の検証では各モデル線分に対して画像線分との対応を調べるため $O(MN)$ かかる．したがって，候補生成と検証を合わせた全体の計算量は $O(M^2N^5)$ となる．

5.3.1節で述べたように，ILJP 数は，特徴フィルタを通すことで大幅に削減できる．実験では，上記 $O(N^4)$ の項は実質的に $O(N)$ 程度になるという結果が得られた．したがって，全体の計算量は実質的に $O(M^2N^2)$ 程度になると考えられる．

なお，回転成分を離散化して全探索しているため，その区画数が上記計算量の係数として乗算される．これは，かなり大きな数となるが，定数であり組み合わせでは効いてこない．また，移動ロボットが活動する屋内環境では，対象物体は水平な床や机の上に置かれていることが多く，姿勢範囲が制約されて探索空間は小さくなる．また，ロボットが地図を持っていれば，物体の姿勢がある程度予測できるため，さらに探索空間は小さくなりうる．このような制約がはたらく環境では，計算時間は十分現実的なものになる．これについては，第 7 章で述べる．

5.4 基礎実験による本方式の有効性の検証

5.4.1 実験条件

実験は，筑波大学校舎内の物体をデジタルカメラで撮影して行った．画像は 320 × 240 に縮小して使った．計算機はノート PC(CPU : PentiumII 400MHz，メモリ 96MB)，OS は Linux，言語は Java を用いた．

実験で用いた物体は，deskG，deskW，door，elevator である．モデルは 3 次元ワイヤーフレームで構成した．図 5.7に使用したモデルを示す．モデルは認識に有用な線分だけを持ち，必ずしも物体の形状を完全に表現するわけではない．たとえば，机は，通常の視点からはおもに前面部分しか見えないため，机モデルは前面部分の線分だけから構成されるものに簡略化した．線分数は，deskG が 13，deskW が 15，door が 21，elevator が 15 である．また，MLJP の個数は，deskG が 21，deskW が 29，door が 25，elevator が 16 である．

物体姿勢の回転成分について，以下のような制約を与えた．まず，各物体は水平な床面に垂直に置かれていると仮定した．すると，物体の姿勢自由度は垂直軸回りの回転の 1 自由度となる．これに，カメラのパンチルトの 2 自由度を併せても，全体では 2 自由度となる．

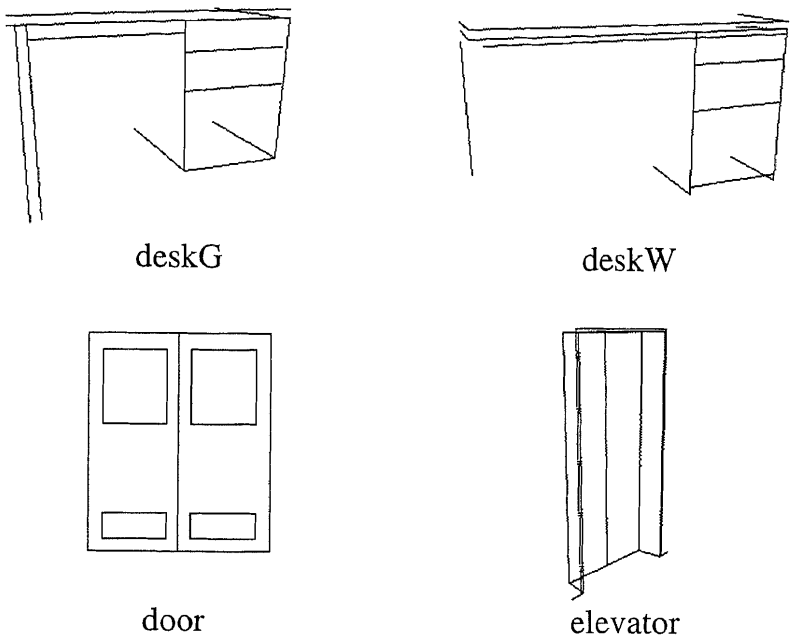


図 5.7: モデル

このため，回転成分の探索はオイラー角の 2 つの角 ψ(スピン) と ϕ(チルト) に対して行った．また，探索においては，ψ は-60〜60 度，ϕ は-30〜30 度の区間に限定し，離散化幅 5 度で区切った．並進成分には制約は何も与えず，並進成分の投票用のビンの幅は 10cm とした．

5.4.2 実験結果

実験では，前述の 4 つの物体のいずれかが写っている 60 枚の画像を用いた．システムの最終目的は，目標物体が画像内に存在するかどうかを判定し，存在すればその姿勢を求めることであるが，今回はさしあたって，目標物体は画像内に存在するという前提で実験を行った．すなわち，実験プログラムは，探すべき目標物体とその目標物体が写っている画像を入力として，目標物体のモデル線分と画像線分の対応を求め，目標物体のカメラ座標系での姿勢を推定するという処理を行った．この場合でも対応空間と姿勢空間の同時探索が必要であり，本方式の検証実験として意味がある．

実験結果を表 5.1に示す．システムが出力した解候補のうちスコアが 1 位の姿勢が実際に正解であった率は平均で約 72%であり，スコア 3 位以内に正解が入っていた率は約 96%である．ただし，この中には，スコアが同じになって正解と不正解の姿勢が同順位であったものも含んでいる．なお，処理時間は 2〜12 秒であった．線分抽出などの画像前処理には，これ以外に 2〜5 秒の処理時間を要した．

実験結果のいくつかについて，推定された姿勢に基づいてモデル投影像を原画像に重ね

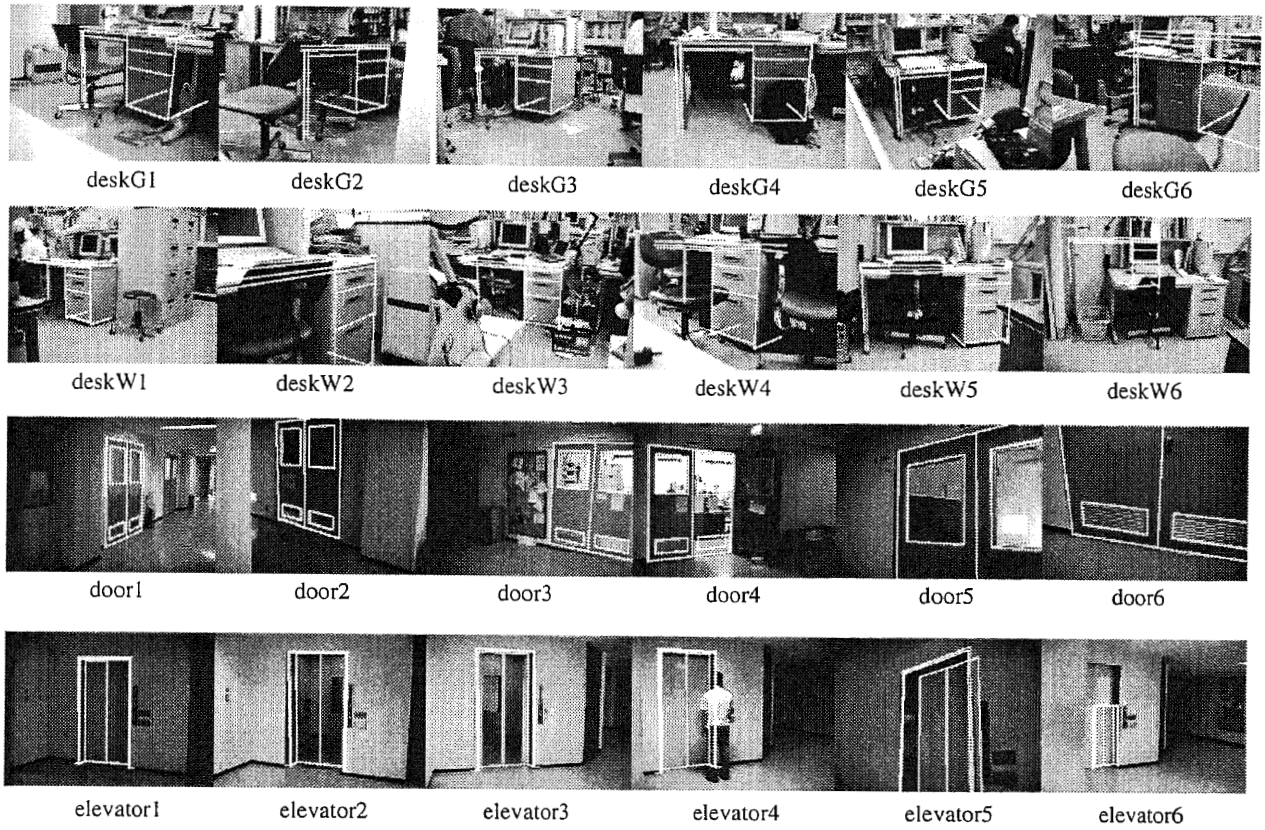


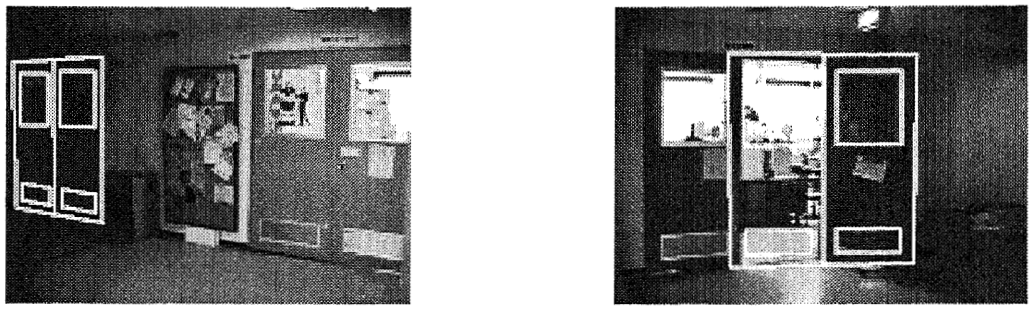
図 5.8: 認識結果

合わせたものを図 5.8に示す．同図で，deskG6，deskW6，elevator6 は認識失敗例であり，それ以外は成功例である．図 5.8から，照明変化，オクルージョン，混雑環境下でもよく認識できていることがわかる．たとえば，照明については，door2 や elevator1 は，他の door 画像や elevator 画像と比べて，直接照明を消してやや暗い状態でとった画像であるが，正しく認識できた．また，オクルージョンについては，deskG2，deskG4，deskW3，deskW4，door4，elevator4 が物体の一部が隠れた顕著な例であるが，正しく認識できている．また，deskW2，door5，door6，elevator5 のようにカメラの視界から物体の一部がはずれた場合も認識できている．このような場合も認識できることは，移動ロボットへの応用にとって好ましい．ただし，deskG6 や elevator6(柱の陰で半分隠れている) のように隠れた部分が多すぎると誤認識を起こす．deskG3，deskG5，deskW3 は混雑環境の例であるが，正しく認識できている．これらの画像では線分数が 300 本を越え (長さ 10 未満の線分は除外している)，ILJP 数は 2000 個程度である．ただし，deskW6 のように，あまり混雑していなくても，モデルと偶然マッチする線分の組み合わせがあると，正解よりもそちらのスコアの方が高くなることがある．これに対処するには，色や領域など線分以外の情報を用いる必要がある．

方式の原理から当然ではあるが，1 枚の画像に同じ物体が複数個あると，それぞれが検

表 5.1: 実験結果

Target	正解が 1 位の個数	正解が 3 位以内の個数
deskG	10/14	13/14
deskW	10/15	14/15
door	13/16	16/16
elevator	10/15	14/15
total	43/60	58/60



door3b　　door4b

図 5.9: 同一物体が複数個ある場合の認識

出される．ただし，それぞれに別のスコアがつけられるため，検出の強さは異なる．例を図5.9に示す．door3b は図 5.8の door3 の奥のドアを検出したものである．手前のドアがスコア 1 位であるのに対し，奥のドアは照明が暗いため 3 位であった．なお，スコア 2 位は手前のドアの姿勢がわずかにずれたものであった．door4b は図 5.8の door4 の開いた方のドア板を検出したものである．閉じたドア板 2 枚を door モデルとして定義したため，このような結果になったが，ドア板の可動範囲を定義すれば，開いたドアとして認識することも可能である (第 7章参照)．

スコア 1 位の解が正解でない場合には，大きく 2 つのパターンがある．1 つは，正解に近いがわずかに姿勢がずれており，スコアが下位のものの方が正解に近い場合である．この場合，スコア上位には類似した姿勢の解が並ぶ傾向がある．このタイプのエラーは，クラスタリングの精密化によって改善できる可能性がある．もう 1 つは，正解とまったく異なる姿勢のものがスコア 1 位となる場合である．この場合は，スコア上位には様々な姿勢の解がランダムに並び，しかも，スコア自体も比較的低いという傾向がある．これは，目標物体の特徴がうまく抽出できず，しかも，他の物体の特徴が偶然マッチした場合に起こると考えられる．

本方式は，特徴として線分だけを用いているが，色や領域など他の情報も併用してフィ

ルタリングを強化すれば，認識率を高めることが可能である．第7章で，エッジの色情報を用いたり，並進成分の制約を加えることで，認識率を高める例を示す．また，この実験では手で持ったデジタルカメラで画像を撮影しているため，カメラの姿勢が多少傾いたものがある．この場合，カメラ姿勢の回転成分にパンチルト角以外の成分も含まれてしまい，目標物体が水平な床面に置かれていても，その回転成分の自由度が2にならないということが起こる．このことが，スコア1位の認識率が約72%とやや低めである原因の1つとなっている．第7章では，ロボットにカメラを載せて，カメラ姿勢の誤差が少ない状態で画像を撮影し，認識実験を行った結果を示す．

5.4.3 特徴フィルタの効果

5.3.1節で述べた特徴フィルタの効果を確認するために，画像線分数と画像線分交差数の関係，および，画像線分数とILJP数の関係を調べた．画像線分数と画像線分交差数の関係を図5.10に示す．画像線分数の増加に対して画像線分交差数はほぼ線形に増えていることがわかる．また，画像線分数とILJP数の関係を図5.11に示す．ILJP数は分散が大きいが，N^4(Nは画像線分数)に比例しているようには見えず，むしろほぼ線形であるように見える．これより，5.3.5節で述べたように，最悪ケースの計算量は$O(M^2N^5)$であるが，実質的には$O(M^2N^2)$程度に見なせると考えられる．

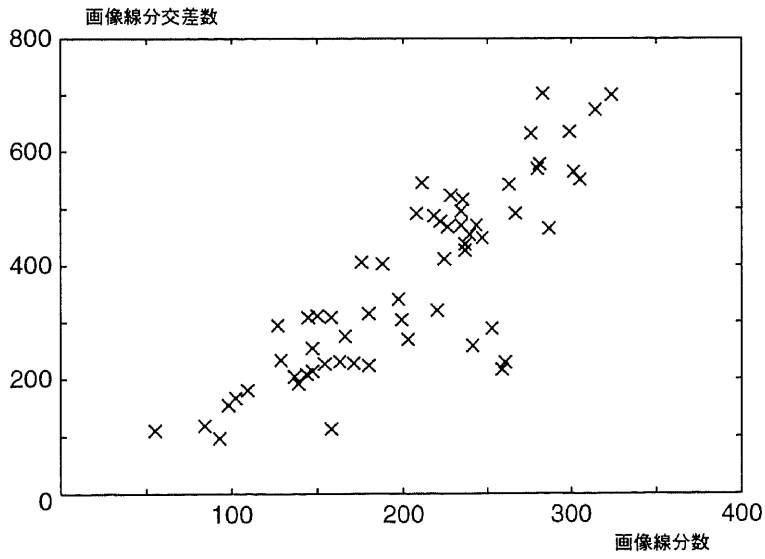


図 5.10: 画像線分数と画像線分交差数の関係

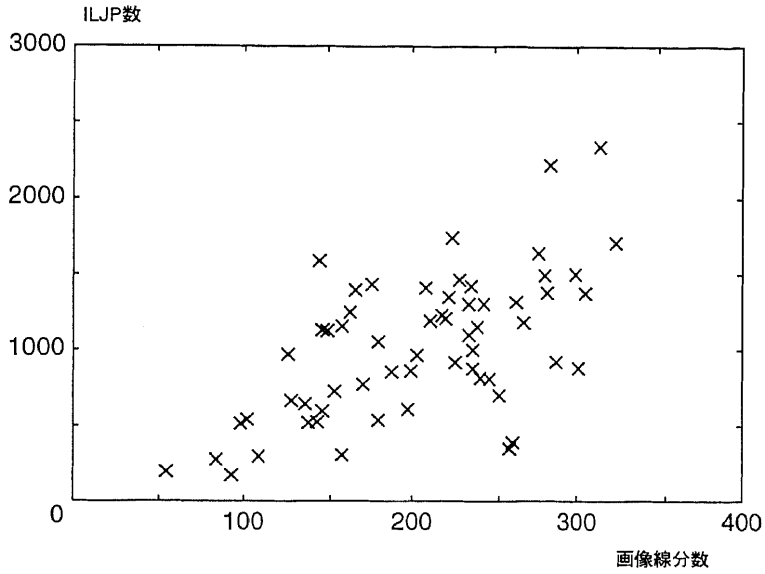


図 5.11: 画像線分数と ILJP 数の関係

5.4.4 画像前処理の影響

原画像からの線分抽出の様子を図5.12および図5.13に示す．エッジ抽出にはCannyオペレータを用いた [10]．抽出されたエッジを曲率変化の大きい点で分割することで，画像線分を求めた．実験では，60 枚の画像に対して，これらの処理のパラメータ値 (エッジ抽出閾値や曲率変化閾値など) は，すべて同じ値に固定した．そのため，照明条件の違いによりノイズエッジが多数とれる場合もあったが，線分長フィルタにより大部分が除かれ，認識への影響は少なかった．この例を図5.13に示す．これは図5.8のelevator1の画像であるが，照明が暗いため，エッジ抽出において大量のノイズエッジが発生している．しかし，線分分割後の線分長フィルタで長さ 10 未満の線分を除去することで，図5.13右のようにノイズの大部分が除かれている．ノイズはまだ多少残っているが，ランダムなものが多いため，その後の特徴フィルタでさらに除去され，最終的に正しい認識結果が得られた．

また，いくつかの画像では，コントラストが弱いためにエッジがかすれて部分的に検出できないことがあったが，オクルージョンと同様に一部の特徴が欠けたまま対処され，正しく認識されることが多かった．

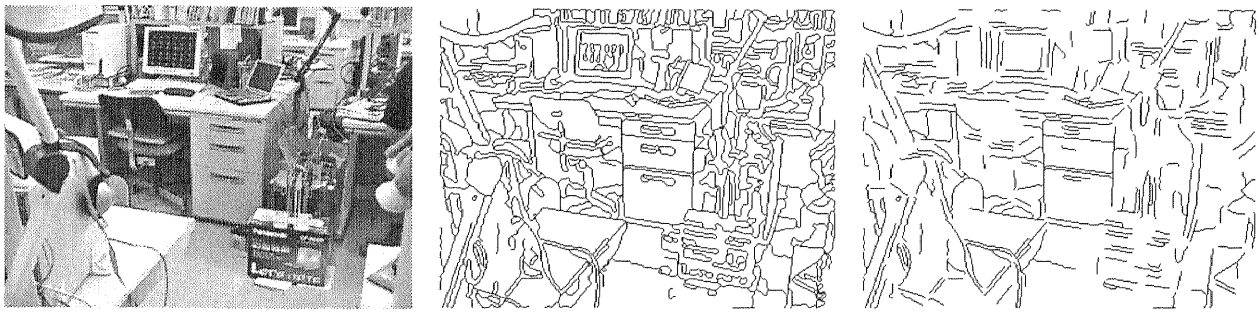


図 5.12: 線分抽出例 (正常照明)

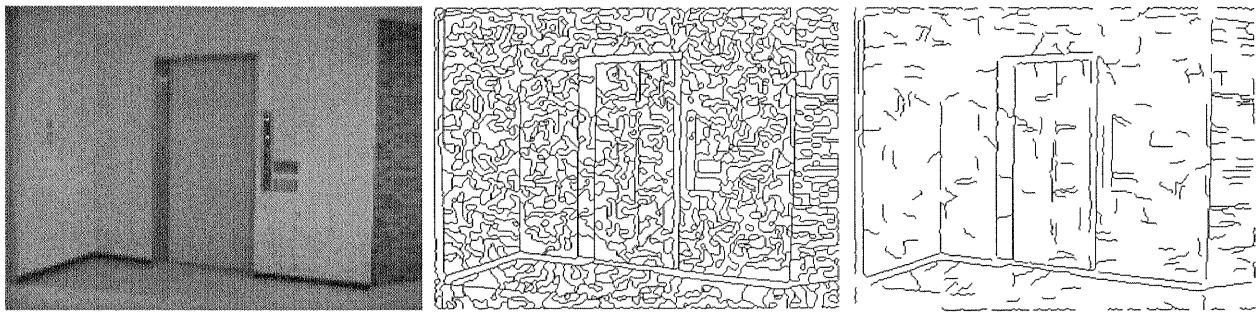


図 5.13: 線分抽出例 (弱照明)

第 6 章

ナビゲーション方式

本章では，これまで説明してきたTGマップでの自己位置推定と単眼ビジョンによる環境認識を組み合わせて，経路に沿って目的地まで走行する方式を述べる．TGマップは不正確さを許すため，その上の走行経路も不正確であり，ロボットが走行時に経路の位置を修正する必要がある．このため，本章では，これまで説明した物体認識と自己位置推定を統合して，環境を認識しながら自己位置を修正し，それに合わせて経路を修正して走行する方式を提案する．

走行経路は，将来的には経路計画機能によって自動生成するが，現状では人間が作成してロボットに与える．

6.1 従来研究と本研究のアプローチ

6.1.1 従来研究

移動ロボットの多くのナビゲーション方式では，まずマップを用いて目的地までの経路を決め，それから，その経路に沿って走行する．その際，経路からはずれないように，適当な地点でランドマークを観測して自己位置を修正する．また，予期しない障害物に衝突しないように，センサで前方監視をする．この他，なめらかに経路に追従するための走行制御方式も重要である．このようにナビゲーションには多くの要素が含まれるが，本章では，とくに走行経路に焦点を当てる．

走行経路の求め方は，用いるマップの種類に依存する．幾何マップでは，ロボットの走行可能領域が2次元で表現されるので，その領域内で実際にロボットが追従して走る線を計算する必要がある．この方法として，可視グラフを用いる方法 [53, 57]，ボロノイ図を用いる方法 [78, 81]，ポテンシャル場のアナロジーを用いる方法 [43] などがある．とくに，前二者は，まず，ロボットが走行可能な線からなるグラフを生成し，次に，目的地に到達する経路をそのグラフから探索する．得られた経路はロボットの大きさが考慮されており，環境変

化や予期しない障害物がない限り，物理的に走行できるものになっている．一方，トポロジカルマップの場合は，マップ自体が経路を表すグラフになっているので，経路計画では，単に目的地に到達する経路を探索するだけでよい [47, 59]．ただし，これで得られた経路はトポロジカルな経路なので，実際に走行する線は実行時に決める必要がある．

これらの中間的な方法として，あらかじめ正確な幾何情報をもった経路地図を与えておく方法がある [65]．すなわち，幾何マップの走行可能領域を通る線をつないだグラフを人手で与えておき，経路計画ではグラフ探索だけを行う．幾何マップができていれば，その上に経路を人手で与えるのはそれほど工数はかからない．この方法は，経路計画の計算量が少なく，しかも，物理的に走行可能な経路が直接的に得られる利点がある．

マップに不確実性がある場合や環境が変化する場合は，経路計画は複雑になる．この場合，作成された経路が実際に通れるかどうかあらかじめわからないため，走行と経路計画を並行して行う必要がある．また，環境を認識して経路が通れるか調べることにもコストがかかるため，走行距離などのコストの他に，認識コストも考慮して最適な経路を求める方法が提案されている [60]．そこでは，ランドマーク観測点の決め方が本質的に重要となる．

6.1.2 本研究のアプローチ

TG マップは不正確さを許すため，走行経路に関して様々な問題をもつ．まず，物体の位置が不正確なため，経路の位置も不正確になる．TG マップで得た経路にそのまま沿って走行しても，物体の位置がずれているため，実際には通れなかったり，物体と衝突したりする可能性がある．このため，本質的に走行と経路計画を並行して行う必要がある．すなわち，走行時に環境を認識して，そのときの状況にあうように経路の修正をする．この経路修正には，以下の 2 つの種類がある．

- トポロジーレベルの修正
  マップ上では通れるはずの領域が，実環境では物体の位置がずれているために通れない場合である．これは，経路を表すグラフのアークが削除されることに相当し，経路を修正するには，グラフの再探索が必要になる．

- 位置レベルの修正
  物体の位置がずれているため，経路の位置を修正しないと，距離が足りなかったり，衝突したりする場合である．トポロジーレベルでは通れるため，グラフの再探索は必要なく，経路の修正は位置の修正だけでよい．

前者はマップが正確であっても，予期しない障害物があれば起こり得る問題であり，多くのナビゲーション方式で検討されている．一方，後者は不正確さを許す TG マップで顕著に現れる問題である．そこで，本論文ではとくに後者の方式について検討する．

経路の位置修正において重要なのは，経路位置の表現とランドマークの観測である．まず，経路位置の表現については，自己位置推定と同じ問題が起こる．すなわち，マップが不正確な場合，走行経路をグローバル座標系で表すと実環境との整合がとれなくなり，いつか走行ができなくなる．また，ランドマーク観測においては，ランドマークとして指定した物体をビジョンで認識するため，その物体が画像内にきちんと写るような観測計画が必要である．しかし，マップが不正確なため，常に観測が成功するとは限らない．

