タスクオリエンテッドアプローチによる自律移動マニピュレータの研究

—— ドアの通り抜けを含む自律ナビゲーションの実現 ——

１９９７年 ３月

永谷 圭司

論文概要

本研究の目的は，移動マニピュレータの自律化技術の研究である．この中でも特に，車輪を用いて移動する屋内用移動マニピュレータの自律動作に関する研究を行った．

この研究を進める上で，筆者らは具体的な研究目標を設定し，その実現を目指すタスクオリエンテッドアプローチ（Task Oriented Approach）という研究方針を採用し，目標タスクを「移動マニピュレータによるドアの通り抜けを含む自律ナビゲーション」と設定した．このタスクを実ロボットによって実環境で実現するため，まずタスクの分析を行い，動作設計及びハードウエア・ソフトウエアの実装を行うことで，目標タスクの実現を目指した．また，構築した実ロボットを用いて実験を行うことにより，移動マニピュレータの自律動作に関する問題点を発見し，これらの問題に対処するための手法や動作の設計に関する数多くの知見を得た．

本論文は，タスクオリエンテッドアプローチに沿って進められた本研究の説明，及びこの研究で得られた移動マニピュレータの自律動作に関する知見や問題点を述べたものである．本概要では，この論文を執筆するにあたり行った研究の概要について以下に述べる．

タスクの設定と分析

本論文では，まずタスクの設定及び分析について説明する．

目標タスクは「移動マニピュレータによるドアの通り抜けを含む自律ナビゲーション」である．具体的に目指すタスクは，片開きドアによって区切られた我々の研究室と廊下における，ロボットの現在位置から目的地までの自律ナビゲーションである．ただし，動作の前提として，ロボットは環境情報，現在位置，及び目的地を有するものとする．

このタスクを実現するロボットの動作は，人間が行うドア開け動作を参考にして以下に示す通り設計した．まずロボットは，走行経路及び動作を計画し，計画した走行経路にしたがって目的地に向かって走行する．この走行経路上にドアが存在する場合，ロボットに

搭載したマニピュレータ及びハンドを用いてドアノブを回し，ドアを開閉して通り抜ける動作を行うことで，目的地に到達する．

以上のように，目標タスクを実現するためのロボットの動作は非常に複雑なので，この動作を以下に示す部分動作に分割し，タスクの分析を行った．

1. ロボットの走行経路と動作の計画

   まずロボットは，現在位置と目的地から走行経路の計画を行う．ただし経路上にドアが存在する場合には，ロボットはこのドアを通り抜ける動作を行う必要があるので，ここでは走行経路だけでなく，ロボットの動作計画も同時に行う必要がある．また，ロボットが動作中，予定外の状況に対処するために動作計画の変更を行うこともできるように，この計画機能はロボットが有する必要がある．

2. ドアの前まで経路追従走行

   次にロボットは，走行機能を用いて計画した経路を走行し，ドアの前に停止する．本研究の場合，ベースロボットに車輪型の移動ロボットを想定しているため，軌跡追従走行に車輪の回転数を計数して自己位置を推定するオドメトリを使用するが，走行に伴ってこの位置には誤差が生ずるため，走行時外界センサを用いた推定位置の修正動作が必要となる．

3. ドアノブの把持動作

   ベースロボットはドアの前に停止した後，マニピュレータを制御し，搭載したハンドを用いてノブを把持する．このとき，オドメトリに生ずるロボットの推定位置の誤差が原因で，ハンドの位置にも誤差が生ずる．そこで，外界センサ（主に視覚センサ）を用いてノブの位置を認識し，把持動作を行う必要がある．

4. ドア開け動作

   次にロボットは，ドア開け動作を行うのだが，マニピュレータの長さに制限があるため，ここでは，走行系とマニピュレータが協調しつつ動作を行う必要がある．また，このドア開けの動作時に，ベースロボットの位置誤差によって，ハンドが把持したノブに大きな力がかかるという問題も生ずる．

5. ドアノブのリリース

   ドアを開けた後，ロボットはノブをリリースし，マニピュレータを初期姿勢に戻す．

6. ドアの裏側まで走行

   次にロボットは，ドアを閉めるため，ドアの裏側にあるノブの前まで走行し停止する．この動作は，走行経路は異なるが,「1. ドアの前まで経路追従走行」と同じ動作である．

7. ドアノブの把持動作 (2. ノブの把持動作と同じ動作)

8. ドアを閉める動作

   次に，ロボットは，マニピュレータとベースロボットの協調動作を行いつつ，ドアを閉める．この協調動作は，基本的には「4. ドア開け動作」と同じで，動作パラメータが異なる．

9. ドアノブのリリース (5. ノブのリリースと同じ動作)

10. 目的地まで経路追従走行

    ドアを通り抜けた後，ベースロボットが有する走行機能を用いて，計画した経路に沿って目的地へ向かう．この動作は,「1. ドアの前まで経路追従走行」と同じであり，動作パラメータのみ異なる．

以上の各部分動作を順次実行することで，目標タスクを実現することができる．

動作設計とロボットシステムの構築

目標タスクを分析した結果，このタスクを実現するのに必要な動作は,「走行経路と動作の計画」,「経路追従走行」,「ノブの把持動作」,「ノブのリリース」,「ドアの開閉動作」に大別できる．この各部分動作を以下のように設計した．

1. 走行経路と動作の計画

   まずロボットは，与えられた現在位置，目的地，環境情報より，ロボットの走行経路及び動作計画を行うことにした．またこの動作計画は，ロボットに搭載したコンピュータシステム上で行うことにした．

2. 経路追従走行

この部分動作では，計画した経路の追従走行を行う．ただし，オドメトリによって生ずるベースロボットの推定位置誤差に対処するため，環境中の平らな壁をランドマークとして，超音波センサを用いてこれを検知し，ロボットが有するランドマークモデルと比較することで，位置修正を行うことにした．

3. ノブの把持動作

この部分動作では，マニピュレータを制御してハンドを移動し，ノブを把持する．ただし，ベースロボットの推定位置の誤差によって生ずるハンドの位置誤差に対処するため，視覚センサを用いてハンドとノブとの相対位置誤差を検出し，ハンドの位置修正を行うことにした．

4. ノブのリリース

この部分動作では，ハンドがノブを離した後，マニピュレータを初期姿勢に戻す．この動作は，マニピュレータの制御によって実現することにした．

5. ドアの開閉動作

この部分動作では，ベースロボットの走行とマニピュレータ制御の協調動作を行うことで，ドアを開閉することにした．また，ベースロボットの位置誤差によってハンドが把持したノブに生ずる力に対処するため，マニピュレータのコンプライアンス制御を行うことにした．また，ドアノブを把持するために修正したマニピュレータの姿勢からも，ベースロボットの位置誤差が推定できるので，この位置推定をドア開け開始時に行うことにした．

以上の設計した動作を実現するため，本研究では，下表に示す機能を搭載するロボットシステムを構築することにした．

(a) 走行経路及び動作の計画機能
(b) ベースロボットの走行機能
(c) 超音波距離センサによるベースロボットの位置推定機能
(d) オドメトリによるベースロボットの位置推定機能
(e) ノブを把持したマニピュレータの姿勢によるベースロボットの位置推定機能
(f) ハンドユニットの指の開閉機能
(g) 視覚センサによるノブの位置認識機能
(h) マニピュレータの姿勢制御機能
(i) 力センサによるハンドの先端のコンプライアンス機能
(j) 超音波距離センサによる走行経路上の障害物の検知機能

これらの機能を実現するためのロボットシステムは，以下に示すハードウエアとソフトウエアで構成される．

ハードウエア

ハードウエアのメカニズムとして，ベースロボットには，筆者らの研究室において研究開発が行われた自律型移動ロボット「山彦てん」を使用することにした．このロボットと同タイプのロボットを使用した，筆者の所属する研究グループで行われている自律移動ロボットの研究を**山彦プロジェクト**と呼ぶ．また，マニピュレータ，及びハンドについては，本研究で設計，及び製作を行った．

ハードウエアのセンサシステムとしては，走行経路上の障害物，及び平らな壁を検知するための超音波距離センサ，コンプライアンス制御を行うための力センサ，ノブの認識を行うための視覚センサを搭載した．

これらのメカニズムを駆動するためのアクチュエータ及びセンサ機能を制御するためのコントローラとしては，山彦プロジェクトで提案された機能分散コントローラを使用した．このコントローラは，各アクチュエータ及びセンサ毎に機能が分散され，各機能を実現するための単位は機能モジュールは呼ばれる．この各機能モジュールは，CPU，RAM，ROM，センサまたはアクチュエータを有し，それぞれ機能毎に独立したシステムを構成する．これらの機能は，機能を統括制御するマスタモジュールによって管理され，ロボットの動作はこのマスタモジュールによって決定される．本研究では，目標タスクに必要な機能を実現するため，(1) 走行制御モジュール，(2) 超音波距離センサモジュール，(3) 位置推定モジュール，(4) マニピュレータ制御モジュール，(5) 視覚センサモジュール，(6) 音声合成モジュー

ル，(7)マスタモジュールをロボットに搭載した．このうち，マニピュレータ制御モジュール及び力センサモジュールについては，本研究で設計・製作を行い，その他のモジュールに関しては山彦プロジェクトで研究開発されたものを使用した．

ソフトウエア

ソフトウエアは，機能分散コントローラの概念にしたがい，マスタモジュール上でロボット全体の動作決定を行うソフトウエアと，各機能モジュール上で動作するソフトウエアの2階層で実現した．

機能モジュール上で動作するソフトウエアは，各機能モジュール上で常に動作するもので，本研究では，マニピュレータ制御モジュール上のソフトウエア，視覚センサモジュール上のソフトウエア，力センサモジュール上のソフトウエアの設計及び実装を行った．また，その他の機能モジュール上のソフトウエアについては，既に山彦プロジェクトで研究開発が行われた既存のものを使用することにした．

一方，マスタモジュール上のソフトウエアについては，目標タスクを実現する動作が複雑なため，まず動作を時間の経過に沿って動作のまとまり毎に分割し，各部分動作を実現するためのソフトウエアの単位をアクションプリミティブと名付けた．本研究で実現したアクションプリミティブは,「経路追従走行」,「ドアノブの把持動作」,「ノブのリリース」,「ドアの開閉動作」である．よって，ロボット全体の動作はアクションプリミティブのシーケンスで表現することができる．そこで，マスタモジュール上のソフトウエアは，これらのアクションプリミティブの集合と，このアクションプリミティブのシーケンスを作成し，これを順次実行する，統括制御ソフトウエアの2階層で構成することにした．

各アクションプリミティブ，及び統括制御ソフトウエアは，それぞれマスタモジュール上のプロセスとして実装した．

ナビゲーション実験

構築したロボットシステムを用いて，実環境におけるドアの通り抜けを含む自律ナビゲーションを実現し，幾つかの経路において数多くの実験を行った．この結果，このロボットシステムを用いてドアの通り抜けを含む自律ナビゲーションの成功率は8割程度であった．また動作中問題が生じたときに，ノブの把持を手伝うといった人間のサポートがあれば，動

作はほとんどの場合成功する．この数字を考えると，目標タスクはほぼ達成されたといって良い．ただし，2割程度の失敗の最大の原因は，ベースロボットの推定位置誤差によるものと考えられる．

検討と評価

タスクオリエンテッドアプローチによって行われた本研究を通じて，移動マニピュレータの自律動作に関する幾つかの知見と問題点が明らかになった．移動マニピュレータの問題点としては，ベースロボットの推定位置誤差の問題，物体を操作する際に生ずる力の問題，ハンドの把持を行う際に起こるスリップの問題，などが挙げられる．またハードウエアの構成に関する知見としては，小型軽量マニピュレータの重要性，小型ハンドユニットの必要性，機能分散コントローラの有用性が明らかとなった．また，ロボットの動作設計に関しては，ソフトウエアを時間の経過に沿った部分動作に分割して設計開発を行うことの有効性，及び動作計画と実行を分けることの利点，といった知見が得られた．研究方針については，タスクオリエンテッドアプローチで行う研究の有用性が確認された．

本概要の最後に，本論文の構成について述べる．

第1章「序論」では，研究の背景，従来の研究とその問題点について述べ，タスクオリエンテッドアプローチで行う本研究の立場を明らかにする．

第2章「タスクの設定」では，研究方針であるタスクオリエンテッドアプローチ，及び実験ロボット工学について説明し，本研究のタスクを明確に定義する．

第3章「タスクの分析」では，目標タスクの分析を行う．

第4章から第9章までが，目標タスクを実現するためのロボットシステムの，ハードウエア，及びソフトウエアに関する構築方法を述べたものである．

まず第4章「動作設計とロボットシステムの構築」では，第3章の分析に基づいて目標タスクを実現するためのロボットの動作設計を行い，この動作を実現するために構築したロボットシステムの全体像について説明する．

第5章「ハードウエア」では，構築したロボットシステムに搭載したハードウエアメカニズム，センサ及びコントローラについて詳しく説明する．

第6章「ソフトウエアの構成」では，目標タスクを実現するための，ロボットの動作を

行うソフトウエアの構成について紹介する.

第7章「機能モジュール上のソフトウエアの実現」では，各機能を制御するための，ソフトウエアの設計及び実装の詳細について報告する.

第8章「アクションプリミティブの実現」では，ロボットの部分動作を制御するための，ソフトウエア群の設計及び実装の詳細について報告する.

第9章「統括制御ソフトウエアの実現」では，ロボットの部分動作を実現するソフトウエア群を統括し，動作全体を制御する統括制御ソフトウエアの設計及び実装について述べる.

第10章「ナビゲーション実験」では，構築したこのロボットシステムを用いて行ったナビゲーション実験の結果と，この結果に対する検討について報告する.

第11章「本研究に対する検討と評価」では，タスクオリエンテッドアプローチ，及び実験ロボット工学に沿って進められた本研究を通じて得られた移動マニピュレータの自律化に関する知見や問題点について述べる.

第12章「結論」は本研究の全体のまとめである.

目 次

図 目 次

表 目 次

第 1章

序論

1.1 研究の背景

近年，自動機械の自律化・知能化に対する要求が高まっている．現在自動機械として実用化されている例としては，自動車工場で稼動する産業用ロボットが挙げられる．この自動機械は，人間の代わりに自動車の組み立て作業や塗装作業などを行い，生産効率や製品の質を向上させると共に，単調な労働から人間を開放してきた．しかし，一般に産業用ロボットの土台は地面に固定されているため，その作業範囲は狭い．またティーチングプレイバックによる単調な作業の繰り返しが基本となるため，予定外の状況に対処する能力が低く，作業内容の変更も比較的困難である．一方，広域で作業を行う自動機械として実用化されている例としては，工場内での物品搬送に使用される自動搬送車が挙げられる．この自動機械は，工場内に張り巡らされたガイドラインに沿って走行するため，その作業範囲は産業用ロボットと比較して遥かに広いが，決められた経路の走行以外の能力は低い．また，これらの実用化された自動機械は，作業環境として工場という限定された環境しか想定されていないため，予定外の状況に対処する能力や自律的な判断を行う能力は低い．

この一方で，様々な環境において，より複雑な作業を自律的に行う自動機械の必要性が，近年高まってきた．これらの自動機械を必要とする環境は，大きく (A) 極限環境 (B) 屋外環境 (C) 屋内環境 (D) 工場内の４つに分類される．

(A) 極限環境とは，宇宙空間や深海，原子炉内など人間が作業を行うには極めて危険な極限環境を指す．この環境では，遠隔で人間の指図した通りの動作を行うこと（テレオペレーション）が自動機械には求められるが，通信に時間のかかる宇宙空間や深海などでは，動作命令と実動作とのタイムラグが大きな問題となる．このような場合，自動機械自身に，状況に対応した高い自律性が求められる．

(B) 屋外環境とは，屋外の不整地や建設現場，山林などを指す．この環境における自動機械には，主に不整地における重たい物資の運搬作業などが求められるが，将来的には作業効率を上げるための自律移動機能の必要性も高まると予想される．

(C) 屋内環境とは，人間が働くオフィスビル，病院，家の中などの屋内環境を指す．このような環境において自動機械には，来客案内，物品の運搬，掃除作業など人間の仕事のサポートを行うことが求められる．この場合，自動機械は，人間と空間を共有するため，センサを用いて周囲の状況を認識し，人間の安全を確保しつつ動作するという，高い自律性を有する必要がある．

(D) 工場内の自動機械には，現在と同様，物資の運搬や製品の製作・加工を行うことが求められるが，この自動機械が高度な自律性を有することで，より多様な仕事を任せることができるようになると予想される．例えば，無人搬送車が搬送物品を自律的に判し，自らこれを搬送することで，作業効率は今以上に向上することが期待される．

このように，現在様々な環境において，自動機械の自律化・知能化に対する要求が高まっており，企業や大学，官庁などの様々な研究機関において，機械の自律化・知能化に関する研究が行われてきた．

特に，屋内環境で動作する自動機械については，(1) 利用範囲が広く社会的に要求が高い，(2) 人間と環境を共有するため高い自律性が要求される，(3) 屋内を対象とするため実験が容易である，といった特徴があるため，現在様々な研究が行われている．我々の研究グループでも，屋内環境において自律走行を行う自律移動ロボットの研究「山彦プロジェクト」が十数年継続して行われてきた[YY87][YSI91]．この研究の特徴は，山彦プロジェクトで研究開発が行われた小型の自律型移動ロボット「山彦」を，研究用のプラットフォームとして使用し，実験主体で研究を進める点である．この山彦ロボットの集合写真を図1.1に示す．このプロジェクトでは，これらの移動ロボットを使用し，ナビゲーション，ポジショニング，走行制御，各種センサ，コントロールアーキテクチャなど，移動ロボットの自律動作に関する研究が行われてきた．

このような背景の下で近年，走行機能を持つ移動ロボットにマニピュレータを搭載したタイプの移動マニピュレータが注目を集めており，現在多くの研究者によって移動マニピュレータの研究が行われている．この移動マニピュレータは，マニピュレータを用いた複雑な作業が行えるという利点と，自ら移動することで作業範囲が広いという利点を兼ね備えている．その一方で，マニピュレータを支える土台自身が移動することで生ずるマニピュレータの先端の位置誤差の問題や，マニピュレータ制御と走行の協調動作の問題など，解

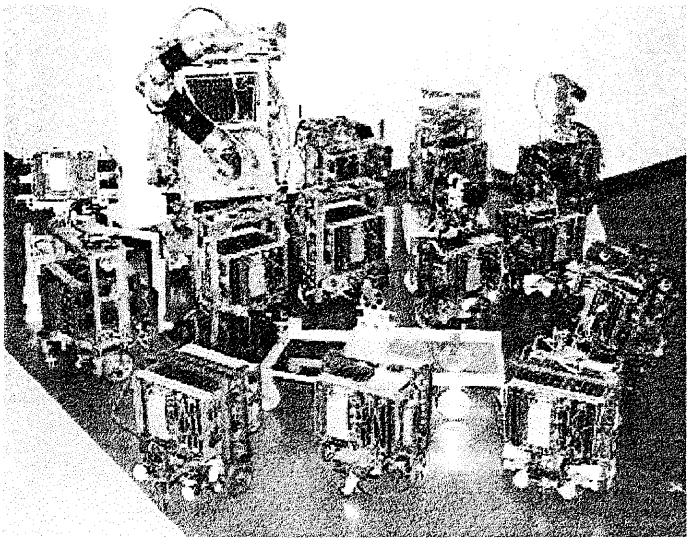

図 1.1: 自律型移動ロボットファミリー（写真）

決すべき点も多い.

1.2 本研究の目的

筆者らは，本研究の目的を，屋内環境で作業を行う自動機械（以後ロボットと呼ぶ）の自律化技術の研究と設定し，**タスクオリエンテッドアプローチ**(TaskOriented Approach)という方針で研究を行った．この研究方針は，ロボットが実現すべきタスク(動作目標)を設定し，そのタスクを実現する過程でロボットが有する問題について考察し，これを解決するためのアルゴリズムや手法を探求するものである．このタスクオリエンテッドアプローチに関する詳細は2.2節に示す．また，タスクを実現する過程で採用した研究方針が，**実験ロボット工学**(Experimental Robotics)である．この方針は，まず実際のロボットシステムを構築し，このシステムを用いて実環境で動作を行うことで，ロボットが有する問題点を発見・克服するという方針である．この研究方針の詳細は2. 3節に示す．本研究は，このタスクオリエンテッドアプローチ，及び実験ロボット工学という2つの研究方針で進められた.