これらの問題に対処するため，本研究では以下のアプローチをとる．

- 経路位置を基準実体の座標系で表す
  ロボット姿勢と同様に，経路位置も基準実体の座標系で表せば，基準実体の近傍では，実環境と整合のとれた経路が得られる．ここで，ロボット姿勢の基準実体と経路の基準実体は一致させる．

- 経路の基準実体をランドマークとする
  経路の基準実体を自己位置推定のランドマークとしても用いる．そうすると，ロボット姿勢の基準実体にするのにも整合性がよい．また，TG マップの実体間の相対姿勢から，ランドマークの存在範囲を計算して，観測計画を立てることができる．

- 走行時に物体の姿勢を確認して経路を修正する
  基準実体が切替わる区間は不正確である可能性が高い．そこで，その区間で重点的に次の基準実体 (ランドマーク) を観測して，経路位置を修正する．

このアプローチにより，物体の位置が不正確な場合でも，経路位置の修正を系統的に行うことができる．以下で，その具体的な方法を説明する．なお，本論文では，初期の走行経路は人手で作成してロボットに与えるとする．

6.2 走行経路

6.2.1 基本構成

走行経路は直線分の列で定義する．この直線分を経路線分と呼ぶ．経路線分は始点と終点の 2 点で定義される．始点，終点とも走行する床面上にあるものとする．走行経路の最後の経路線分を除けば，経路線分の終点は，次に続く経路線分の始点と一致する．各経路線分には，ロボットの走行速度を指定することができる．

経路線分の走行には，山彦の走行制御システムである Spur の直線追従コマンド spur_line_LC を用いる [36]．spur_line_LC は，ロボットの現在姿勢から指定した直線に漸近するように，なめらかな軌跡をたどる．このため，経路線分の列で経路を定義しておけば，なめらかな走

行を実現できる．ただし，経路線分の終点を乗り換え点にして，その点で次の経路線分への直線追従コマンドを実行すると，オーバーシュートが発生するため，乗り換え点を終点のやや手前に設定する．

前節で述べたように，TG マップ上のロボット姿勢は基準実体からの相対姿勢で表されるため，経路線分の位置も基準実体からの相対姿勢で表すと都合がよい．すなわち，経路線分の始点と終点の座標を，基準実体とそこからの相対姿勢で表すことにする．図 6.1にその例を示す．同図で，点 A～D の基準実体は e1，点 E～F の基準実体は e2 である．このようにすると，基準実体の周辺の経路線分は，その基準実体に対して正確な位置に作成できる．ロボットは，基準実体を認識してそこからの自己位置を推定し，その基準実体に対する経路線分に乗る．経路線分に正しく乗っている限り，ロボットは基準実体にぶつからずに走行することができる．

実体の中間にある区間では，経路線分は 2 つの基準実体にまたがることになる．この場合，経路線分は始点と終点で異なる基準実体をもつ．図 6.1の経路線分 DE は，その例である．このような基準実体をまたがる経路線分では，その走行中に基準実体を切替える処理が必要になる．これについては後で述べる．

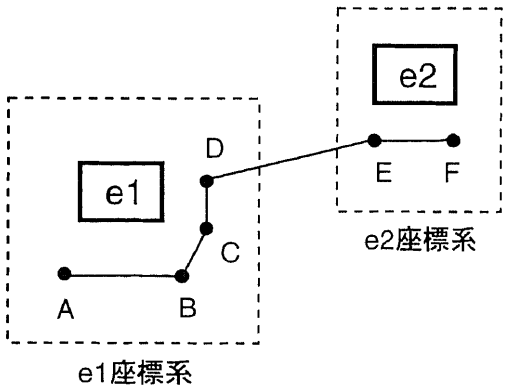


図 6.1: 走行経路

6.2.2 カメラ視線

ロボットがランドマークやタスクで扱う目標物体を見つけるためには，カメラが適切な方向に向いている必要がある．このためのカメラ操作の指示を与えるために，カメラ視線情報を各経路線分上に定義する．カメラ視線情報として，ナビゲーション用と一般タスク用の 2 種類がある．前者は，目標物体を見つけた後，ロボットの自己位置推定まで行う．後者は，目標物体を見つけるだけで，その結果はタスク側に渡される．

カメラ視線は，経路線分上の点を始点とし，目標物体付近の点を終点とする線分である．ただし，始点，終点とも高さをもつ．始点の高さにはロボットのカメラの高さを設定し，目標点の高さには目標物体を最もよく撮影できる高さを設定する．また，適切な大きさで目標物体を撮影するために，カメラのズーム係数を指定できる．図 6.2にカメラ視線の例を示す．同図で，a1～a5 がカメラ視線である．

カメラ視線情報には，そのカメラ視線での撮影によって認識すべき目標物体クラスを指定する．本研究で開発した物体認識方式は，画像中の任意の物体を認識するのではなく，指定した目標物体を見つけるものであるため，画像から探す物体のクラスをあらかじめ指定しておく必要がある．図 6.2の a1,a2 の目標物体は e1，a3,a4,a5 の目標物体は e2 である．ナビゲーション用のカメラ視線で目標物体として指定できるのは，そのカメラ視線がある経路線分の終点の基準実体のみであるとする．この制約は，基準実体の切替えのない経路線分では，現経路線分の基準実体の認識が目的であり，また，基準実体の切替えのある経路線分では次の経路線分の基準実体の認識が目的であることを意図している．図 6.2の a1,a2,a5 は前者，a3,a4 は後者の例である．なお，一般タスク用のカメラ視線では，任意の目標物体を指定できる．

ロボットは，経路線分の走行中にカメラ視線の始点に到達すると，経路線分の方向とカメラ視線の方向からカメラのパンチルト角を計算して，カメラをカメラ視線の終点に向ける．そして，画像を撮影して，指定された目標物体を画像内から見つける．なお，現状では，カメラ視線の処理においてロボットは一時停止する．原理的には，走行しながら撮影と認識を行うことも可能であるが，この場合は，認識中に次のカメラ視線までに到達した場合の対処，および，自己位置推定中に走行した分の補正 [58] などの処理が必要になる．

カメラ視線情報は，1 本の経路線分に任意個設定できる．TG マップの不正確さの度合の大きい領域では，目標物体がすぐには見つからない可能性が高いため，複数のカメラ視線を設定して何度もリトライできるようにする．これは，照明やオクルージョンなどの悪条件によって物体認識に失敗した場合にも有用である．

6.3 走行経路が与えられた場合の自己位置推定

第 4章で述べた自己位置推定法は，初期姿勢も経路も与えない条件下で考えられたものであった．そこでは，ロボットは目標物体を知らず，たまたまカメラで撮影された物体に基づいて自己位置推定を行うという問題設定であった．ところが，初期姿勢が与えられ，所与の走行経路に沿ってナビゲーションを行うという条件下では，目標物体を絞り込めるため，制約がない場合に比べて自己位置推定処理を軽減できる．

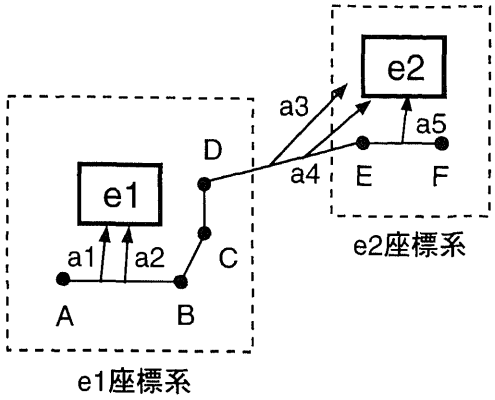


図 6.2: カメラ視線の例

6.3.1 自己位置推定の分類

ロボットは経路線分を走行中にカメラ視線の始点に到達すると，その目標物体の画像を撮影して認識し，自己位置推定を行う．このような自己位置推定は，ロボットの現在姿勢の基準実体，走行中の経路線分の始点の基準実体，終点の基準実体，カメラ視線の目標物体の4つの組み合わせによって分類できる．まず，6.2.2節のカメラ視線の定義から，カメラ視線の目標物体と，そのカメラ姿勢がある経路線分の終点の基準実体は一致する．また，走行中の経路線分の始点の基準実体とロボット姿勢の基準実体は常に一致するようにできる．実際，基準実体が切替わる経路線分で自己位置推定を行えば，次の経路線分の始点の基準実体とロボット姿勢の基準実体は同一になる．基準実体が切替わる経路線分で自己位置推定に失敗した場合でも，自己位置修正をしないまま，ロボット姿勢を強制的に次の経路線分の基準実体の座標系に変換すればよい．

以上のことから，走行経路が与えられた場合の自己位置推定は以下の2つに分類できる．

- 基準実体が切り替わる経路線分での自己位置推定
  経路線分の始点の基準実体とロボット姿勢の基準実体が同じであり，また，経路線分の終点の基準実体とカメラ視線の目標物体が同じである．

- 基準実体が切り替わらない経路線分での自己位置推定
  経路線分の始点および終点の基準実体，ロボット姿勢の基準実体，カメラ視線の目標物体がみな同じである．

これらについて，次節以降で順次説明する．

6.3.2 基準実体が切替わる経路線分での自己位置推定

始点と終点の基準実体が異なる経路線分上に設定されたカメラ視線によって画像を撮影し，目標物体を認識して自己位置推定を行う場合を考える．TG マップでは実体間の相対姿勢は不正確でよいため，このような基準実体をまたがる経路線分は，長さも方向も正確である保証はない．このため，この経路線分でのロボットの自己位置推定は本ナビゲーション方式において本質的に重要である．

以下に，自己位置推定の手順を示す．ここで，前述のカメラ視線の定義から，カメラ視線の目標物体は経路線分の終点の基準実体と同じである．また，これ以前の自己位置推定は成功しており，それによりロボット姿勢の基準実体は経路線分の始点の基準実体と同一になっているとする．

(1) 経路線分の走行

経路線分の始点と終点の基準実体が異なるので，始点と終点の位置の座標系が異なる．このため，終点の位置を始点の基準実体の座標系に変換してから走行する．

(2) カメラ操作

カメラ視線も経路線分の始点の基準実体の座標系に変換し，それに基づいてカメラを操作し，画像を撮影する．

(3) 自己位置推定

撮影した画像から目標物体を認識し，そのロボット座標系での姿勢を推定する．その物体姿勢に基づいて，第 4章で述べた自己位置推定アルゴリズムを適用して，ロボット姿勢を計算する．この時点で，ロボット姿勢の基準実体は目標物体，すなわち，経路線分の終点の基準実体に置き換えられる．

第 4章でのアルゴリズムでは，目標物体と同じクラスの実体すべてに対して確率計算をしていた．ところが，ロボットの現在姿勢と TG マップの相対姿勢アーク情報から，目標物体の存在範囲を計算することができる．すると，カメラ視線情報に基づいて撮影した画像にどの物体が写っているかを絞り込むことができ，確率計算を行う対象を大幅に減らすことができる．これは，経路が与えられた走行における自己位置推定の利点である．

(4) 経路線分の修正

得られたロボット姿勢を始点とし，元の経路線分の終点を終点とする経路線分を生成して，走行経路に挿入する (元の経路線分の後ろに追加する)．得られたロボット姿勢の基準実体は元の経路線分の終点の基準実体と同一であるから，この新しい経路線分では，始点と終点の基準実体は一致する．

この手順における自己位置推定と経路修正の例を図6.3に示す．同図で，(b) と (c) は不正確なマップであり，実体 e1 と e2 の間隔が (a) の実環境より狭い．同図 (b) は，実環境での認識よって得た実体 e2 からの相対姿勢と使って自己位置推定を行った結果である．同図 (c) は，自己位置推定で得たロボット姿勢を始点として，新しい経路線分を生成した結果である．この例では，自己位置修正によりロボットと実体 e1 が重なっているが，これはマップの表示上の問題であり，走行には支障がない．

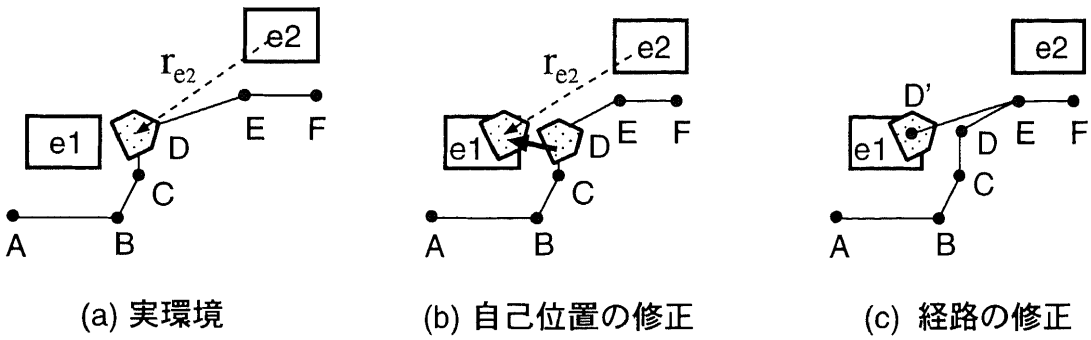


図 6.3: 自己位置推定と経路修正の例

6.3.3 基準実体が切替わらない経路線分での自己位置推定

始点と終点の基準実体が同一である経路線分上に設定されたカメラ視線によって画像を撮影し，目標物体を認識して自己位置推定を行う場合である．このような経路線分は基準実体に対してはほぼ正確な位置にあるので，この経路線分でのロボットの自己位置推定はロボット姿勢の調整のために行われる．

以下に，自己位置推定の手順を示す．ここで，前述のカメラ視線の定義から，カメラ視線の目標物体は経路線分の (始点および終点) の基準実体と同じである．また，ロボット姿勢の基準実体も経路線分の基準実体と同じである．

(1) 経路線分の走行
経路線分の座標系とロボット姿勢の座標系が同じなので，ロボットは経路線分にそのまましたがって走行する．

(2) カメラ操作
カメラ視線の座標系もロボット姿勢の座標系と同じなので，そのままカメラを操作し，画像を撮影する．

(3) 自己位置推定
撮影した画像から目標物体を認識し，そのロボット座標系での姿勢を推定する．次に

その物体姿勢に基づいて，自己位置推定アルゴリズムを適用して，ロボット姿勢を計算する．

ここで，ロボット姿勢の基準実体はすでに経路線分の基準実体と同一なので，基準実体の切替は必要ない．そのため，第4章の自己位置推定アルゴリズムは必要なく，従来のマルコフ姿勢推定を用いればよい．ただし，これは，目標物体の近くに同じクラスの物体がなく，画像に写った物体が目標物体であると判断できる場合に成り立つ．このためには，ロボットの現在姿勢とTGマップの相対姿勢アーク情報から，目標物体やそれと同クラスの物体の存在範囲を計算し，重なりがないかどうかを調べる必要がある．

(4) 経路線分の修正

得られたロボット姿勢を始点とし，元の経路線分の終点を終点とする経路線分を生成して，走行経路に挿入する (元の経路線分の後ろに追加する)．得られたロボット姿勢の基準実体は元の経路線分の基準実体と同一であるから，新しい経路線分でも始点と終点の基準実体は一致する．

6.3.4 目標物体の存在範囲

TGマップは不正確でよいため，マップ上で目標物体に向けてカメラ視線を設定しても，実環境ではその先に目標物体がなく，画像に写らない可能性がある．すると，ロボットは目標物体を見つけることができず，自己位置推定に失敗してしまう．これを防ぐには，目標物体の存在可能な範囲を求めて，その範囲を網羅するように複数のカメラ視線を設定することが考えられる．この様子を図6.4に示す．同図で，実体e2の存在可能範囲を楕円で表している．カメラ視線a1～a5はこの範囲をほぼ網羅しているので，a1～a5のどれかで実体e2の撮影に成功することが期待される．

目標物体の存在範囲は，TGマップでの実体間の相対姿勢の誤差を用いて計算できる．すなわち，現在のロボット姿勢の基準実体から目標物体までの相対姿勢アークをたどって，各アークの誤差を累積していけばよい．このとき，ロボット姿勢自体にも誤差があるので，この誤差も上記の誤差累積に含める必要がある．たとえば，図6.4で，ロボットが経路線分DEにいるときのロボット姿勢の基準実体がe1であるとすると，e1とe2の間に相対姿勢アークが張られていれば，ロボットから見たe2の存在範囲を計算できる．

目標物体と同じクラスの実体がマップ内に存在する場合，その実体の存在範囲も計算して，目標物体の存在範囲と重なりがないかどうかを調べる必要がある．重なりがある場合は，6.3.2節や6.3.3節で述べた自己位置推定において，目標物体とだけ融合するのではなく，重なりのある実体も融合対象にしなければならない．この場合は，第4章で述べた自己位置推

定アルゴリズムがそのまま適用でき，基準実体の同定とそこでの相対姿勢を同時に計算することになる．

この存在範囲によるカメラ視線の作成は，ロボットが走行時の観測計画で自動的に行うのが理想である．人手で走行経路とカメラ視線を作成する場合は，TG マップエディタにおいて，ロボットの現在姿勢から見た目標物体の存在範囲を表示できる機能があるとよい．そうすると，走行経路の作成者は，その存在範囲を網羅するようなカメラ視線を作成することができる．ただし，人手による経路作成時にはロボットの現在姿勢の誤差を知ることはできないため，適当な誤差を仮定して，上記計算を行うことになる．この機能は現在の TG マップエディタでは実装しておらず，残された課題である．

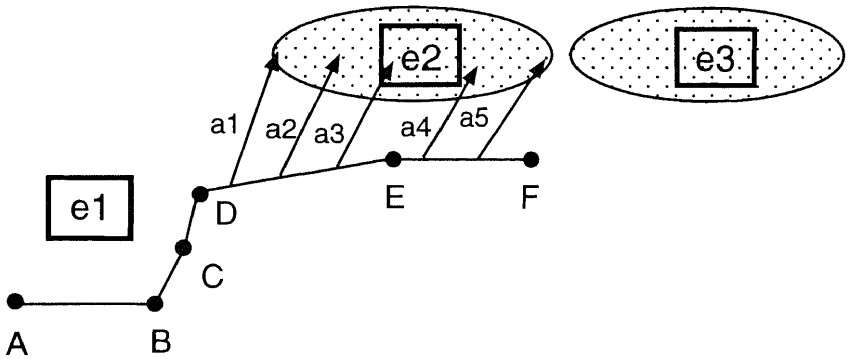


図 6.4: 目標物体の存在範囲とカメラ視線

6.4 走行経路に沿ったナビゲーション

6.4.1 山彦の座標系とロボット姿勢の対応

ロボット姿勢は基準実体とそこでの相対姿勢の対で表される．これに基づいて山彦を走行させるには，山彦の座標系へのマッピングが必要である．山彦の走行制御システムである Spur は，ロボット姿勢をグローバル座標 (GL 座標) と任意に設定したローカル座標 (LC 座標) の両方で表すことができる [37]．本方式では，これらの座標系を以下のように用いる．

- LC 座標
  山彦の LC 座標を，経路線分の基準実体のローカル座標系に対応させる．すると，山彦の LC 座標での姿勢が，ロボット姿勢の基準実体の座標系での相対姿勢にそのまま対応する．

  この対応づけには，Spur_set_pos_LC コマンドや Spur_set_LC_on_LC コマンドを用いる．

- GL 座標
  山彦の GL 座標を TG マップのグローバル座標系に対応させる．GL 座標での姿勢は，ロボットの走行軌跡として有用である．

6.4.2 ナビゲーションの手順

ナビゲーション手順を以下に示す．最初に (1) と (2) を行い，その後，(3) と (4) を繰り返して目標点まで移動する．

(1) 初期姿勢
走行経路の最初の経路線分の始点と進行方向で決まる初期姿勢にロボットを置く．ただし，正確な位置に置く必要はなく，大まかでよい．

(2) 初期基準実体の認識
最初の経路線分上のカメラ視線は，経路線分の基準実体，すなわち，ロボット初期姿勢の基準実体を認識するために設定しておく．ロボットは，そのカメラ視線の始点に到達すると，6.3.3節で述べた手順で自己位置と経路を修正する．

(3) 経路線分の走行
カメラ視線の始点か経路線分の終点に到達するまで，ロボットは経路線分の方向に進む．カメラ視線の始点に到達した場合は，6.3.2節あるいは6.3.3節で述べた手順で自己位置と経路を修正する．

(4) 経路線分の乗り換え

- 基準実体の切替えがない経路線分から次の経路線分への乗り換え
  Spur の直線追従コマンドにより，次の経路線分にそのまま乗り換える．
- 基準実体の切替えがある経路線分から次の経路線分への乗り換え
  次の基準実体を認識するためのカメラ視線が設定されている場合は，6.3.2節で述べた手順で自己位置と経路線分を修正する．新しい経路線分と次の経路線分では基準実体の切替えがないので，Spur の直線追従コマンドでそのまま乗り換えできる．

  次の基準実体を認識するためのカメラ視線が設定されていない場合は，自己位置修正は行わず，ロボット姿勢を次の経路線分の基準実体の座標系に直接座標変換する．これにより，ロボット姿勢の期待値は次の経路線分の始点に一致するが，その分散は新旧の基準実体間の相対姿勢誤差により拡大する．このような乗り換えは，新旧の基準実体間の相対姿勢誤差が小さい区間で行わないと，障害物にぶつかったりしてナビゲーションが失敗する可能性が高くなる．

6.5 基礎実験による本方式の有効性の検証

本ナビゲーション方式の有効性を検証するために，基礎実験を行った．実験環境は筑波大学の実験室で，ロボットは図6.6の走行経路を2つの机を認識して走行する．このTGマップでは，実環境と比べて机1と机2の相対姿勢に0.5〜1[m]程度の誤差がある．なお，走行経路は，走行経路エディタを用いて作成した．走行経路エディタは，TGマップの上にGUIから対話的に経路を作成するツールである．その詳細は，付録B.2節を参照のこと．

図6.5に机の認識結果を示す．図6.6に，ロボットの走行の様子を示す．実験は実機を用いて行ったが，同図に示す結果は，実験で得た画像を用いて，タスク実行用GUIであるタスクランチャのシミュレーションモードで再現したものである．タスクランチャについては，付録B.3節を参照のこと．

同図(a)は，ロボットが点Aを出発し，点Pまで走行して机1の画像を撮影したときのロボット姿勢である．初期姿勢の誤差を大きく設定してあるため，ロボット姿勢のサンプルがやや広く分布している．同図(b)は，机1を認識して自己位置修正を行った結果である．ロボット姿勢が点Pからずれており，ロボットはここから点Bをつなぐ新しい経路線分を生成して，同図(c)に示すようにその経路線分を走行した．同様に，机2の認識によって，同図(d)から(e)のようにロボット姿勢を修正し，そこから点Dに向かう新しい経路線分を生成して，同図(f)のようにその経路線分を走行した．

以上の基礎実験により，本ナビゲーション方式が有効に機能することが確認できた．より複雑な経路の走行実験については，第8章で述べる．

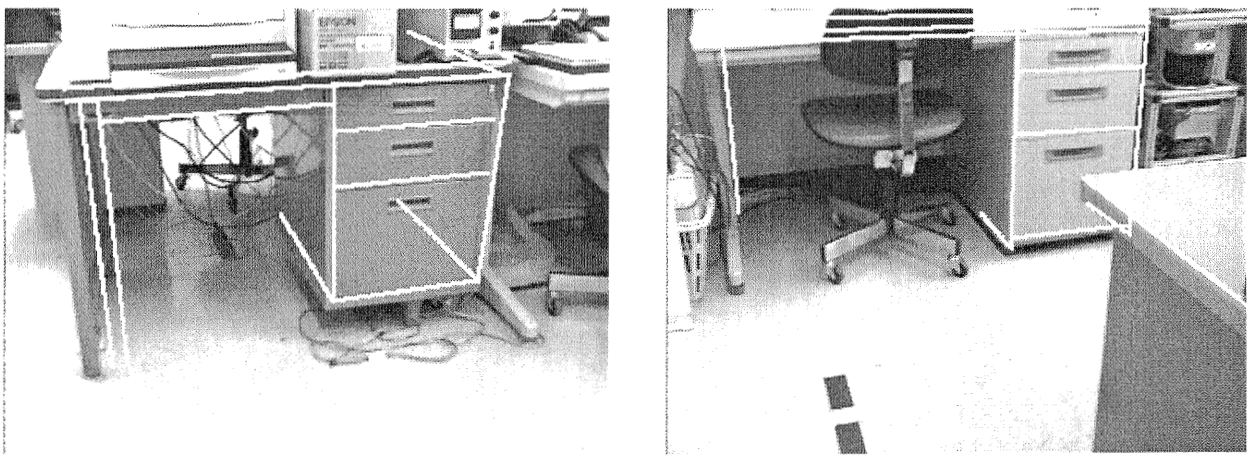
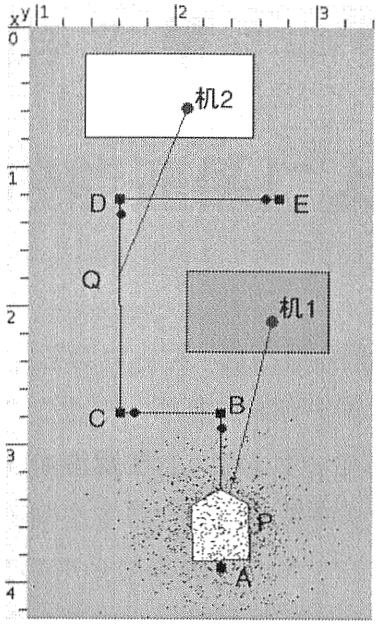
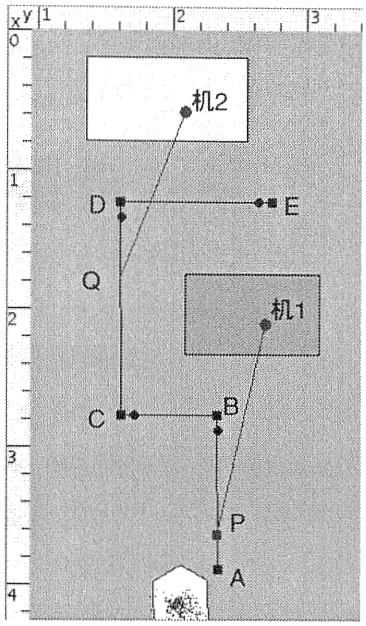


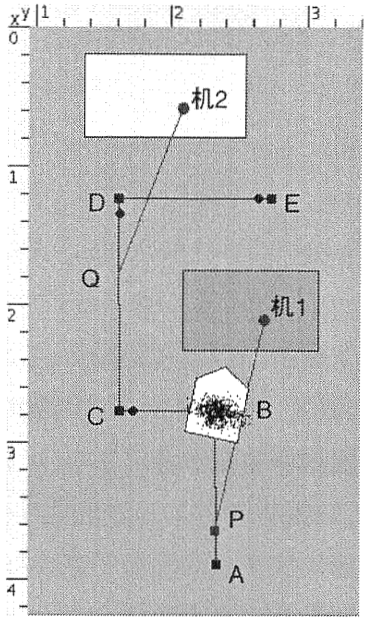
図6.5: 机1(左)と机2(右)の認識結果



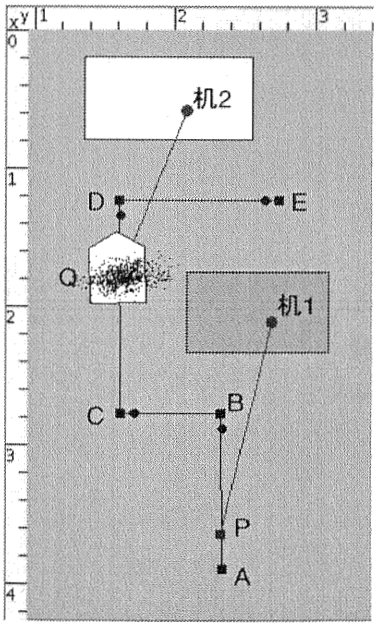
(a) 机1の認識前



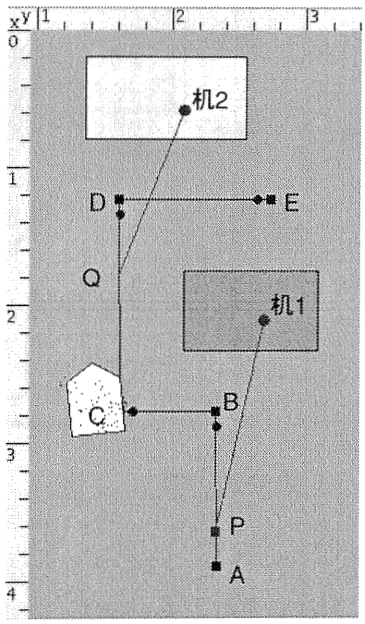
(b) 机1の認識による自己位置修正



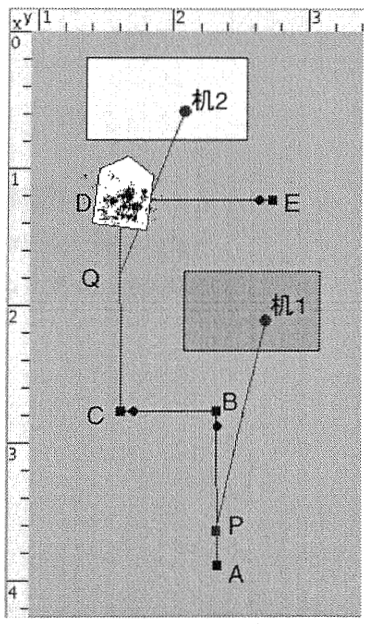
(c) 経路を修正して走行



(d) 机2の認識前



(e) 机2の認識による自己位置修正



(f) 経路を修正して走行

図 6.6: 実験結果

第 7 章

機能連係による相乗効果

TG マップ，自己位置推定，物体認識は互いに補完しあう．これらの機能が互いに連係することで，単独で実行するときよりも性能が向上する．この関係を図 7.1に示す．たとえば，物体認識では，TG マップからの姿勢制約や自己位置推定の結果によって，処理効率や精度が向上する．自己位置推定の精度は TG マップの精度や走行経路に依存し，これをもとに，TG マップ作成やランドマーク設定の指針が得られる．また，物体認識と自己位置推定の結果に基づいて物体のより正しい姿勢を求めて，TG マップを更新することもできる．本章では，これらについて実験結果を踏まえて説明する．

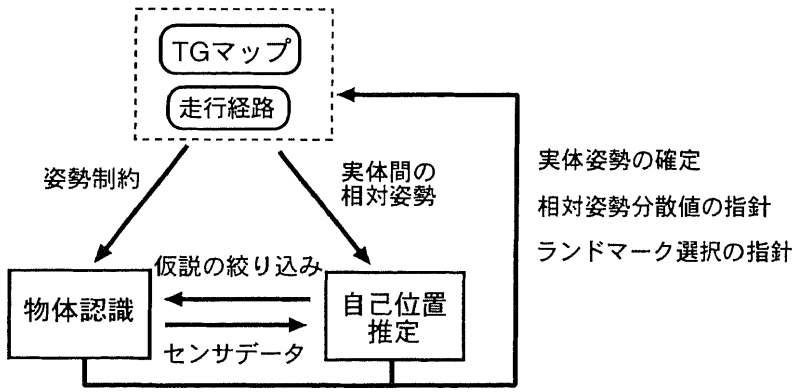


図 7.1: 機能の連係

7.1 物体認識の性能向上

7.1.1 基本方針

TG マップは不正確でよいため，ナビゲーションにおいて誤った自己位置推定をすると，ロボットが迷ったり障害物に衝突するなどの可能性があるため，ランドマークの認識率は非

常に高いことが要求される．ところが，物体認識を単独で行った場合に，正解が1位になる率は第5章の基礎実験では67〜81%であり，ナビゲーションのためには十分とは言えない．しかも，ナビゲーション中は，移動障害物により目標物体が一時的に隠されたり，マップが不正確なために撮影した画像に目標物体が写っていない場合があったりするため，ナビゲーション中の実質的な認識率はもっと低くなる可能性がある．

これらの問題に対処するために，認識処理の過程で有望な解候補をできるだけ残しながら，さまざまなフィルタをかけて，ありえない候補を削除していくという戦略をとる．認識の誤りには，間違ったものを解と見なす場合 (false positive) と，正解を見落す場合 (false negative) がある．処理の過程で false negative になると，それ以降で正解を得る可能性は無くなるため，なるべく有望な候補は残すようにするのである．ただし，ほとんどの候補は false positive であるから，それを少しでも減らしていくことが必要である．

第5章で述べた物体認識も基本的にこの方針を採用しており，認識結果として多くの候補を出力できるようになっている．そこで，マップから得られる制約情報で物体認識の候補を絞り込むことで，物体認識の性能向上を図る．具体的には，ロボット姿勢と TG マップの記述から目標物体の姿勢範囲を予測し，それによる制約条件を用いて物体認識結果を絞り込む．これには，以下の2つの方法がある．

- 事前の姿勢制約による探索範囲の絞り込み
  姿勢制約により，あらかじめ探索範囲が明確に絞り込める場合である．とくに，本方式では，物体姿勢の回転成分がこれに適する．
- 事後の認識候補の絞り込み
  物体認識で候補を出力した後，ロボット姿勢と矛盾する姿勢にある候補を削除する．

両者は，前処理か後処理かが違うだけで，本質的には同じである．処理のしやすさに応じて使い分ける．

7.1.2 姿勢制約による探索範囲の絞り込み

物体認識においては物体の姿勢空間を探索するため，物体姿勢に関する事前知識があれば，探索範囲を狭めることができ，処理時間と認識率の両方に寄与する．本研究で開発した物体認識方式は，回転成分と並進成分を分解して探索するため，それぞれに対して独立に制約をきかせることができる．

利用できる事前知識には，物体が通常置かれる状態についてのデフォルト知識と TG マップの2種類がある．

(1) 回転成分の制約

第 5章で述べた物体認識方式では，回転成分を離散化して全探索するため，回転成分の制約により探索範囲を直接的に限定することができる．すなわち，目標物体のカメラに対する回転成分の範囲があらかじめわかれば，その範囲だけ離散化して探索すればよい．

- 物体に対する標準的な制約

  たとえば，ドア，机，椅子など屋内に存在する多く設備や家具は，その座標系の z 軸を床面に垂直に設定すれば，ふつうの状態では z 軸まわりの回転自由度だけをもつ．これにより，回転成分の探索範囲を大幅に限定できる．

  いま，カメラ操作としてパン，チルトだけ許すこととし，その回転成分を $E^c =< \theta_1, \phi_1, 0 >$ とする．θ_1, ϕ_1 はそれぞれパン角，チルト角であり，ロボットにとって既知の値である．一方，上記のように z 軸回りの回転自由度だけをもつ物体の姿勢を $E^m =< \theta_2, 0, 0 >$ とする．θ_2 は未知の値である．すると，カメラ座標系での物体姿勢の回転成分は次のようになる．

$$R(E) \quad = \quad R_y(-\phi_1) R_z(\theta_2 - \theta_1) \tag{7.1}$$

  ここで，$R_y(a)$ は，y 軸の回りに角度 a だけ回転する回転行列である．R_z も同様である．オイラー角 $E =< \theta, \phi, \psi >$ に対する回転行列は $R(E) = R_z(\theta) R_y(\phi) R_z(\psi)$ であるから，これと式 (7.1) を比較すると，$E =< \theta, \phi, \psi >=< 0, -\phi_1, \theta_2 - \theta_1 >$ となる．ここで，θ_1, ϕ_1 は既知であるから，探索は ψ 成分に関してだけ行えばよい．実際には，カメラ姿勢の誤差を考慮して ϕ 成分についても，$-\phi_1 \pm \delta$ の範囲を探索するが，それでも計算量は大幅に減る．

- TG マップによる制約

  TG マップから物体の姿勢が制約できる場合，物体認識を促進することができる．一般に，屋内においては，物体の並進成分 (位置) の測定を広範囲に精度よく行うのは難しいが，物体の回転成分の範囲については測定しやすいことが多い．通常，部屋や廊下は長方形をなし，机やテーブル，キャビネット，ドアなどの物体は壁や他の物体と平行や垂直に置かれることが多いため，その相対的な回転成分を知ることは容易である．

  これにより，回転成分の探索範囲を大幅に狭められる．たとえば，式 (7.1) において，θ_2 も既知となるため，誤差を考慮しても，$-\phi_1 \pm \delta$ および $\theta_2 - \theta_1 \pm \delta$ の範囲を探索すればよく，計算量はさらに減る．

探索範囲の限定により計算量が減り，処理時間が短縮されるが，それと同時に，誤認識 (false positive) も減る．つまり，処理時間と認識率の両方が向上する．

(2) 並進成分の制約

第5章で述べた物体認識方式では，並進成分は投票およびクラスタリングによって求める．あらかじめ，目標物体の並進成分の範囲がわかっていれば，その近傍だけで投票やクラスタリングを行うことで候補を限定することができる．

- 物体の置かれている高さ (z 成分)
  物体が置かれている平面の高さがわかると，z 成分の探索範囲は誤差分だけに限定できる．たとえば，床面に置かれている物体は z=0，高さ h の机上に置かれている物体は z=h として，誤差を考慮した近傍領域だけに候補を限定する．

- 物体の置かれている位置 (xy 成分)
  原理的には，第6章で述べた方法で目標物体の存在範囲を調べれば，xy 成分も絞り込むことができる．ただし，TG マップが不正確さを許すことと，ロボット姿勢も誤差をもつことから，目標物体の xy 成分の範囲は広くなる．また，処理もやや面倒なため，現状では，次節のロボット姿勢との融合で実現する．

7.1.3 ロボット姿勢との融合による false positive の削除

ロボットの自己位置と TG マップをもとにロボットから見た目標物体の相対姿勢がわかれば，それと物体認識から得られた目標物体の推定姿勢とを融合することで，ありえない認識候補を削除できる．ロボットから見た目標物体の相対姿勢は，目標物体が現在のロボット姿勢の基準実体であるときは直接計算できる．そうでないときは，ロボット姿勢の基準実体から目標物体への相対姿勢アークをたどって計算することになる．どちらにしても，目標物体の相対姿勢はある確率分布をもつ．

目標物体に関する上記 2 つの相対姿勢の融合は，自己位置推定アルゴリズムを流用して行う．そのため，ロボットから見た目標物体の相対姿勢ではなく，目標物体から見たロボットの相対姿勢に対して融合処理を行う．融合処理はモンテカルロ法を用いる．以下に手順を示す．

(1) 現在のロボット姿勢の基準実体を現在の目標物体に切替える
  この切替えによって，ロボット姿勢は目標物体の座標に変換されて拡散する．すでに基準実体が目標物体である場合はこの処理は必要ない．

(2) 物体認識の結果から，目標物体を基準実体としたロボット姿勢を求める
  物体認識の結果は，ロボットから見た目標物体の相対姿勢なので，これから逆算して，目標物体から見たロボットの相対姿勢を求める．

物体認識の結果は，上位 N 個の候補を用いる．各候補に対するロボット姿勢の確率分布を正規分布で表す．各候補には物体認識のスコアが付与されており，そのスコアを重みとした正規分布の線形和で，全体のロボット姿勢の確率分布を生成する．

(3) (1) のロボット姿勢と (2) のロボット姿勢の確率分布を融合する
融合するには，2 つの確率分布の積を求める．具体的には，まず，(1) のロボット姿勢の各サンプルを (2) のロボット姿勢の確率密度関数に適用して，サンプルの重みを計算する (サンプルの姿勢に対して決まる確率値を重みとする)．次に，この重みに基づいてリサンプルしたサンプル集合を新しいロボット姿勢とする．

ロボット姿勢が何も残らなければ，物体認識に失敗したものとして，以前のロボット姿勢に戻す．

この処理の様子を図 7.2に示す．同図 (a) は，実体 e1 を基準にした現在のロボット姿勢の分布である．楕円が分布を表す．同図 (b) は，物体認識により得た実体 e1 を基準にしたロボット姿勢の仮説である．同図 (c) は，両者を融合した結果である．(b) の仮説 h1〜h4 のうち，(a) のロボット姿勢の分布に重なるのが h1 だけであり，融合の結果，h1 が新しいロボット姿勢として得られる．この過程で，ロボット姿勢が求まるだけでなく，仮説 h2〜h4 が false positive として削除され，h1 が正しい解として残る．h1 から逆算すれば e1 の姿勢も決まる．

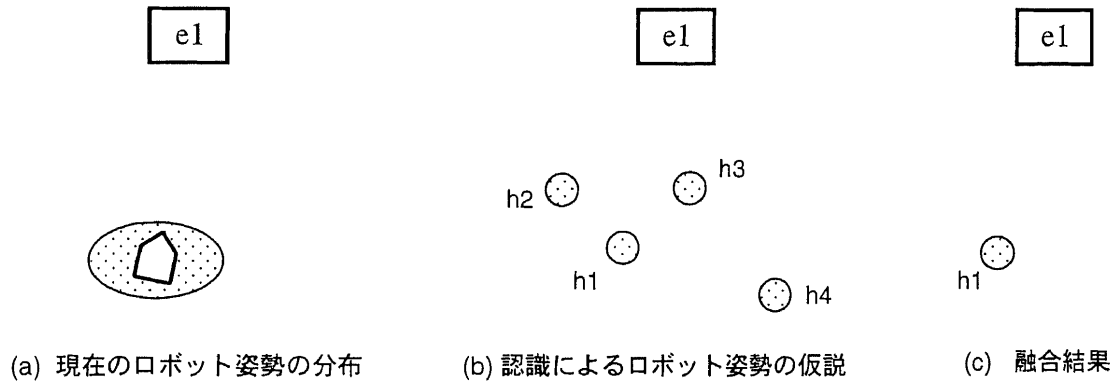


図 7.2: ロボット姿勢との融合による false positive の削除

7.1.4 認識の二重化による false positive の削除

物体認識を異なるカメラ角で二回行い，その結果を融合することで false positive を取り除くことができる．この方法で有効なのは，ランダムな false positive の削除である．ランダムな false positive は，あるカメラ角において，画像線分のランダムな組み合わせがモデル線

分と偶然一致した場合に生じる．これは，そのときのカメラから見た環境中の3次元の線分の並び方や，そのときの画像輝度による画像線分の抽出具合などに依存する．画像輝度は，照明条件の他にカメラのホワイトバランスや自動露光調節によっても微妙に変化する．カメラ角を変えて画像を2回撮影して，それぞれで認識を行えば，ランダムなfalse positiveが両方に現われる確率は低いため，ほぼ取り除くことができる．

図7.3にこの様子を示す．2回の認識により，(a)と(b)のようなロボット姿勢の仮説(h11～h14，h21～h24)が得られたとする．両者を融合すると，(c)のように姿勢がほぼ重なるh11とh21が残る．他の仮説は重なりがないので削除される．

カメラ角をわずかに変えただけでは，カメラから見た環境中の3次元の線分の並び方があまり変化しないため，本来は走行してカメラの位置も変えた方がよいが，1つの物体の認識に特定の走行距離がかかるのを避けるため，カメラ角だけを変えることにしている．

前節のロボット姿勢と融合する方法では，目標物体から見たロボット姿勢の誤差範囲が広い場合にfalse positiveが残ることがあるが，この認識の二重化を併用することでかなり絞り込むことができる．しかも，この処理もロボット姿勢に還元して行えるので，前節の方法と一体化して実装することができる．以下に手順を示す．

(1) カメラ視線で設定されたカメラ角で画像を撮影する

(2) 撮影した画像に対して，前節の方法で，ロボット姿勢と融合しながら物体認識を行う

(3) (2)で融合に成功したら，カメラ角をわずかに変えて画像を撮影する

(4) 撮影した画像に対して，同様に，ロボット姿勢と融合しながら物体認識を行う

認識の二重化を行うと，false negativeの確率が増える．すなわち，2枚の画像のどちらかがfalse negativeとなれば，全体の認識がfalse negativeになる．これに対しては，6.2.2節で述べたようにカメラ視線を複数本設定しておき，認識に失敗した場合には，次のカメラ視線でリトライすることで対処する．

7.1.5 実験

(1) 姿勢制約の効果

実際にロボットがタスクを実行する状況で物体認識の実験を行った．この実験の大きな特徴は姿勢制約を加えたことであるが，それ以外に，第5章での基礎実験と異なる点を以下に示す．

(1) ロボットに搭載したカメラで撮影
第5章の実験では，デジタルカメラを用いて人手で撮影したが，本実験では，ロボットにパンチルトカメラを載せた状態で撮影した．

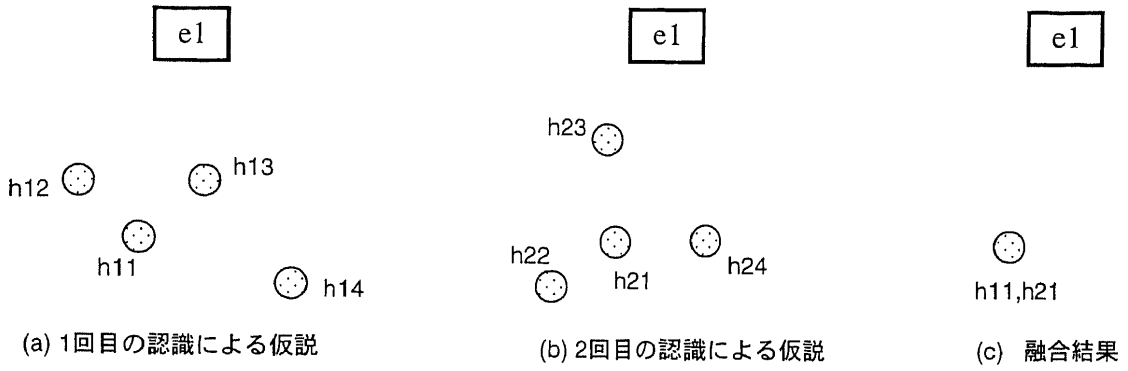


図 7.3: 認識の二重化による false positive の削除

(2) 姿勢制約の利用

第 5章の実験では，姿勢の回転成分を 2 自由度にするという制約だけを用いていたが，本実験では，7.1.2節で述べた種々の姿勢制約を与えた．

(3) エッジの色情報の利用

本実験では，エッジ周辺の色情報をエッジ選択のフィルタとして用いた．ただし，実装の簡単さのため，エッジ色はモデル全体で 1 つとした．

認識対象の物体は 10 個であり，各物体ごとに 50〜70 枚の画像を撮影して，認識実験を行った．図 7.4に実験結果を示す．各物体のモデルを図 7.4の左端に，認識結果例をその右に示す．認識結果は，原画像にモデルを投影して重ね合わせたものである．

実験では，以下の 2 つの条件を設定した．

- 基本制約

  回転成分の自由度は 2 自由度とし，その探索範囲は，ψ(スピン) は-80〜80 度，ϕ(チルト) は-30〜30 度とした．離散化幅 5 度で区切った．

  並進成分の z 方向の制約として，物体が置かれた水平面の高さを与え，これによって投票のビンの範囲を制限した．10 個の物体のうち，teabox だけは机の高さ (70[cm]) に置かれ，それ以外は床面 (0[cm]) に置かれているとした．ビンの幅は 10[cm] とした．

- マップによる姿勢制約

  マップ上の物体の方位，ロボットの方位，および，カメラのパンチルト角から回転成分の探索範囲を $\psi_0 \pm 20$[度] および $\phi_0 \pm 10$[度] とした．ただし，ψ_0，ϕ_0 は，それぞれ，予測される物体姿勢の回転成分である．

実験で得られた認識率を表 7.1に示す．マップによる姿勢制約がある方が認識率が上がっていることがわかる．これは，探索範囲が制限されたため，false positive が減ったことによ

る．また，MLJP 数が多いほど認識率がよい傾向がある．これは，MLJP 数が多ければ，オクルージョンやノイズによって多少の特徴 (ILJP) が消失しても，残った ILJP で認識ができるためである．

表 7.1の各物体を分析すると，以下の傾向がある．

- cabinet,deskG,deskW は認識率が非常に高い
  これは，MLJP 数が多く，しかも，物体の前面に集中しているためと思われる．

- deskS は置かれた状態に対する依存性が高い
  deskS は脚がパイプのため，脚の向こうの背景が見える．このため，エッジ抽出において，脚のエッジが背景のエッジと交わって分断され，認識に失敗することが起こる．逆に，背景が壁や床のように単調な環境では，認識率は極めて高い結果が得られた．

- doorL, doorM も認識率が高い
  MLJP 数はそれほど大きくないが，全体のサイズが大きいので，エッジが抽出しやすく，消失するエッジが少ないためと考えられる．

  ただ，doorM は縦長の窓をもつため，カメラ角によっては，全部が画像内に収まらず，ILJP が大きく失われることがあった．

- refrig(冷蔵庫) は認識率が悪い
  この原因は，冷蔵庫，その背後の壁，および，隣接する物体がほぼ同じ色であるため，正面から撮影した画像において，エッジが抽出できなかった部分があったためである．MLJP 数が少ないため，このエッジ消失が致命的になったと考えられる．

  斜めから撮影した画像では，エッジがよく抽出され認識率はずっと高かった．

- teabox は認識率が悪い
  これは，サイズが 20[cm] 程度と他の物体よりも小さく，探索空間の離散化の仕方が適合しなかったことがあげられる．また，MLJP が物体の前面，側面，上面に分散しており，1 つのカメラ角で見える MLJP 数は比較的少ないことも原因として考えられる．

このように，物体によって認識率の高低があり，また，置かれた状態に依存しやすいものもある．ナビゲーションのランドマークとして選ぶ際には，物体そのものの性質と物体が置かれている状態，および，カメラ角を勘案して，適切なものを選ぶのが望ましい．

処理時間を表 7.2に示す．これは，画像の前処理から認識までを通した処理時間である．マップによる制約がある場合の方が，処理が 2〜3 倍速くなっていることがわかる．回転成分の探索範囲は 12 分の 1 になっているのに対して，処理時間の減少率が小さいのは，前処理 (画像微分から ILJP 抽出まで) の時間が姿勢の探索範囲によらず一定だからである．前処理

には，0.3〜0.5 秒かかっている．また，解の候補は探索範囲に一様に分布するのでなく，正解の近くに分布するため，その付近の処理に時間がかかることも原因の 1 つである．

マップ制約がある場合は，だいたい画像 1 枚につき 1 秒以内で認識ができており，ロボットが停止して処理するには，実用上問題のない速さである．ただし，ナビゲーション中は，これに画像キャプチャ時間が加わり，また，ロボットの動作制御プログラムが並列して走るため，実際の処理時間はもう少し長くなる．