このタスクオリエンテッドアプローチに沿って，筆者らは，目標タスクを「移動マニピュレータによるドアの通り抜けを含む自律ナビゲーション」と設定した．ただし，このタスクの条件として，環境情報及び移動マニピュレータの現在位置は既知とする．このタスク

を実現するための移動マニピュレータの動作は以下のようになる．

まず移動マニピュレータは，現在位置から目的地までの経路を計画し，この経路に沿って走行する．ただし，走行経路上にドアが存在する場合，このロボットは，搭載したマニピュレータとハンドを用いて，自律的にこのドア通り抜ける動作を行う．このことから，この目標タスクには，ロボットの動作計画，走行とマニピュレータ制御の協調動作，物体を操作する際の力制御，更にはロボットのアーキテクチャといった，ロボットの自律化に関する研究の主要なサブテーマが多数含まれることがわかる．このタスクを実現する過程で，移動マニピュレータが有する問題点を発見し，移動マニピュレータの自律化に必要な機能やアルゴリズムの研究を行うことが本研究の第一の目的である．

筆者らはこのタスクの下，実験ロボット工学に沿って移動マニピュレータのシステムを構築し，実環境においてこのタスクの実現を図った．ただし，ここでの実環境とは，大学やオフィスビル内の部屋や廊下など，**ロボットのために整備を行っていない屋内環境**を指す．

一方，設定したタスクを実現することで，これまでドアが開いていない限り移動ロボットが通行不能だったところが通行可能となり，屋内環境における移動ロボットのナビゲーション領域は大きく拡大する．したがって，設定したタスクの実現自体が，本研究の第二の目的である．

1.3 従来の研究と本研究の立場

本研究に関連する従来の研究分野としては，移動ロボットのナビゲーションに関する研究，移動ロボットの経路計画に関する研究，移動マニピュレータに関する研究，ロボットの動作計画に関する研究，ロボットシステムの構成法に関する研究，などが挙げられる．ここでは，これらの研究分野における従来の研究の概要，並びにこれらに対する本研究の立場を，各研究分野毎に明らかにする．

1.3.1 移動ロボットのナビゲーションに関する研究

従来の研究と問題点

安定した移動ロボットのナビゲーションには，正確なポジショニング能力（自己位置推定能力）が要求される．このポジショニングを行う方法として，大きく内界センサを用いる方法と外界センサを用いる方法の2つがある．内界センサを用いる方法としては，車輪の回転数を測定して自己位置を推定するオドメトリ法や，ジャイロによる位置推定法など

が一般的である．これらの方法は，ロボットの筐体内に搭載したセンサのみを用いて，常に位置推定を行うことができるという利点を持つ反面，車輪のスリップやジャイロのドリフトによって誤差が生ずるという位置誤差の問題も存在する．一方外界センサを用いた位置推定法には，GPSを用いた位置推定[SHW94]，環境内に設置されたリフレクタの位置をレーザを投光することで検知し位置推定を行う方法[NOY95]，視覚センサを用いて環境内のランドマークを獲得し位置推定を行う方法[LC96]，超音波距離センサを用いて環境中の壁を認識し，位置修正を行う方法[YN95]などが挙げられる．これらの外界センサを用いたポジショニングを行う場合，ランドマークをセンサで検出してポジショニングを行うため，位置推定方法は環境に大きく依存する．しかし，ランドマークを捕らえることさえできれば，環境中のロボットの位置は精度良く求まる．したがって，外界センサを用いたポジショニングと内界センサをを用いたポジショニングを組み合わせて使用することで，ロボットの正確なポジショニングを行うことが現在の主流である．

一方，移動ロボットのナビゲーションは，ポジショニングとも密接な関係がある．このナビゲーションに関する研究は，屋内環境では，カメラで得た画像上の縦線を抽出し，その縦線をランドマークとしてロボットを走行させる研究[OOY96]，予め走行経路の画像情報を抽象化して記憶し，その情報と実センサ情報をテンプレートマッチングすることでナビゲーションを行う研究[MII96]，環境中に配置したリフレクタをレーザを照射することで検知しつつ走行を行う方法[NOY95]，環境中の平らな壁をランドマークとしてナビゲーションを行う方法[YN95]などが挙げられる．屋外環境では，視覚センサを用いて道路のエッジを検出して道路を追従し，この追従行動を組み合わせて屋外ナビゲーションを行う研究[NM95]，視覚センサを用いて道路を判断し走行するNAVLABの研究[C.E89]，屋外に存在する街路樹をランドマークとしてナビゲーションを行う研究[MOY95]などが挙げられる．

このように，ナビゲーションに関する研究はロボットの動作環境に大きく依存し，対象とする環境に則したナビゲーション方法が提案されてきた．しかし，屋内環境だけを考えても，ドアの閉まっている部屋や階段など，ナビゲーションが不可能な領域は未だに広い．特にドアの通り抜けのように，ロボットがアクティブに環境に働きかけて動作範囲を広げるナビゲーションの研究は，ほとんどなされていない．

本研究の立場

従来研究からも分かる通り，ナビゲーションの研究は，タスクや動作環境に大きく依存するため，各々の環境に適したナビゲーションアルゴリズムが必要である．本研究では，通

常走行を行う部分では，環境中の平らな壁をランドマークとした移動ロボットのナビゲーション手法[YN95]を取り入れ，これにドアの通り抜け機能を追加して，屋内環境における広域ナビゲーションの実現を目指すことにした．

1.3.2 移動ロボットの経路計画に関する研究

従来の研究と問題点

移動ロボットの経路計画に関する研究は，最短経路探索問題などの応用例として人工知能の分野などで数多く行われてきた．近年では，実ロボットを念頭に置いた経路計画の研究が行われている．Khatibは，ポテンシャル法を用いたマニピュレータや移動ロボットの経路計画法を提案した[Kha86]．Zelinskyらは，ディスタンストランスフォームを用いた経路計画法を提案し，ポテンシャル法におけるローカルミニマ問題を解決した[ZY93]．池上らは，ポテンシャル法とセンサ利用を融合した経路計画法を提案した[TS92]．小森谷らは，屋内環境において，ランドマークの性質を考慮した走行経路およびランドマーク観測計画法を提案した[KOT92]．Chosetらは，ボロノイダイアグラムを用いたセンシングと経路計画法を提案した[CB96]．また筆者らは，ロボットの衝突の危険性を評価関数とした移動ロボットの経路計画法を提案し，山彦ロボット（図1.1）を用いて実環境で計画した経路を走行する実ロボットシステムを実現した[NY93]．

このように，ロボットの経路計画に関する研究は数多く行われているが，従来の経路計画の研究の多くは，ロボットを念頭に置いているにもかかわらず，その評価をシミュレーションで終わらせているものが多い．このため，提案された経路計画が実環境でどれだけ有効であるかについては，深く議論できていないのが現実である．

本研究の立場

現実の屋内環境について考えると，一般にロボットが走行できる領域はそれほど広くない．本研究で対象とする環境も，移動範囲はある程度限定されるため，通過できる経路は限られる．このような環境では，経路はグラフネットワークで大抵表現できるため，実際問題として屋内環境の走行を行うための複雑な経路計画は必要ない．そこで本研究では，グラフで表現した屋内における走行可能経路網を予め構築し，そこから経路選択を行うことで経路計画を行うことにした．

1.3.3 移動マニピュレータに関する研究

従来の研究と問題点

研究の背景でも述べた通り，近年移動マニピュレータを用いた研究が，盛んに行われるようになった．

この研究には，特にマニピュレータを動かした際の台車の安定を確保するための，移動マニピュレータのダイナミクスや安定性に関する制御の研究が多い．見浪らは，マニピュレータの運動が台車に及ぼす影響を台車の駆動力で補償する研究を行った[見浪93]．山本らは，マニピュレータと台車との干渉を考慮しつつ，手先と台車の参照点を目標起動に追従させることで，ロボットの安定化を図る研究を行った[YX92]．Dubowsky らは，ダイナミクスを考慮した移動マニピュレータの研究を行った[DT87]．菅野らは，移動マニピュレータのゼロモーメントポイント(ZMP)を考慮し，安定性を確保する研究[HSK94]を行った．

このような，移動マニピュレータのダイナミクスや安定性に関する制御の研究は，非常に重要であると考えられるが，対象とする移動マニピュレータが一般的または抽象的であるため，提案された理論やアルゴリズムが現実にどのようなタスクに応用でき，その場合にどの程度有効であるかは，あまり深く議論されていない．

一方，移動マニピュレータの応用に関する研究として，ドア開け動作に関する研究が幾つか行われてきた．日立製作所の岩本らは，クローラを使用した移動マニピュレータによるドア開け動作を実現した(関連研究は[岩本88])．ただしこの実験には，ロボットのサイズに合わせて設計・製作されたドアが使用された．また，ミュンヘン工科大学の Schmidt らは，大学の屋内環境におけるドア開け動作を実現した（関連研究は[AS94]）．この実験では，レーザ灯台を用いてロボットのポジショニングを行い，ハンドル式のノブを回すことでドア開け動作が行われた．ただし，ノブを回しドアを押し開ける部分は，センサフィードバックを行わずにオープンループで行っているため，ロバスト性は低い．また，東北大学の中野らは，全方向移動車に搭載したマニピュレータを用いた，ドア開け動作の設計及び実装を行った[佐々95]．この研究の特徴は，手先のコンプライアンスをハードウエアで実現した点，及びドアを開ける際，ドアが閉まらないようにベースロボット本体でドアを支えるという点である．

ドア開け以外の，移動マニピュレータの応用に関する研究として，Khatib らは2台の移動マニピュレータによる物体の協調搬送を実現した[OKK+96]．この研究の特徴は，台車の移動に伴って把持物体に生ずる位置誤差を，マニピュレータのコンプライアンスと台車の移動によって吸収する点である．また，対象物の表面にマニピュレータの先端を押し付

けて移動マニピュレータを走行させる研究[YX93]や，ロボットより大きな物体を動かすための移動マニピュレータによる押し動作に関する研究[栗栖95]も行われてきた．

以上に挙げた通り，移動マニピュレータに関する研究は現在盛んであるが，基礎的な動作に関する研究が多く，応用を目指した研究は比較的少ない．また，移動マニピュレータによるドアの押し開けといった応用研究については，環境をロボットに合せることで実現されたものはあるが，実環境を対象としたものは，ほとんど見受けられない．

本研究の立場

以上のように，移動マニピュレータのダイナミクスに関する研究は数多く行われており，制御法の例題としては興味深いものがある．しかし現実問題として，マニピュレータに対して台車が十分に重ければ，マニピュレータがどのような姿勢をとっても移動マニピュレータは転倒せず，制御性や安定性に関する問題はあまり生じない．よって本研究では，台車に比較的重い移動ロボットを採用し，マニピュレータの制御に関しては，標準的な制御法のみを使用することにした．

一方，移動マニピュレータの応用に関する研究については，近年ドア開けをテーマにした研究が行われるようになってきた．この理由として，ドア開けのテーマが「半固定された対象物のハンドリング」や「走行とマニピュレータ制御の協調」といった研究上興味深い問題を含むことが挙げられる．しかし，従来の研究のほとんどが，ロボットのために準備した環境において実現が図られてきた．これに対し本研究では，実環境に則した動作を目指し,「ロボットには必要な機能を搭載するが，実環境は改造しない」という方針で研究を行うことにした．また，ドアの通り抜け動作自体を研究目的とするのではなく，あくまでロボットの動作領域を広げるためのナビゲーション中の一機能と位置づけた．したがって，このナビゲーション動作を実現する過程で，実環境で働く移動マニピュレータを実用化するために必要な問題点やアルゴリズムが明確になる．また，ドアの通り抜け動作自体が，自律ロボットの標準問題のひとつであるため，この動作の実現自体にも大きな意味がある．

1.3.4 ロボットの動作設計に関する研究

従来の研究と問題点

ロボットの動作設計についても，様々な方法が提案されてきた．井上らは，マニピュレータの制御を行う作業を幾つかのレベルに分解し動作を実現するマニピュレータ制御システ

ムを構築した[小笠84]．この研究では，マニピュレータによる複雑な動作を，一番下の動作レベルの“Move to”命令，及びハンドの開閉命令を組み合わせることで実現した．Khatibらは，まず複雑な動作を分割し，各部分動作を実現し，組み合わせることで，ロボットの複雑な動作を実現する方法を提案した[Kha95]．鈴木らは，ロボットの状態がセンサ情報によって推移する状態遷移型ロボット言語“ROBOL/0”を提案し，この言語による移動ロボットの障害物回避動作を実現した[SY89].

このように，ロボットの動作計画に関する従来の研究に共通することは，ロボットの複雑な動作を計画・実行するために，その動作を分割したり階層化して考えることである．この分割や階層化は，複雑な問題を考える場合には自然な発想であるが，目標タスクやロボットのシステムに大きく依存し，どの方針が適当であるかは一概には言えない．

またBrooksは，移動ロボットの動作を反射の組み合わせで表現し，動作計画を行わないロボットの制御法の提案をした[Bro86]．この流れで，近年ロボットの動作を行うために，詳細な計画を必要としないリアクティブな制御が注目されている（例えば[MCR+93]）．この制御は，センサからの情報を直接アクチュエータにフィードバックするもので，人間の動作に例えれば反射動作をロボット上に実現するものである．したがって，この制御には，動作を行う上で環境やロボット自身の厳密なモデルを構築する必要が無く，またロバストであるという利点がある．しかし，この制御は，センサの性能に大きく依存し，実環境を対象とした場合，現存する一般的なセンサでは十分なパフォーマンスが得られない．

本研究の立場

井上らが提案したシステム[小笠84]のように，単一機能のシーケンスで動作が実現するものについては，機能に着目した動作分割が有効である．しかし，本研究のように，マニピュレータ制御系と走行制御系の協調動作については，動作を機能毎に分割することができない．そこで本研究では，ロボットの動作を時間に沿った動作のまとまり毎に分割し，目標動作を各部分動作の組み合わせで表現することにした．

また，本研究のように，実環境において複雑な動作の実現を目指す場合，センサの能力は限られるためリアクティブな動作は実現が困難である．よって本研究では，環境及びロボットのモデルを基に動作計画を行い，これに沿ったロボットの動作を行うといったモデルベースで動作設計を行うことにした．

1.3.5 ロボットのアーキテクチャに関する研究

従来の研究と問題点

ロボットのアーキテクチャの研究で最も有名なものが，Brooksが提唱するサブサンプションアーキテクチャである．このアーキテクチャは反射などの動作を積み上げていき，高度な動作を実現するもので，Brooksは，このアーキテクチャを使用した昆虫型ロボットによる障害物回避動作を実現した[Bro86]．しかし，このアーキテクチャには，ロボット全体を統括制御するパートが無く，複雑な動作をシーケンシャルに行うといった動作には向いていない．またThorpeらは，UNIXのネットワークを利用してNAVLABという自律走行車内にLANを形成し，ワークステーションによる自律走行実験を行った[C.E89]．またChatilaらは，移動ロボットのコントローラを制御レイヤから意思決定レイヤまでの幾つかのレイヤを用いて構成した[R.C91]．また，山彦プロジェクトでは，ロボットの各機能毎に独立のコントローラが制御を行う機能分散コントローラの提案を行った[SIY93]．現在，図1.1に示す山彦ロボットは，全てこのコントローラで稼動している．

本研究の立場

本研究では，基本的には，山彦プロジェクトで提案された機能分散コントローラを使用することにした．コントローラを機能毎に分散することは，新しい機能の搭載，デバッグ，メインテナンスを考えると非常に有益である．また，この機能分散コントローラを使用することで，山彦プロジェクトの研究資産を利用できるという利点がある．そこで，この機能分散コントローラをベースに，移動マニピュレータのコントローラを構成することにした．

1.3.6 従来の研究のまとめ

本節で述べた通り，移動マニピュレータの自律化に関連する研究テーマは多岐にまたがっており，個々の研究テーマは他の研究分野に密接に関連がある．よって，各テーマ毎に一般的な研究を目指すのではなく，むしろ本研究のように現実的なタスクを設定し，実験主体で広く研究を行う立場も必要であると，筆者らは考える．

1.4 本論文の構成

序論の最後に，本論文の構成について述べる．

第2章「タスクの設定」では，研究方針であるタスクオリエンテッドアプローチ，及び実験ロボット工学について説明し，本研究のタスクを明確に定義する．

第3章「タスクの分析」では，目標タスクの分析を行う．

第4章から第9章までが，目標タスクを実現するためのロボットシステムの，ハードウエア，及びソフトウエアに関する構築方法を述べたものである．

まず第4章「動作設計とロボットシステムの構築」では，第3章の分析に基づいて目標タスクを実現するための，ロボットの動作設計を行い，この動作を実現するために構築したロボットシステムの全体像について説明する．

第5章「ハードウエア」では，構築したロボットシステムに搭載したハードウエアメカニズム，センサ及びコントローラについて詳しく説明する．

第6章「ソフトウエアの構成」では，目標タスクを実現するためのロボットの動作を行うソフトウエアの構成について紹介する．

第7章「機能モジュール上のソフトウエアの実現」では，各機能を制御するための，ソフトウエアの設計及び実装の詳細について報告する．

第8章「アクションプリミティブの実現」では，ロボットの部分動作を制御するための，ソフトウエア群の設計及び実装の詳細について報告する．

第9章「統括制御ソフトウエアの実現」では，ロボットの部分動作を実現するソフトウエア群を統括し，動作全体を制御する統括制御ソフトウエアの設計及び実装について述べる．

第10章「ナビゲーション実験」では，構築したこのロボットシステムを用いて行ったナビゲーション実験の結果と，この結果に対する検討について報告する．

第11章「本研究に対する検討と評価」では，タスクオリエンテッドアプローチ，及び実験ロボット工学に沿って進められた本研究を通じて得られた移動マニピュレータの自律化に関する知見や問題点について述べる．

第12章「結論」は本研究の全体のまとめである．

第 2章

タスクの設定

2.1 はじめに

本研究は，タスクオリエンテッドアプローチ，及び実験ロボット工学という研究方針に沿って進められてきた．本章では，まずこれらの研究方針について説明する．ここでは，このタスクオリエンテッドアプローチの方針に沿って，目標タスクを「移動マニピュレータによるドアの通り抜けを含む自律ナビゲーション」と設定した．そこで本章では，このタスクが対象とする環境を明確に定義し，本タスクのより具体的な設定を行う．

2.2 タスクオリエンテッドアプローチ

まず，**タスクオリエンテッドアプローチ**による研究の進め方と特徴について説明する．

現在，ロボットの研究分野において広く行われている研究方針は，研究の核となる一般的な理論やアルゴリズムを提案し，それをシミュレーションまたは実ロボットに搭載して実験を行うことにより，提案した理論やアルゴリズムの有効性を確かめるというものである．この研究方針を，ここでは「シーズオリエンテッドアプローチ (Seeds Oriented Approach)」と呼ぶことにする．この一般的な理論やアルゴリズムは，実際には具体的なタスクを想定し，そのタスクを意識したものが導出されるのだが，研究方針の核はあくまで理論やアルゴリズムである．このシーズオリエンテッドアプローチにしたがって，現在までにロボットのための様々な理論やアルゴリズムが考案されてきた．これらの研究成果を組み合わせれば，知能的な動作を行う自律汎用ロボット（Versatile Robot）がすぐにでも完成するという期待が持てる．しかし現実には，各研究に適した限られた条件下においてロボットの基本的な動作しか実現されておらず，実用段階におけるロボットの自律化・知能化は程遠

い．この理由としては，(1) ロボットの動作が作業内容や環境に大きく依存すること，(2) 各研究が分野毎に独立に進められてきたこと，(3) 研究者によって自律化技術に対する問題意識が異なること，などが挙げられる．

このような状況から，より高度な自律動作を目指すため，本研究では具体的なタスク（作業目標）を設定し，この実現を目指すという方針で研究を進めることにし，この研究方針を「タスクオリエンテッドアプローチ(Task Oriented Approach)」と名付けた．このタスクに沿って行う研究の過程で，ロボットが有する問題に対する理解を深めることができ，現実問題に則した理論や手法，アルゴリズムを導出することができる．このような具体的な研究を積み重ねることで，ロボットの自律化技術の研究が進むと筆者らは考える．このタスクオリエンテッドアプローチが目指すところのイメージを図2.1に示す．

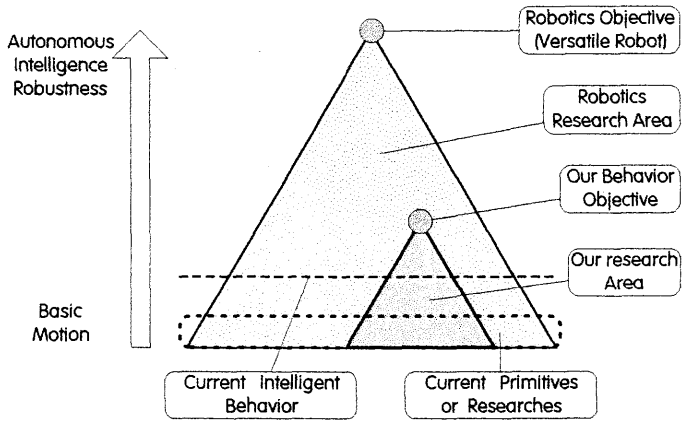

図 2.1: 我々の研究目標の位置付け

図2.1の解説：この図のピラミッドの頂点はロボット工学の究極の目標の一つである汎用ロボット(Versatile Robot)である．また，このピラミッドは，目標である汎用ロボットを実現するための研究分野全体(Robotics Research Area)であり，ピラミッドの底辺が目標を実現するための要素技術である．現在，要素技術の研究は幅広く行われているが，これらを積み上げて実現されるロボットの自律動作はなかなか向上しない（図中では Current Intelligent Behavior のレベル）．これに対し，タスクオリエンテッドアプローチでは，限られた制約条件の下で，より高度な自律動作をタスクとして設定し（Our Behavior Objective），このタスクの実現に必要な研究を行う(Our Research Area).