表 7.1: 認識率

		基本制約のみ		マップ制約あり	
対象	MLJP 数	1 位	3 位以内	1 位	3 位以内
cabinet	29	83.6% (46/55)	96.4% (53/55)	98.1% (54/55)	100% (55/55)
deskG	21	92.5% (49/53)	98.1% (52/53)	96.2% (51/53)	100% (53/53)
deskW	29	98.3% (57/58)	98.3% (57/58)	98.3% (57/58)	98.3% (57/58)
deskS	28	85.9% (61/71)	88.7% (63/71)	86.9% (61/71)	90.1% (61/71)
doorL	12	92.6% (50/54)	94.4% (51/54)	94.4% (51/54)	94.4% (51/54)
doorM	12	82.1% (32/39)	89.7% (35/39)	89.7% (35/39)	92.3% (36/39)
teabox	24	47.1% (33/70)	65.7% (46/70)	60.0% (42/70)	80.0% (56/70)
corner	6	71.4% (35/49)	75.5% (37/49)	79.6% (39/49)	79.6% (39/49)
elevator	16	69.0% (40/58)	89.7% (52/58)	75.9% (44/58)	93.1% (54/58)
refrig	11	58.1% (32/55)	70.9% (39/55)	65.5% (36/55)	83.6% (46/55)

表 7.2: 処理時間

対象	基本制約のみ	マップ制約あり
cabinet	3.59 秒	1.66 秒
deskG	2.33 秒	1.02 秒
deskW	1.21 秒	0.54 秒
deskS	0.71 秒	0.41 秒
doorL	1.84 秒	0.66 秒
doorM	0.60 秒	0.38 秒
teabox	1.20 秒	0.57 秒
corner	1.03 秒	0.48 秒
elevator	1.48 秒	0.50 秒
refrig	1.38 秒	0.53 秒


cabinet
deskG
deskW
deskS
doorL
doorM
teabox
corner
elevator
refrig

図 7.4: 認識例

(2) false positive の削除

ロボット姿勢との融合と認識の二重化による false positive の削除の例を図 7.5と図 7.6に示す．対象物体は机であり，図 7.5に物体認識で得られた候補のいくつかを示す．このように認識候補には，正解の他に false positive が含まれている．図 7.6(a) は，机の物体認識前のロボット姿勢である．(b) は，1 回目の物体認識の各候補から求めたロボット姿勢の仮説である．この認識結果には false positive が多く，ロボット姿勢は広い範囲に散らばっている．(c) は，(a) と (b) を融合した結果である．(a) の散らばりが大きいため，融合結果にもまだ多少散らばりがある．(d) は，2 回目の物体認識の各候補から求めたロボット姿勢の仮説である．この認識結果では false positive は少なめで，しかも (b) とは散らばり方が異なっている．(e) は，(c) と (d) を融合した結果である．(e) では，(c) の融合結果よりも散らばりが減っていることがわかる．

この例から，ロボット姿勢との融合により false positive が減ることがわかる．また，認識の二重化を用いると，さらに false positive を減らせることが確認できる．

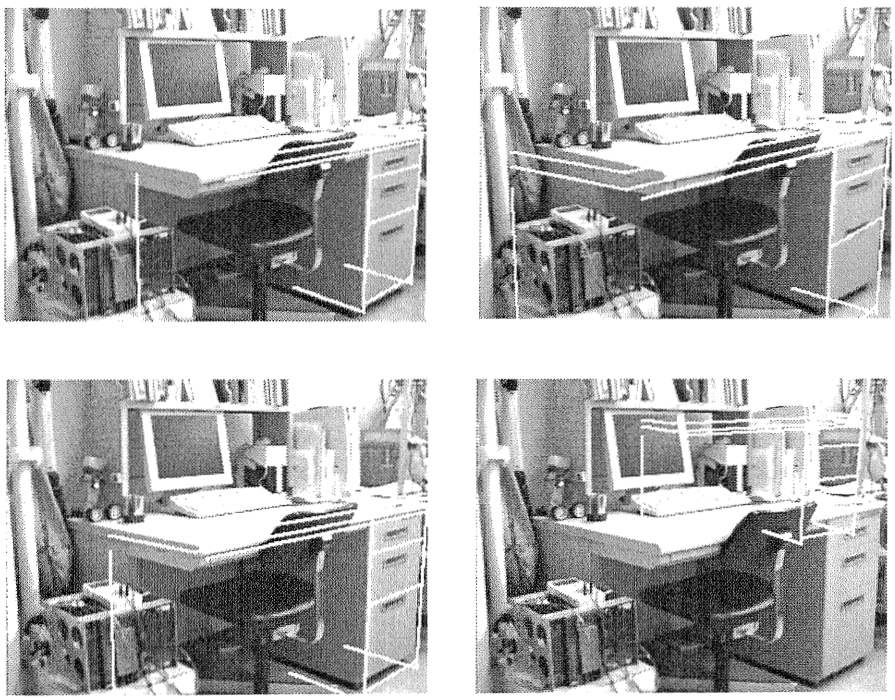
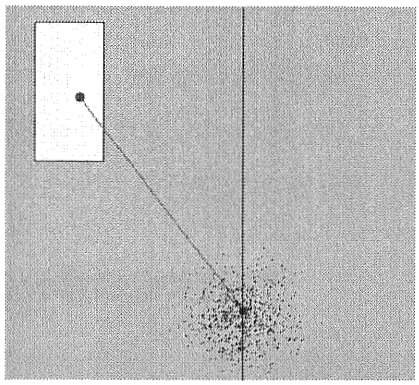
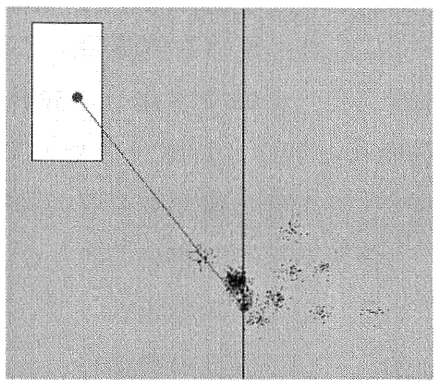


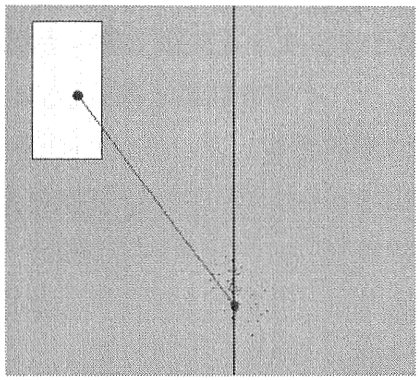
図 7.5: 机の認識例： 右下の画像は明らかな false positive



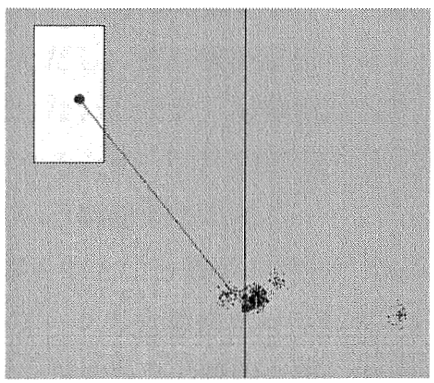
(a) 現在のロボット姿勢



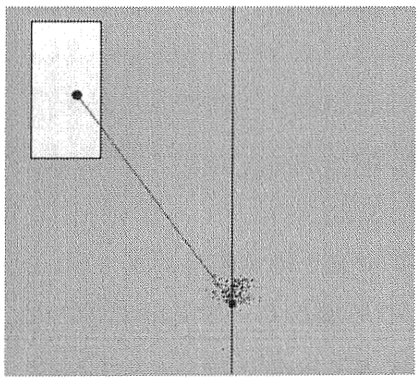
(b) 1回目の認識によるロボット姿勢の仮説



(c) (a)と(b)の融合結果



(d) 2回目の認識によるロボット姿勢の仮説



(e) (c)と(d)の融合結果

図 7.6: false positive の削除

7.2 自己位置推定の精度向上

自己位置推定の精度は，TG マップの精度や走行経路，ランドマーク観測計画に依存する．ここでは，精度のよい自己位置推定ができるための TG マップや経路の作成法を考える．また，同じクラスの実体が複数あるために自己位置が複数得られる状況での対処法を検討する．

7.2.1 相対姿勢アークの分散の設定指針

TG マップの精度は実体間の相対姿勢アークの分散の値で決まる．TG マップの作成においては，この値の厳密な測定はしないにしても，目視などによっておおむね妥当と思われる値を設定することが期待される．ただし，どの程度の値が望ましいかという指針があれば，TG マップの作成がしやすくなる．ここでは，その定性的な検討を行う．

相対姿勢アークの分散は，ロボットの自己位置推定において，基準実体を切替えたときにロボット姿勢の分布を拡散させてマップの誤差を補正し，センサデータとの融合を可能にする役割をもつ．この分散が小さすぎると，データ融合が失敗して自己位置推定ができなくなる．したがって，分散値の下限は，このデータ融合が成功する限界で決まる．一方，分散が大きすぎると，目標物体だけでなく，同じクラスに属する他の実体とも融合する可能性が生じる．このため，他の実体とのデータ融合が起こらないかどうかが分散値の上限の 1 つの目安となる．ただし，他の実体が近くにありすぎる場合は避けようがないため，真の上限が決まるわけではない．なお，分散値が大きいと，基準実体の切替えにおいてロボット姿勢を大きく拡散させるため，7.1.3節で述べた方法で物体認識の false positive を削除することが難しくなるという問題もおこる．

以上の考察をもとに分散値を適切に定めればよいが，データ融合がうまくできるかどうかは，相対姿勢アークの分散値だけでなく，走行におけるランドマークの選び方にも依存する．たとえば，ランドマークの間隔を離しすぎると，ランドマーク間で経由する実体の数が増えて相対姿勢アークの分散が累積され，不必要に融合する実体の数が増える．このため，走行経路上で適切な間隔でランドマークを選ぶことが重要である．また，近接したランドマークでは不必要に相対姿勢の分散を大きくしないことも重要である．近接していれば，その相対姿勢の測定も容易なので，誤差も小さくできることが期待される．

7.2.2 走行経路の作成指針

前節の考察から，走行経路の作成において，ランドマークの選び方が重要である．第 6 章で述べたナビゲーション方式では，ランドマークは必ず経路線分の基準実体となるので，ランドマークをいかに選ぶかが経路作成の決め手となる．また，この際，実体間の相対姿勢アークの分散が適切かどうかをチェックして，必要ならば TG マップを修正する．

以下に，ランドマーク選択の指針を示す．この指針で選んだランドマークを基準実体にした経路線分を連結することで，走行経路を作ればよい．ただし，基準実体の中にはランドマークにならないものがあってもよい．

- 同じクラスの実体と存在範囲の重なりが小さいこと
  周囲にクラスが同じで姿勢が近い実体が存在しないことが，一番望ましい．このとき，距離が近くても方位が違えば問題はない．存在する場合は，なるべく近くに1つ前のランドマークを設けて，そのランドマークからの相対姿勢をできるだけ正確にしておく．

- 前のランドマークとの相対姿勢の誤差が小さいこと
  この方が，基準実体の切替えにおけるロボット姿勢の拡散が小さくなり，false positive をより多く削除できる．

- 経路上からランドマークを観測しやすいこと
  カメラに対する物体の向きと距離が適切で，画像に適切な大きさで写るものを選ぶ．

走行経路の最初のランドマークと最後のランドマークは，それぞれ，走行経路の始点と終点の近くの物体で上記の条件を満たすものを選ぶ．

7.2.3 同じクラスの実体が近接している場合

前節のように適切に相対姿勢アークを設定しても，同じクラスの複数個の実体が非常に近くに存在して，存在範囲が重なることがありうる．そうすると，ロボット姿勢の仮説が複数個得られてしまう．たとえば，ドア板が二枚一組になっていたり，同じ机が並んでいる場合にこれが起こるが，室内では珍しいことではない．

このような場合でも，各実体の認識がきちんとできれば，自己位置推定により，一時的に複数個になった仮説を1個に収斂させることができる．すなわち，ある地点で実体 e1 を認識して，e1 および e2 を基準にしたロボット姿勢の仮説をたて，少し走行したところで実体 e2 を認識して，やはり，e1 および e2 を基準にしたロボット姿勢の仮説をたて，両者を自己位置推定の処理過程で融合すれば，ロボット姿勢は1個になる (図 4.4参照)．

ところが，1枚の画像から e1 と e2 の両方が認識できると，この処理を一度に行うことができる．この様子を図 7.7に示す．同図 (a) でロボットが e1 と e2 を両方認識したとする．e1 の推定姿勢が q1，e2 の推定姿勢が q2 であるとする．ロボットはこの時点では，q1 と q2 が，e1 と e2 のどちらの姿勢であるかわからない．そこで，両方の仮説を混合して融合する．(b) は，e1 の姿勢が q1 あるいは q2 であるとした仮説であり，(c) は，e2 の姿勢が q1 あるいは q2 であるとした仮説である．これらを融合すると，B と C が同じ姿勢なので生き残る．これにより，1回の画像撮影で実体の同定が確実に行え，また，ロボット姿勢も一意に決めることができる．

この方法では，1 枚の画像で片方の実体しか認識されない場合は，融合結果の仮説が残らないため，少し走行して再試行する必要がある．

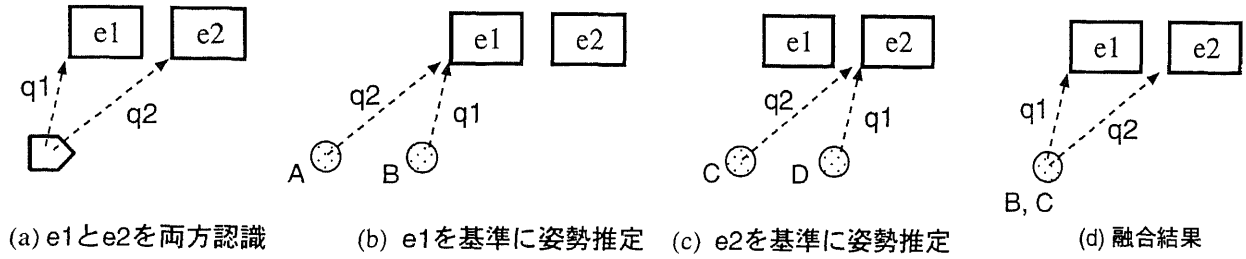


図 7.7: 近接した同クラスの実体の識別

7.2.4 実験

(1) 相対姿勢アークの誤差が自己位置推定に及ぼす影響

相対姿勢アークの標準偏差をどの程度に設定すると，自己位置推定が正しく行われるかを評価する実験を行った．この問題をきちんと分析するには様々な条件を考慮する必要があるが，今回は，おおまかな定性的傾向を得ることを目的とした簡易実験を行った．

実験環境は筑波大学 L 棟 302 号室の一部で，冷蔵庫と机 (灰色) を対象物体とする．実験では，この 2 つの物体を TG マップに記述し，経路を設定して，ロボットを走らせる．ロボットは，最初に冷蔵庫，次に机を認識し，最終的に机を基準実体としたロボット姿勢を求める．このとき，マップにおける冷蔵庫と机の相対姿勢 (x 方向の距離) を変化させ，どの程度までロボットが正しく自己位置推定ができるかを調べる．冷蔵庫と机の相対姿勢アークの標準偏差は固定であり，x 方向を 50[cm]，y 方向を 10[cm]，θ を 3[度] とした．なお，冷蔵庫と机の画像はロボットを実際に走らせて撮影したが，この実験での自己位置推定は，その画像を用いた走行シミュレータで行った．また，実環境での，冷蔵庫と机の間の距離は約 1.5[m] である．図 7.8に冷蔵庫と机の認識結果を示す．

図 7.9に冷蔵庫と机の相対姿勢を変えた場合の自己位置推定の結果を示す．この図は，ロボットが机を認識して，机を基準実体としたロボット姿勢を求めた直後の様子を表している．同図 (a) は両者の x 方向の距離が 3[m] の場合，(b) は両者の x 方向の距離が 1[m] の場合である．どちらも自己位置推定に成功している．実際，(a)，(b) とも，ロボットと机の間隔は約 2[m] であり，机を基準実体にしたロボット姿勢は両者で一致していることがわかる．なお，x 方向の距離が 3[m] を大きく超えた場合は自己位置推定に失敗することが多かった．このことから，相対姿勢の標準偏差が 50[cm] のとき，冷蔵庫と机の距離を，実測 1.5[m] に対して，マップ上で 3[m] としても自己位置推定できたことになる．すなわち，標準偏差 50[cm] に対

する許容誤差は，約 1.5[m] と考えられる．正規分布で 3σ で 97%をカバーできることから見ても，おおむね妥当な値であると考えられる．実際は，もとのロボット姿勢の分布の広がりもあるため，許容範囲はもう少し広くなる．

このことは，認識前のロボット姿勢と認識によるロボット姿勢の仮説の重なりを考えるとわかりやすい．図 7.9で経路上に楕円形に広がっている点群は，ロボットが机を認識する前のロボット姿勢の分布である．机を認識して自己位置推定をする際に，冷蔵庫から机に基準実体を切替えたことで，ロボット姿勢の分布は拡散して，図のような楕円形になる．また，ロボット上に密集した点群は，認識によるロボット姿勢の仮説である．図 (a)，(b) とも，認識前のロボット姿勢と認識によるロボット姿勢仮説の重なりがあるので，自己位置推定に成功した．図 (a) よりも机を遠くに離すと，この重なりがなくなり，自己位置推定に失敗する．

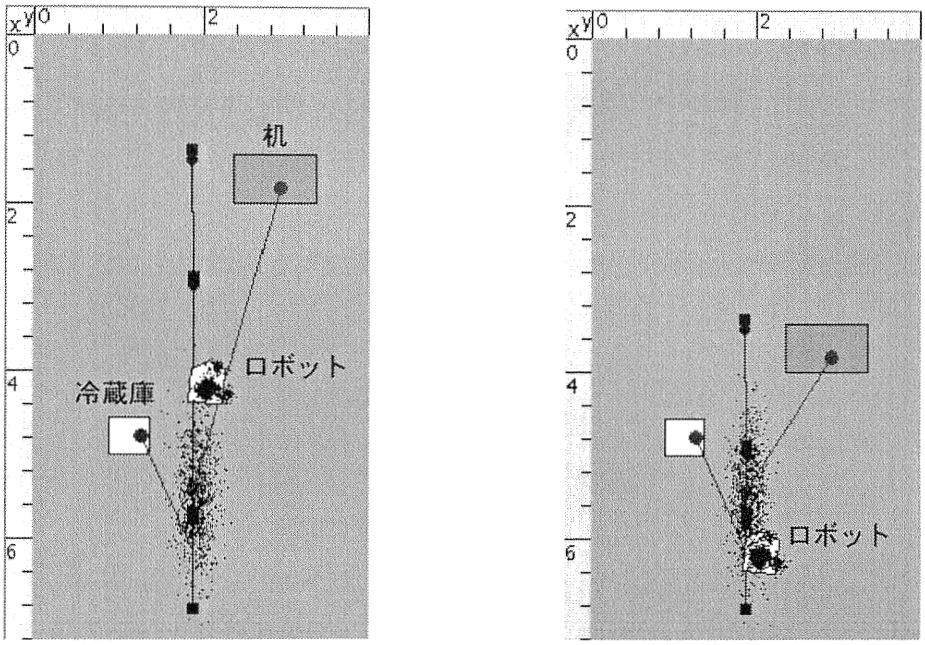
これらから得た相対姿勢アークの分散の設定指針を以下に示す．

(1) 物体間の相対姿勢に対して誤差がどのくらいあるか推量する
このためには，誤差が相対姿勢の何%くらいかあるかを考えるのがわかりやすい．たとえば，相対姿勢が 2[m] 程度であれば，目視でもそれほど大きな誤差はありえない．誤差を 50%とすると，1[m] の誤差まで想定することになる．相対姿勢が 20[m] 程度であれば，目視や記憶では，相当大きな誤差が生じうる．しかし，誤差を 50%とすると，10[m] の誤差まで考えることになり，絶対的な許容量はかなり大きくなる．

(2) 想定した誤差の 1/3 程度の値を相対姿勢の標準偏差とする
上記の実験結果から，標準偏差の 3 倍程度までの誤差ならば，自己位置推定は成功する．そこで，誤差の 1/3 程度の値を目安とする．

ただし，この指針はあくまで目安である．上述した標準偏差に対する許容誤差は，拡散前のロボット姿勢の分布にも依存するし，データ融合の際の確率の下限閾値にも依存する．また，前述のように，近傍に同じクラスの実体が存在すると，それとの識別が問題となる．これらに対する定量的な分析は，残された課題である．

図 7.8: 冷蔵庫と机の認識



(a) 机と冷蔵庫の間隔が 3 m　　(b) 机と冷蔵庫の間隔が1m

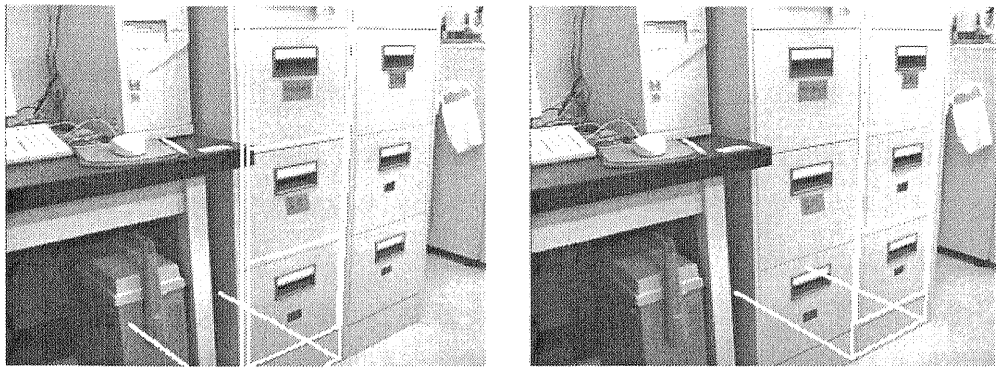
図 7.9: 冷蔵庫と机の相対姿勢を変えた場合の自己位置推定

(2) 同じクラスの実体が近接している場合

同クラスの実体が近接している場合の自己位置推定の実例を図 7.10と図 7.11に示す．この例では，幅 50cm のキャビネットが 2 つ並べて置かれている．このような配置では，1 枚の画像に両方のキャビネットが写ることが多く，その場合，図 7.10のように，たいていは両方とも認識される．

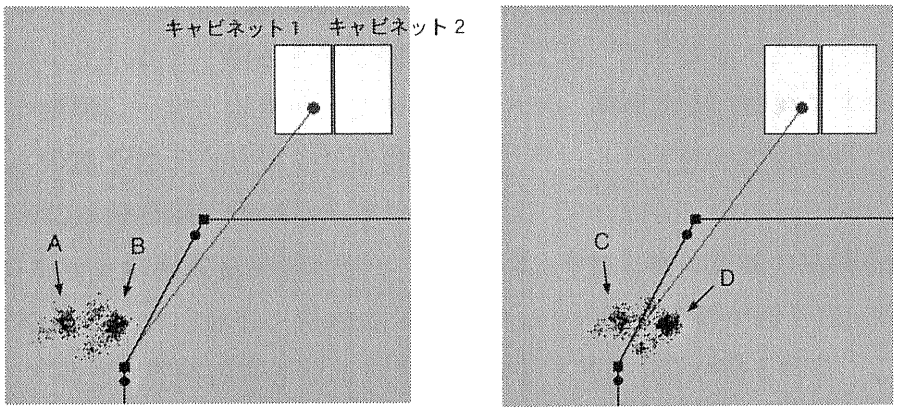
図 7.11は 7.2.3節の方法で自己位置推定を行った例である．同図 (a) は，キャビネット 1

を基準にして求めたロボット姿勢の仮説である．多少のちらばりがあるが，AとBが主要な仮説である．Aはキャビネット2を認識したことによる姿勢仮説，Bはキャビネット1を認識したことによる姿勢仮説である．(b) は，キャビネット2を基準にして求めたロボット姿勢の仮説である．多少のちらばりがあるが，CとDが主要な仮説である．Cはキャビネット2を認識したことによる姿勢仮説，Dはキャビネット1を認識したことによる姿勢仮説である．(c) は，認識前のロボット姿勢に (a) を融合した結果である．(a) としか融合していないので，仮説AとBの両方が生き残っている．(d) は，認識前のロボット姿勢に (a) と (b) の両方を融合した結果である．(a) と (b) を融合したことで，重なりあう仮説BとCが生き残り，正しいロボット姿勢が得られていることがわかる．



(a) キャビネット 1 を認識　　(b) キャビネット 2 を認識

図 7.10: 近接したキャビネットの認識



(a) キャビネット1を基準にした仮説　　(b) キャビネット2を基準にした仮説

(c) ロボット姿勢と(a)の融合結果　　(d) ロボット姿勢と(a)と(b)の融合結果

図 7.11: 近接したキャビネットの識別

7.3 TG マップへの反映

7.3.1 可動物体の姿勢推定

環境中には，ある制約のもとで姿勢が変化する物体が存在する．たとえば，ドア板は土台 (ドア枠) に対して回転する構造をもち，両者の間の角度をパラメータとして土台からの相対姿勢を定義できる．また，窓や引出しなどは土台に対して平行移動する構造をもち，並進パラメータで姿勢を定義できる．このような可動物体の姿勢をマップにあらかじめ正確に記述することは不可能であり，通常は，姿勢が変化しない土台の姿勢だけを記述することになる．ところが，ドア板などは，ロボットが走行する際に，その姿勢を推定しなければ，衝突する可能性がある．

このようなパラメータ化された動きをもつ物体の姿勢推定は，自己位置推定によるロボット姿勢の推定値と，物体認識から得たロボット姿勢の仮説を融合することで実現できる．この様子を図 7.12に示す．同図 (a) は，ロボットがドア板を認識したときの様子である．物体認識によってロボットとドア板の相対姿勢が決まっても，ドア板とドア枠の角度 α が未知なため，ロボット姿勢は一意には決まらない．そのため，物体認識の結果だけでは，ロボット姿勢は同図 (b) のようにドーナツ上に分布する (ただし，ドア板の回転中心がドア板の端にあるため，厳密には分布は円形にならない)．

ここで，ロボットの現在姿勢がわかっていれば，それと物体認識の結果を融合することで角度 α を求めることができる．たとえば，ロボットの現在姿勢が図 7.12(c) のようにドア枠から q1 だとすると，物体認識による仮説と融合して同図 (d) が得られる．ここで，ロボットとドア枠の角度はロボットの現在姿勢から得られ，ロボットとドア板の角度は物体認識から得られるので，ドア板とドア枠の角度 α が確定する．実際には，これらの計算は自己位置推定の枠組の中で確率的に行われる．

7.3.2 TG マップの更新

走行経路に沿ってロボットがナビゲーションを行うと，走行中の物体認識で得た物体の推定姿勢とロボットの走行軌跡とから，物体のより正確な姿勢がわかる．これに基づいて，TG マップを更新することができる．以下に，その手順を示す．

(1) 走行軌跡の生成

ナビゲーションにおいて，ロボットの現在姿勢を定期的に取得する．ロボットの現在姿勢は，山彦の Spur_get_pos_GL を用いる．とくに，カメラ視線の始点に到達して自己位置修正を行うと，基準実体の切替えで座標変換が行われるため，山彦の GL 座標も不連続になる．そこで，カメラ視線の始点に到達すると，自己位置推定の前後の山彦の GL 座標を取得する．

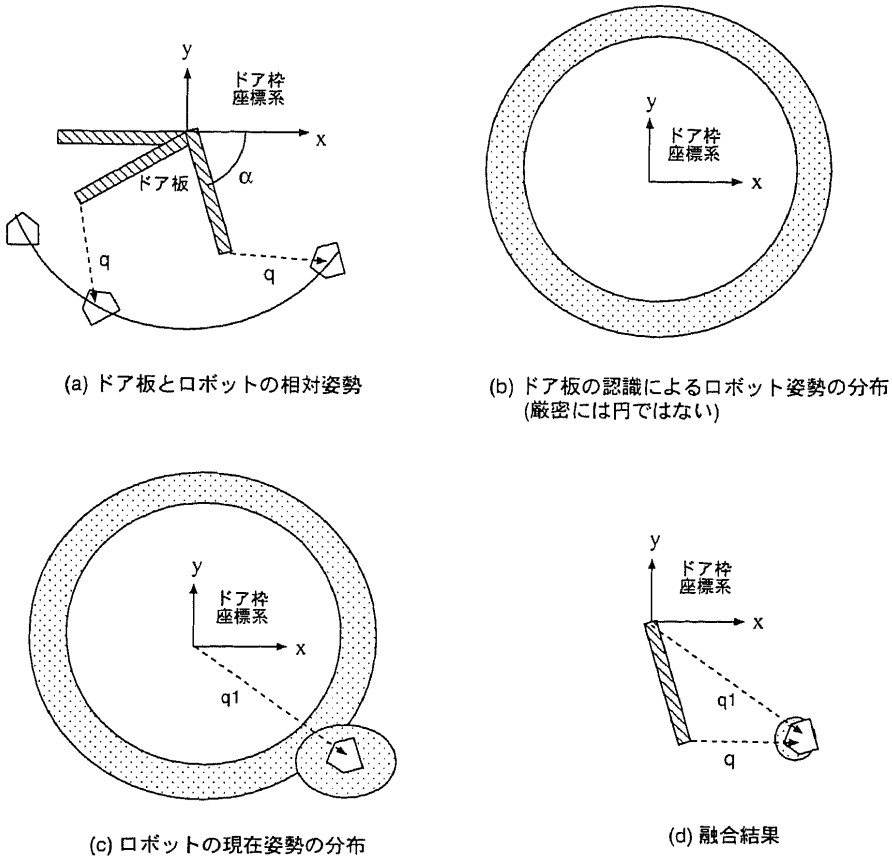


図 7.12: ドアの姿勢推定

この結果，たとえば，図 7.13(a) のような走行軌跡が得られる．同図では，点 P1 において，実体 e2 を認識して自己位置推定をした結果，ロボット姿勢が点 P2 に修正された様子を表している．

(2) 走行軌跡の復元

カメラ視線の始点で不連続になった部分を補正して，連続した走行軌跡を作成する．これは，自己位置推定前後のロボット姿勢がグローバルには同じになるように，軌跡全体を移動することで行う．

図 7.13(b) にこの様子を示す．自己位置推定中はロボットが静止しているので，同図 (a) の P1 と P2 は本来は同じ点である．そこで，両者が一致するように，P2 から先の軌跡を移動する．

(3) 認識した物体の配置

認識した物体のロボットに対する相対姿勢の原点を，復元した走行軌跡上のカメラ視線の始点に移すことで，物体のマップ上での姿勢を計算する．この姿勢を用いて，マッ

プ上の物体姿勢を更新する．

図 7.13(b) の実体 e2 は，このようにして姿勢が更新されている．

この更新処理は，TG マップ上のすべて実体ではなく，ナビゲーションで観測された実体にしか適用されない．このため，この更新によって実体間の相対姿勢に矛盾が生じる可能性がある．これを防ぐには，観測された実体の更新に伴って，それと相対姿勢アークが定義されている未観測の実体をいっしょに更新することが考えられる．しかし，実体の相対姿勢関係がループをなしている場合は，影響の波及をどの程度行えばよいか判断するのは難しい．また，実体姿勢の更新によって，部屋の大きさを変えたり，壁の長さを変えるといった処理も必要になる可能性がある．このような様々な依存関係を考慮したマップ更新は，残された課題である．

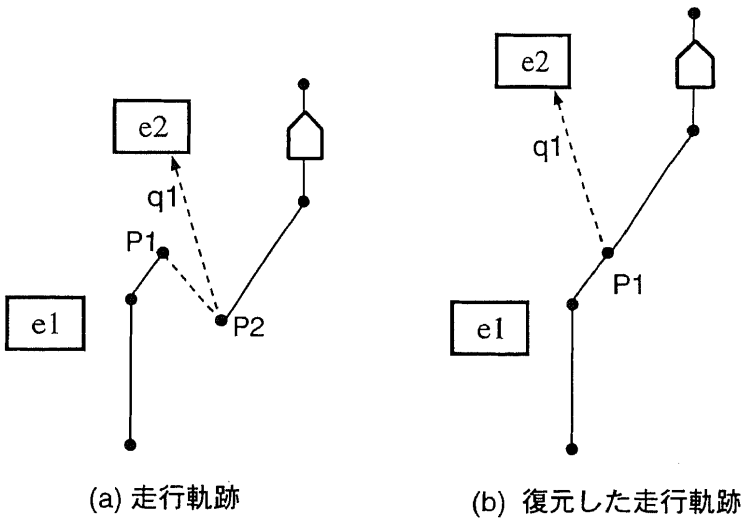


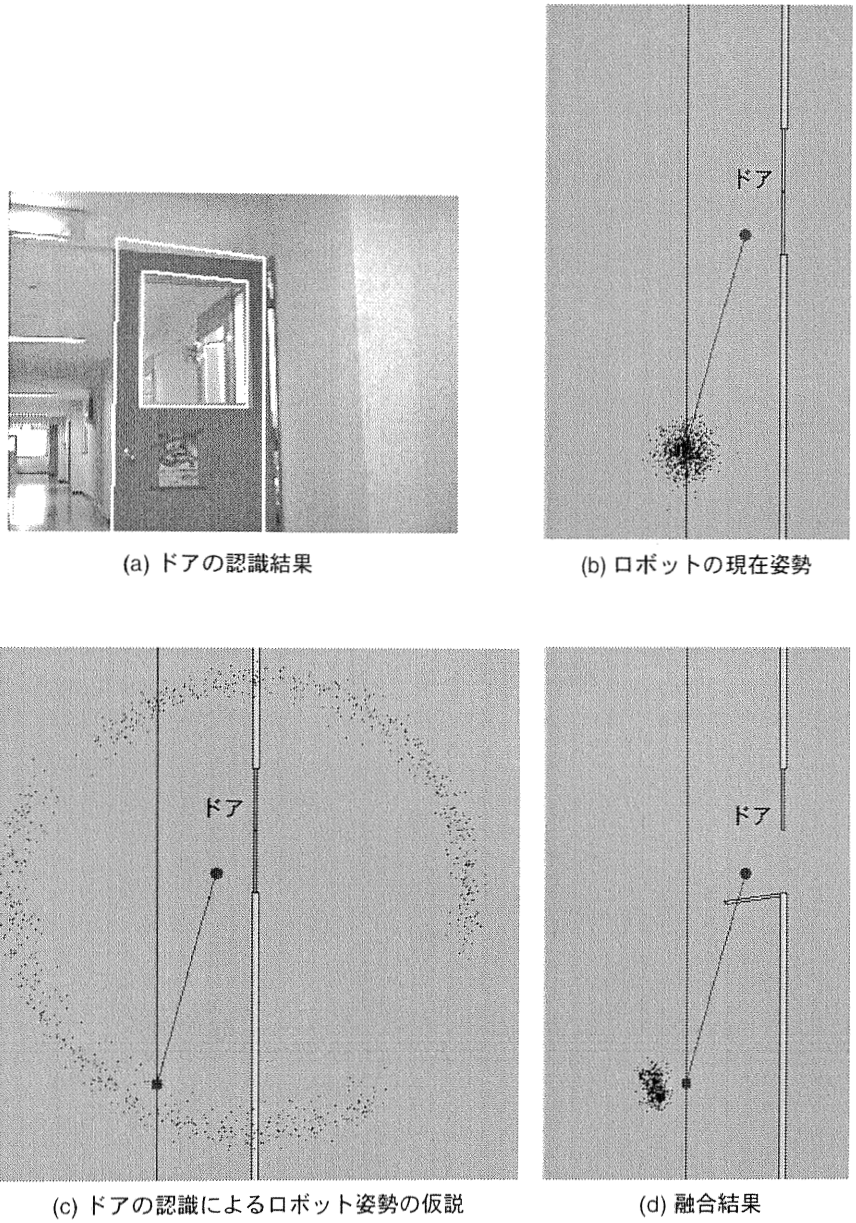
図 7.13: TG マップの更新

7.3.3 実験

(1) ドア板の姿勢推定

ロボット姿勢と物体認識結果の融合により，ドア板の姿勢を計算する実例を図 7.14に示す．同図 (a) は，ドア板の認識例である．(b) は，認識前のロボット姿勢である．この位置でドアを認識し，(a) の結果を得た．(c) は，ドア板の認識によって得たロボット姿勢の仮説である．ドア板が回転するため，ドーナツ上に分布する．ここでは，ドア板の回転角 α の範囲を，ドア板を閉じた状態を 0 度として，0〜150 度としている．ドア板は裏面からも同じように認識でき，回転中心がドア板の端にあるため，図のような少しいびつなドーナツ形にな

る．(d) は，(b) と (c) を融合した結果である．融合の結果，ロボット姿勢とドア板の姿勢の両方が確定したことがわかる．



(a) ドアの認識結果　(b) ロボットの現在姿勢

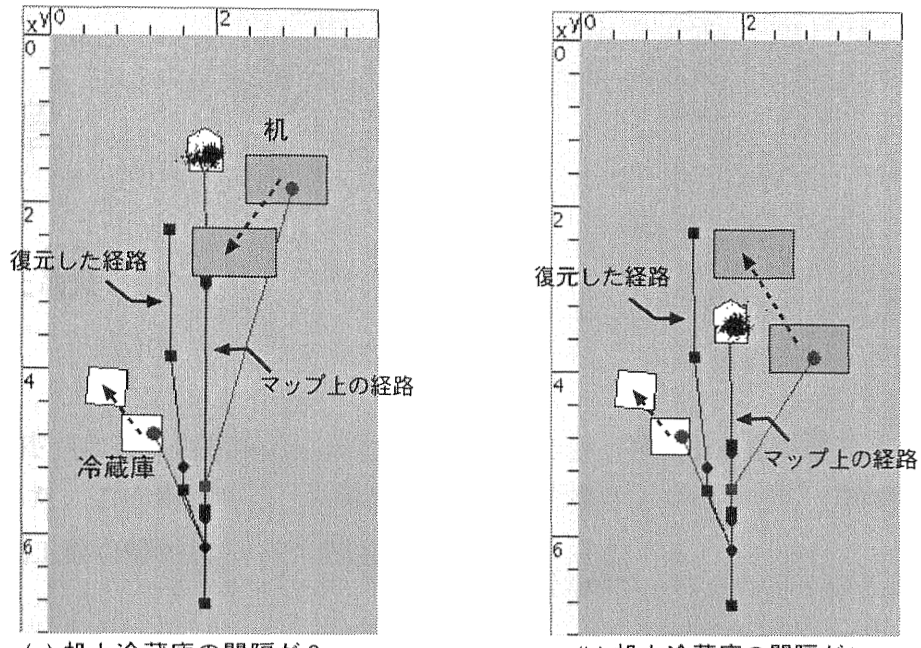
(c) ドアの認識によるロボット姿勢の仮説　(d) 融合結果

図 7.14: ドア板の姿勢推定

(2) マップ更新

7.2.4節で述べた冷蔵庫と机の実験結果を用いて，マップ更新を行った例を図 7.15 に示す．この図は，復元された走行軌跡と更新された冷蔵庫と机の位置を示している．同図 (a) と (b) で，冷蔵庫と机の間の距離は 2[m] の違いがあるが，更新後の距離はどちらも約 1.5[m] であ

り，両者がよく一致していることがわかる．



(a) 机と冷蔵庫の間隔が 3 m　　(b) 机と冷蔵庫の間隔が1m

図 7.15: マップ更新の例

第 8 章

総合実験

これまで述べた方式の有効性を確認するために，第 2 章で述べた目標タスクに対する総合実験を行った．この実験では，各機能を連係して目標タスクを実行し，環境変化への対処とマップ作成工数の軽減について評価を行った．

8.1 実験条件

8.1.1 実験環境と TG マップ

実験環境は，筑波大学 L 棟 3 階の L302-1 号室，L307 号室，および，その間の廊下である．見つける物は，紅茶箱 (7.1.5節の teabox) である．ロボットには，「L307 号室の小机の上に載っている紅茶箱を見つけよ」という指示を走行経路の形で与えるものとする．

このタスクのために作成した TG マップを図 8.1に示す．同図 (a) に主な物体の配置を示す．各部屋には他にも物体が置かれているが，マップにはロボットの走行に強く関連するものだけを記述している．図の左および上の xy 値が距離を表す．このマップの大きさはおよそ 10m × 14m であるが，これは不正確であり，実際の環境はもう少し大きい．

同図 (b) に物体間の相対姿勢アークを示す．このマップは目視で作成したものであり，随所に 30[cm]〜2[m] 程度の誤差がある．L302-1 号室は物体が近接して並んでいるため，相対姿勢の誤差は比較的小さい．とくに机など隣接しているものの誤差は小さいため，相対姿勢アークの標準偏差はデフォルト値 ($\sigma_x = 10[cm], \sigma_y = 10[cm], \sigma_\theta = 3$[度]) のままにしている．やや離れたキャビネットと机や，ドアと机の間の相対姿勢アークについてだけ標準偏差を 30[cm] にした．廊下や L307 号室は，物体の間隔がやや広いため，誤差が大きい．とくに，L302-1 号室のドアと L307 号室のドア，および，L307 号室のドアと小机の間には，約 2[m] の誤差がある．この部分の相対姿勢アークの標準偏差は，1[m] と大きめにした．

以上のように，相対姿勢アークの多くはデフォルト値で済み，他のものも大雑把な値ですむため，マップ作成の工数は小さい．図 8.1のマップは，TG マップエディタを用いて，約

10 分で作成できた.

8.1.2 タスク実行の手順と走行経路

タスク実行の具体的な手順を以下に示す.

(1) L302-1 号室のスタート点 (図 8.3の P18 付近) から開始する.

(2) L302-1 号室の中を走行し，ドアを通って廊下に出る.

(3) 廊下を走行し，ドアを通って L307 号室に入る.

(4) L307 号室の中で，指定された小机まで移動して，その上に載っている紅茶箱を見つける.

(5) L307 号室の中を走行し，ドアを通って廊下に出る.

(6) 廊下を走行し，ドアを通って L302-1 号室に入る.

(7) L302-1 号室の中を走行し，スタート点に戻る.

このタスクを行うための走行経路を図 8.3に示す. 経路線分から各物体に伸びている線分はカメラ視線である. 走行距離は，このマップでは 20[m] 程度であるが，実環境では約 30[m] である. 走行においてランドマークとして用いる物体は，キャビネット (7.1.5節の cabinet, 認識 2 回)，灰色の机 (deskG, 1 回)，ドア (doorL, 4 回)，小机 (deskS, 1 回)，紅茶箱 (teabox, 1 回)，白い机 (deskW, 2 回) の 6 種類，のべ 11 個である. 7.1.4節で述べた認識の二重化のため，各物体について認識を 2 回行うので，認識回数は最小で 22 回である. 認識や自己位置推定に失敗するとリトライするため，実際はもっと多くなる可能性がある.

8.1.3 機能の連係

第 7章で述べた機能の連係については，以下のように用いた.

- 姿勢制約の利用
  走行経路中のすべての物体認識に対して，7.1.2節の方式を採用した.

- false positive の削除
  走行経路中のすべての物体認識に対して，7.1.3節と 7.1.4節の方式を採用した.

- 近接した実体による自己位置推定
  L302-1 号室のキャビネットに対して，7.2.3節の方式を採用した.

- 稼働物体の姿勢推定

  7.3.1節で述べた方法はこの実験では採用せず，ドアはあらかじめ開いて，図8.1のような姿勢にあるものとした．これは，ロボットの位置によっては，ドアが見えないことがあり，その場合に「存在しない」ことを判断するのが難しいという問題があるためである．

- マップ更新

  7.3.2節で述べた方法で，認識した物体の配置を更新した．

8.2 実験結果

ロボットは各ランドマーク物体を正しく認識し，自己位置推定を行って，走行に成功した．走行速度は経路線分によって異なるが，10〜20[cm/秒] である．図8.4に，走行軌跡を示す．基準実体の切替えに伴って，グローバル座標で見たロボット姿勢が大きくずれた部分の主なものに☆印をつけてある．このずれが大きい部分ほど，マップの誤差が大きい．たとえば，点 Q1 から Q2 にとんでいる部分は，基準実体を切替えたことによってロボット姿勢が修正されたことを表している．

図8.5に，復元した走行軌跡と物体姿勢の更新結果を示す．これから，マップがどの程度ずれていたかがわかる．ただし，更新された物体姿勢は必ずしも正確ではなく，まだ誤差が残っている．とくに，向きのばらつきが大きく，これによって経路の先の物体姿勢に影響がおよぶことがある．この原因は，物体認識で回転成分がわずかにずれた候補が複数個出力されるためである．

物体認識の結果は，上位5個の候補までを用いた．図8.6に物体認識の結果を示す．認識時間は画像と目標物体に依存するが，だいたい1〜5[秒] であった．false positive の削除を行うために認識を2回行うため，処理時間が余分にかかるが，マップからの姿勢制約を利用することで相殺した．

図8.7に，走行経路の各点におけるロボットの走行の様子を示す．実験の各試行において，走行経路のゴール点とロボットが実際に到着したゴール点のずれは，おおむね 30[cm] 以内に収まった．

8.3 実験の評価

この総合実験の目的は，本研究のアプローチの妥当性の評価である．この実験を通して得た評価と知見を以下に示す．

TG マップの目的は，環境変化に対するロバスト性とマップ作成工数の軽減であった．ま

ず，作成工数については，今回の実験環境程度の広さでは，TG マップは 10 分程度で容易に作成することができた．作成工数の大幅な削減に寄与したのは，メジャーによる計測をいっさい行わずに済んだことである．相対姿勢アークの誤差も，多くのものはデフォルト値で済ませているため，その設定に苦労することはなかった．また，室内は，机やキャビネットが規則正しく並んでいるので，その配置は TG マップエディタで簡単に行うことができた．マップ作成の容易さは，当然，環境の複雑さと広さに依存するが，上記の利点は，他の多くの屋内環境にあてはまると考えられる．

環境変化に対するロバスト性については，2 つの観点で評価できる．1 つは，前述のようにマップの不正確さと環境変化は表裏一体なので，それ自体が環境変化と見なせるという観点である．本実験で用いたマップには随所に 30[cm]〜2[m] 程度の誤差があり，その中でナビゲーションに成功したので，この観点では，十分にロバストであると言える．もう 1 つの観点は，この条件下で実環境に変化を加えるとどうなるかという観点である．これについては，変化を比較的与えやすいロボットの初期位置と L307 号室においた小机の位置について試した．その結果，ロボットの初期位置を 50[cm] 変えても，ナビゲーションは成功した．また，L307 号室の小机の位置を 1〜2[m] 変えても，小机が画像にきちんと写れば成功した．このことから，本研究で想定した程度のロバスト性はほぼ達成できたと思われる．

ただし，これは全般的なロバスト性を意味するわけではない．走行経路については，その作成自体はエディタで簡単にできるが，実際に走らせてみると目標物体以外の物体にぶつかって手直しが必要になることがあった．この問題は，基本的には，経路周辺の全物体が見えないと解決できない．これを物体認識で行うのは現状では困難なので，従来の障害物検出技術を用いるのが現実的であると思われる．

物体認識の成功率は高い．本実験の試行を何度か行ったときも，画像に物体が写っているのに認識に失敗した例は，紅茶箱を除けば，ほとんどなかった．紅茶箱については，発見できなかったり，発見できても線分の対応がずれていたことが散見された．また，ドアに関しては，姿勢推定の精度が足りず，わずかに姿勢の違う候補が複数個出力されることがあった．物体認識での姿勢推定に誤差があると，ロボットの方位がずれて，それ以降のカメラ視線がそれて目標物体が画像に写らないことがある．これによってナビゲーションが失敗することが何度かあった．この場合，リトライしても，カメラ視線が同じ方向を向いたままで失敗することが多い．これに対処するには，物体認識の姿勢推定の精度を上げる他に，周囲を見まわして目標物体を能動的に探す動作が必要であろう．

また，自己位置推定においては，モンテカルロ法の解のばらつきによって，ロボット姿勢がずれることがあった．これは走行シミュレーションでも観察されたので，ロボット走行や認識のばらつきとは別のものである．モンテカルロ法のサンプル数，サンプルからの代表姿勢の取り方，確率の下限閾値などが，自己位置推定法の検討すべき課題として残されている．