さて，タスクオリエンテッドアプローチで重要なのは，タスクの設定である．例えば，実現が容易である単純なタスクならば，タスクオリエンテッドで研究を行う必要はない．一方，実現に程遠いタスクは，タスクを実現するまで研究を持続させることが困難である．ま

た，タスクの難易度が高くても，実現価値の低いタスクならば，有用な研究とは言えない．これらを踏まえて，本研究では，**移動マニピュレータによるドアの通り抜けを含む自律ナビゲーションの実現** というタスクを設定した．ただし対象とする環境を，人間と共存する環境（オフィスビルや大学構内）とし，ロボットのためにこの環境の改造を行わないという方針を設けた．

このタスクは，移動ロボットのナビゲーションの一機能にすぎないが，走行制御，マニピュレータの制御，センサデータの処理や融合，動作計画など，多くの研究要素を組み合わせることで，初めてこのナビゲーション動作が実現可能となる．したがって，このタスクの実現は，ロボットの自律化に関する一つの研究例題として極めて有用であると考えられる．

一方，現実問題として，移動ロボットの作業領域がドアによって仕切られている場合，ロボットは何らかの方法でドアを開けない限り，これらの空間を往来することはできない．この場合，マニピュレータを搭載した移動ロボットが，自分の力でドアを開けて通り抜けることができれば，ロボットのための整備を行っていない屋内環境でも，その動作範囲は飛躍的に拡大する．この点からも，本タスクは実現そのものに大きな意味を持ち，タスクオリエンテッドアプローチのタスクとして十分適当であると考えられる．

最後にもう一つ，タスクオリエンテッドアプローチの例として，かつてアメリカのNASAが行ったアポロ計画を紹介する．この計画では，「人間を月に送り込む」というタスクが設定され，このタスクを実現するため様々な研究や技術開発が行われた．それぞれの研究は，目標タスクの実現を目指して進められ，人間が月に到達し，目標タスクが実現すると共に計画そのものは終了した．しかし，この間におけるコンピュータ技術や極限作業技術の進歩には目覚しいものがあり，またそこで培われた技術は，現在様々な分野に広く普及している．この例からも，タスクオリエンテッドアプローチの有用性が伺える．

2.3 実験ロボット工学

次に**実験ロボット工学**による研究の進め方と，この研究方針の特徴について説明する．

ロボットの応用が期待されるのは，実環境（リアルワールド）である．ただし実環境という言葉が持つ意味は，ロボットのタスクに依存する．例えば，極限作業ロボットの研究における実環境は宇宙空間や海中であり，また産業用ロボットにとっての実環境は工場内の限られた領域である．本研究では，実環境を (1) 物理法則に支配された (Physical Constraint)，(2) ロボットのための整備をしない (Unstructured Environment)，(3) 人間と空間を共有

する屋内環境（Indoor Environment with Human）と定義した．具体的には，大学構内やオフィスビル内，または家の中などである．

この実環境で動作するロボットの研究を行うため，本研究では，まず実ロボットを構成して，可能なところから実環境中でロボットを動かし，その過程で理論やアルゴリズムを探求するという研究方針を採用し，この研究方針を「実験ロボット工学」と名付けた．この方針は，実環境におけるロボットの問題点を発見するという点で，非常に有効である．

山彦プロジェクトでは，この実験ロボット工学という方針に沿って，実ロボットである山彦ロボット（図1.1）を使用し，移動ロボットのためのナビゲーション，センサ，制御など数多くの研究が行われてきた．実験を実環境で行うことで，シミュレーションやトイワールド（Toy World: ロボットの実験を行うために構成した人工的な環境）での実験では分かりにくい，実環境に則した数多くの問題が明確となり，これに対処するために数多くのアルゴリズムが提案されてきた．このように，実験を行うことで問題点を明確にしつつ研究を進める点が実験ロボット工学の特徴である．

2.4 研究手順

核となる理論やアルゴリズムから目標動作を導き出すシーズオリエンテッドアプローチに対し，タスクを決めてそれに向かって研究を進めるという点で，タスクオリエンテッドアプローチはトップダウン的なアプローチである．これに対し，実験ロボット工学は実環境でロボットを動かすことで動作やアーキテクチャ，アルゴリズムなどを積み上げてゆくという点で，ボトムアップ的なアプローチである．現実問題として，ドアの通り抜け動作を実現するためには，トップダウン的にタスク設定及び目標動作の設計を行い，その後実環境でロボットを動かしながら，ボトムアップ的に問題点を一つ一つ克服することが必要であると考えられる．そこで，これらのタスクオリエンテッドアプローチと，実験ロボット工学という方針に沿って，本研究は以下に示す手順で進められた．

まずタスクを設定し，このタスクを実現するためのアルゴリズムや動作設計を行った後，これらを実ロボットに搭載し実環境で実験を行う．この実験過程で問題となった点を解析し，アルゴリズムや動作設計の変更を行う．この手順を繰り返すことで，設定したタスクの実現を目指すことにした．この手順を図2.2に示す．

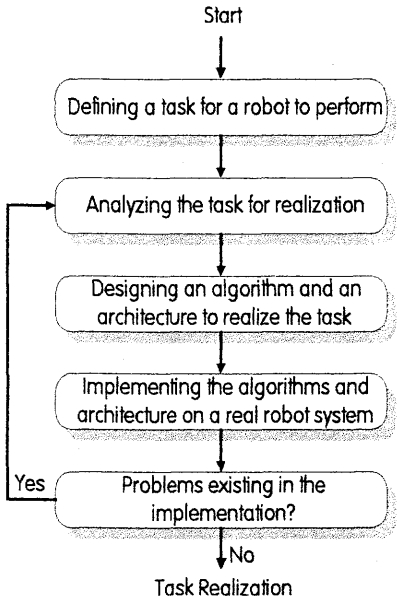

図 2.2: 研究手順

2.5 タスクが対象とする環境

ドアの通り抜けを含む自律ナビゲーションというタスクが実現するかどうかは，対象とする環境に大きく依存する．そこで本研究では，対象とする環境の設定を具体的に行った．

対象とする環境は我々の大学の研究室，及びその部屋に接続する廊下に限定した．この環境の概観を図2.3に示す．この環境の床面はほぼ平らで，凹凸があるとしても$5mm$以下である．また，部屋と廊下は片開きドアで接続され，ドアは通常閉まっている．このドアは，ノブを回すことで支えが外れ，その状態でノブを押すか引くことによってドアを開けることが可能となる．ドアにはバネが付いており，支えがなければバネの力によってドアが閉まる機構が備わっている．ただし，ドアが90度開いた場合，ドアを支える機構が作動し，ドアは力を加えない限りその状態を保持する．つまり，ドアの開き角が90度以内であれば，開いているドアは自動的に閉まるが，ドアを90度以上開けると，支えを外してもドアはその状態を保持する．ドアのサイズは，縦$200cm$，横$90cm$，幅$5cm$である．ドアのノブは円筒形で，高さ約$100cm$，ドアの端から約$5cm$のところに中心がくる．ノブの半径は約$2.5cm$でドア面から約$6cm$のところにノブの先端がくる．この環境及びドアは，日本の大学やオフィスビルなどの屋内環境では，ごく一般的なものである．

図 2.3: 環境（筑波大学 3L-402 の扉と付近の廊下）

2.6 タスクの設定

前節で設定した環境の下，より具体的なタスクの設定を行った．

タスクの前提として，ロボットは自立型（アクチュエータ，センサ，バッテリ，コントローラなど全ての機能をロボットが有する）の移動マニピュレータであり，現在位置，目的地，及びドアの位置やランドマークの位置などの環境情報を有するものとした．この条件の下で，目標タスクをより具体的に「移動マニピュレータが自らの力でドアを通り抜け現在位置から目的地まで自律ナビゲーションをおこなうこと」と設定した．この動作のイメージを図2.4に示す．

さて，自律移動ロボットによるドアの通り抜けは，全てのドアを自動ドアにしたり，無線通信によってロボットからドアを開ける信号を出すなど，ドアの方に工夫をすることによっても実現することができる．しかし，現存する大学やオフィスビルなどの設備をロボットだけのために改造することは現実的ではない．むしろ，複雑な動作を必要とするが，搭載したマニピュレータを用いてロボット自身が人間と同じ手順でドアを開ける動作を実現する方が，このロボットが適用できる範囲は広がる．したがって本タスクには，搭載したマニピュレータを用いたロボット自らが行うドアの通り抜け動作を盛り込んだ．

また予定外の状況（例えば走行経路上に障害物が存在する場合や，ドアに鍵がかかっている状況）にも対処すること，動作中人に危害を加えないことや，物を壊さないことも，本

タスクで実現すべき項目である.

一方，将来複数ロボットが同一環境で活動する状況を考えた場合，環境の状態を変えないということが，複数の自律ロボットが協調する際の必須条件となると予想される．そこで，ドアを開けた後にこれを閉めるという動作を行うことも，本タスクに含めることにした．したがって，開けたドアを閉めるというロボットの動作は，礼儀正しい動作であるという以上に，自律ロボットにとっては重要な動作である.

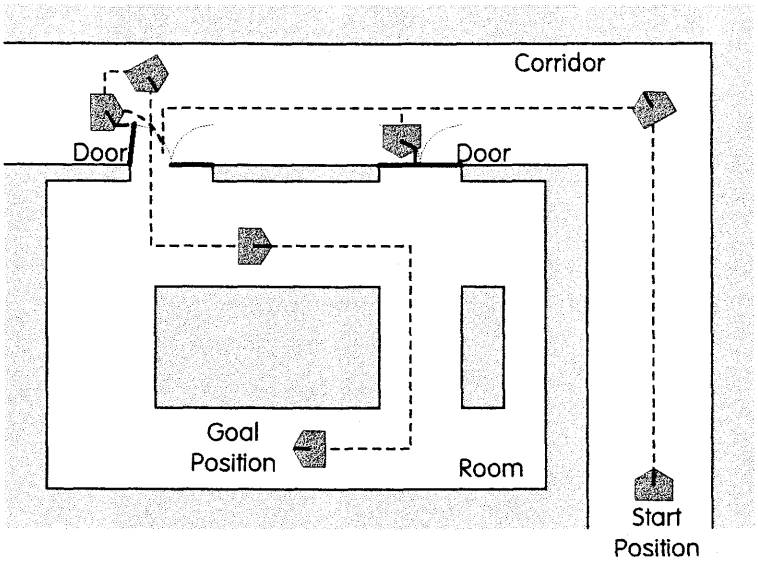

図 2.4: 目標タスクのイメージ

2.7 まとめ

本章では，本研究で採用した研究方針であるタスクオリエンテッドアプローチ，及び実験ロボット工学の手順や特徴について述べた．また，本研究で具体的に設定した環境及びタスクについて説明した.

第 3章

タスクの分析

3.1 はじめに

タスクオリエンテッドアプローチでは，タスクを設定した後，そのタスクの分析を行う．本研究では，2.6節で設定したタスク「ドアの通り抜けを含む自律ナビゲーション」の分析を行うため，このタスクを実現する動作を幾つかの部分動作のシーケンスで表現した．本章では，この各部分動作毎に行った分析について報告する．

3.2 ロボットの動作の分割

目標タスク「移動マニピュレータによるドアの通り抜けを含む自律ナビゲーション」を実現する動作を分析するため，この複雑な動作を幾つかの部分動作に分割した．以後，本論文では，特に注釈のない限り移動マニピュレータをロボットと呼ぶことにする．

さて，実環境におけるドアは，人間が開閉するように設計されているため，ロボットによるドアの通り抜け動作も，人間が行うドアの通り抜け動作と同様の手順で行うことができる．この動作は，表3.1に示すように，時間の経過に沿って動作のまとまり毎に分割した部分動作のシーケンスで表現できる．したがって，この部分動作のシーケンスを順次実行することで，目標タスクは実現される．次節より，表3.1に示した各部分動作の分析について述べる．

(1)	ロボットの走行経路と動作の計画
(2)	ドアの前まで経路追従走行
(3)	ノブの把持動作
(4)	ドア開け動作
(5)	ドアノブのリリース
(6)	ドアの裏側まで走行
(7)	ノブの把持動作
(8)	ドア閉め動作
(9)	ドアノブのリリース
(10)	目的地まで経路追従走行

表 3.1: ドアの通り抜けを含む自律ナビゲーションを実現するための部分動作のシーケンス

3.3 ロボットの走行経路と動作の計画

まず，表3.1中の(1)「ロボットの走行経路と動作の計画」に関する分析を行う．目標タスクは，自律ロボットによる自律走行なので，ここでは動作を開始する前にロボット自身が，与えれた環境モデル，現在位置，目的地から走行経路及びドア開けなどの動作の計画を行う必要がある．

さて，実環境でロボットが動作する以上，予定外の状況が発生した場合，ロボットはその状況を認識し対処しなければならない．例えば，経路上のドアに鍵がかかっている場合，この状況を認識せずに計画にしたがってドア開け動作を行えば，ロボットはこの動作に失敗するだけでなく，ドアを破壊する可能性もある．したがって，ロボットは，動作中センサを用いてロボットの周囲の状況を監視することが必要不可欠である．仮に，動作の継続が不可能であると判断した場合，ロボットは現在の動作を中止し，経路及び動作の再計画を行わなければならない．したがって，走行経路及び動作の計画を行う能力は，ロボット自身が有する必要がある．

3.4 経路追従走行の分析

次に表3.1中の(2)，(6)，(10)で行われれる動作「経路追従走行」に関する分析を行う．ここでは，マニピュレータの土台となる移動ロボット（以下ベースロボットと呼ぶ）が計画した走行経路にしたがって走行する．

さて，移動マニピュレータの制御を行う際，如何に精度良くベースロボットの位置推定を行い正確に目標経路を走行できるかが，ひとつの重要な鍵である．なぜならば，ベースロボットの位置誤差は，直接マニピュレータの先端の位置誤差に影響するからである．したがって，ベースロボットは，できる限り正確に目標経路を走行する必要がある．

さて，屋内環境における車輪型移動ロボットの位置推定には，通常オドメトリが使用される．この理由として，オドメトリは移動ロボットの駆動輪にエンコーダを備え付けるだけで容易に実現できるという点と，屋内という地面が平らな環境では，オドメトリによって比較的精度の良いロボットの位置推定が行えるという点が挙げられる．特に，本研究のように，実環境でベースロボットを走行させる場合，環境中にガイドラインやランドマークを設置する必要の無いオドメトリによる位置推定は有効である．

しかし，このオドメトリを用いた位置推定には，車輪のスリップや地面の小さな凹凸により，走行するにしたがって推定位置に誤差が生ずるという問題がある．この誤差は，走行距離が短い場合は問題にならないが，走行距離が長くなるにしたがって誤差は累積し，ベースロボットは目標経路を大きく逸れる場合も起こり得る．そこで，ロボットがオドメトリをベースにした走行を行う場合，外界センサを用いた推定位置の修正を同時に行う必要がある．

3.5 ドアノブの把持動作の分析

次に表3.1中の(3)，(7)で行われる「ドアノブの把持動作」に関する分析を行う．この動作において，ロボットは，マニピュレータを制御して，マニピュレータの先端に搭載されるハンドによりノブを把持する．

仮にロボットが，ハンドの位置とノブの位置を正確に把握していれば，マニピュレータの姿勢制御によって，ロボットはノブを把持することができる．しかし現実には，ハンドの位置に誤差が含まれるため，マニピュレータの姿勢制御のみでノブを把持することは不可能である．このハンドの位置誤差の原因には，以下に挙げるものが考えられる．

1. ベースロボットの推定位置の誤差

   ベースロボットが走行するのに伴い，オドメトリによるベースロボットの推定位置には誤差が生ずる．経路追従走行の分析でも述べた通り，この誤差は環境中のランドマークを検出し，ロボットが有するランドマークモデルと比較することで減少させることができるが，ある程度の誤差は避けられない．このベースロボットの位置誤差が

原因で，ハンドの位置にも誤差が生ずる．

2. ベースロボットの傾き

一般に移動ロボットは，走行面の凹凸による本体の振動を吸収したり，小さな段差を乗り越えるため，サスペンション機能を有する．このため，床面の状況やマニピュレータの姿勢に応じて，ロボット本体が前後または左右に少し傾く場合があり，これが原因でハンドの位置にも誤差が生ずる．

3. マニピュレータの精度

一般に，移動ロボットに搭載するマニピュレータは小型軽量を目指して製作されるため，マニピュレータの各関節の剛性はそれほど高くない．このため，据置型のマニピュレータと比較してマニピュレータの先端の位置ぎめ精度は低く，これが原因でハンドに位置誤差が生ずる場合もある．

以上より，移動マニピュレータによるノブの把持動作を行う際には，ハンドの位置誤差が問題となる．そこで，この把持動作を実現するためには，外界センサなどを使用して，ハンドに対するノブの相対位置を検出する必要がある．

3.6 ノブのリリース動作の分析

表3.1中の (5), (9) で行われれる動作「ノブのリリース」の動作では，ロボットは把持したノブを離し，走行の妨げにならないように，マニピュレータを初期姿勢に戻す必要がある．

3.7 ドア開け動作・ドア閉め動作の分析

最後に表3.1中の (4), (8) で行われれる「ドア開け動作・ドア閉め動作」に関する分析を行う．この動作において，ロボットは，ハンドで把持したノブを操作してドアを開閉し，ドアの通り抜け動作を行う．

一般にロボットに搭載するためのマニピュレータは小型で，その長さには限りがある．したがって，移動マニピュレータによってドアを開け閉めする際，ベースロボットは走行しつつマニピュレータがドアを操作するといった，走行とマニピュレータ制御の協調動作が必要となる．

一方，ドアの押し開け・引き開け動作を行う際，ベースロボットは走行するため，ドアの通り抜け動作中にもオドメトリには誤差が生ずる．しかし，ドアの片側は環境中に拘束されているため，ノブを把持したハンドの位置をずらすことができないので，ノブに大きな力がかかる恐れがある．したがって，搭載するマニピュレータには，この力を吸収する機構が必要となる．

3.8 まとめ

本章では，目標タスク「移動マニピュレータによるドアの通り抜けを含む自律ナビゲーション」を実現するために，ロボットが行う動作を時間の経過に沿って動作のまとまり毎に分割し，各部分動作の動作分析を行った．この分析より，目標タスクを実現するためには，ベースロボットの経路追従走行，ベースロボットの位置推定，マニピュレータの姿勢制御，マニピュレータの先端にかかる力の吸収機構，外界センサによるノブの認識機能，ベースロボットとマニピュレータの協調動作，などを実現する必要があることが明らかとなった．次章では，ここで分析したタスクに基づいて行った動作設計について紹介し，構築した実ロボットシステム全体の説明を行う．

第 4章

動作設計とロボットシステムの構築

4.1 はじめに

本研究では，第3章で行ったタスクの分析に基づいて，タスクを実現するためのロボットの動作を設計した．本章の前半では，設計したこのロボットの動作について部分動作毎に説明する．また，この複雑なタスクを実現するためのシステムには，メカニズム，センサ，コントローラといったハードウエアから，動作を決定しハードウエアをコントロールするためのソフトウエアまで必要である．本研究では，これらのハードウエア・ソフトウエアを実現し，これらを1つのロボットシステムに統合して，タスクの実現を目指した．本章の後半では，本研究で構築したロボットシステムの全体像について説明する．

4.2 ドアの通り抜けを含む自律ナビゲーションの動作設計

第3章の分析に基づいて，タスクを実現するためのロボットの大まかな動作を，以下のように設定した．

まずロボットは，走行経路及び動作の計画をした後，計画した経路の追従走行を行う．ただし，走行経路上にドアが存在する場合，ロボットはノブを把持してドアを開けた後，裏側に回り込んでドアを閉め，再び目的地に向けて経路の追従走行を行う．

この一連の動作を，表3.1に示すように，時間の経過に沿って動作のまとまり毎に部分動作に分割し，各部分動作の設計を行った．次節より，この設計に関する詳細について述べる．

4.2.1 走行経路と動作の計画

2.6節のタスクの設定でも述べたように，ロボットは，環境情報，現在位置，目的地が予め与えられるため，ロボットはナビゲーションを始める前に，自ら走行経路及び動作を計画することにした．この計画と実行を分けた理由は，動作中のロボットの計算機の負荷を軽減するためである．

また，ロボットがナビゲーション中，センサ情報より計画の遂行ができないことが検知された場合（例えばドアに鍵がかかっている場合など）には，ロボットは直ちに動作を停止し，その地点からの動作の再計画を行うことにした．このため，走行経路及び動作の計画機能はロボットに搭載することにした．

4.2.2 経路追従走行

この部分動作では，計画した走行経路に沿って，ベースロボットが正確に追従走行することが求められる．ここでは，基本的にはオドメトリを用いて推定したベースロボットの位置にしたがって，経路追従走行を行うことにした．ただし，オドメトリを用いて位置推定を行う際には，タイヤのスリップや地面の凹凸が原因で生ずるベースロボットの位置の累積誤差が問題となる（図4.1-(a)参照）．この累積誤差に対処するため，外界センサを用いて環境中のランドマークを獲得し，ロボットが有するランドマークモデルと比較することでベースロボットの推定位置の修正を行うことにした．さて，環境が屋内であるため，ロボットの動作範囲内には超音波を良く反射する平らな壁が複数存在する．そこで，外界センサとして超音波距離センサを使用し，環境中の平らな壁をランドマークと見なして，この壁を用いたベースロボットの位置修正を行うことにした．この位置修正を壁に沿った経路上で繰り返し行うことで，目標経路に沿った精度の良い経路追従が実現する（図4.1-(b)参照）．またベースロボットは，走行経路上の障害物に衝突しないように，常に超音波距離センサで前方を監視し，障害物が走行経路上に存在することが検知された場合には，直ちに走行を中止し，この障害物が取り除かれるまで待機することにした．