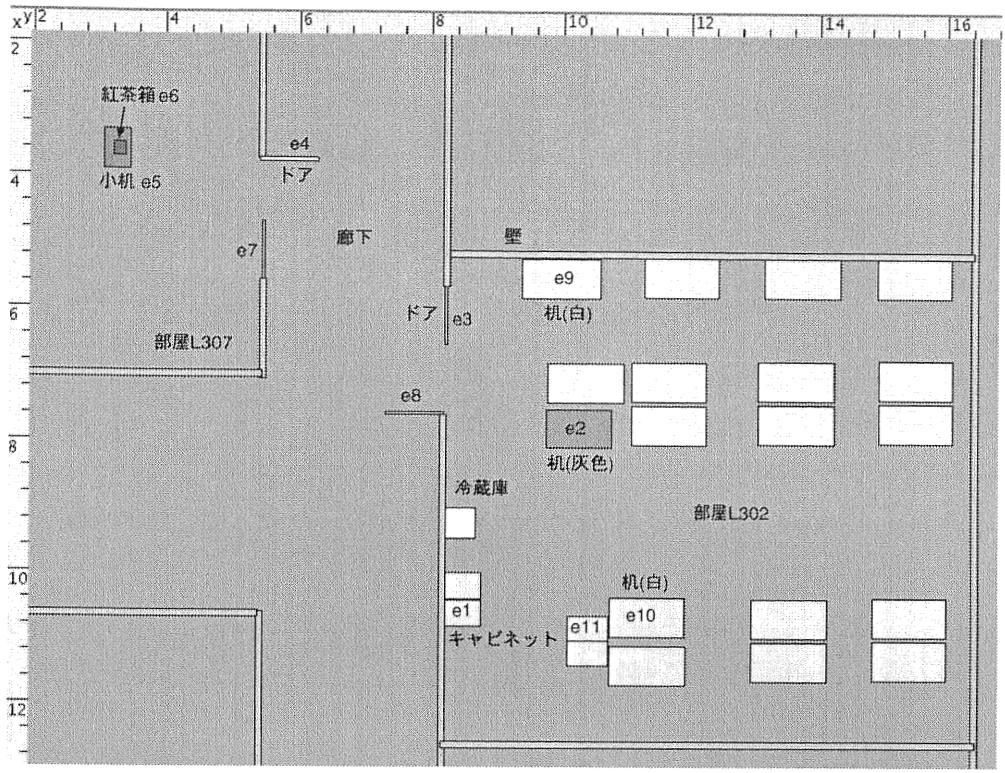


図 8.1: 実験で用いた TG マップ：実体の配置

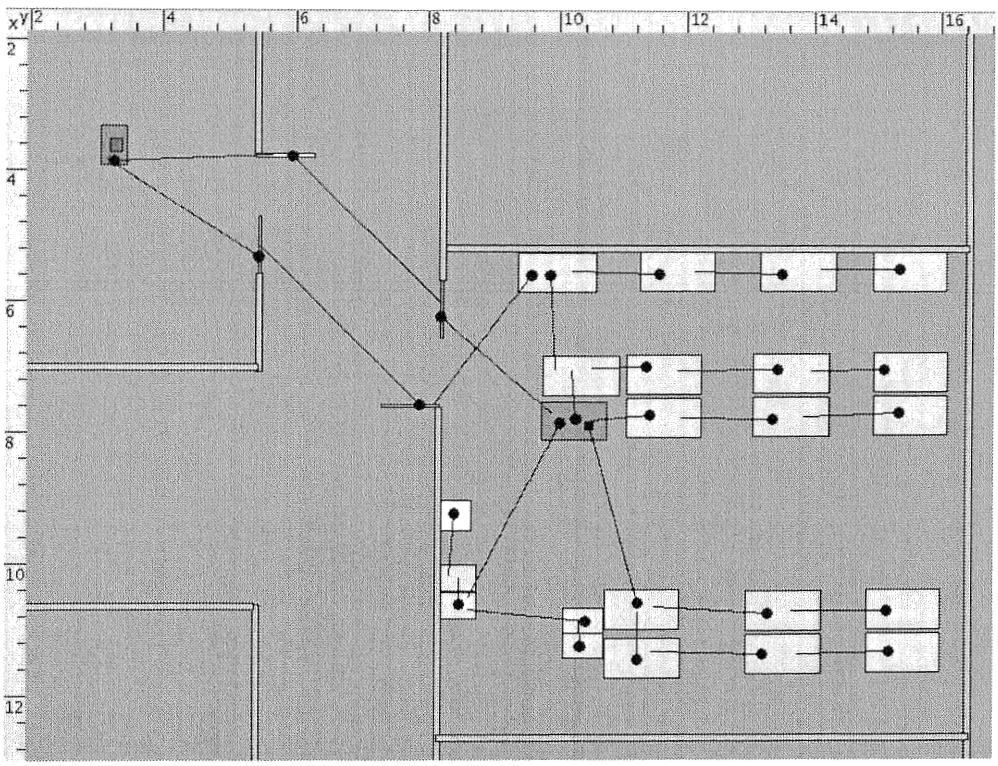


図 8.2: 実験で用いた TG マップ：相対姿勢アーク

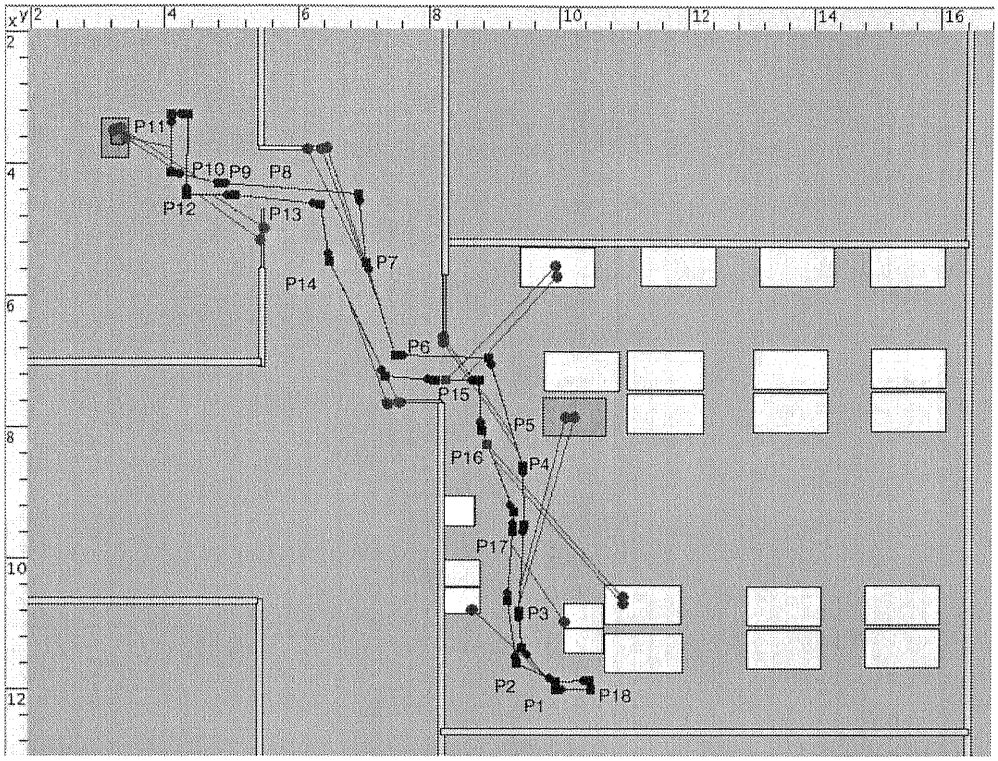


図 8.3: 走行経路

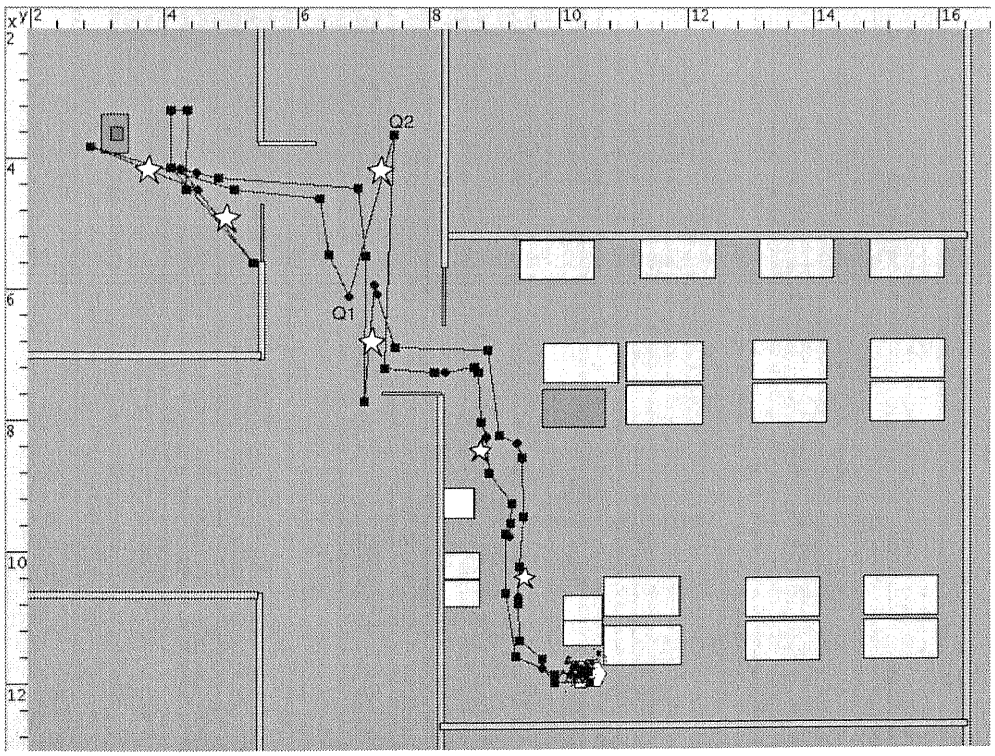


図 8.4: 走行軌跡

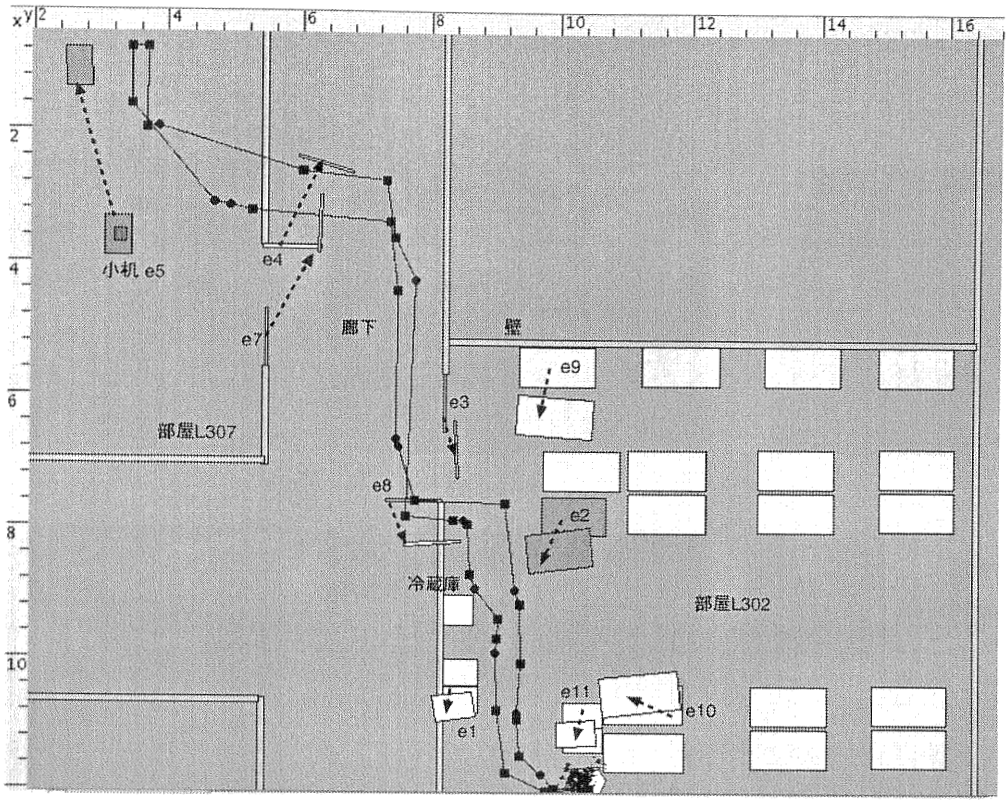


図 8.5: 修正した走行軌跡と実体配置の更新

図 8.6: 物体認識の結果

図 8.7: タスク実行の様子: () 内は走行経路上の点を表す

第 9 章

考察

第 1 章で述べたように，本研究の中心課題は，不正確さを許すマップを前提とした環境認識であった．環境認識として，とくに，ロボットの自己位置推定と環境内の物体姿勢の推定を対象とした．そして，これを実現するために，不正確さを許すマップの表現方法，その上での自己位置推定法，および物体認識方式を提案し，それらを統合したナビゲーション方式を提案した．

本章では，これらの目的が本研究でどの程度実現できたかを総括する．また，残された課題についても言及する．まず，中心課題である環境認識について考察し，その後，TG マップ，自己位置推定，物体認識，ナビゲーション方式の各技術について述べる．

9.1 不正確さを許すマップに基づく環境認識について

第 2 章で述べたように，マップに不正確さを許す主な利点は，環境変化への対処とマップ作成工数の軽減であった．マップ作成工数については次節で考察することとし，本節では環境変化への対処について分析する．

本研究では，不正確さを許すマップを用いて環境認識を行うために，物体をマップの構成単位とする，物体姿勢を他の物体からの相対姿勢で定義する，物体姿勢に誤差を陽に与える，という方針のもとで，TG マップを提案した．そして，物体を単位に認識を行い，ロボット姿勢を基準実体とそこからの相対姿勢の対で表すことで，不正確なマップの上でのロボットの自己位置推定が可能になった．また，7.3節で述べたように，自己位置推定によるロボット姿勢の修正が反映された走行軌跡を用いて，環境内の物体の姿勢を求めることができた．ロボットの自己位置推定と環境内の物体姿勢の推定が 2 つとも実現できたので，本論文で定めたマップの不正確さへの対処は達成したと言える．

マップの不正確さへの対処で本質的な役割を果たすのは，実体を基準にしたロボット姿勢の表現である．これにより，マップと実環境のずれによって，グローバル座標系でロボット姿勢を表すと一貫性がとれなくなる問題を解決できた．また，基準実体を切替える時に，

物体間の相対姿勢の誤差をロボット姿勢の分布に上乗せして拡散することで，不正確なマップでもデータ融合がうまく行えるようになった．

ロボット姿勢を求める過程で，個体同定と姿勢推定もできる．これを用いると，環境内で位置が変化した物体を見つけることができる．これは正確なマップを用いる場合にも有効である．正確なマップを用いるナビゲーションでは，通常，位置が変化しにくい物体だけでマップを構成する．そして，それらの物体をランドマークとして自己位置推定を行うのが一般的である．このような場合は，ロボット姿勢をグローバル座標系で表すことが可能だが，位置が変化しやすい物体を見つけることは簡単ではない．これまで述べてきたように，安定した物体認識のためにはマップからの予測を用いることが有効だが，物体の位置が変化しやすい場合は，それを補正して予測を行う仕組みが必要になる．本方式では，この予測が自己位置推定でのデータ融合の中に自然に組み込まれているので，位置が変化しやすい物体の同定にも有効である．

なお，環境変化に対処することは，本質的にはマップを逐次更新していくことである．そこでは，1.1.3節で述べたように，ロボットの自己位置推定と物体の姿勢推定を同時に行うことが必要になる．これは，SLAM(Simultaneous Localization and Mapping) と呼ばれ，さかんに研究が行われている．本研究もマップ更新を含むので，一種の SLAM を行っているととらえることができる．ただし，多くの SLAM 研究では，最尤推定などを用いて誤差が最小になるようにロボット姿勢と物体姿勢を決めていく [12, 50, 81, 96]．本研究では，このような最適な更新は行っておらず，今後の課題として残されている．

9.2 TG マップについて

まず，マップに不正確さを許すもう 1 つの利点であるマップ作成工数の軽減について考察する．TG マップでは物体を部品化して，床面上に配置する方式をとっている．物体姿勢は不正確でよいため，実環境の計測がほとんど必要なく，簡単にマップを作成できる．実際に，総合実験で用いた規模のマップならば，TG マップ作成エディタを用いて 10 分程度で作れることを確認した．部品の充実は課題であるが，屋内にある物体の多くは，床面への投影像を長方形で表せるので，2 次元マップであれば部品の作成は容易である．

このように，TG マップは作成工数が小さくて済むため，ロボットによるマップ構築にも応用できると考えられる．すなわち，全くの未知環境でロボットがマップを自動構築するのは負荷が大きいので，人間が与えた大雑把な初期マップをもとに，ロボットがマップを精密化していくことでマップを構築する．このようにして作成されたマップでは，個々の物体が分離されて識別可能になっているため，従来の環境の形状だけを記述したマップに比べて有用性が高い．

TG マップの特徴として，実体の姿勢をすべて相対的に計算することがあげられる．これ

には，次のような利点がある．もし，TG マップにおいて実体姿勢をグローバル座標系で表すと，実体間の相対姿勢の誤差が累積されて，推定姿勢は実際よりもはるかに広い範囲に拡散する．このため，実体間の幾何学的制約がぼやけて，ロボットの姿勢推定の精度は著しく悪くなる．これに対して，実体姿勢を相対座標で計算すれば，誤差はほぼ最小限ですみ，実体間の幾何学的制約をうまく生かすことができる．これは，実体姿勢の共分散行列で考えると，相関項を利用していることに相当する．TG マップはグラフ表現なので，注目している実体間の相関項だけを抽出することができ，全実体姿勢の共分散行列を扱うのに比べて，次元数がはるかに少なくて済むという利点がある．

TG マップの構成要素である実体には領域と物体があるが，本論文ではほとんど物体だけを扱ってきた．しかし，領域をうまく活用すれば，広い環境を階層的に表現できるので，マップの作成効率が向上すると期待される．また，現状では相対姿勢アークを物体間に直接張ることで相対姿勢を設定しているが，物体の相対姿勢をそれを含む領域に対してだけ定義するようにすれば，相対姿勢の設定は非常に容易になる．この場合，物体間の相対姿勢とその誤差は，一方の物体から領域を経由して他方の物体に姿勢の座標系を変換すれば計算できる．このような領域中心の相対姿勢定義と現在用いている相対姿勢アークとを併用することで，より柔軟な TG マップの作成が可能になると期待される．これについては，今後の課題として残されている．

9.3 自己位置推定について

本研究の自己位置推定の目的は，不正確なマップの上で実環境と整合のとれたロボット姿勢を得ることであった．これを実現するために，ロボット姿勢をグローバル座標系ではなく，基準実体とそこでの相対姿勢の対で表現し，移動につれて基準実体を次々と切替えるようにマルコフ姿勢推定を拡張する方式を提案した．そして，実際にこの方式を実装して実験を行い，マップが不正確な場合でも，自己位置推定が正しく行われることを確認した．

実装ではモンテカルロ法を用いているが，本方式は，物体単位でサンプルをまくので，領域全体にサンプルをまく従来のモンテカルロ法より効率がよいという特徴がある．とくに，走行経路が決まっていれば，ランドマークの物体を限定できるので，第 4章のように，マップ内の同クラスのすべての実体に対して計算する必要がなくなり，計算効率は非常によい．

残された課題として，モンテカルロ法の改良がある．これにはまず，モンテカルロ法で得られたサンプルの分布からロボット姿勢を１つに決定する問題がある．現在は，単純にサンプル分布の重心をロボット姿勢の代表値にしているが，本来は，サンプルのクラスタリングを行ったうえで確率が最大となるロボット姿勢を求める必要がある．また，モンテカルロ法では乱数を用いるため，試行によりばらつきがある．現状では，このばらつきによるロボット姿勢のずれがナビゲーションに悪影響を与えることがある．上記のようなロボット姿勢の

代表値の計算とともに，精度と効率を考慮した適切なサンプル数の設定が重要である．

9.4 物体認識について

目標タスクを行うには，指定された物体やナビゲーションに用いるランドマークを，不正確なマップを用いて見つけなくてはならない．しかも，ロボットが活動する環境では，照明条件、オクルージョン、混雑環境などの問題が多く現われる．

本論文では、この問題に対処するために，多段階フィルタリングによりロバストに物体認識と姿勢推定を行う方式を提案した．とくに，姿勢空間分解法により段階的に照合を行うことで、大幅に計算量を削減した．この方式の特徴の 1 つは，姿勢の回転成分を離散化して全探索し，フィルタリングによって候補を絞り込んでいくため，姿勢方程式の直接計算を最小限にできる点である．本方式で方程式を解くのは，2 点の対応から並進成分を求める部分であるが，非常に単純な線形計算であり，安定して解を求めることができる．これは，4 次方程式や数値解法など複雑な計算を必要とする姿勢推定法に比べて優位な点であると思われる．

また，この方式では，姿勢の回転成分や物体の高さなど，不正確なマップからでも比較的正確に得られる情報を制約条件として使って，認識率と処理時間を向上できる．7.1節で述べたように，実際に実験でその有効性を示した．目標物体や画像に依存するが，マップからの制約を用いれば，認識率はほぼ 80〜98%，処理時間は 0.5〜2 秒であり，総合実験によって実環境でのナビゲーションに使えることが確認された．

7.1節で示したように，認識率は目標物体によりばらつきが大きいが，傾向として，特徴の数が多いほど認識率は高い．このことから，物体モデルをもっと精密にすれば，認識率が向上すると考えられる．また，現在の画像は，解像度が 320 × 240，階調が 16bit であり，画質が十分ではない．この改善によっても認識率が向上すると思われる．

残された課題として，姿勢推定の精度向上がある．現在は，姿勢がわずかに異なる候補が多数出力されるため，ロボットの自己位置推定の誤差が大きくなることがある．精度を高めるには，候補を絞りながら回転成分の離散化や並進成分の投票ビンを段階的に細かくしていく方法が考えられる．また，本方式で求めた姿勢値と特徴対応に対して，ニュートン法などの数値解法による姿勢推定法を適用して高精度の解を求める方法も有効であろう．

9.5 ナビゲーション方式について

不正確なマップを用いたナビゲーションでは，マップから得られた走行経路も不正確であるため，ロボットが走行時に経路を修正する必要がある．このため，ロボット姿勢と同様に経路位置も基準実体からの相対位置で表すことにし，ロボットが基準実体を認識して，その姿勢が得られた時点で経路位置を修正する方式を提案した．そして，シミュレーションお

よび実験で，この方式が有効に機能することを確認した．本方式は，基準実体をベースにしているので，ロボットの自己位置推定と非常に整合性がよい．

残された課題として，経路計画がある．現在は，経路は人手で与えて，ロボットは経路の位置だけを修正する．ロボットに自律性を持たせるには，目的地を与えてそこまでの経路を計画する機能が必要である．この場合，トポロジー的な探索と物理的に通行可能な領域の探索の両方が必要になる [87]．通行可能領域の確認は安全な走行に不可欠であり，このためには，走行中に基準実体の周辺の物体を認識して，その位置を確認する必要がある．これにともなって，計画していた経路が通行不能であることがわかり，経路のトポロジー的な再計画が必要になる可能性がある．これは，マップ修正と経路修正が関係する興味深い課題である．

第 10 章

まとめ

本論文では，移動ロボットが環境の不確実性に対処してロバストに行動することを目的として，マップの不確実性に焦点を当て，不正確さを許すマップの構成方法，不正確なマップにおける移動ロボットの自己位置推定法，単眼ビジョンを用いた物体認識方式，および，これらに基づいたナビゲーション方式を提案した．これらの各要素技術について試作システムを作成し，実験によってその有効性を確かめた．また，移動ロボット「山彦」のうえに統合システムを構築して，指定した物を見つけてくるという目標タスクを実現することで，これらの方式が実環境で有効に機能することを確認した．

従来の移動ロボットのナビゲーションの研究では，オドメトリと外界センサの不確実性に対処したものがほとんどであり，不正確なマップを前提とした研究は少ない．本研究は，物体を環境認識の処理単位として，確率をベースにした従来の枠組をマップの不正確さを扱えるように拡張したものであり，その点に新規性がある．本研究で提案した方式は，マップの構成，ロボット姿勢，経路走行のすべてにおいて，物体を基準にした相対姿勢を用いて表現する．そして，物体の近傍ではその相対姿勢で行動し，その物体から離れていくと，次の物体に基準を移すことで，ロボットの走行領域を広げていく．離れた物体間ではマップの誤差が大きくなるが，物体認識と自己位置推定によってその誤差を補正する．このようにロボット姿勢の基準を次々と切替えることで，局所領域内でのロボット姿勢を常に正しく保つ点に本方式の特徴がある．

本研究の中心課題は不確実性であるが，人間がふつうに活動する環境の中で種々の物体を認識して行動するロボットの端緒を開いたことも重要な成果である．従来の移動ロボットは場所を認識することが主流であり，環境内の様々な物体を認識して行動するものはほとんどなかった．物体認識を行う場合でも，専用に整備された認識しやすい条件が設定されていた．物体単位で環境を認識できれば，マップの不正確さに対処できるだけでなく，物体モデルに種々の知識を付加することで，環境に対する様々な推論や予測が可能になる．これが実現すると，ロボットが行うタスクの記述や実行制御が飛躍的に高度化することが期待できる．

本研究で実現した物体認識は限られたクラスのものであるが，このようなロボットの高機能化に向けても，本研究は大きく貢献したと信ずる．

謝辞

本研究は，工学博士 油田信一 筑波大学機能工学系教授の御指導のもとで行われたものです．同教授には，研究を進めるに際して，昼夜を問わず適切かつ熱心な御指導と御助言を頂きました．

筑波大学機能工学系知能ロボット研究室の諸氏には，本研究のためのシステム開発や環境整備において多くの御協力を頂きました．また，研究打合せや研究室主催のシンポジウムなどでの討論を通じて，多くの有意義な御指摘と御助言を頂きました．

また，工学博士 宮本定明 筑波大学機能工学系教授，工学博士 中村裕一 筑波大学機能工学系助教授，工学博士 相山康道 筑波大学機能工学系講師，工学博士 三浦純 大阪大学工学研究科助教授には，本論文をまとめるにあたって，適切な御指導と御助言を頂きました．

以上の皆様に対して，ここに深く感謝いたします．

付録 A

システムアーキテクチャ

本章では，本研究で開発したシステムの構成を説明する．システムのソフトウェアの実装にあたっては，拡張性を高めるためにオブジェクト指向言語 Java を用いて機能の部品化を図った．また，操作性を高めるために，GUI を用いたユーザインタフェースを充実させた．しかも，様々なコマンドを，GUI だけでなくプログラムからも実行できるインタフェースを提供し，ロボットの行動プログラムの記述性の向上を図った．

A.1 ハードウェア構成

研究に用いた移動ロボットの構成を図 A.1に示す．このロボットは，筑波大学知能ロボット研究室で開発された移動ロボットプラットフォーム「山彦」の上に，ノート PC，パンチルトカメラを載せて構成されている．図 A.2にハードウェア構成の詳細を示す．

各構成要素の役割を以下に示す．

- 山彦
  直線追従走行，円弧追従走行，自転，停止などの動作を行う．前後左右に 1 個ずつ超音波センサをもち，障害物を検出することができる．また，前方下部のバンパにより，障害物との接触を検出できる．本システムでは，これらのセンサは用いず、ノート PC からコマンドを受けて経路に沿った走行だけを行う．

- パンチルトカメラ
  パン (左右首振り)，チルト (上下首振り)，ズームなどの機能をもつ．また，ホワイトバランス機能や自動露光調節機能もある．本システムでは，ノート PC からコマンドを受けて，環境内の目標物体の画像を撮影する．