以上より，この動作を行うためにロボットに搭載する機能は，ベースロボットの走行機能，超音波距離センサ及びオドメトリを用いたベースロボットの位置推定機能，走行経路上の障害物の認識機能である．

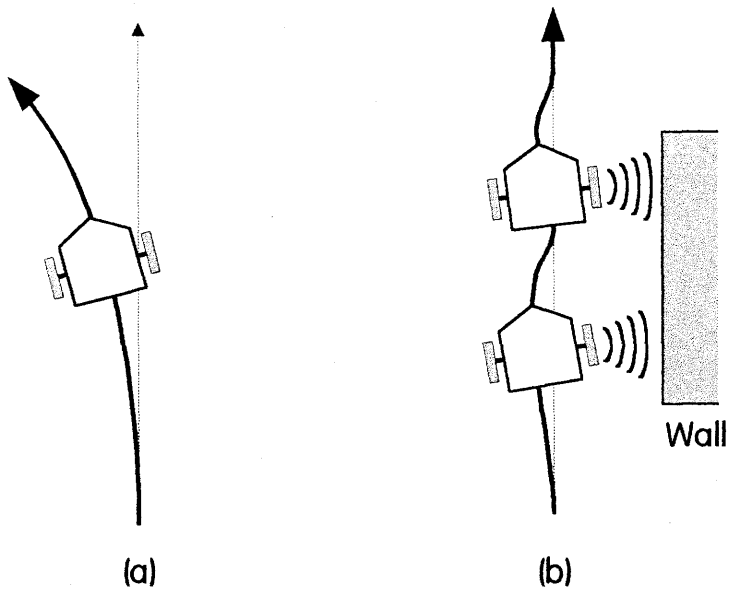

図 4.1: オドメトリの誤差の問題とベースロボットの位置修正

4.2.3 ノブの把持動作

3.5節の動作分析で述べた通り，ノブの把持動作で問題となるのはハンドの位置誤差である．この誤差は，強力な外界センサを用いてベースロボットの位置を精度良く推定することで多少減少させることはできるが，ロボットが走行する限りゼロにはならない．そこで，ハンドに視覚センサを搭載することで，ノブとの相対位置を認識し，その情報をマニピュレータの制御にフィードバックすることで，ノブの把持動作を行うことにした．具体的な設計を以下に述べる．

まずロボットは，ノブから少し離れたところにハンドが来るようにマニピュレータを制御する．次に視覚センサを用いてハンドとノブの相対位置を検出し，ノブを把持する方向にハンドが移動するように，マニピュレータを制御する．このノブの位置認識とハンドのアプローチを繰り返すことで，ノブを把持できる位置にハンドが制御され，この後ノブの把持及び回転を行う．

以上より，この動作を行うためにロボットに搭載する機能は，マニピュレータの姿勢制御機能，視覚センサを用いたノブの位置認識機能，ハンドの指の開閉機能である．

4.2.4 ノブのリリース動作

ノブのリリースは，ハンドがノブを離し，マニピュレータをベースロボットの走行の障害とならない姿勢に戻す動作である．この動作は，ベースロボットが停止した状態で行うことができるため，ハンドがノブを離した後，マニピュレータの姿勢制御のみで行うことにした．

以上より，この動作を行うためにロボットに搭載する機能は，マニピュレータの姿勢制御機能，及びハンドの指の開閉機能である．

4.2.5 ドア開け動作・ドア閉め動作

3.7節の分析でも述べた通り，ドア開け動作・ドア閉め動作を行う際，マニピュレータの長さに限りがあるため，走行とマニピュレータ制御の協調動作は必須である．また，ベースロボットの推定位置誤差が原因で，把持したノブに力が生ずるという問題もある．そこで，ベースロボットとマニピュレータの協調動作を行いドアを開閉しつつ，力センサによるハンドの先端のコンプライアンス制御を実現することにした．また，ドアが閉まっていれば，ノブの位置は空間中に固定されるため，ノブを把持するマニピュレータの姿勢から，ベースロボットの位置誤差が推定できる．そこで，ドア開け動作を開始する時点で，このベースロボットの位置推定も行うことにした．

以上より，この動作を行うためにロボットに搭載する機能は，ベースロボットの走行機能，ノブを把持した姿勢からベースロボットの位置を推定する機能，マニピュレータの姿勢制御機能，力センサによるマニピュレータの先端のコンプライアンス機能である．

4.3 ロボットシステムの構築

4.2節の動作設計より，目標タスクを実現するために，表4.1に示す機能を実現することにした．

これらの機能を実現するためのロボットのメカニズム，センサ，コントローラ，及びソフトウエアについて設計及び実装を行い，これらを統合することで，目標タスクを実現するための1つのロボットシステムを構築した．本節では，このロボットシステムの概略について説明する．

(a)	走行経路及び動作の計画機能
(b)	ベースロボットの走行機能
(c)	超音波距離センサによるベースロボットの位置推定機能
(d)	オドメトリによるベースロボットの位置推定機能
(e)	ノブを把持したマニピュレータの姿勢からベースロボットの位置を推定する機能
(f)	ハンドの指の開閉機能
(g)	視覚センサによるノブの位置認識機能
(h)	マニピュレータの姿勢制御機能
(i)	力センサによるマニピュレータのコンプライアンス機能
(j)	超音波距離センサによる走行経路上の障害物の検知機能

表 4.1: 目標タスクを実現するためロボットに搭載する機能

4.3.1 メカニズム

まず，構築したロボットシステムのメカニズムについて説明する．

表4.1-(b)に示した走行機能を実現するためには，マニピュレータの土台となる走行機能を有するベースロボットが必要である．さて，ロボットとマニピュレータの協調動作を考えた場合，ベースロボットには，走行する向きに対する拘束を考えなくてすむ全方向移動ロボットが有利である．しかし，目標タスクの基本はナビゲーションであり，長距離走行を行う際に生ずるオドメトリ誤差を考慮すると，全方向移動ロボットはベースロボットとして適当でない．そこで本研究では，左右に取り付けられた駆動輪による走行機能を有する山彦プロジェクトで設計・製作が行われた「自律型移動ロボット山彦てん」を使用することにした．このベースロボットについては5.2.1節で詳しく説明する．

また，表4.1-(h)，表4.1-(i)に示したマニピュレータの制御を行うため，ベースロボットには7関節マニピュレータを搭載した．また，表4.1-(f)に示したハンドの指の開閉を行うため，ロボットハンド（以下これをハンドと呼ぶ）をマニピュレータの先に搭載した．このマニピュレータ及びハンドについては，本研究で設計及び製作を行った．これらの詳細は，5.2.2節及び5.2.3節で説明する．

4.3.2 センサ

次に，本ロボットに搭載したセンサについて述べる．

外界センサとしては，表4.1-(g)に示したノブの位置認識を行うため，視覚センサをハン

ド上部に設置した．また表4.1-(i)に示したコンプライアンス制御を実現するため，力センサをマニピュレータの先端に搭載した．また表4.1-(c)に示した位置推定機能，及び表4.1-(j)に示した走行経路上の障害物を検知する機能を実現するため，超音波距離センサをベースロボットの周囲8個所に搭載した．

一方，内界センサとしては，表4.1-(d)に示したオドメトリによる位置推定機能を実現するため，ベースロボットの駆動輪にはロータリエンコーダを取り付け，オドメトリシステムを実現した．また表4.1-(h)に示したマニピュレータの姿勢制御や，表4.1-(e)に示したベースロボットの位置推定を行うため，マニピュレータの姿勢を検知するためのロータリエンコーダを，7関節マニピュレータの各関節を駆動するモータに取り付けた．これらの各センサに関する詳細は，5.4節で詳しく述べる．

4.3.3 コントローラ

次にメカニズムやセンサを制御するためのロボットのコントローラについて説明する．

ロボットの各機能を分散開発できるように，本研究におけるロボットのコントローラは機能分散コントローラ[SIY93]で構成した．この機能分散コントローラでは，各機能はCPU，RAM，ROMによって構成されたコンピュータシステムを有し，この上で動作する専用のソフトウエアによって，センサまたはアクチュエータが独立に制御される．この機能の単位を機能モジュールと呼ぶ．

表4.1に示す機能を実現するため，ロボットには，走行制御モジュール，位置推定モジュール，マニピュレータ制御モジュール，力センサモジュール，視覚センサモジュール，超音波距離センサモジュール，音声モジュールを搭載した．これらの機能モジュールは，ロボット全体の動作を決定するために搭載したマスタモジュールによって統括管理される．このマスタモジュールと各機能モジュール間の通信は，双方向メモリと専用バスを用いて実現された．ただし，力センサモジュールとマニピュレータ制御モジュール間，及び走行制御モジュールと位置推定モジュール間の通信はトランスピュータリンクを通じて行う．この機能分散コントローラに関する詳細は，5.5節で詳しく述べる．

4.3.4 ソフトウエア

最後に機能分散コントローラを制御し，ロボットの全体の動作を実現するためのソフトウエア構成について説明する．

ロボットのコントローラを機能分散で構成したため，ロボットのソフトウエアも，機能

モジュール上で動作するソフトウエア（第7章）と，マスタモジュール上で動作するソフトウエア（第8章・第9章）の2階層で構成することにした．

各機能モジュール上のソフトウエアは常に動作し，それぞれの機能を担当する．例えば，走行制御ソフトウエアは，常にロボットを目標軌跡に追従させるように動作し，また測距ソフトウエアは，超音波距離センサを用いて環境中の物体までの距離を常に計測する．一方マスタモジュール上のソフトウエアは，各機能を統括し，ロボット全体の動作を決定するものである．ここでは，このソフトウエアを更に2階層に分割し，各部分動作を実現するためのアクションプリミティブというソフトウエア群と，この部分動作を統括する統括制御ソフトウエアで実現した．4.1-(a)に示した走行経路及び動作の計画機能は，この統括制御ソフトウエア上で実現する．

4.3.5 システムインテグレーション

以上に挙げたメカニズム，センサ，コントローラ，ソフトウエアを，機能分散のコンセプトにしたがって，ベースロボット「山彦てん」上に搭載した．この構成を図4.2に示す．このシステムでは，マスタモジュール上の統括制御ソフトウエアは，各部分動作を実現するソフトウエア「アクションプリミティブ」を実行する．このアクションプリミティブは，担当する部分動作を実行するため，バスを通じて各モジュール上のソフトウエアと通信し，動作コマンドの送信やセンサ情報の受信を行う．各機能モジュールのソフトウエアは，アクションプリミティブからの動作コマンドにしたがってロボットのアクチュエータの制御を行い，またセンサから得られた環境情報をアクションプリミティブに返す．

以上に示したように，機能分散のコンセプトにしたがって機能を分散し，これを統合することで，目標タスクを実現するための実ロボットシステムを構築した．

4.4 まとめ

本章では，第3章のタスクの分析にしたがって，目標タスクを実現するために設計したロボットの動作について説明し，その動作を実現するために構築したロボットシステムの概要について述べた．

この構築したロボットシステムのハードウエア及びコントローラについては第5章で述べる．またソフトウエアについては第6章から第9章にかけて詳しく述べる．

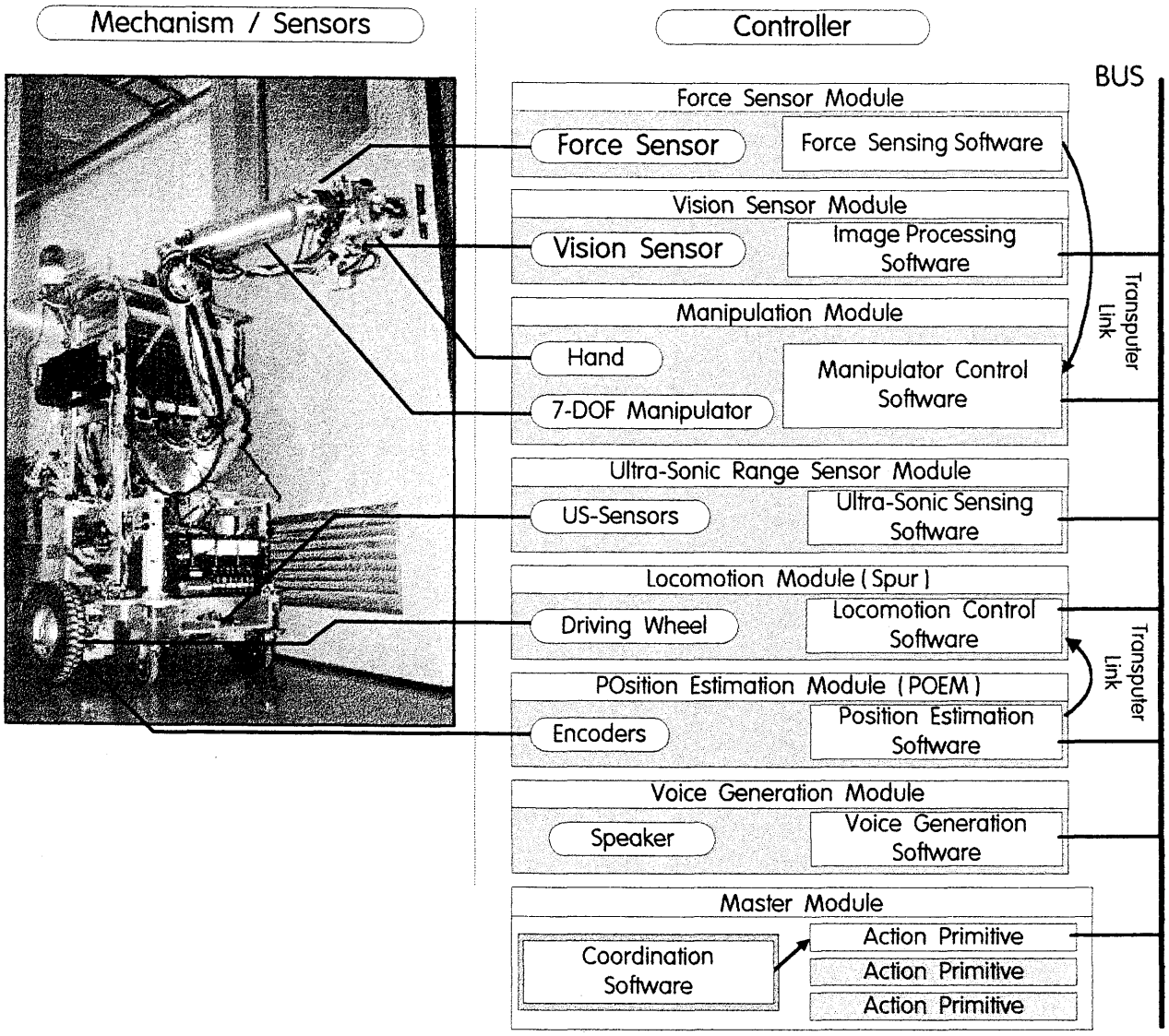

図 4.2: ロボットのシステムインテグレーション

第 5章

ハードウエア

5.1 はじめに

本章では，4.3節の「ロボットの構築」で概略を説明したロボットシステムの，特にハードウエアに関する詳細について説明する．

本研究で使用するベースロボットには，山彦プロジェクトで研究開発が行われた「自律型移動ロボット山彦てん」を使用し，このベースロボットに搭載する小型軽量マニピュレータとロボットハンドの設計及び製作を行った．これらのロボットのメカニズムについては5.1節で述べる．このメカニズムを駆動するアクチュエータにはDCモータを使用した．5.3節では，このDCモータの駆動方式について紹介する．また外界センサとして，このロボットには，ランドマークである壁までの距離の測距，及び経路上の障害物の検知を行う超音波距離センサ，コンプライアンス制御を行うための力センサ，ノブの位置を認識するための視覚センサを搭載した．5.4節では，これらのセンサの仕様について述べる．以上のメカニズムやセンサを制御するためのロボットのコントローラには，機能分散コントローラを採用した．5.5節では，このコントローラの構成を紹介する．

5.2 ロボットのメカニズム

5.2.1 ベースロボット「自律型移動ロボット山彦てん」

本研究では，走行機能，マニピュレータ，各種センサ，コントローラ及びロボットの電源を搭載するためのベースロボットとして，山彦プロジェクトにおいて研究・開発が行われてきた「自律型移動ロボット山彦てん」を使用した．以下このロボットの概観，特徴，走行方式，アクチュエータ，コントロールボード，及び電源ユニットについて述べる．

概観

「山彦てん」は，車輪型移動ロボットである．このロボットのサイズは，高さ約 $90cm$，幅約 $70cm$，奥行き約 $50cm$ であり，バッテリーを含めた重さは約 $60kg$ である．したがって，重量が $20kg$ 程度の小型マニピュレータであれば，これを搭載する能力は十分にある．小型マニピュレータを搭載した山彦てんを前面から見た概観を図5.1に示す．

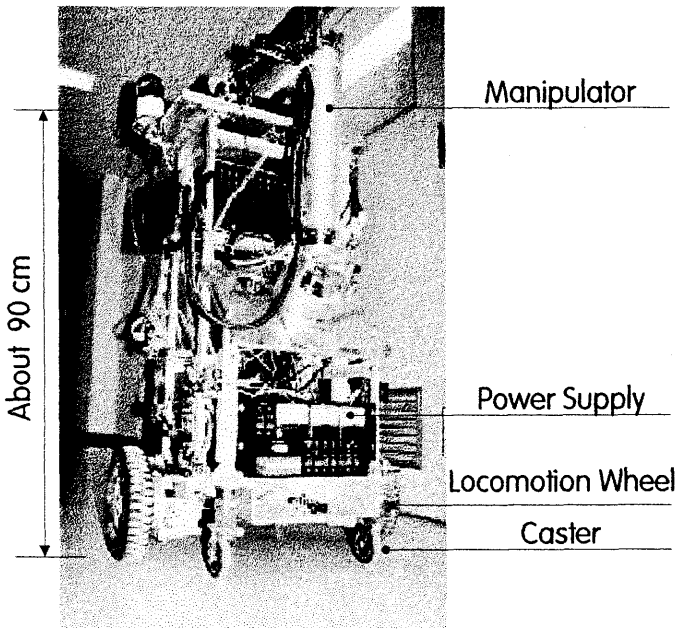

図 5.1: 自律型移動ロボット山彦てん（写真）

特徴

このベースロボットの特徴は，アクチュエータ・コントローラ・電源などを全てロボット内部に有する自立型（Self Contained）ロボットであるという点である．また，もう一つの特徴は，搭載したコントローラによって，ロボット自ら動作を決定し，自律的に動作を行うことができる自律型ロボットであるという点である．

走行方式

このロボットは，左右に備え付けられた駆動輪を用いて段差の無い屋内環境を走行する能力を有する．また，左右の駆動輪の回転差を用いて進行方向を操舵する，パワーホイー

ルドステアリング（Power Wheeled Steering）を有する．ロボットの本体は，この駆動輪のほかに4つのキャスタによって支えられる．また左右の駆動輪にはロータリエンコーダが搭載され，これを用いて位置推定を行うオドメトリ機能を有する．ベースロボットの駆動輪及びキャスタの概観を図5.2に示す．

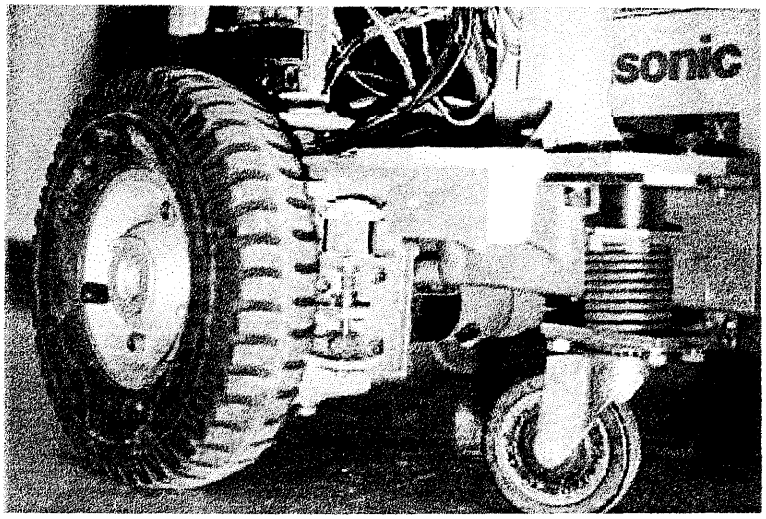

図 5.2: 山彦てんの駆動輪及びキャスタ（写真）

アクチュエータ

ベースロボットの駆動輪のアクチュエータは，三洋電気製の$24V45W$ のDCモータである．このモータの回転を 60：1 に減速することでベースロボットの駆動輪は駆動される．また，モータの制御には，PWMスイッチング方式と呼ばれるDCモータ制御方式を使用している．このモータ制御方式及びモータドライバの詳細については5.3節で述べる．

コントロールボード

ロボットのコントローラは，各機能毎に$25cm \times 15cm$のボード上に実装され，これらのボードは，ロボット後部のラックの溝に沿って，縦置きに挿入される（図5.3参照）．これらのボードは，100ピンのコネクタによって，ロボットのバス及び電源を供給するバックボーンボードに接続する．このコントローラに関する詳細は，5.5節で示す．