- ノート PC
  山彦，パンチルトカメラを含めたシステム全体の制御を行う．カメラ画像からの物体認識，ロボットの自己位置推定，ナビゲーション制御などを行う．

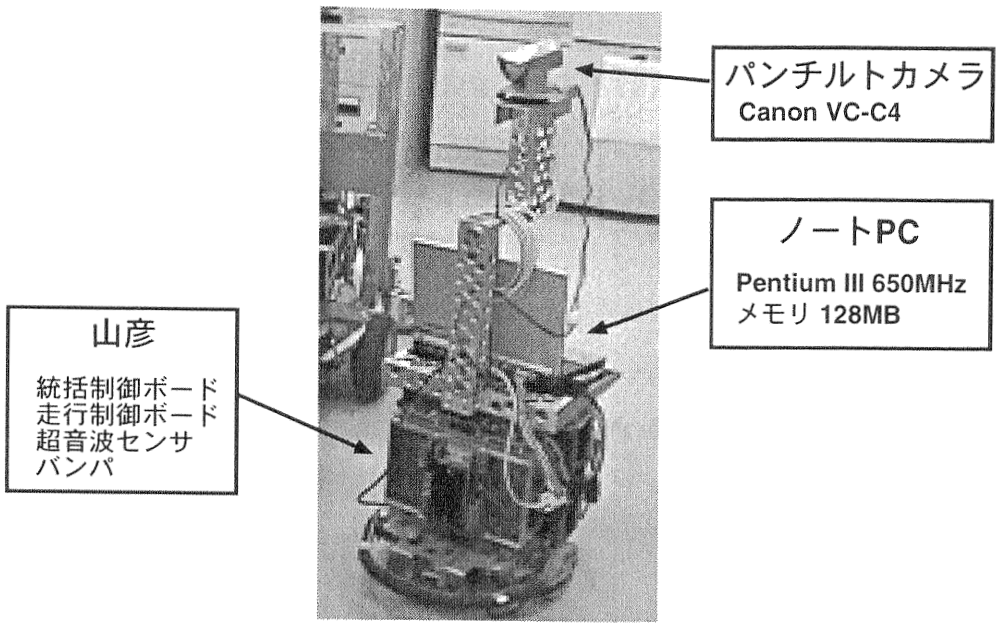


図 A.1: ロボット構成 (2 号機)

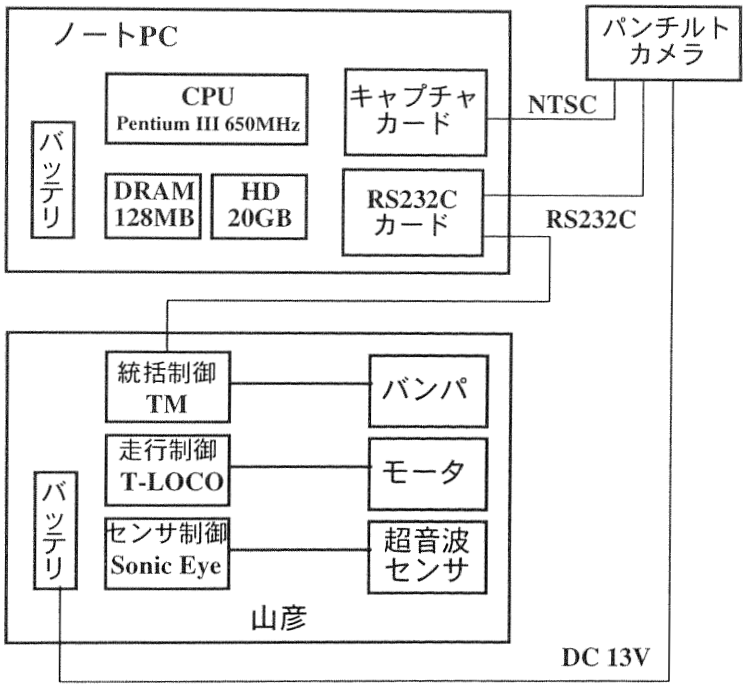


図 A.2: ハードウェア構成

A.2 ソフトウェア構成

ソフトウェア構成を図 A.3に示す．山彦側のソフトウェアは，筑波大学知能ロボット研究室で開発されたロボット用 OS である Liberos [61] の上に，C 言語を用いて作成した．ノート PC 側のソフトウェアは，Linux の上に Java (IBM JDK 1.3) を用いて作成した．

同図で網掛けしたものが，本研究で開発したソフトウェアである．それぞれ，以下に説明する．

- 走行制御部
  ノート PC から山彦の走行を制御するためのプログラムである．ノート PC 側の走行制御プログラムとその操作 GUI，山彦側の走行コマンドサーバと PC 通信プログラムからなる．

- カメラ制御部
  ノート PC からパンチルトカメラの操作を行うプログラムである．カメラ制御プログラムと画像キャプチャプログラム，および，その操作 GUI からなる．

- TG マップ部
  TG マップに基づいてナビゲーションを行うためのプログラム群である．TG マップの基本データ構造，自己位置推定プログラム，経路走行プログラム，TG マップエディタ，経路作成エディタ，タスク起動 GUI(タスクランチャ) からなる．

- 物体認識部
  単眼カメラの画像から物体認識を行うためのプログラム群である．物体認識を行うための様々な画像処理オペレータとそれを起動する GUI(画像処理ワークベンチ) からなる．

- 動作スクリプト部
  ロボットの行動プログラムを記述するための雛型となるプログラム群である．具体的には，ロボット動作の基盤となる Java オブジェクトが定義されている．動作の開始/終了/同期の制御を記述する関数インタフェースをもつ．タスクのためのロボット動作は，このクラスを継承して作成する．また，走行経路からロボットの走行プログラムを自動生成する機能を含む．

- 基本 GUI
  マップ表示画面，画像表示画面，コマンド実行画面などから構成される．

この他に，TG マップ用の 2 次元モデルや物体認識用の 3 次元モデルの定義プログラムがある．また，RS232C 通信や行列計算などのユーティリティプログラムもある．

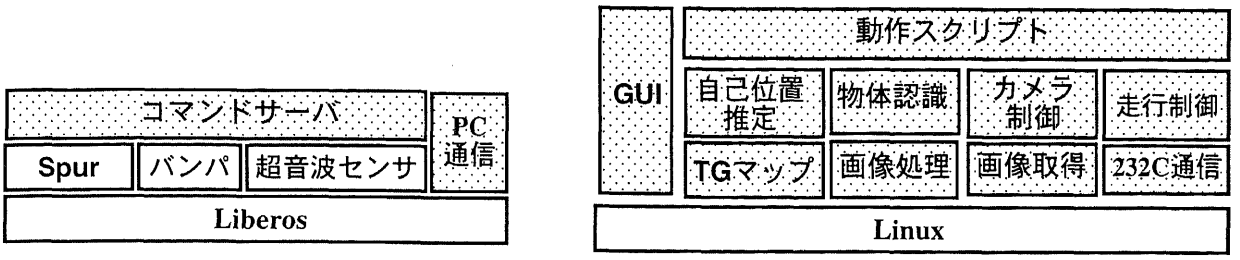


図 A.3: ソフトウェア構成

表 A.1: プログラムの規模

サブシステム	規模 (行)	マシン	言語	備考
走行制御	2.1K	PC	Java	
コマンドサーバ	1.3K	山彦	C	PC 通信プログラム (asagiri2) を含む
カメラ制御	2.8K	PC	Java, C	画像キャプチャを含む
TG マップ	12.1K	PC	Java	自己位置推定，マップエディタ，経路エディタ，タスク起動 GUI を含む
物体認識	17.5K	PC	Java	画像処理，起動 GUI を含む
動作スクリプト	2.0K	PC	Java	
基本 GUI	3.5K	PC	Java	
モデル定義	5.9K	PC	Java	マップ用モデル，物体認識用モデル
その他	1.4K	PC	Java, C	RS232C 通信，ユーティリティなど
合計	50K			

付録 B

ツール

本研究で開発した，TG マップや走行経路を作成するためのエディタ，ロボットのタスク起動 GUI，画像処理 GUI について説明する．

B.1 TG マップエディタ

TG マップを作成するための専用エディタである．主な機能を以下に示す．

- 実体の生成
  実体モードにおいて，実体クラスを一覧からマウスで選択すると，そのインスタンスが生成されて，マップ上に配置される．その後，実体の姿勢を変えるには，後述の編集機能を使う．

- 相対姿勢の設定
  相対姿勢モードにおいて，マウスで 2 つの実体をクリックすると，それらの実体の間に相対姿勢アークが張られる．相対姿勢アークは，実体間の相対姿勢とその誤差をデータとして持つ．このうち，相対姿勢はマップ上の 2 つの実体の姿勢から自動的に計算される．編集によってどちらかの実体の姿勢が変更された場合も，相対姿勢は自動更新される．

  相対姿勢の誤差は，ユーザが数値で与える．そのデフォルト値は (10[cm], 10[cm],3[度]) である．

- 編集機能

  - 平行移動
    マウスでマップ上の実体をドラッグすることで，実体の平行移動ができる．

- 回転
  マウスでマップ上の実体を選択し，回転角度を指定するスライダを操作することで，実体の回転ができる．
- スケール
  マウスでマップ上の実体のハンドル (実体輪郭上の小さな黒い正方形) を動かすことにより，実体のサイズを変えることができる．現在，これが可能なのは Wall の長さだけである．
- グルーピング
  マウスでマップ上の複数の実体を選択し，Group ボタンを押すと，それらの実体が 1 つにグルーピングされる．UnGrp ボタンを押すと，もとの実体に分解される
- 複製
  マウスでマップ上の実体を選択し，Dup ボタンを押すと，その実体の複製が生成される．
- コピー/ペースト
  マウスでマップ上の実体を選択して Copy ボタンを押すと，その実体がクリップボードに登録される．Paste ボタンを押すと，その実体の複製がマップ上に配置される．異なるマップから実体を複製したい場合に有用である．

この他，実体の削除や表示順序の変更 (一番上，一番下) を行うことができる．

- 表示機能
  スクロール，拡大縮小表示ができる．また，実体名の表示/非表示，実体のローカル座標系の表示/非表示，相対姿勢の誤差範囲の表示/非表示の選択ができる．

TG マップエディタの画面例を図 B.1に示す．

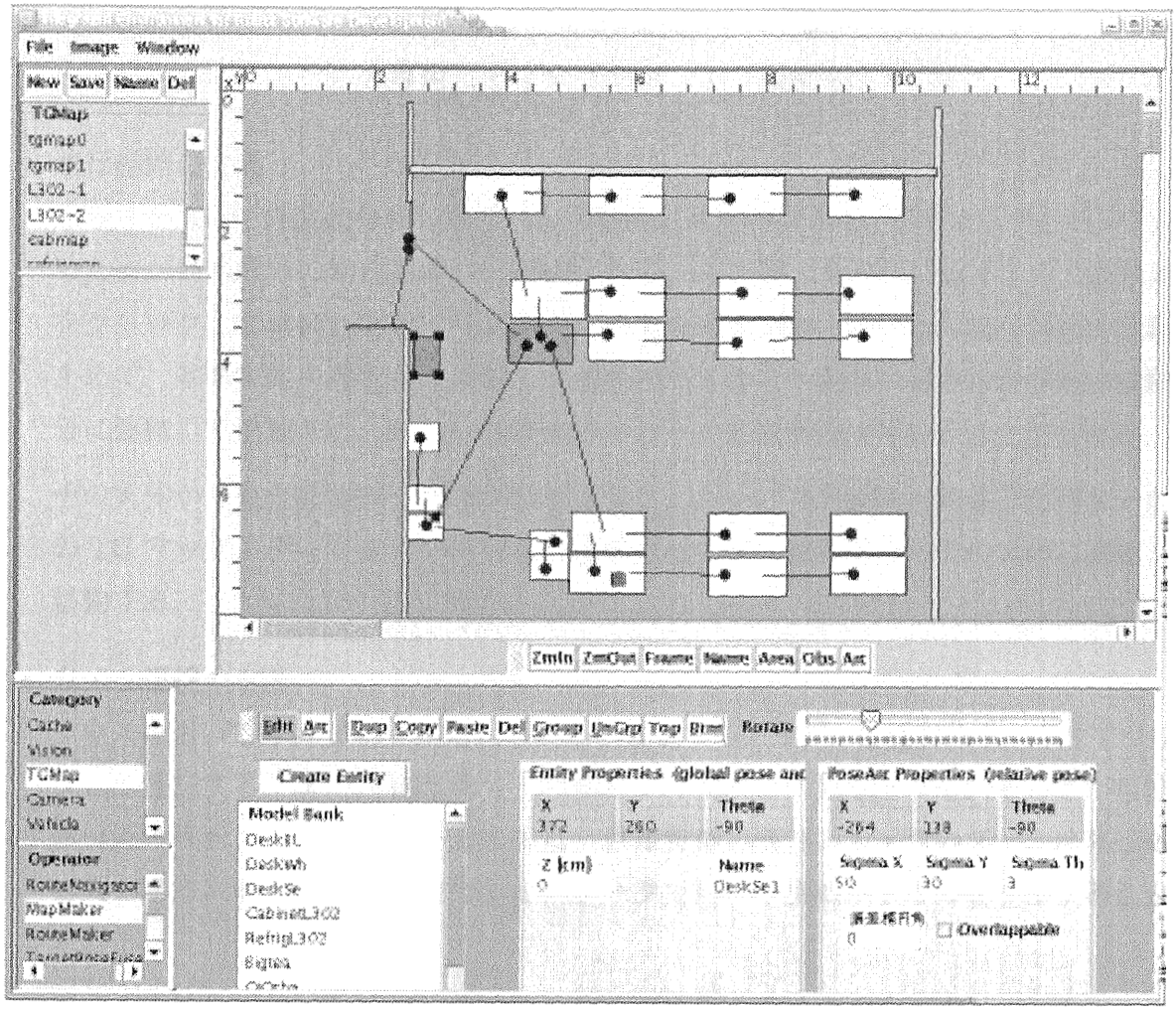


図 B.1: TG マップエディタの画面

B.2 走行経路エディタ

走行経路を作成するための専用エディタである．主な機能を以下に示す．

- 走行経路の定義
  マウスを用いて経路線分をつなぐことで走行経路を作成する．TG マップ上でクリックした点が経路線分の始点あるいは終点となる．この際，あらかじめマウスで基準実体をクリックして選択しておく．基準実体を変える場合は，いったん走行経路の定義を止めて，基準実体を選択しなおして定義を再開すると，新しい経路線分が前の経路線分に連結して追加され，新しい基準実体のもとで定義を続けていくことができる．

  その実体から見た経路線分の相対位置は変化しない．走行経路を作成した後で，前述の TG マップエディタを用いて実体を移動させると，経路線分もその実体に追随して移動する．

- カメラ視線の定義
  マウスを用いてカメラ視線の始点と終点を指定する．まず，目標実体をマウスでクリックして選択する．次に，経路線分上の点をクリックして始点とする．そして，目標実体付近の点をクリックして終点とする．目標実体の高さとカメラのズーム比率は数値で入力する．また，後述の走行シミュレータで用いる画像ファイルを登録することができる．

  カメラ視線を定義した後で，経路線分や目標実体を移動させると，カメラ視線もそれに追随して移動する．

- 走行経路の登録管理
  1 つの TG マップに複数の走行経路を定義できる．作成した走行経路は一覧に登録される．

走行経路エディタの画面例を図 B.2に示す．

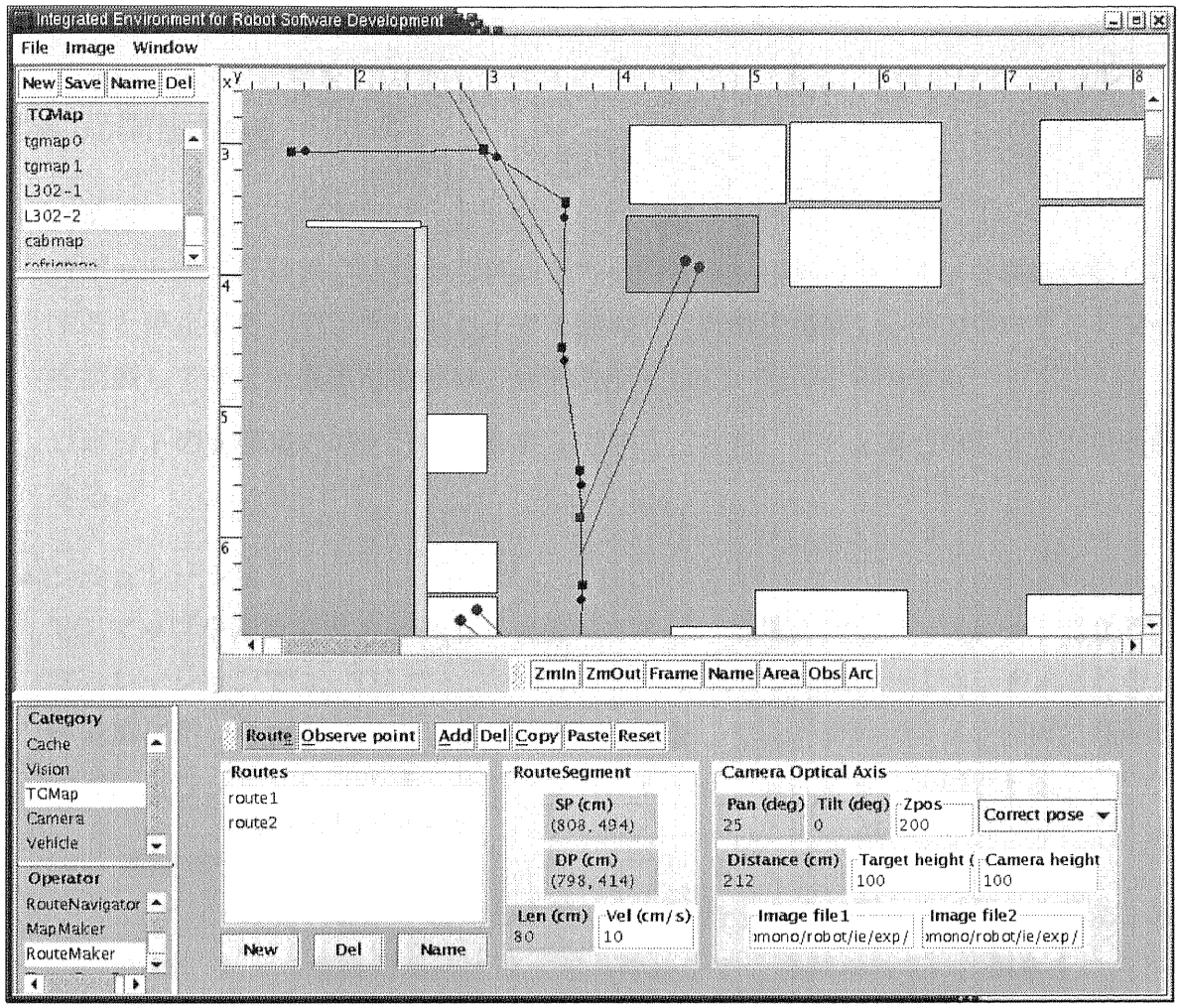


図 B.2: 走行経路エディタの画面

B.3 タスクランチャ

ロボットのタスク起動をGUIから操作するプログラムである．実機走行とシミュレーションの両方が可能である．

- 実機走行
  ロボットが走行経路の始点から終点まで走行する．移動の際，カメラ視線の始点に達すると停止して，パンチルトカメラを操作して画像を撮影し，その中のランドマークを認識して自己位置推定を行い，経路を修正して，走行を再開する．

  ロボットの実環境での走行軌跡を表示できる．また，それに基づいてマップ更新ができる．

- シミュレーション
  TG マップや走行経路のデバッグ支援のために，実機走行をシミュレートできる．シミュレーションでは，TG マップ上でロボットの走行がアニメーション表示される．物体認識に用いる画像は，あらかじめ実機で撮影して走行経路のカメラ視線に登録しておく．走行シミュレータは，以下の 2 つのモードで実行することができる．

  - 連続実行モード
    ロボットが走行経路を始点から終点まで走行する様子をシミュレートする．移動の際，カメラ視線の始点に達すると停止して，登録されていた画像をロードし，その中のランドマークを認識して自己位置推定を行い，経路を修正して，走行を再開する．

    ロボットの実環境での走行軌跡を表示できる．それに基づいてマップ更新ができる．

  - ステップ実行モード
    マウスでボタンを押すたびに，ロボットが走行経路上を一定区間だけ走行する．移動区間の単位は，ロボットの現在位置から経路線分の終点まで，あるいは，カメラ視線の始点までである．カメラ視線の始点に移動した場合，登録されていた画像をロードして，その中のランドマークを認識して自己位置推定を行い，経路を修正する．

    このモードでは，物体認識から推定したロボット姿勢の仮説を表示できる．また，基準実体の切替えによるロボット姿勢分布の拡散の様子を表示できる．これらは，相対姿勢アークのデバッグなどに有用である．

タスクランチャの画面例を図 B.3に示す．

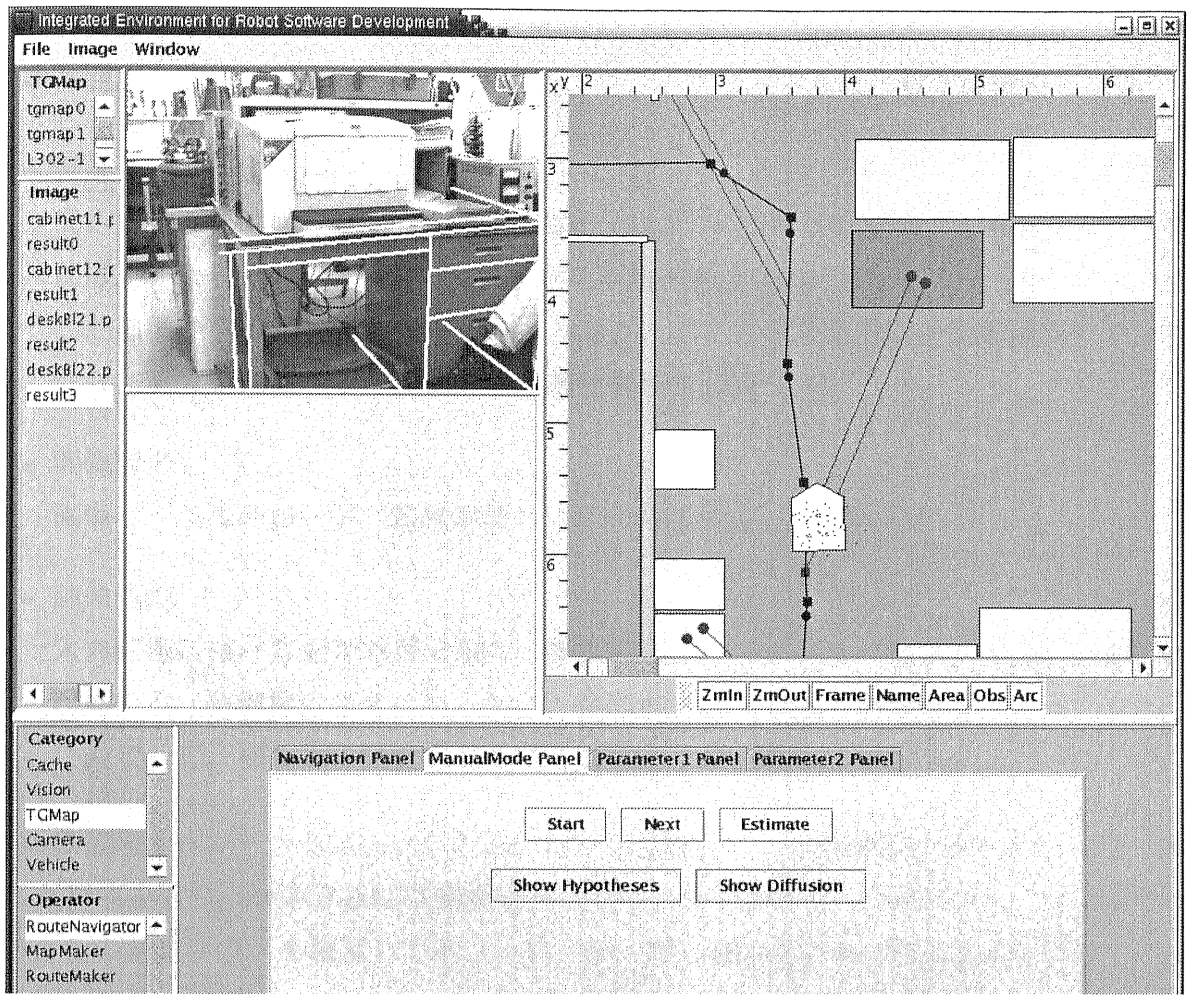


図 B.3: タスクランチャの画面

B.4 画像処理ワークベンチ

物体認識や画像処理をGUIで対話的に行うプログラムである．以下の機能をもつ．

- 画像表示
  入力画像と処理画像を表示するための2つのウインドウをもつ．入力画像ウインドウには，画像一覧からマウスで選択された画像を表示する．また，カメラからキャプチャした画像やファイルからロードした画像も，ここに表示される．処理画像ウインドウには，画像処理オペレータによる処理結果が表示される．

- 画像管理
  画像ファイルのロード，処理画像のファイルへのセーブを行う．

- 画像処理
  入力画像に様々な画像処理を施す．画像処理オペレータを一覧からマウスで選択して起動する．各画像処理オペレータにも専用のGUIがあり，種々のパラメータを対話的に設定して，画像処理を行うことができる．