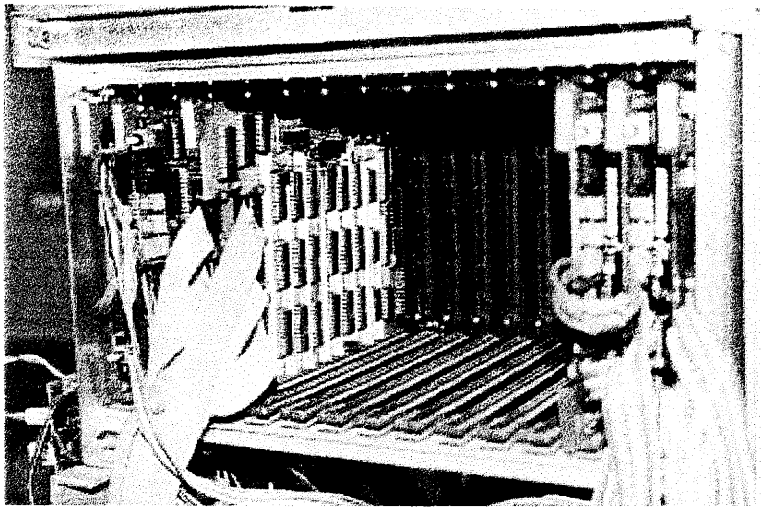

図 5.3: コントローラが配置されるラック（写真）

電源ユニット

ロボットの電源は，直列に接続した2つの普通自動車用12V鉛蓄電池を使用する．電源ユニットはスイッチングレギュレータを用いて，この24Vの電源から，5V系，±12V系，±15V系，24V系 の電源を作る．各電源の用途は以下の通りである．コントロールボード上のコンピュータやロジック回路には，5V系の電源を使用する．視覚センサに使用するカメラなどの電源には，±12V系の電源を使用する．力センサのオペアンプの電源には，±15V系の電源を使用する．各モータを駆動するための電源としては，24V系の電源を使用する．

この電源ユニットの概観を図5.4に示す．電源ユニットの前面には，メインスイッチ，各電圧に対応した電源のブレーカー及び電流計が配置されている．また，バッテリは，この電源ユニットの後部に位置する．

5.2.2 移動ロボット搭載用小型マニピュレータ

ドアを開閉する動作を実現するためのメカニズムとして，人の腕とほぼ同じサイズの小型マニピュレータの設計及び製作を行い，ベースロボットに搭載した．

一般に市販の据置型マニピュレータは，先端の位置精度を上げるため，各関節の剛性を上げる努力がなされており，しかもベース部の重量は重い方が，マニピュレータ全体が安

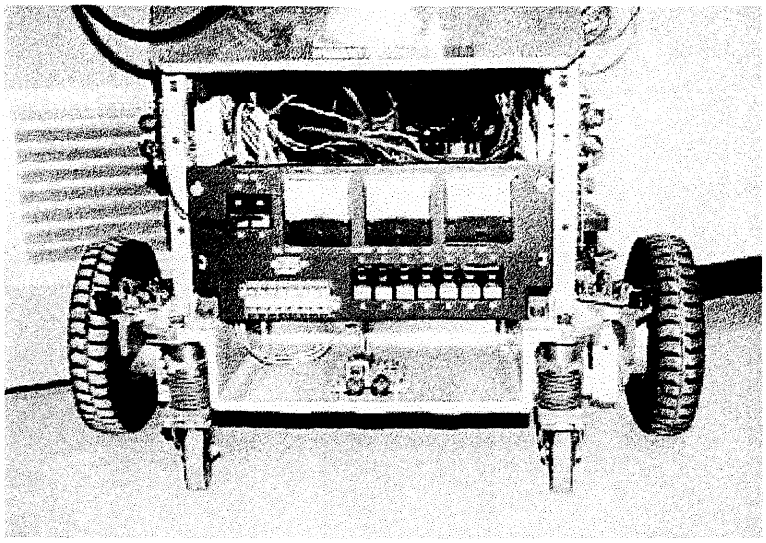

図 5.4: ロボットの電源ユニット（写真）

定する．したがって，市販の人の腕と同サイズのマニピュレータは，サイズが小さいにも関わらず非常に重い．これに対し，移動ロボットに搭載するマニピュレータは，高い剛性をそれほど必要とせず，むしろ軽量であることが望まれる．そこで本研究では，搭載するマニピュレータを自作することにした．

移動ロボット搭載用小型マニピュレータの設計

マニピュレータは，自律型移動ロボット「山彦てん」に搭載することを念頭に，軽量化に重点を置き，マニピュレータのメカニズムの設計を行った．

1. マニピュレータのサイズと関節の配置

   マニピュレータの長さは，マニピュレータの先端に搭載する力センサの幅まで含めて，人の腕の長さより少し長い93cmとした．また，今後の移動マニピュレータや冗長マニピュレータの研究にも適用できるように，関節の配置を人の腕と同じ7自由度とした．マニピュレータのサイズおよび関節の配置を図5.5に示す．

   このマニピュレータの関節配置の特徴は，マニピュレータの第6関節の先に力センサを搭載し，そのセンサの先にノブを回転するための第7関節を搭載した点である．これにより，ノブの回転とは独立に，力センサより下の6つの関節（第1関節～第6関節）によってコンプライアンス制御が実現できる．

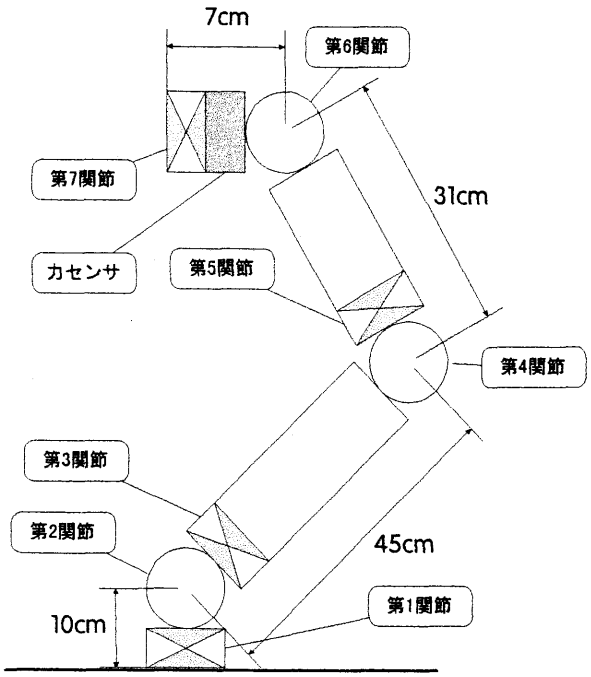

図 5.5: 移動ロボット搭載用マニピュレータのサイズ及び関節の配置

2. 各関節の必要トルクの算出

本研究におけるマニピュレータの役割は，物の運搬ではなくドアノブの操作であるため，マニピュレータの可搬重量は小さくても構わない．そこで，本マニピュレータの可搬重量を$2kg$と設定し，必要トルクを計算するため，マニピュレータの重量配分が図5.6に示すようになると仮定した．ただしこの仮定は，モータの選定を行うために，設計段階の初期に行ったものなので，図5.5の設計と比較してマニピュレータの長さが若干異なっている．

このマニピュレータの各関節に必要なトルクを計算したので，以下にこの計算手順について述べる．

(a) 重力に対抗するためのトルク

まず，各関節が自重を支えるために必要なトルクについて計算する．ただしこのトルクはマニピュレータの姿勢によって変化する．そこで各関節にかかる自重が最大となる姿勢をマニピュレータがとった際に，その関節を支えるために

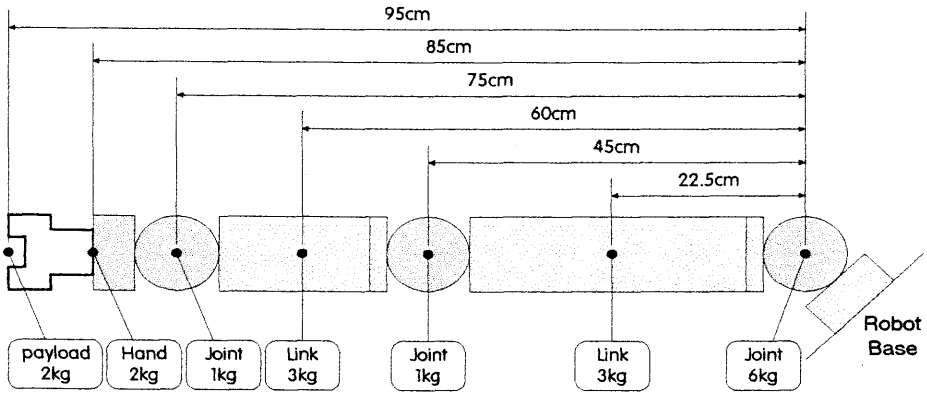

図 5.6: マニピュレータの重量配分

必要なトルクを，その関節の「重力に対抗するためのトルク」と定義し，このトルクの計算を行った．

例えば，第2関節（マニピュレータの根本）にかかるマニピュレータの自重が最大となるマニピュレータの姿勢は，図5.6で示される姿勢である．したがって，第2関節の重力に対抗するためのトルクτ_{g2}は，以下の計算式より

$$\begin{aligned}\tau_{g2} &= (0.225m \times 3.0kg + 0.45 \times 1.0 + 0.6 \times 3.0 + 0.75 \times 1.0 \\ &\quad +0.85 \times 2.0 + 0.95 \times 2.0) \times 9.8N/kg = 71.3Nm \end{aligned} \tag{5.1}$$

となる．以下同様に計算を行うと第1関節及び第3関節から第6関節までの重力に対抗するためのトルクは，

$$\begin{aligned}\tau_{g1} &= (0.225 \times 3.0 + 0.45 \times 1.0 + 0.6 \times 3.0 + 0.75 \times 1.0 + \\ &\quad 0.85 \times 2.0 + 0.95 \times 2.0) \times \sin(\pi/4) \times 9.8 = 50.4Nm \end{aligned} \tag{5.2}$$

$$\begin{aligned}\tau_{g3} &= (0.15 \times 3.0 + 0.3 \times 1.0 + 0.4 \times 2.0 + 0.5 \times 2.0) \\ &\quad \times 9.8 = 25.0Nm \end{aligned} \tag{5.3}$$

$$\begin{aligned}\tau_{g4} &= (0.15 \times 3.0 + 0.3 \times 1.0 + 0.4 \times 2.0 + 0.5 \times 2.0) \\ &\quad \times 9.8 = 25.0Nm \end{aligned} \tag{5.4}$$

$$\tau_{g5} = (0.1 \times 2.0 + 0.2 \times 2.0) \times 9.8 = 5.88Nm \tag{5.5}$$

$$\tau_{g6} = (0.1 \times 2.0 + 0.2 \times 2.0) \times 9.8 = 5.88Nm \tag{5.6}$$

$$\tau_{g7} = 0.0Nm \tag{5.7}$$

と計算される．ここで，τの添え字の数字が関節の番号である．また，図5.6に示すように，ベースロボット「山彦てん」のマニピュレータの設置面は地面に対して45°の角度を持つため，第1関節の重力に対抗するためのトルクには，$\sin(\pi/4)$が乗じられる．

(b) 目標時間内に目標角速度に到達するためのトルク

次に，目標時間内に目標角速度に到達するための各関節に必要なトルクについて計算する．ただしここでは，目標角速度に到達する目標時間を0.5秒とした．

まず，各関節の目標角速度を表5.1の上段に示す通り設定した．ただし第1，第2関節が急激に動くと危険なため，これらの関節の目標角速度は比較的小さく設定した．ここから，この目標角速度に0.5秒で到達するための目標角加速度は表5.1の下段に示す通り計算される．

	第1,第2関節	第3,第4関節	第5,第6関節
目標角速度 (ω)	$45deg/s$	$90deg/s$	$120deg/s$
目標角加速度 ($\dot{\omega}$)	$1.57rad/s^2$	$3.14rad/s^2$	$4.19rad/s^2$

表 5.1: 各関節の目標角速度と目標角加速度

一方，各関節における最大慣性モーメントは，図5.6より以下の通り計算される．まず第1，第2関節における慣性モーメントI_{12}は，腕が真っ直ぐに伸びた場合が最大で，

$$\begin{aligned} I_{12} &= \int_0^{0.45} \frac{3.0}{0.45}x^2dx + \int_{0.45}^{0.75} \frac{3.0}{0.3}x^2dx + 0.45^2 \times 1.0kg \\ &\quad +0.75^2 \times 1.0 + 0.85^2 \times 2.0 + 0.95^2 \times 2.0 \\ &= 5.32kgm^2 \end{aligned} \tag{5.8}$$

となる．同様に第3,第4関節における肘から先の慣性モーメントI_{34}，及び第5,第6関節に関する手首から先の慣性モーメントI_{56}は，それぞれ，

$$\begin{aligned} I_{34} &= \int_0^{0.3} \frac{3.0kg}{0.3m}x^2dx + 0.3^2 \times 1.0 + 0.4^2 \times 2.0 + 0.5^2 \times 2.0 \\ &= 1.18kgm^2 \end{aligned} \tag{5.9}$$

$$\begin{aligned} I_{56} &= 0.1^2 \times 2.0 + 0.2^2 \times 2.0 \\ &= 0.1kgm^2 \end{aligned} \tag{5.10}$$

となる．そこで，回転に関する運動方程式 $N = \dot{\omega} I$ を用いて，表5.1及び式(5.8)-(5.10)の最大慣性モーメントより，各関節の目標時間内に目標角速度に到達するためのトルクは，

$$\tau_{s1} = \tau_{s2} = I_{12} \times \dot{\omega} = 1.57 \times 5.32 = 8.35Nm \quad (5.11)$$

$$\tau_{s3} = \tau_{s4} = I_{34} \times \dot{\omega} = 3.14 \times 1.18 = 3.71Nm \quad (5.12)$$

$$\tau_{s5} = \tau_{s6} = I_{56} \times \dot{\omega} = 4.19 \times 0.10 = 0.42Nm \quad (5.13)$$

と計算される．

(c) 各関節の必要トルク

各関節に要求される最大トルクは，重力に対抗するためのトルクと，目標時間内に目標角速度に到達するためのトルクの合計値で表すことができる．したがって，式(5.1)-(5.7)，及び式(5.11)- (5.13)より，このマニピュレータの各関節に必要なトルクは，表5.2に示す通りとなる．

関節	重力に対抗するトルク	目標角速度を出すトルク	各関節の必要トルク
1	$50.40Nm$	$+8.35Nm =$	$58.75Nm$
2	$71.30Nm$	$+8.35Nm =$	$79.65Nm$
3	$25.00Nm$	$+3.71Nm =$	$28.71Nm$
4	$25.00Nm$	$+3.71Nm =$	$28.71Nm$
5	$5.88Nm$	$+0.42Nm =$	$6.30Nm$
6	$5.88Nm$	$+0.42Nm =$	$6.30Nm$

表 5.2: マニピュレータの各関節の必要トルク

移動ロボット搭載用小型マニピュレータの製作

前節で行ったマニピュレータの関節の配置，及び各関節に必要なトルクの計算に基づいて，ベースロボット「山彦てん」に搭載するための小型マニピュレータの製作を行った．

1. マニピュレータの材質

マニピュレータの部材には，軽量かつ加工が容易なアルミ材を使用した．

モータのブラケットやギアの取り付け部，マニピュレータを支えるパーツなどは，旋盤及びフライス盤を使用してアルミを削り出すことによって製作した．またマニピュ

レータのリンクには，アルミパイプを使用した．

2. モータの選定

各関節に搭載するモータの選定は，基本的には表5.2に示した各関節の必要トルクの2倍以上の停動トルクが得られる，モータとギアの組み合わせの中から行った．モータ本体はMAXON社のモータを使用した．選定した各関節の減速比及びスペックを表5.3に示す．また，第7関節のモータに関しては，ノブを回転することができるトルクさえ出せれば良いので，できるだけ小さいモータを選定した．

関節	モータ	停動トルク (mNm)	減速比	効率	出力トルク (Nm)	必要トルク (Nm)
1	15V 90W	899	960:1	50%	432	58.75
2	15V 90W	899	520:1	50%	234	79.65
3	24V 20W	241	520:1	50%	63	28.71
4	24V 20W	241	708:1	50%	85	28.71
5	24V 20W	241	520:1	50%	63	6.3
6	18V 6W	41	256:1	50%	5.2	6.3
7	24V 4.5W	35	84:1	50%	1.5	α

表 5.3: 選定した各関節のモータのスペック及び減速比

この表から分かる通り，第1関節と第2関節はマニピュレータ全体を支えるため，要求トルクの2倍以上のモータを搭載した．逆に小型化を図るため，第6関節のモータ及び減速ギアの出力トルクは，要求トルクとほぼ同じものを搭載した．

3. 各関節の機構

マニピュレータをできるだけスリムにするため，モータはリンクの内側に埋め込むことを目標とした．

第1関節については，モータの配置の都合上，歯付きベルトを用いてその回転軸とは離れた位置からこの関節を駆動することにした（図5.7参照）．ただし，この関節のモータの減速には，ハーモニックギア及びプラネタリギアを使用した．第2関節については，ハーモニックギアとプラネタリギアを用いて減速した後，スパイラルベベルギアを用いてモータの駆動方向を変えることにした．また，MAXONのモータは縦長であるため，第3関節から第6関節を駆動するモータは，リンクにモータが縦に入るようにして，リンクの内側に埋め込むことにした．ここで，モータ軸と関節の回転軸の向きが異なる関節（第4関節，第6関節）については，プラネタリギアを用いて減速したのち，ベベルギアを用いて駆動方向を変えることにした．また，モータ軸と

関節軸が同一直線上にある関節（第3関節，第5関節）については，プラネタリギア及びハーモニックギアで減速して直接駆動することにした．また第7関節では，モータの配置スペースの問題から，モータ及びプラネタリギアを力センサの脇に搭載し，ハンドの土台を駆動することにした．マニピュレータの根元にあたる第2関節及び第3関節の写真を図5.8に示す．

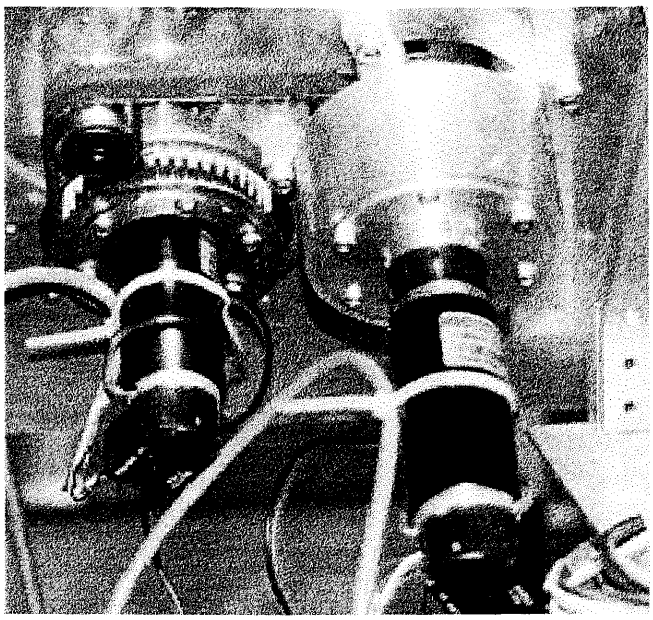

図 5.7: 第1関節及び第2関節の駆動モータ及び第1関節を駆動する歯付きベルト—マニピュレータの土台を裏側から見たところ（写真）

4. マニピュレータの概観

組みあがったマニピュレータの全体重量は約$14kg$であり，目標重量の$20kg$をかなり下回ることができた．また，第3関節から第6関節のモータがリンクの中に入っているので，全体としてスリムに仕上がった．ハンドを搭載したマニピュレータの写真を図5.9に示す．

製作した小型マニピュレータの評価

実際に各モータに電流を流して動作を行うと，第6関節のトルクは，見積もりよりも大きく，マニピュレータの動作に支障はなかった．この理由として，この関節では，$18V$の

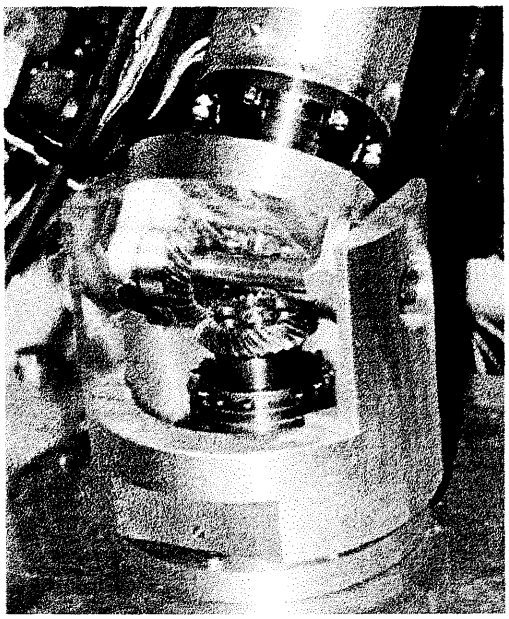

図 5.8: スパイラルベベルギアを用いた第2関節とハーモニックギアを用いた第3関節（写真）

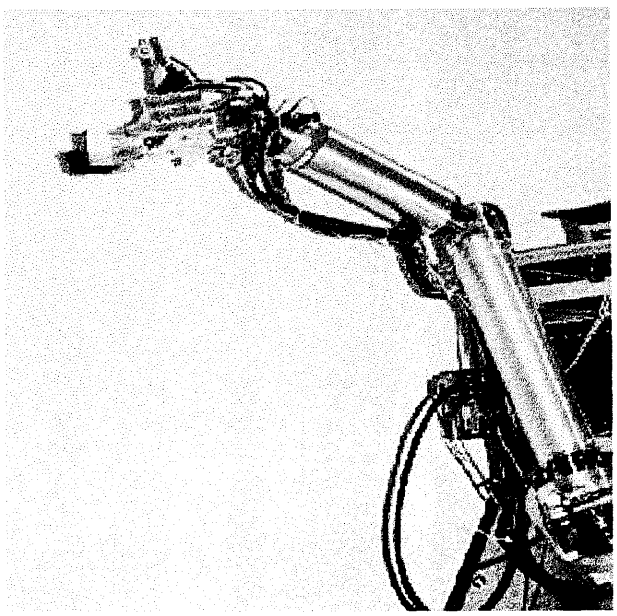

図 5.9: 搭載用小型軽量マニピュレータ（写真）

モータを24Vで動作するため，実際の停動トルクが見積もりより大きくなったということ，また減速による効率の低下も，実際には 50% より高いということが考えられる．

一方，第4関節など，駆動にベベルギアを採用している関節については，スパイラルタイプのベベルギアを使用したにも関わらず，ギアのかみ合わせの部分で予想以上のバックラッシュが起こった．また，減速の最終段をプラネタリギアで行う関節については，ギアの中で起こるバックラッシュも予想以上に大きかった．特に，第6関節（手首の関節に相当）で生ずるバックラッシュは大きく，マニピュレータの先端の位置精度が著しく低下した．逆に最終段の減速をハーモニックギアで行った第3，第5関節については，バックラッシュはほとんど見受けられなかった．このことから，今後このタイプのマニピュレータを製作する際，全てのモータの減速の最終段は，ハーモニックギアを採用し，ベベルギアは使用すべきでないということが明らかになった．

また，ベルトで駆動する第1関節については，ベルトの強度の見積もりを誤ったため，このベルトの寿命が短いという問題が生じた．現在，このベルトは約2ヶ月毎に交換している．

以上より，製作したマニピュレータは，先端の位置精度が悪い点など問題も多いが，移動ロボット搭載型の必須条件である小型軽量という要求は十分に満たしているので，本研究で使用することにした．また，小型マニピュレータを製作するための知見を得るための，プロトタイプとしての役割については十分に果たしたといえる．

5.2.3 ロボットハンド

ノブの把持機能を実現するためのメカニズムとして，製作した小型マニピュレータに搭載可能なロボットハンド（以下ロボットハンドを単にハンドと表記する）の設計・製作を行い，マニピュレータの先端に搭載した．

ロボットハンドの設計

目標タスクを実現するためにハンドに求められる能力は，ノブを把持する能力である．そこで，マニピュレータに搭載するハンドは，スライド型の2指ハンドとした．

対象とするノブの形はシリンダー状なので，指の内側はノブの外形に合せてカーブさせることにした．また指のストローク幅については，ノブの把持及びノブのリリースができる幅とした．また，視覚センサをこのハンドに搭載するため，ハンド上部にはCCDカメラを搭載するためのスペースを作ることにした．

ロボットハンドの製作

以上で述べた設計にしたがって，ハンドの製作を行った．

ハンドの部材にはアルミを使用し，サイズは全長約18cm，幅14cm，指の長さ 10cm とした．また指の内側は，ノブの形状にあわせてカーブさせた．指の開閉には，DCモータをウォームギアで減速して指をスライドさせる機構を採用した．この機構によって，指の最大開き幅は 7cm，最小開き幅は 4cm で，指のストローク幅は 3cm となり，ノブの把持及びリリースを行うのに十分なストローク幅を確保した．また，モータにはエンコーダを搭載し，指の開き幅を任意に設定できるようにした．このハンドの全体重量は約 4kg である．

CCDカメラ，及び照明用ライトを搭載したハンドの概観を図5.10に示す．

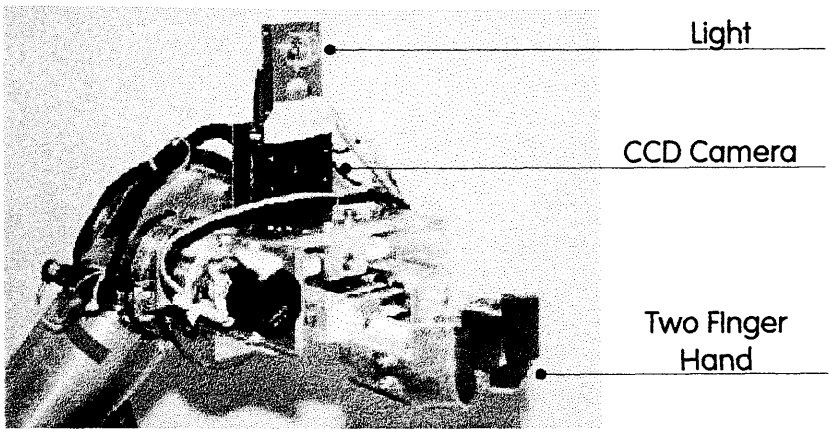

図 5.10: 2指ハンド及びCCDカメラ（写真）

製作したロボットハンドの評価

製作した小型マニピュレータの可搬重量は 設計段階で2kg なので，計算ではこの約4kg のハンドを支えるトルクは出せない筈である．しかし，マニピュレータの各モータのスペックを大きく見積もったおかげで，このハンドの重量でも，現在マニピュレータの動作に支障はないが，マニピュレータの負荷を少しでも軽減するため，ハンドの更なる軽量化を目指し，空気アクチュエータを用いた軽量ハンドの製作を検討中である．

5.3 モータドライバとモータ制御

ベースロボット，マニピュレータ，ハンドといったメカニズムを駆動するためのアクチュエータには，DCモータを使用した．これらのDCモータの制御を行うため，本研究では山彦プロジェクトで提案されたPWMスイッチングによるソフトウエアサーボ方式を採用した[IY91]．以下に，本研究で使用するモータドライバ，及びこのソフトウエアサーボの概略について述べる．

5.3.1 モータドライバ

DCモータの駆動には，トランジスタブリッジ回路を用いたモータドライバを使用した．この回路を図5.11に示す．

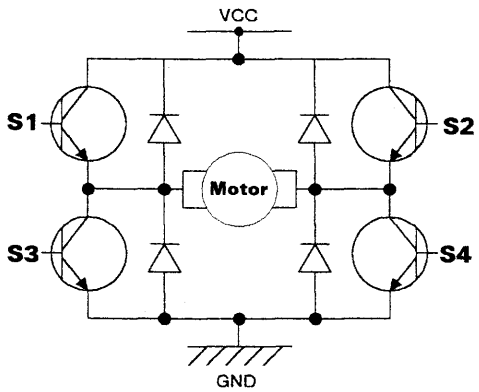

図 5.11: トランジスタブリッジ回路によるモータドライバの構成

この図において，トランジスタスイッチのS1及びS4がONになった場合，図のモータには左から右に向かう電流が流れる．逆にS2及びS3がONになった場合，モータには図の右から左に向かう電流が流れる．このように，この回路では，トランジスタのON-OFFの組み合わせで，モータに流れる電流の方向を決定することができる．また，モータが止まる際にモータの逆起電力によって発生する電流は，ダイオードを伝ってバッテリを充電する向きに流れる．

山彦プロジェクトでは，使用するモータの大きさに合せて，このトランジスタブリッジを用いた数種類のモータドライバの設計・製作が行われた．ベースロボットの駆動輪用のモータ，及びマニピュレータの第1関節と第2関節を駆動するモータは，大電流を必要とす

るため，FET4つでトランジスタブリッジを構成したタイプのモータドライバを使用した（図5.12右側）．このモータドライバは，電流を最大10Aまで流すことができる．また，その他のモータをドライブするためのモータドライバには，トランジスタブリッジが内包されたパッケージIC「L6203」を用いたモータドライバを使用した（図5.12左側）．このモータドライバは，電流を最大5Aまで流すことができる．

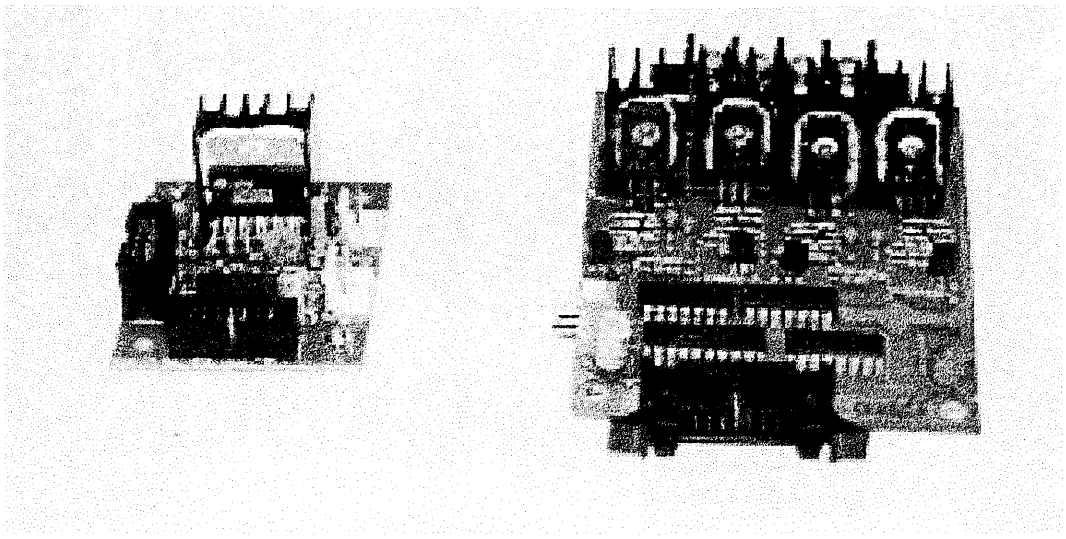

図 5.12: モータドライバ（写真）

5.3.2 ソフトウエアサーボ

前節で示したモータドライバを用いたDCモータの制御には，PWMスイッチングを用いたソフトウエアサーボ方式を採用した．この方式は，飯田らによって提案されたもので[IY91]，山彦プロジェクトで開発されたロボットの，DCモータの制御に広く用いられている．以下に，このソフトウエアサーボの概略について説明する．

モータのPWMスイッチング方式では，モータに流す電流を直接制御するのではなく，トランジスタスイッチのON-OFFのパルス幅を変えることでモータを制御する．このスイッチングの1周期に対するトランジスタのONの時間の割合はPWM比と呼ばれ，この比を変化させることでモータに流れる電流を変化させることが本方式の特徴である．

さて，DCモータを流れる電流は，駆動回路の電源電圧及びPWM波の基本周期を予め定めておけば，モータの回転数とPWM比によって一意に決まる．ただし，モータの回転

数を一定にした場合でも，目標電流に対するPWM比は非線型となる．そこで本方式では，予め各モータ毎にテーブルを作成して目標電流とモータの回転数に対するPWM比を決定する．このテーブルを使用することで，回路中にアナログ回路を使用せずにモータに流れる電流の制御を行うことができ，DCモータの力制御が可能となる．また，ハードウエアとしてはトランジスタブリッジのみですむため，アナログ部分での電力消費が少ないという利点も持つ．このソフトウエアサーボの構成を図5.13に示す．

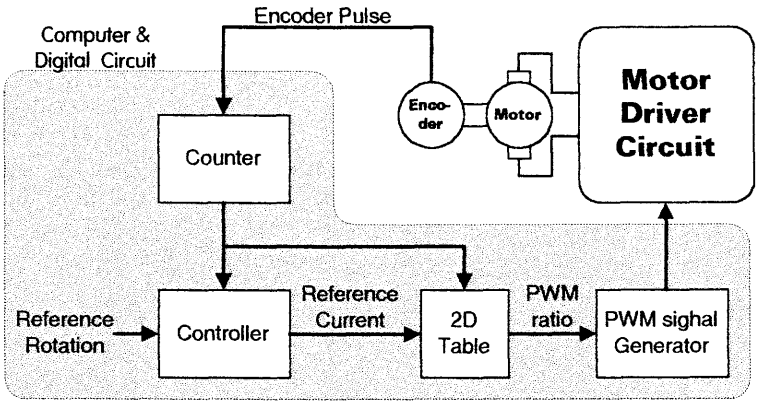

図 5.13: DCモータのソフトウエアサーボ

5.4 センサ

4.3.2節でも述べた通り，ドアの通り抜けを含む自律ナビゲーションを実現するための外界センサとして，ベースロボットの位置推定機能を実現するための**超音波距離センサ**，ノブの位置認識を行うための**視覚センサ**，マニピュレータのコンプライアンス制御を実現するための**力センサ**をロボットに搭載することにした．以下に，これらのセンサの仕様と性質に関する詳細を述べる．

5.4.1 超音波距離センサ「HiSonic」

ベースロボットに搭載する超音波距離センサには「HiSonic」を使用することにした．

この「HiSonic」は，山彦プロジェクトにおいて研究・開発が行われた超音波距離センサで，超音波の反射を用いて環境中の物体までの距離を測定するものである[OOY95]．このセンサは，現在山彦プロジェクトで開発された全ての山彦ロボットに標準的に搭載されて

いる．このセンサの特徴は，移動ロボットが走行中に障害物を発見することを念頭に開発されたため，センサの指向性が市販の超音波距離センサより広いという点である．

HiSonic は，電圧昇圧回路，及び超音波送受信回路から構成される．この HiSonic 及びコントロールボードの概観を図5.14に示す．なお，このコントロールボードについては，5.5.2節の項目で説明する．

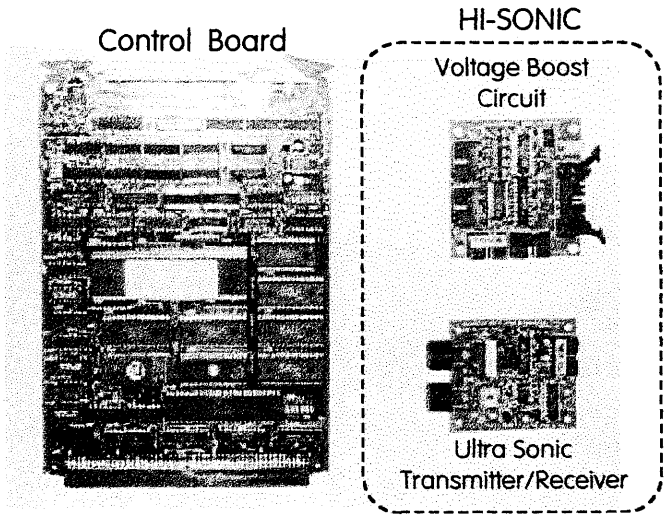

図 5.14: 超音波距離センサ「HiSonic」及びコントロールボード（写真）

HiSonic の電圧昇圧回路では，短時間で圧電素子を十分に充分に振動させるために，ダイオードブリッジ回路を用いて $24V$ の電圧が約 $120V$ まで昇圧される．一方，超音波送受信回路では，トランスを用いてこの電圧が $600V$ まで昇圧され，この電圧をトランスデューサの圧電素子にかけることで，障害物の検出には十分な超音波が送信される．この超音波が対象物で反射し，この反射波がレシーバに受かるまでの時間を計測することで，対象物までの距離が測定できる．超音波の周波数は $40kHz$，センサの指向性は約 $30deg$ であり，最大測距距離は約 $5m$ である．

ここでは，この HiSonic の超音波送受信回路を，ベースロボットの周囲8箇所に搭載した．この配置は，ベースロボットが走行中，経路上の予定外の障害物を検知するためにロボットの前方に向けたものが4箇所，ベースロボットがカーブを走行する際の経路上の障害物を検知するために斜め前方に向けたものが2箇所，また環境中のランドマークである平らな壁までの距離を検知するため，ベースロボットの左右に向けたものが2箇所とした．ベースロボットに搭載した HiSonic の概観（右前方）を図5.15に示す．

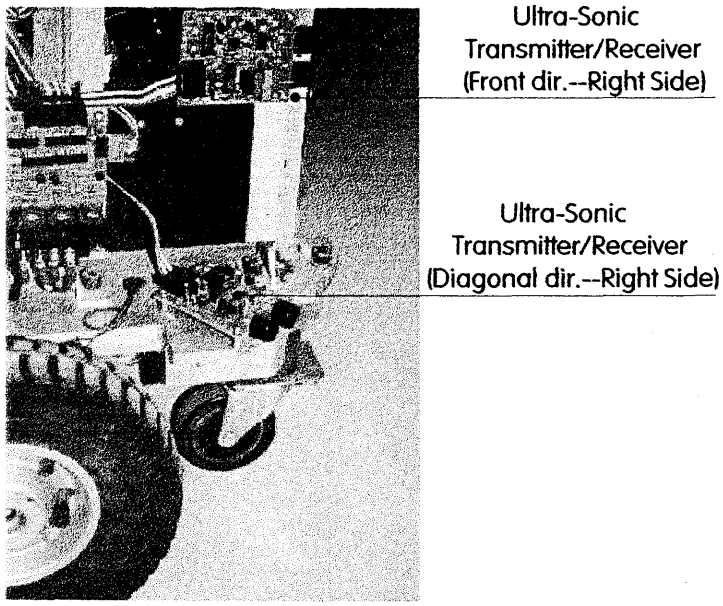

図 5.15: HiSonic をベースロボットの右前方に搭載したところ（写真）

5.4.2 視覚センサ

ノブの位置を認識するための視覚センサはハンド上部に搭載した．ハンドに搭載されたCCDカメラ及びライトの写真を図5.10に示し，以下にこの視覚センサのスペックについて述べる．

この視覚センサシステムは，ハンドに搭載したCCDカメラ，得られた画像を取り込み，処理を行う画像処理ボード，対象物に光を当てるためのライトによって構成される．CCDカメラには，WATEC社の「WAT-906」という小型オンボードカメラを使用した．このカメラは，視野角が約60度あり，得られた画像情報をNTSC信号で出力する．この画像情報は，画像処理ボード上でA/D変換された後，このボード上のフレームメモリに格納される．このフレームメモリの記憶容量は，256×240画素の画像が2枚分あり，各画素は256階調のグレイスケールである．また，多少暗い環境でも画像の輝度値の差を出すために，CCDカメラの上部にライトを搭載した．このライトは，通常車のウインカーに使用される$12V5W$のものである．

5.4.3 力センサ

マニピュレータのコンプライアンス制御を行うために，力センサをマニピュレータの第6関節と第7関節の間に搭載した．この力センサはニッタ製の6軸力覚センサ「UFS-3012-25」である．

この力センサシステムは，力センサ本体と専用のA/Dコンバータユニット，及び力情報を処理するコントローラによって構成される．力センサ及びA/Dコンバータユニットを図5.16に示し，センサのスペックを表5.4示す．

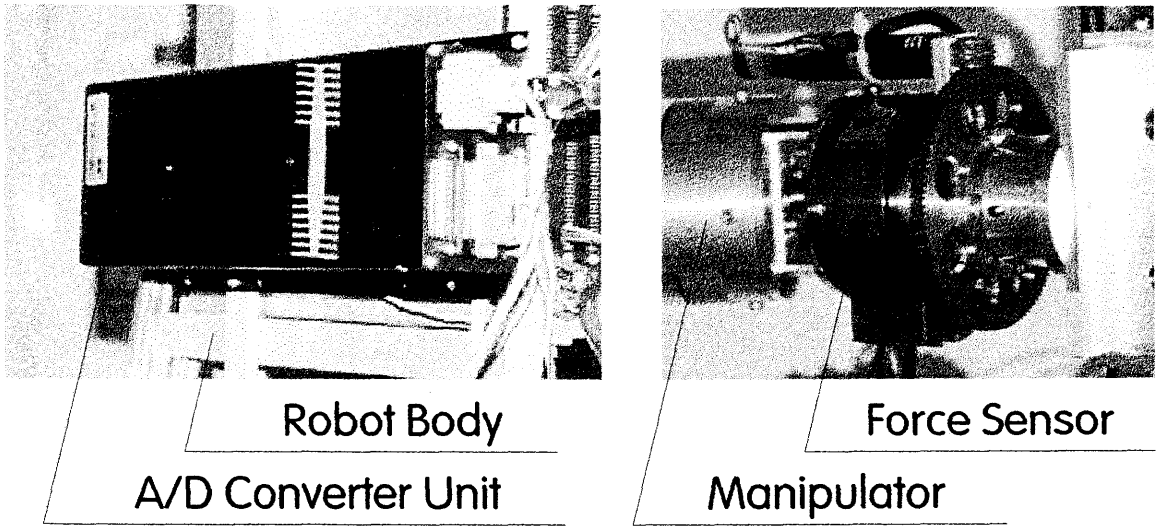

図 5.16: 力センサ及びA/Dコンバータユニット（写真）

サイズ:	外形 $75mm$, 幅 $31.8mm$
重量:	$310g$
定格荷重:	x,y軸方向 $11kgf$
	z軸方向 $22kgf$
	各軸回りの
	モーメント $85kgf \cdot cm$

表 5.4: 力覚センサ「UFS-3012-25」のスペック

このセンサによって得られた力データは，専用ケーブルを通じて専用のA/Dコンバータユニットに送られて処理が行われる．処理された力情報は，この専用ユニットからシリアル

通信によってコントロールボードに転送される．コントロールボードに関する詳細は5.5.2節で説明する．

力情報のデータフォーマットは，初めの3バイトがシンクロ用の固定データ，次の1バイトがマスク用バイト，次の16バイトがデータバイト，最後の1バイトがリターンコードである[ニッ95]．データバイトの要素は，各軸方向の力 (F_x, F_y, F_z)，モーメント (M_x, M_y, M_z) 及び全体ベクトル (F_v, M_v) であり，各データは2バイトであり2の補数で表される．ただし，全体ベクトルについてはこのセンサでは使用しない．これより，1サイクルに要する力情報はヘッダを含めて21バイトとなるので，このシリアル通信を9600bps で行った場合，1サイクルの情報を送信するのに要する時間は 17.5ms となる．