画像処理オペレータはJavaオブジェクトで定義され，処理機能と操作用GUIが一体となった構成をもつ．システムには画像処理オペレータの雛型となる基底クラスが用意されており，その基底クラスを継承して新しいオペレータの追加を容易に行えるようになっている[86]．

画像処理オペレータとして，画像微分，エッジ抽出，エッジ分割，エッジグルーピング，エッジ色抽出，物体認識，3次元モデル表示，文字認識，表色系変換，階調変換，二値化，ガウスフィルタなど，様々なものを用意している．

画像処理ワークベンチの画面例を図B.4に示す．

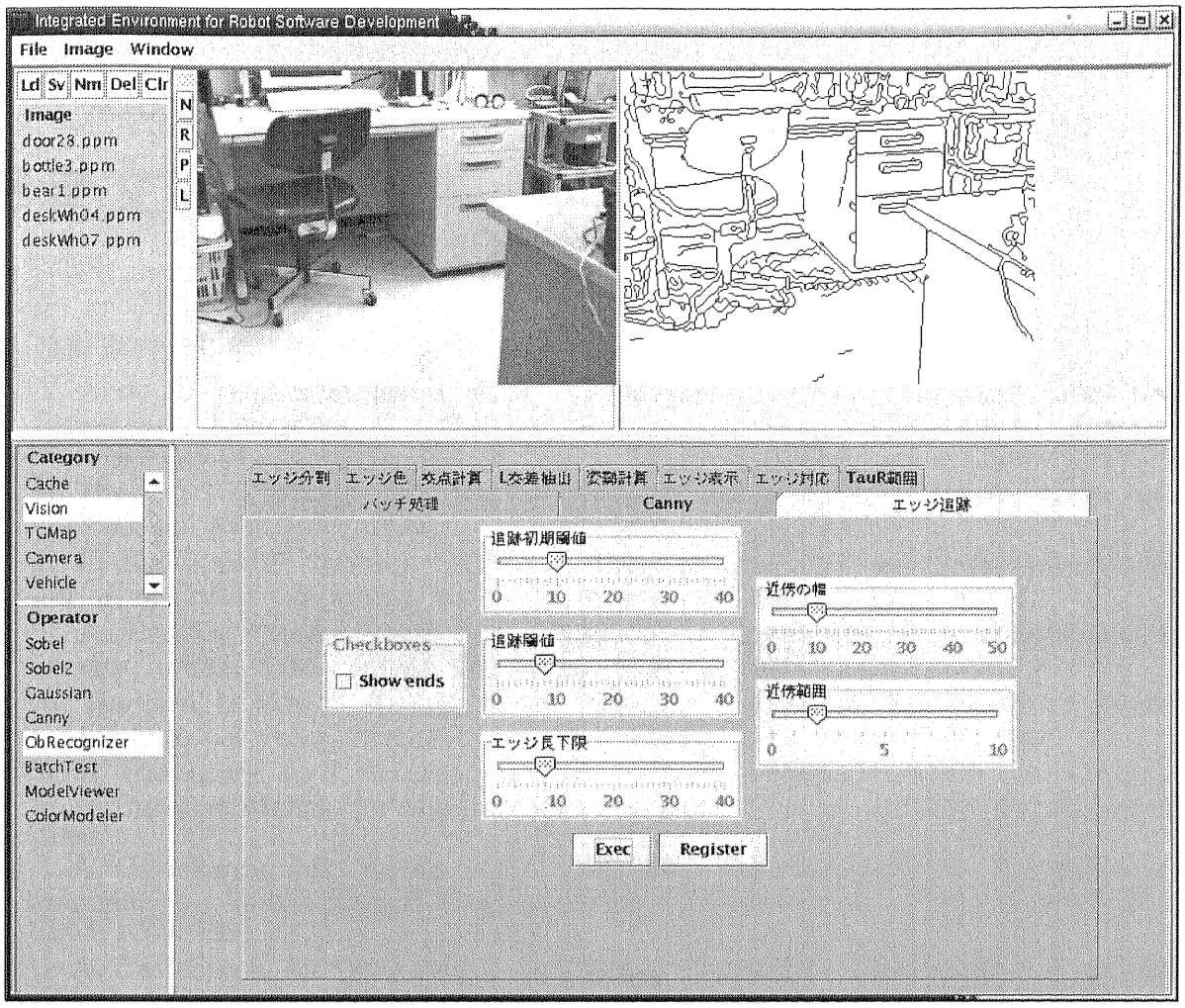


図 B.4: 画像処理ワークベンチ

参考文献

[1] 浅野郡司: "知能ロボット", 東京電機大学出版局, 1990.

[2] Y. Ando, T. Tsubouchi, and S. Yuta: "Development of Ultra-Wide-Angle Laser Range Sensor and Navigation of a Mobile Robot in a Corridor Environment," *Journal of Robotics and Mechatronics*, Vol. 11, No. 1, pp. 25-32, 1999.

[3] R. Basri and E. Rivlin: "Homing using Combinations of Model Views," *Proc. of IJCAI'93*, pp. 1586–1591, 1993.

[4] R. C. Bolles and P. Horaud: "3DPO: A Three-Dimensional Part Orientation System," *Int. J. of Robotics Research*, Vol. 5, No. 3, pp. 3-26, 1986.

[5] R. A. Brooks: "Model-Based Three-Dimensional Interpretations of Two-Dimensional Images," *IEEE Trans. PAMI*, Vol. PAMI-5, No. 2, 1983.

[6] W. Burgard, D. Fox, D. Hennig, and T. Schmidt: "Estimating the Absolute Position of a Mobile Robot Using Position Probability Grids," *Proc. of AAAI'96*, 1996.

[7] A. R. Cassandra, L. P. Kaelbling, and J. A. Kurien: "Acting under Uncertainty: Discrete Bayesian Models for Mobile-Robot Navigation," *Proc. of IROS'96*, pp. 963–972, 1996.

[8] J. A. Castellanos, J.D. Tardós, and G. Schmidt: "Building a Global Map of the Environment of a Mobile Robot: The Importance of Correlations," *Proc. of ICRA'97*, pp. 1053–1059, 1997.

[9] J. A. Castellanos, M. Devy, and J. D. Tardós: "Towards a Topological Representation of Indoor Environments: A Landmark-based Approach," *Proc. of IROS'99*, pp. 23–28, 1999.

[10] J. Canny: A Computational Approach to Edge Detection, *IEEE Trans. PAMI*, Vol. 8, No. 6, pp. 679-698 (1986).

[11] S. Chen and W. Tsai: "A Systematic Approach to Analytic Determination of Camera Parameters by Line Features," *Pattern Recognition*, Vol. 23, No. 8, pp. 859-877, 1990.

[12] I. .J. Cox and J. J. Leonard: "Modeling a dynamic environment using a Bayesian multiple hypothesis approach," *Artificial Intelligence* 66, (1994), 311-344.

[13] H. I. Christensen, N. O. Kirkeby, S. Kristensen, L. Knudsen, and E. Granum: "Model-driven vision for in-door navigation," *Robotics and Autonomous Systems* 12 (1994) 199-207.

[14] H. Bulata and M. Devy: "Incremental Construction of a Landmark-based and Topological Model of Indoor Environments by a Mobile Robot," *Proc. of ICRA'96*, pp. 1054–1060, 1996.

[15] M. Dhome, M. Richetin, J. L. Lapresté, and G. Rives: "Determination of the Attitude of 3-D Objects from a Single Perspective View," *IEEE Trans. on PAMI*, Vol. 11, No. 12, pp. 1265-1278, 1989.

[16] G. Dudek and D. Jugessur: "Robust Place Recognition using Local Appearance based Methods," *Proc. of ICRA2000*, pp. 1030-1035, 2000.

[17] G. Dudek and M. Jenkin: "Computational Principles of Mobile Robotics," Cambridge University Press, 2000.

[18] H. F. Durrant-Whyte and J. J. Leonard: "Navigation by Correlating Geometric Sensor Data," *Proc. of IROS'89*, pp. 440–447, 1989.

[19] Eberst, C., Andersson, M., and Christensen, H. I.: "Vision-Based Door-Traversal for Autonomous Mobile Robots," *Proc. of IROS2000*, 2000.

[20] A. Elfes: "Sonar-Based Real-World Mapping and Navigation," *IEEE J. of Robotics and Automation*, Vol. RA-3, No. 3, pp. 249-265, Jun. 1987.

[21] O. D. Faugeras and M. Hebert.: "The Representaion, Recognition, and Locating of 3-D Objects," *Int. J. of Robotics Research*, Vol. 5, No. 3, pp. 27-52, 1986.

[22] M. A. Fischler and R. C. Bolles: "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Communications of ACM*, Vol. 24, No. 6, pp. 381-395, 1981.

[23] D. Fox, W. Burgard, S. Thrun, and A. B. Cremers: "Position Estimation for Mobile Robots in Dynamic Environments," *Proc. of AAAI'98*, 1998.

[24] D. Fox, W. Burgard, F. Dellaert, and S. Thrun: "Monte Carlo Localization: Efficient Position Estimation for Mobile Robots," *Proc. of AAAI'99*, 1999.

[25] W. E. L. Grimson: "Localizing Overlapping Parts by Searching the Interpretation Tree," *IEEE Trans. on PAMI*, Vol. PAMI-9, No. 4, pp. 469-482, 1987.

[26] W. E. L. Grimson: "Object Recognition by Computer: The Role of Geometric Constraints", MIT Press, 1990.

[27] R. M. Haralick, et. al.: "Pose Estimation from Corresponding Point Data," *IEEE Trans. on SMC*, vol. 19, No. 6, pp. 1426-1446, 1989.

[28] R. M. Haralick, C. Lee, K. Ottenberg, and M. Nölle: "Review and Analysis of Solutions of the Three Point Perspective Pose Estimation Problem," *International Journal of Computer Vision*, Vol. 13, No. 3, pp. 331-356, 1994.

[29] A. Hauck and N. O. Stöffler: "A Hierarchic World Model Supporting Video-Based Localization, Exploration and Object Identification," *Proc. of ACCV'95*, pp. III-176–III-180, Dec. 1995.

[30] P. Hébert, S. Betgé-Brezetz, and R. Chatila: "Decoupling Odometry and Exteroceptive Perception in Building A Global World Map of a Mobile Robot: The Use of Local Maps," *IEEE Proc. of ICRA'96*, pp. 757-763, 1996.

[31] J. J. Heuring and D. W. Murray: "Visual head tracking and slaving for visual telepresence," *Proc. of ICRA'96*, pp. 2908-2914, 1996.

[32] R. Horaud, B. Conio, and O. Leboulleux: "An Analytic Solution for the Perspective 4-Point Problem," *Computer Vision, Graphics, and Image Processing*, Vol. 47, No. 1, pp. 33-44, 1989.

[33] R. Horaud, F. Dornaika, B. Lamiroy, and S. Christy: "Object Pose: The Link between Weak Perspective, Paraperspective, and Full Perspective," *International Journal of Computer Vision*, Vol. 22, No. 2, pp. 173-189, 1997.

[34] I. Horswill: "The Polly System," in *Artificial Intelligence and Robile Robots*, pp. 125-139, AAAI Press/MIT Press, 1998.

[35] D. P. Huttenlocher and S. Ullman: "Recognizing Solid Objects by Alignment with an Image," *International Journal of Computer Vision*, Vol. 5, No. 2, pp. 195-212, 1990.

[36] S. Iida and S. Yuta: "Vehicle Command System and Trajectory Control for Autonomous Mobile Robots," *Proc. of IROS'91*, pp. 212-217, 1991.

[37] 飯田重喜: "車輪型自立移動ロボットの走行制御システムに関する研究", 筑波大学工学研究科博士論文, 1991.

[38] 池内: "物体認識と認識プログラムの自動生成", 人工知能学会誌, Vol. 4, No. 1, pp. 30-42, 1989.

[39] D. W. Jacobs: "Matching 3-D Models to 2-D Images," *International Journal of Computer Vision*, Vol. 21, No. 1/2, pp. 123-153, 1997.

[40] P. Jensfelt, D. J. Austin, O. Wijk, and M. Andersson: "Feature Based Condensation for Mobile Robot Localization," *Proc. of ICRA2000*, pp. 2531–2537, 2000.

[41] D. Jung and A. Zelinsky: "Integrating Spatial and Topological Navigation in a Behaviour-based Multi-Robot Application," *Proc. of IROS'99*, pp. 323-328, 1999.

[42] T. Kanbara, J. Miura, and Y. Shirai: "Selection of Efficient Landmarks for an Autonomous Vehicle," *Proc. of IROS'93*, pp. 1332-1338, 1993.

[43] O. Khatib: "Real-time Obstacle Avoidance for Manipulatros and Mobile Robots," *Int. J. Robotics Res.*, Vol. 5, No. 1, pp. 90-98, 1989.

[44] 喜多, D. L. Wilson, J. A. Noble, 喜多: "汎用 3D-2D レジストレーション手法とその応用例", 情報処理学会研究報告 CVIM120-10, pp. 73-80, 2000.

[45] S. Koenig and R. G. Simmons: "Xavier: A Robot Navigation Architecture Based on Partially Observable Markov Decision Process Models," in *Artificial Intelligence and Mobile Robotics*, AAAI Press/MIT Press, pp. 91–122, 1998.

[46] Kosaka, A. and Pan, J.: "Purdue Experiments in Model-Based Vision for Hallway Navigation," *Proc. of Workshop on Vision for Robots in IROS'95*, pp. 87-96, (1995).

[47] B. Kuipers and Y-T. Byun: "A Robot Exploration and Mapping Strategy based on a Semantic Hierarchy of Spatial Representation," *Robotics and Autonomous Systems*, vol. 8, pp. 47–63, 1991.

[48] Y. Lamdan, J. T. Schwartz, and H. J. Wolfson: "On Recognition of 3-D Objects from 2-D Images," *Proc. of ICRA'88*, pp. 1407-1413, 1988.

[49] J. J. Leonard, H. F. Durrant-Whyte, and I. J. Cox: "Dynamic Map Building for an Autonomous Mobile Robot," *Proc. of IROS'90*, pp. 89–96, 1990.

[50] J. J. Leonard and H. F. Durrant-Whyte: "Simultaneous Map Building and Localization for an Autonomous Mobile Robot," *Proc. of IROS'91*, pp. 1442–1447, 1991.

[51] S. Linnainmaa, D. Harwood, and L. S. Davis: "Pose Determination of a Three-Dimensional Object Using Triangle Pairs," *IEEE Trans. on PAMI*, Vol. 10, No. 5, pp. 634-647, 1988.

[52] Y. Liu, T. S. Huang, and L. D. Faugeras: "Determination of Camera Location from 2-D to 3-D Line and Point Correspondences," *IEEE Trans. on PAMI*, Vol. 12, No. 1, pp. 28-37, 1990.

[53] Y. Liu and S. Arimoto: "A Flexible Algorithm for Planning Local Shortest Path of Mobile Robots Based on Reachability Graph," *Proc. of IROS'90*, pp. 749-756, 1990.

[54] D. G. Lowe: "Three-Dimensional Object Recognition from Single Two-Dimensional Images," *Artificial Intelligence*, Vol. 31, pp. 355-395, 1987.

[55] D. G. Lowe: "Fitting Parametarized Three-Dimensional Models to Images," *IEEE Trans. on PAMI*, Vol. 13, No. 5, pp. 441-450, 1991.

[56] D. G. Lowe: "Object Recognition from Local Scale-Invariant Features," *Proc. of ICCV*, 1999.

[57] T. Lozano-Perez and M. A. Wesley: "An algorithm for planning collision-free paths among polyhedral obstacles," *Comm. ACM*, Vol. 22, pp. 560-570, 1979.

[58] S. Maeyama, A. Ohya, and S. Yuta: "Non-Stop Outdoor Navigation of a Mobile Robot -Retroactive Positioning Data Fusion with a Time Consuming Sensor System-," *Proc. of IROS'95*, pp. 130-135, 1995.

[59] M. J. Matarić: "Integration of Representation into Goal-Driven Behavior-Based Robots," *IEEE Trans. on Robotics and Automation*, Vol. 8, No. 3, pp. 304–312, June, 1992.

[60] J. Miura and Y. Shirai: "Vision and Motion Planning for a Mobile Robot under Uncertainty," *Int. J. Robotics Res.*, Vol. 16, No. 6, pp. 806-825, 1997.

[61] Takeshi Miyai, and Shin'ichi Yuta: "Design and Implementation of Distributed Controller and its Operating System for Autonomous Mobile Robot Platform," *Proc. of International Conference on Field and Service Robotics (FSR'97)*, pp. 342-347, 1997.

[62] P. Moutarlier and R. Chatila: "Incremental Free-Space Modelling From Uncertain Data by An Autonomous Mobile Robot," *Proc. of IROS'91*, pp. 1052–1058, 1991.

[63] H. Murase and K. Nayar: "Visual Learning and Recognition of 3-D Objects from Appearance," *International Journal of Computer Vision*, Vol. 14, No. 1, pp. 5-24, 1995.

[64] 中野, 谷内田, 辻: "3 次元モデルを用いた機械部品の認識", 信学技法 PRL79-85, pp. 95-102, 1980.

[65] 小倉知樹, 油田信一: "自立型移動ロボット「山彦」における自律屋内走行システムの標準機能化ー独立した地図管理・経路計画系とシステムライブラリによる実環境内走行制御系の実現ー", 第4回ロボティクスシンポジア, pp.359-366,1999.

[66] K. Ohba and K. Ikeuchi: "Detectability, Uniqueness, and Reliability of Eigen Windows for Stable Verification of Partially Occluded Objects," *IEEE on PAMI*, Vol. 19, No. 9, pp. 1043-1048, 1997.

[67] A. Ohya, A. Kosaka, and A. Kak: "Vision-Based Navigation of Mobile Robot with Obstacle Avoidance by Single Camera Vision and Ultrasonic Sensing," *Proc. of IROS'97*, pp. 704–711, 1997.

[68] 岡村耕二： "Linux でマルチメディア 第6回 Linux から利用する IBM Smart Capture Card", *Software Design*, pp. 68-75, Sep. 1998.

[69] T. Q. Phong, R. Horaud, A. Yassine, and P. D. Tao: "Object Pose from 2-D to 3-D Point and Line Correspondences," *International Journal of Computer Vision*, Vol. 15, pp. 225-243, 1995.

[70] S. I. Roumeliotis and G. A. Bekey: "Bayesian estimation and Kalman filtering A unified framework for Mobile Robot Localization," *Proc. of ICRA2000*, pp. 2985-2992, 2000.

[71] 査, 生田, 長谷川: "スケールに不変な特徴量を用いた距離画像の位置合わせ", 第18回日本ロボット学会学術講演会, pp. 413-414, 2000.

[72] S. Se, D. Lowe, and J. Little: "Vision-based Mobile Robot Localization And Mapping using Scale-Invariant Features," *Proc. of ICRA2001*, pp. 2051-2058, 2001.

[73] A. Selinger and R. Nelson: "A Perceptual Grouping Hierarchy for Appearance-based 3D Object Recognition," *Computer Vision and Image Understanding*, Vol. 76, No. 1, pp. 83-92, 1999.

[74] S. Simhon and Gregory Dudek: "A Global Topological Map formed by Local Metric Maps," *Proc. of IROS'98*, pp. 1708–1714, 1998.

[75] Y. Shirai: "Robot Vision Research: Past and Future Roles," *International Journal of Robotics Research*, Vol. 18, No. 12, pp. 1185-1200, 1999.

[76] P. Suetens, P. Fua, and A. J. Hanson: Computational Strategies for Object Recognition, *ACM Computing Surveys*, Vol. 24, No. 1, pp. 5-61, 1992.

[77] Y. Sumi and F. Tomita: "3D Object Recognition Using Segment-Based Stereo Vision," *Proc. of ACCV'98*, 1998.

[78] O. Takahasi and R. J. Schillinger: "Motion Planning in a Plane using Generalized Volonoi Diagrams," *IEEE Trans. Robotics Automation*, Vol. 5, No. 2, pp. 143-150, 1989.

[79] S. Thrun, et. al.: "Integrating Topological and Metric Maps for Mobile Robot Navigation: A Statistical Approach," *Proc. of AAAI'98*, 1998.

[80] S. Thrun, et. al.: "Map Learning and High-Speed Navigation in RHINO," in *Artificial Intelligence and Mobile Robotics*, AAAI Press/MIT Press, pp. 21–52, 1998.

[81] S. Thrun: "Learning metric-topological Maps for Indoor Mobile Robot Navigation," *Artificail Intelligence* 99 (1998) 21-71.

[82] S. Thrun: "Probabilistic Mapping Of An Environment By A Mobile Robot," *Proc. of ICRA'98*, pp. 1546–1551, 1998.

[83] N. Tomatis, I. Nourbakhsh, K. Arras, and R. Siegwart: "A Hybrid Approach for Robust and Precise Mobile Robot Navigation with Compact Environment Modeling," *Proc. of ICRA2001*, pp. 1111–1116, 2001.

[84] M. Tomono and S. Yuta: "Mobile Robot Navigation in Indoor Environments using Object and Character Recognition," *Proc. of ICRA2000*, pp. 313–320, 2000.

[85] M. Tomono and S. Yuta: "A Framework for Indoor Navigation based on a Partially Quantitative Map," *Proc. of IROS2000*, 2000.

[86] 友納, 油田： "ビジョンベース・ナビゲーションのためのオブジェクト指向フレームワーク", 日本ロボット学会学術講演会, 2000.

[87] 友納, 油田： "幾何情報とトポロジー情報を融合したマップに基づく屋内ナビゲーションシステム", 第 6 回ロボティクスシンポジア予稿集, pp. 340–347, 2001.

[88] T. Tsubouchi and S. Yuta: "Map Assisted Vision System of Mobile Robots for Reckoning in a Building Environment," *Proc. of ICRA'87*, pp. 1978–1984, 1987.

[89] I. Ulrich and I. Nourbakhsh: "Appearance-Based Place Recognition for Topological Localization," *Proc. of ICRA2000*, pp. 1023–1029, 2000.

[90] Y. Watanabe and S. Yuta: "Position Estimation of Mobile Robots with Internal and External Sensors Using Uncertainty Evolution Technique," *Proc. of ICRA'90*, pp. 2011–2016, 1990.

[91] M. D. Wheeler and K. Ikeuchi: "Sensor Modeling, Probabilistic Hypothesis Generation, and Robust Localization for Object Recognition," *IEEE Trans. PAMI*, Vol. 17, No. 3, 1995.

[92] Y. Yagi, K. Shyouya, and M. Yachida: "Environmental Map Generation and Egomotion Estimation in a Dynamic Environment for an Ominidirectional Image Sensor," *Proc. of ICRA2000*, pp. 3493–3498, 2000.

[93] B. Yamauchi, A. Schultz, and W. Adams: "Mobile Robot Exploration and Map-Building with Continuous Localization," *Proc. of ICRA'98*, pp. 3715–3720, 1998.

[94] S. Yuta, S. Suzuki, and S. Iida: "Implementation of A Small Size Self- Contained Autonomous Robot - Sensor, Vehicle Control, and Description of Sensor Based Behavior -," R. Chatila et al. Eds, *Experimental Robotics II*, pp. 344-359, Springer-Verlag, 1993.

[95] L. Zhang and B. K. Ghosh: "Line Segment Based Map Building and Localization Using 2D Laser Rangefinder," *Proc. of ICRA2000*, pp. 2538–2543, 2000.

[96] L. Zhang and B. K. Ghosh: "Geometric Feature based 2½D Map Building and Planning With Laser, Sonar and Tactile Sensors," *Proc. of IROS2000*, pp. 115–120, 2000.

研究業績

査読のある論文

[1] 友納, 油田： "不正確さを許すマップにおける移動ロボットの自己位置推定", 日本ロボット学会誌, (2002 年 4 月掲載予定).

国際会議

[1] M. Tomono and S. Yuta: "Mobile Robot Localization based on an Inaccurate Map", Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS2001), pp. 399–405, 2001.

[2] M. Tomono and S. Yuta: "A Framework for Indoor Navigation based on a Partially Quantitative Map," Proc. of the IEEE/RSJ International Conference on Intelligent Robots (IROS2000), pp. 626–632, 2000.

[3] M. Tomono and S. Yuta: "Mobile Robot Navigation in Indoor Environments using Object and Character Recognition," Proc. of the IEEE International Conference on Robotics and Automation (ICRA2000), pp. 313–320, 2000.

[4] M. Tomono: "An Event Notification Framework based on Java and CORBA," Proc. of the 6th IFIP/IEEE International Symposium on Integrated Network Management (IM'99), pp.563–576, 1999.

学会等における口頭発表

[1] 友納, 油田： 不正確さを許すマップと物体認識による屋内ナビゲーション, 第 7 回ロボティクスシンポジア, (2002 年 3 月発表予定).

[2] 友納, 油田： 不正確さを許すマップにおける移動ロボットの自己位置推定, 日本ロボット学会学術講演会, 2001.

[3] 友納, 油田： 姿勢空間分解による 3 次元物体認識, 日本ロボット学会学術講演会, 2001.

[4] 友納, 油田：幾何情報とトポロジー情報を融合したマップに基づく屋内ナビゲーションシステム, 第 6 回ロボティクスシンポジア予稿集, pp. 340–347, 2001.

[5] 友納, 油田：”パラメータ空間分割による段階的モデルマッチングを用いた 3 次元物体認識”, 情報処理学会研究報告 2001-CVIM-125, pp. 71–78, 2001.

[6] 友納, 油田：”ビジョンベース・ナビゲーションのためのオブジェクト指向フレームワーク”, 日本ロボット学会学術講演会, 2000.