5.5 コントローラ

アクチュエータ及び各種センサを制御するためのコントローラには，基本的には山彦プロジェクトで提案された，自律移動ロボットのための機能分散コントローラを採用することにした[KY85]．本節では，まずこの機能分散コントローラの概念を紹介する．

本研究では，この機能分散の概念に沿って，ロボットシステムを構築した．この機能分散コントローラでは，各機能を実現するための単位を機能モジュールと呼ぶが，タスクを実現するために必要な機能のうち，既に山彦プロジェクトで研究開発が行われた機能モジュールについては既存のものを使用した．また，新たに搭載する機能については，機能モジュールの設計及び製作を行った．

5.5.1 機能分散コントローラの概念

機能分散コントローラは，センサやアクチュエータなどの各機能毎にコントローラを分散し，ロボットの動作を管理するものである．分散されたロボットの機能を実現する単位は機能モジュールと呼ばれ，基本的に1つの機能に対して1つの機能モジュールが割り当てられる．各機能モジュールは，各機能を実現するためのメカニズムやセンサ，CPU，メモリ，リアルタイムOS，及び専用ソフトウエアで構成される．このため，各機能モジュール毎に独立に開発，改良，デバッグ，メンテナンスを行うことが可能である．これらの機能モジュールは，マスタモジュールと呼ばれる統括制御コントローラによって一括管理される．したがって，ロボットの動作を決定するソフトウエアは，このマスタモジュール上に集約される．マスタモジュールと各機能モジュール間の通信には，バスを用いた非同期

通信が使用される．この機能モジュールで構成される機能分散コントローラの概念図を図5.17に示す．

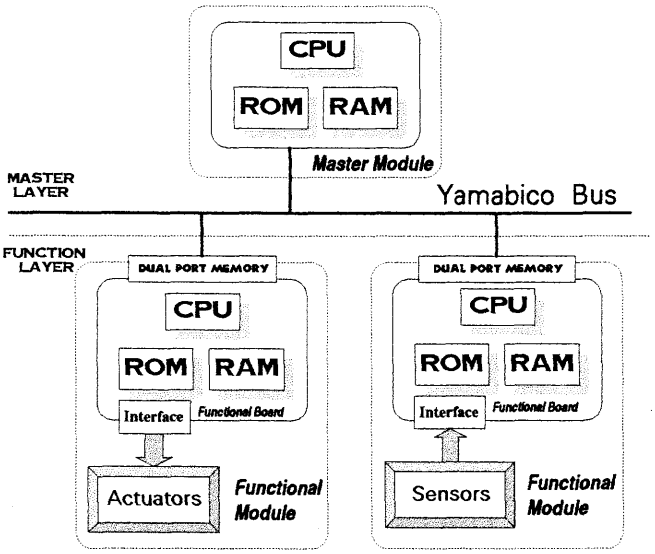

図 5.17: 機能分散コントローラの概念図

この機能分散コントローラの長所は，ロボットに必要な機能を機能モジュール単位で逐次追加することができるという点である．そこで山彦プロジェクトでは，全ての山彦ロボットにこの機能分散コントローラが実装され，自律ロボットに関する数多くの実験が行われてきた．その結果，特に各機能の開発，デバッグ，メンテナンスに関してこの機能分散コントローラの有効性が確認されてきた．

5.5.2 搭載するモジュール

機能分散コントローラの概念に沿って，ロボットシステムのコントローラを構築した．

目標タスクを実現するためにロボットに搭載する機能は，表4.1に示したものであるので，本研究では表5.5に示すモジュールをロボットに搭載することにした．ただし，ハンドの指の開閉機能はモータ制御に関するものなので，マニピュレータ制御モジュールに割り当てた．また，行動プログラムのデバッグを行うのに便利な音声合成モジュールも，搭載する機能モジュールに加えた．

ここで示したモジュールのうち，走行制御モジュール，位置推定モジュール，超音波距離センサモジュール，視覚センサモジュールについては，山彦プロジェクトで研究・開発

が行われた既存のモジュールを使用することにした．また，マニピュレータ制御モジュール及び力センサモジュールについては，本研究で新たに設計・製作を行った．以下にこれらのモジュールに関する詳細を述べる．

モジュール		担当する機能
マスタモジュール	(a)	走行経路及び動作の計画機能
走行制御モジュール	(b)	ベースロボットの走行機能
超音波距離センサモジュール	(c)	超音波距離センサによるベースロボットの位置推定機能
	(j)	走行系路上の障害物検知機能
位置推定モジュール	(c)	超音波距離センサによるベースロボットの位置推定機能
	(d)	オドメトリによるベースロボットの位置推定機能
	(e)	ノブを把持したマニピュレータの姿勢によるベースロボットの位置推定機能
マニピュレータ制御モジュール	(e)	ノブを把持したマニピュレータの姿勢からベースロボットの位置を推定する機能
	(f)	ハンドの指の開閉機能
	(h)	マニピュレータの姿勢制御機能
	(i)	マニピュレータのコンプライアンス制御
視覚センサモジュール	(g)	ノブの位置認識機能
力センサモジュール	(i)	マニピュレータのコンプライアンス制御
音声合成モジュール	(j)	デバッグ情報を発声

表 5.5: 搭載する機能に対応する機能モジュールとマスタモジュール

マニピュレータ制御モジュール

1. 構成

マニピュレータ制御モジュールは，7関節小型マニピュレータ，ハンド，8つのモータドライバ，及びこれらを制御するコンピュータボードで構成した．

このうち，マニピュレータ及びハンドについては，5.2.2節，5.2.3 節で示したように設計・製作を行った．またモータドライバに関しては，5.3節で示したように山彦プロジェクトで開発された既存のものを使用した．また，コンピュータボードは，山彦

プロジェクトで製作された2チャンネルのDCモータ制御を行う走行制御モジュール用コンピュータボード“T-LOCO”（図5.18右側）に，6チャンネルのDCモータを制御するための拡張ボード（図5.18左側）を搭載することで実現した．この拡張ボードはラッピングで製作した．

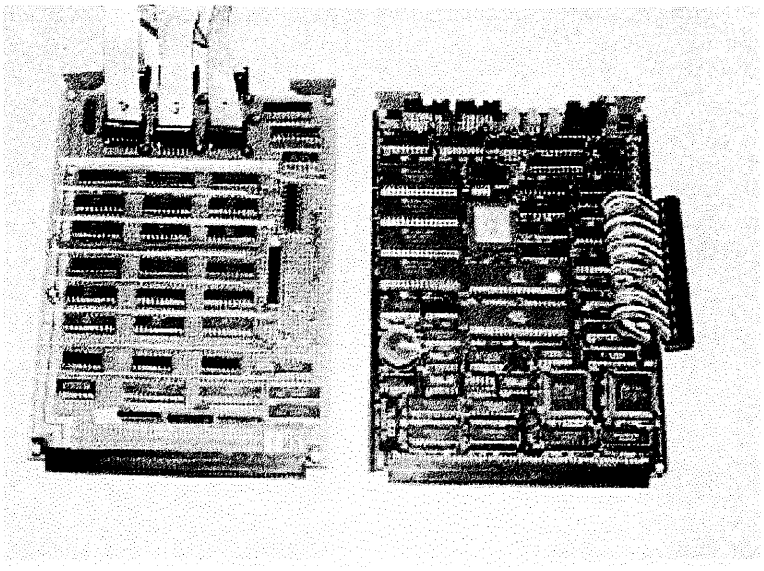

図 5.18: マニピュレータ制御用コンピュータボード（写真）

2. コンピュータシステム

本モジュールのコンピュータシステムには，CPUとしてトランスピュータT805を使用した．またこのシステムには，外部メモリとして512 *kbyte* のRAMと256 *kbyte* のROM，マスタモジュールと通信用を行うための *2kbyte* の双方向メモリ（Dual Port Memory），8チャンネルのDCモータを制御するためのPWMジェネレータM66240，ロータリエンコーダの信号を受けるアップダウンカウンタを搭載した．

3. 他のモジュールとのインタフェース

本モジュールは，コンピュータボード上の双方向メモリを用いて，専用バスを通じてマスタモジュールとの通信を行うことにした．また，コンプライアンス制御を行うため，トランスピュータリンクというCPU直結の高速シリアル通信を用いて，力センサモジュールから直接力情報を受け取ることにした．

力センサモジュール

1. 構成

   力センサモジュールは，6軸力覚センサ，専用のA/Dコンバータユニット，及びこれらを制御するコンピュータボードで構成した．

   このうち，6軸力覚センサ及び専用のA/Dコンバータユニットについては，5.4.3節で既に説明した．またコンピュータボードについては，既存のマスタモジュール用コンピュータボード（T-Master）を改造して実現した．

2. コンピュータシステム

   本モジュールのコンピュータシステムには，CPUとしてトランスピュータT805を使用し，外部メモリとして512*kbyte* のRAMと256 *kbyte* のROMを搭載した．また，専用のA/Dコンバータユニットからの力情報を獲得するため，通信用シリアルIC D71051を搭載した．

3. 他のモジュールとのインタフェース

   本モジュールは，マニピュレータのコンプライアンス制御を行うため，トランスピュータリンクを通じてマニピュレータ制御モジュールに力情報を送信することにした．

既存のモジュール

表5.5に示したロボットに搭載するモジュール中，山彦プロジェクトで既に研究開発が行われ，現在山彦ロボット上で稼動中の既存のモジュールは，走行制御モジュール，位置推定モジュール，超音波距離センサモジュール，視覚センサモジュール，音声合成モジュール，及びマスタモジュールである．以下に各モジュールのスペック，及び動作の概略について述べる．

1. 走行制御モジュール

   走行制御モジュールは，2機の走行用DCモータ，これをドライブする2つのモータドライバ，モータドライバに信号を送るコンピュータボードで構成される．

   コンピュータボードのCPUは，トランスピュータT805である．またこのボードは，外部メモリとして512 *kbyte* のRAMと256 *kbyte* のROM，DCモータを制御する

ための PWM ジェネレータ M66240，マスタモジュールとの通信を行うための 2*kbyte* の双方向メモリ（Dual Port Memory）を有する．

このモジュールは，双方向メモリを通じてマスタモジュールから送られる走行コマンドを解析し，走行用モータを制御することで，ベースロボットを走行させる．ただしこの走行制御には，トランスピュータリンクを通じて位置推定モジュールから随時送られるベースロボットの推定位置情報を使用する．

2. 位置推定モジュール

位置推定モジュールは，走行用 DC モータに搭載されたロータリエンコーダ，及びこのデータを受け取るコンピュータボードで構成される．

コンピュータボードの CPU は，トランスピュータ T805 である．またこのボードは，外部メモリとして 512*kbyte* の RAM と 256*kbyte* の ROM，ロータリエンコーダからの信号を受ける 2 チャンネルのカウンタ，マスタモジュールとの通信を行うための 2*kbyte* の双方向メモリを有する．

このモジュールは，ロータリエンコーダを用いて駆動輪の回転数を計数するオドメトリによって，常にベースロボットの位置を推定する．また，外界センサを用いて推定したベースロボットの位置情報を，双方向メモリを通じてマスタモジュールから非同期に獲得し，この推定位置情報とオドメトリで推定したベースロボットの位置情報を最尤推定を用いて融合することで，ベースロボットの推定位置の修正を行う．このベースロボットの推定位置情報は，トランスピュータリンクを通じて直接走行制御モジュールに送られ，走行制御に利用される．またマスタモジュールも，双方向メモリを通じて，この推定位置情報を参照することができる．

3. 視覚センサモジュール

視覚センサモジュールは，WATEC 社製小型 CCD カメラ，カメラの上部に搭載したライト，及びこれを処理するコンピュータボードで構成される．

コンピュータボードの CPU は，トランスピュータ T805 である．またこのボードは，外部メモリとして 512 *kbyte* の RAM と 256 *kbyte* の ROM，画像を取り込むための A/D コンバータ，及び 1 フレームが 60*kbyte* の画像 2 枚分のフレームメモリ，マスタモジュールとの通信を行うための双方向メモリを有する．

視覚センサの動作については，既に5.4.2節で説明した．マスタモジュールは，双方向メモリを通じて，この画像処理結果を参照することができる．

一方，多少暗い環境でも対象物の輝度値の差を出すため，画像処理を行う際に，CCDカメラ上部に搭載したライトを点灯するが，このライトのON-OFF制御は後述するマスタモジュールのパラレルポートを通じて行うことにした．

4. 超音波距離センサモジュール

超音波距離センサモジュールは，HiSonic（電圧昇圧回路と超音波送受信回路），及びこの情報を処理するコンピュータボードで構成される．

コンピュータボードのCPUは68000（MOTOROLA）である．またこのボードは，外部メモリとして64 *kbyte* のRAMと64 *kbyte* のROM，超音波の反射波が返ってくるまでの時間を測定するため，3つのタイマ用IC HD6340，マスタモジュールとの通信を行うための双方向メモリを有する．

超音波距離センサの動作については，既に5.4.1節で説明した．この距離計測は60ms毎に行われ，その都度計測データは更新されるため，マスタモジュールは双方向メモリを通じて常に最新の距離データを参照することができる．

5. 音声合成モジュール

音声合成モジュールは，音声を発するスピーカ，及び音声をジェネレートするコンピュータボードで構成される．

コンピュータボードのCPUは，TMP68301（TOSHIBA）である．またこのボードは，外部メモリとして256 *kbyte* のRAMと384 *kbyte* のROM，音素片を用いて音声合成を行うための専用ICであるSC-02，マスタモジュールとの通信を行うための双方向メモリを有する．

この音声合成モジュールは，RAMまたはROM上にPCM録音された音声情報を再生する方式と，音素片を合成し音声をジェネレートする方式で音声を発声することができる．再生する音声情報の指定や合成する音素片の指定は，双方向メモリを通じてマスタモジュールから行われる．

6. マスタモジュール

マスタモジュールは，各機能モジュールを統括制御するコンピュータボード（mm-KEI）で構成される．このボードは，山彦ロボットの統括制御用のボードとして，1992年に筆者らによって設計・製作が行われた．このボードの概観を図5.19に示す．

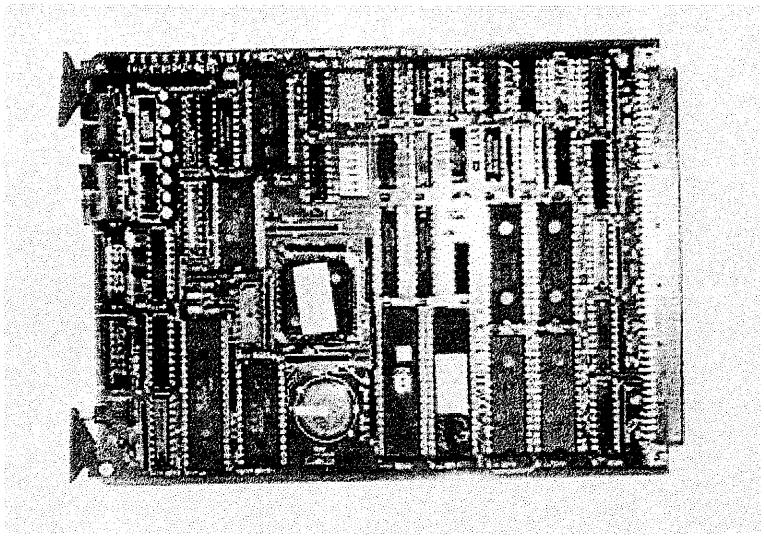

図 5.19: マスタモジュール用コンピュータボード（写真）

コンピュータボードの CPU は， 68000（MOTOROLA）である．またこのボードは，外部メモリとして 512*kbyte* の RAM と 256*kbyte* の ROM，汎用のタイマ用IC HD6340，16 ビットパラレルポートとパラレル入出力用IC HD6321，2チャンネルのシリアルポートとシリアル通信用IC HD6350 を有する．

本モジュールは，シリアルポートを通じてダウンロードされたロボットの行動プログラムにしたがって，各機能モジュールと通信することで，ロボットの自律動作を実現する．この通信は，コンピュータボードが専用バスを通じて他の機能モジュールの双方向メモリに直接アクセスすることで実現される．

5.5.3 機能分散コントローラの実装

各機能モジュールをマスタモジュールが統合することで，機能分散コントローラを実装した．本ロボットシステムの機能分散コントローラの構成を図5.20に示す．

マスタモジュールと各機能モジュール間の通信には，山彦プロジェクトで開発された「山彦バス」を使用した．このバスは，16bit のアドレスバス，16bitのデータバス，コントロー

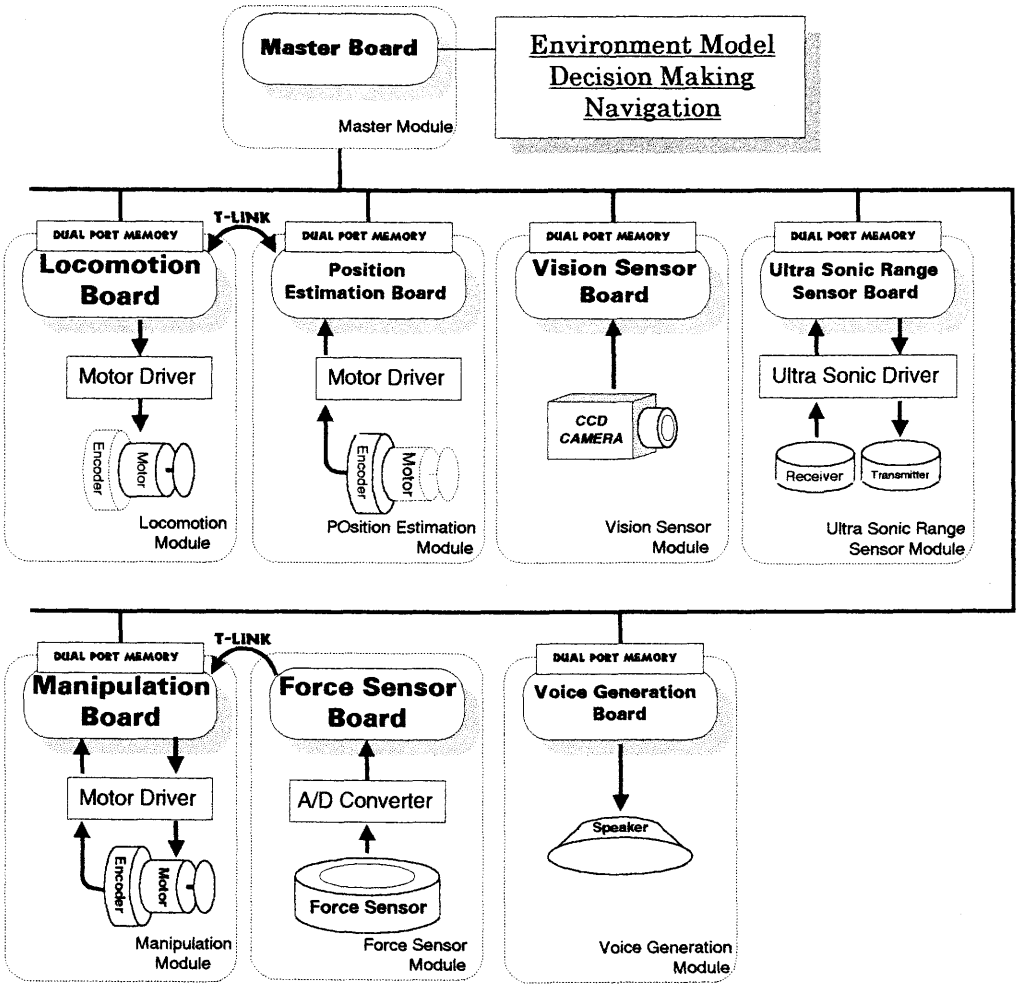

図 5.20: 山彦てんの機能分散コントローラの構成

ルバス，及び電源ラインから構成される．各モジュールのコンピュータボードがラックに挿入され，100 ピンのコネクタによってバス接続されている様子を図5.21に示す．

5.6 まとめ

本章では，構築したロボットシステムのハードウエアの構成要素であるメカニズム，モータ制御，センサ，コントローラの詳細について述べた．

ベースロボットには，基本的に山彦プロジェクトで研究開発が行われた「自律型移動ロボット山彦てん」の筐体，センサ，コントローラなどを使用した．このベースロボットに，

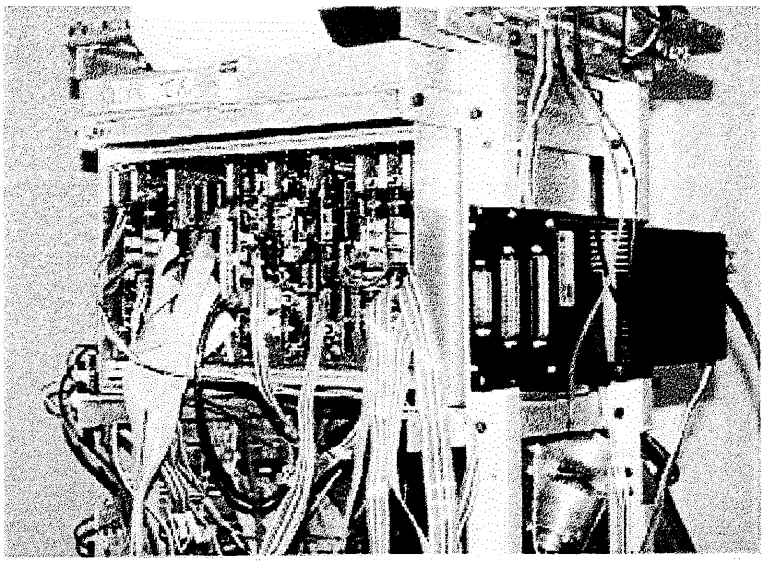

図 5.21: ボードラックとコンピュータボード（写真）

新たに製作した小型マニピュレータ，ハンド，マニピュレータ制御用コンピュータボード，市販の力センサ及び視覚センサを搭載することで，ロボットシステムのハードウエアを構築した．このように，従来のロボットシステムに新しい機能を逐次追加できるという点が，5.5節で紹介した機能分散コントローラの特徴である．

次章より，目標タスク「ドアの通り抜けを含む自律ナビゲーション」を実現するために，このロボットシステムに搭載したソフトウエアについて詳しく述べる．

第 6章

ソフトウエアの構成

6.1 はじめに

本章より第9章までは，ロボットシステムに実装したソフトウエアの詳細について述べる．

本ロボットシステムでは，機能分散コントローラの概念にしたがい，ロボットを制御するソフトウエアを，機能モジュール上のアクチュエータやセンサなど各機能を制御するソフトウエアと，マスタモジュール上のロボットの動作を決定するソフトウエアの2階層で構成した（図6.1参照）．本章では，ロボットに搭載したこれらのソフトウエアの構成について説明する．

6.2 機能モジュール上のソフトウエア

まず，各機能モジュール上に実装したソフトウエアの，一般的な動作について述べる．

各機能モジュール上で動作するソフトウエアは，マスタモジュール上の統括制御ソフトウエアとは独立に常に動作する．例えば，走行制御モジュール上のソフトウエアは，マスタモジュール上のソフトウエアから送られる走行コマンドにしたがい，走行制御モジュール上で常にベースロボットの走行制御を行う．また，超音波距離センサモジュール上のソフトウエアは，超音波距離センサを用いてロボットの周囲の物体までの距離情報を常時測定する．これらは，人間が別のことを考えながら，バランスを取って二足歩行を行ったり，五感を働かせることができることに相当する．ただし，視覚センサモジュール上のソフトウエアについては，画像処理に時間がかかるため，マスタモジュールからのリクエストがあった場合のみセンシング動作を行う．また，音声合成ソフトウエアも，発声コマンドがマスタモジュールから送られた場合のみ動作を行う．したがって，これらのソフトウエア

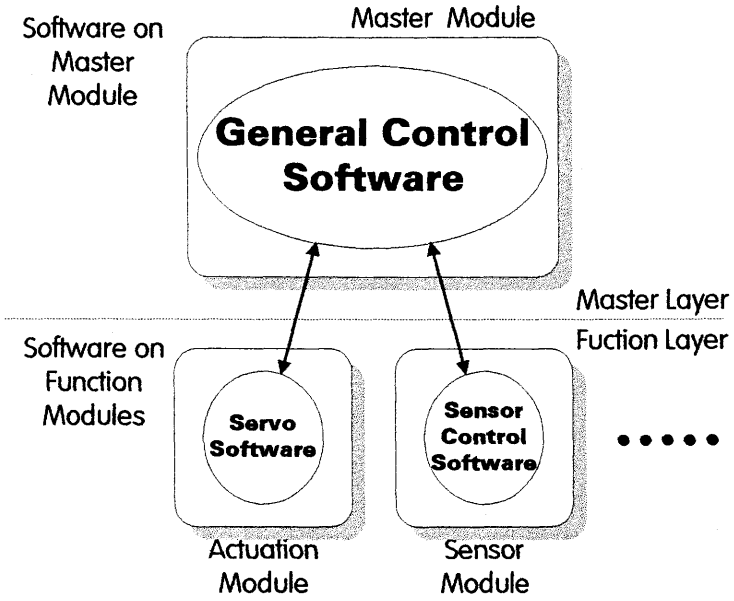

図 6.1: ロボットのソフトウエア構成

は，通常マスタモジュール上のソフトウエアからの通信待ち状態で待機する．

機能モジュール上のソフトウエアとマスタモジュール上のソフトウエアとの通信は，各機能モジュール上に搭載された双方向メモリの指定アドレスに，必要な情報を非同期に読み書きすることで実現される．マスタモジュールによる読み書きは，山彦バス（5.3.3節参照）を通じて行われる．ただし，力センサモジュールとマニピュレータ制御モジュール，及び位置推定モジュールと走行制御モジュール上の各ソフトウエア間の通信は，双方向メモリではなく，トランスピュータリンクを通じて直接行われる．

各機能モジュール上のソフトウエアの詳細については，第7章で述べる．

6.3 マスタモジュール上のソフトウエア

マスタモジュール上に実装したソフトウエアは，各機能モジュール上のソフトウエアとバスを通じて通信を行うことで，各機能を統括制御し，ロボットの動作を決定するものである．

目標タスクを実現するためのロボットの動作が複雑なため，まずこの動作を表3.1に示した通り，時間の経過に沿って動作のまとまり毎に分割し，各部分動作を実現するソフトウ

エアを実装した．本研究では，この各部分動作を実現するソフトウエアの単位を**アクションプリミティブ**と名付けた．これより，目標タスクを実現するためのアクションプリミティブは，経路追従走行，ノブの把持動作，ノブのリリース動作，ドアの開閉動作の4つとなる．これらの各アクションプリミティブの詳細については，第8章で詳しく説明する．

ロボットの動作は，これらのアクションプリミティブのシーケンスで表現できるので，このシーケンスを作成し順次実行する統括制御ソフトウエアもマスタモジュール上に実装した．この統括制御ソフトウエアについては，第9章で詳しく説明する．このように，マスタモジュール上のソフトウエアは，アクションプリミティブと統括制御ソフトウエアの2階層で構成される．

6.4 まとめ

本研究では，機能分散コントローラにしたがって，ロボットに搭載するソフトウエアを機能モジュール上のソフトウエアとマスタモジュール上のソフトウエアの2階層で実現した．更にマスタモジュール上のソフトウエアについては，部分動作を実現するアクションプリミティブというソフトウエア群と，これらを統括する統括制御ソフトウエアの2階層で構成した．これにより，ロボット全体の動作は，統括制御ソフトウエアが作成したアクションプリミティブのシーケンスを順次実行することで実現される．本章では，このソフトウエア構成の全体の枠組みについて説明した．

以後，第7章では各機能モジュール上の個々のソフトウエアの詳細について述べる．第8章では，部分動作を実現する個々のアクションプリミティブの詳細ついて述べる．第9章では統括制御ソフトウエアに関する詳細について述べる．

第 7 章

機能モジュール上のソフトウエアの実現

7.1 はじめに

各機能モジュールには，表7.1に示すソフトウエアを搭載することにした．このうち，(1)，(2)，(4)，(7) のソフトウエアについては，山彦プロジェクトで開発された既存のソフトウエアを使用することにした．また，残りの (3) マニピュレータ制御ソフトウエア，(5) ノブ位置認識ソフトウエア，(6) 力センサソフトウエアについては，本研究で設計及び実装を行った．

	ソフトウエアの名称	ソフトウエアを搭載する機能モジュール
(1)	走行制御ソフトウエア (Spur)	走行制御モジュール
(2)	位置推定ソフトウエア (POEM)	位置推定モジュール
(3)	マニピュレータ制御ソフトウエア	マニピュレータ制御モジュール
(4)	測距ソフトウエア	超音波距離センサモジュール
(5)	ノブ位置認識ソフトウエア	視覚センサモジュール
(6)	力センサソフトウエア	力センサモジュール
(7)	音声合成ソフトウエア	音声合成モジュール

表 7.1: 各機能モジュール上のソフトウエア

本章では，これらの機能モジュール上のソフトウエアについて，新たに実装したものについては設計と実装について説明し，また既存のものについては動作説明を行う．

7.2 マニピュレータ制御ソフトウエア

表5.5に示す通り，マニピュレータ制御モジュールが担当する機能は，マニピュレータの姿勢制御機能，マニピュレータのコンプライアンス制御機能，ノブを把持したマニピュレータの姿勢によるベースロボットの位置推定機能，及びハンドの指の開閉機能である．そこで，これらの機能を実現するマニピュレータ制御ソフトウエアをマニピュレータ制御モジュール上に実装した．ただし，「ノブを把持したマニピュレータの姿勢からベースロボットの位置を推定する機能」については，このソフトウエアで検知したマニピュレータの姿勢情報を受けて，マスタモジュール上のソフトウエアで実現することにした．これらの機能は，マニピュレータ及びハンドのアクチュエータである8つのDCモータを制御することで実現されるため，このソフトウエアのベースは，これらのDCモータの制御である．本節では，このマニピュレータ制御ソフトウエアの設計及び実装について説明する．

7.2.1 マニピュレータ制御ソフトウエアの設計

マニピュレータの姿勢制御

マニピュレータの姿勢制御は，マニピュレータの関節角度がマスタモジュール上のソフトウエアから送られる目標角度となるように，DCモータを制御することで実現した．ただしこのモータ制御には，基本的にはPID制御器を使用した．

マニピュレータのコンプライアンス制御

マニピュレータのコンプライアンス制御は，マニピュレータの先端（以後手先と呼ぶ）の目標位置を，手先にかかる力の方向に力に比例した量だけ移動させることで実現した．ただし，マニピュレータの関節は7つあるため，ここでは第7関節を固定し，6関節マニピュレータとすることで冗長に関する問題を回避した．以下に，コンプライアンス制御の具体的な制御方法について述べる．

手先に加わる力によって手先を移動させる場合，各関節の目標角度からの変化量$d\theta$と，手先にかかる力$[f_x\ f_y\ f_z\ 0\ 0\ 0]^T$との間には，ヤコビ行列とコンプライアンス行列を用いて(7.1) 式に示す関係が成り立つ[P.P81].

$$\begin{bmatrix} f_x \\ f_y \\ f_z \\ 0 \\ 0 \\ 0 \end{bmatrix} = [\mathrm{C}]^{-1} \begin{bmatrix} j_{11} & j_{12} & \cdots & j_{16} \\ j_{21} & j_{22} & \cdots & j_{26} \\ \cdots & & & \\ j_{61} & & & j_{66} \end{bmatrix} \begin{bmatrix} d\theta_1 \\ d\theta_2 \\ d\theta_3 \\ d\theta_4 \\ d\theta_5 \\ d\theta_6 \end{bmatrix} \tag{7.1}$$

ただし，j_{mn}はヤコビ行列のm行n列の要素であり，f_x，f_y，f_zは，手先の座標系における手先に加わる力を示す．また，コンプライアンス行列Cは対角行列であり，各対角要素は手先の各座標軸に関して，単位あたりの力に対する手先の目標位置を移動させる量を表す定数である．この値が大きいと，小さな力でも手先が大きく移動するため，手先はより柔らかく制御される．

次に，ドアを開閉する際のマニピュレータの制御について考察する．ドアの開閉動作時には，ドアの重みやドアに取り付けられたバネによって，ドアを操作する方向とは逆向きの力が手先にかかる．このため，ノブの軸方向にコンプライアンス制御を行った場合，手先の目標位置は，ドアを操作する方向とは逆向きに移動し，ドアを押し開ける（または引き開ける）力がドアには十分に伝わらない．したがって，ドアの開閉動作時には，図7.1に示すように，手先のノブの回転軸に垂直な平面上に拘束されたコンプライアンス制御を行うことにした．この場合のコンプライアンス行列Cは，

$$C = diag(c_1, c_2, 0, 0, 0, 0)$$

となる．但し c_1 及び c_2 は，単位あたりの力に対して手先の目標位置を移動させる量である．

一方，力センサを搭載する位置はハンドの根元なので，力センサにはハンドの重量がかかる．この力をキャンセルするため，手先にかかる力情報を，コンプライアンス制御を始めた瞬間の力データからのオフセットで表すことにした．ただしこの方法では，マニピュレータの姿勢が変化した場合に，力情報は正しくなくなる．この重力補償に関する問題については今後の課題である．

DCモータの制御器

マニピュレータの姿勢制御，及びマニピュレータのコンプライアンス制御を実現するため，DCモータの制御にはPID制御を使用した．ただし，マニピュレータが急激に動作しないよう各モータの回転スピードを押さえるため，微分項の入力は常に0とした．また，各

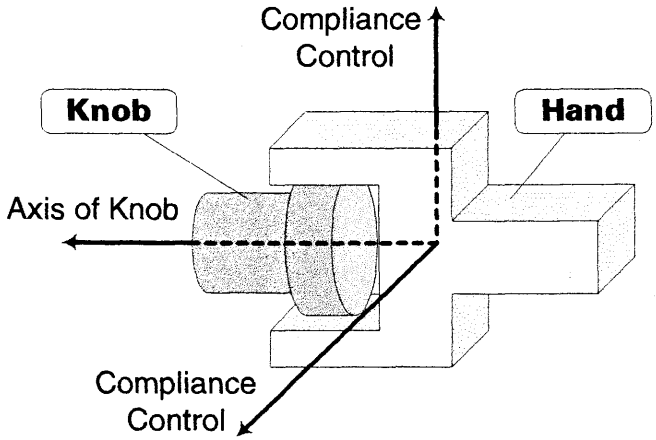

図 7.1: マニピュレータの先端のコンプライアンス制御

モータに入力される目標角は，マスタモジュールから送られる目標角に，手先にかかる力情報によって手先を移動させるための必要回転角を加えたものとした．このDCモータの制御器を図7.2に示す．

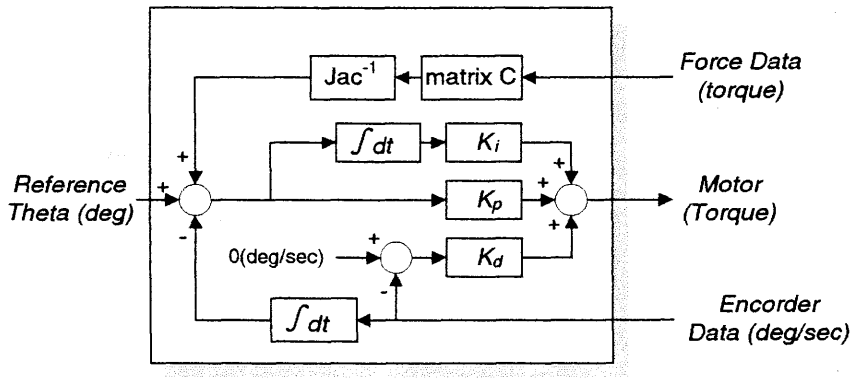

図 7.2: DCモータの制御器

ハンドの指の開閉制御

ハンドの指の開閉制御は，ハンドの指の開き幅がマスタモジュール上のソフトウエアから送られる目標幅となるように，DCモータを制御することで実現した．この制御には，図7.2の制御器から，力センサの入力を外したものを使用した．

入出力情報

以上より，マニピュレータ制御ソフトウエアを動作するため，マスタモジュールから送られる必要情報は，マニピュレータの各関節の目標角度，ハンドの指の開き幅，手先の柔らかさを決定するコンプライアンス行列，手先にかかる力情報である．また，マスタモジュール上のソフトウエアが参照するマニピュレータ制御ソフトウエアからの出力情報は，マニピュレータの各関節の現在角度，及び手先にかかる力情報とした．

7.2.2 マニピュレータ制御ソフトウエアの実装

マニピュレータ制御ソフトウエアは，DCモータを制御するサーボプロセス，力センサからの入力を監視する力センサ監視プロセス，ヤコビ行列を用いて各関節の必要変位角を計算する行列計算プロセス，マスタモジュールとの入出力を担当するインタフェースプロセスによって構成した．このソフトウエア構成を図7.3に示す．

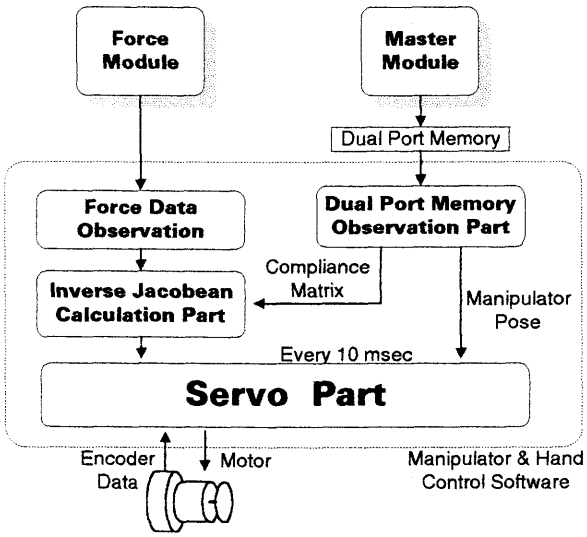

図 7.3: マニピュレータ制御ソフトウエアの構成

トランスピュータのプロセスのプライオリティ (優先度) には，ハイ (High) とロー (Low) の2種類のプライオリティがあるので[inm89]，サーボプロセスについては，プロセスのプライオリティをハイに設定することで，$10ms$のサーボループを確保した．その他の各プロセスは，ロープライオリティで実行することにした．このためサーボプロセス以外の各プロセスは，トランスピュータが持つハードウエアの機能によって自動的にタイムスライス

され，サーボループのバックグラウンド処理として並列に動作する．以下に実装した各プロセスの詳細について述べる．

サーボプロセス

サーボプロセスは，8つのDCモータの制御を行うプロセスである．

このプロセスは，各モータのロータリエンコーダから得たモータの回転角速度，及び力情報監視プロセスが受けた力情報より，設計したモータの制御器（図7.2）を用いて各モータに与えるトルクを計算する．次に，このトルクを出力するのに必要な目標電流からモータテーブルを用いて各々のモータに対するPWM比を求め，この比をPWMジェネレータに設定する（このソフトウエアサーボについては，5.3.2節及び[IY91]を参照）．これにより，モータドライバを通じて各々のモータに電流が流れる．ここで示した一連の動作は，マニピュレータ制御モジュール上で$10ms$毎にハイプライオリティで実行される．

力情報監視プロセス

力情報監視プロセスは，力センサモジュールからの力情報を受けるためのプロセスである．5.4.3節でも示した通り，力情報はトランスピュータリンクを通じて約$18ms$毎に力センサボードから送られてくる．本プロセスでは，逐次この情報を受け取り，力情報を内部変数に代入する．

行列計算プロセス

行列計算プロセスは，力情報によって移動させる手先の目標移動量を各関節の必要回転角に変換する計算を行うプロセスである．具体的には，ヤコビ行列の計算と，(6.1) 式の6元連立一次方程式の計算を行う．

本プロセスはロープライオリティで実行されるので，この計算はサーボループのバックグラウンド処理で行われる．このときの1回のヤコビ行列の計算及び方程式の計算にかかる時間は，実際にソフトウエアを動かしたところ約$20ms$であった．

インタフェースプロセス

インタフェースプロセスは，マスタモジュール上のソフトウエアとの通信を行うプロセスである．この通信は，マニピュレータ制御モジュールに搭載した双方向メモリを用いて行う．

本プロセスは，通常マスタモジュール上のソフトウエアからの入力待ちであり，入力の確認は$10ms$毎に特定の番地をポーリングすることで行われる．マスタモジュール上のソフトウエアからの入力は，前述した通り，マニピュレータの各関節の目標角度，コンプライアンス行列，搭載ハンドの指の開き幅である．各入力情報は，マスタモジュール上のソフトウエアから，バスを通じて双方向メモリ上の指定アドレスに書き込まれるので，本プロセスはこのアドレスを参照することで，制御パラメータなどを更新する．一方，マスタモジュール上のソフトウエアから，手先にかかる力情報やマニピュレータの各関節の現在角度などの問い合わせが来た場合，本プロセスはこれらソフトウエア内部の情報を，双方向メモリ上の指定のアドレスに書き込む．

マニピュレータ制御コマンド

マニピュレータ制御ソフトウエアとマスタモジュール上のソフトウエアとの通信を行うため，表7.2に示すマニピュレータ制御コマンドシステムをマスタモジュール上にC言語の関数として実装した．したがって，マスタモジュール上のソフトウエアは，プログラム中でこのコマンドを発行することで，マニピュレータ制御ソフトウエア上の制御パラメータの変更や，マニピュレータの状態の獲などを行うことができる．

Set_comp_mat_arm(comp_init);	comp_init で示したコンプライアンス行列の対角成分を設定
Set_ref_angle_arm(ref_angle);	ref_angle で示したマニピュレータの目標関節角度を設定
Set_hand_width(d);	ハンドの指の目標幅をd mm に設定
Get_ref_angle_arm(now_angle);	マニピュレータの現在の関節角度をnow_angleに代入
Get_force_data(force_data);	力センサから得られた力情報をforce_dataに代入

変数の型： double ref_angle[7], now_angle[7], comp_init[6], force_data[6], d;

表 7.2: マスタモジュール上に実装したマニピュレータ制御コマンド

7.3 ノブ位置認識ソフトウエア

表5.5に示す通り，視覚センサモジュールが担当する機能は，ノブの位置認識機能である．そこで，この機能を実現するノブ位置認識ソフトウエアを，視覚センサモジュール上に実装した．本節では，このソフトウエアの設計及び実装について説明する．

7.3.1 ノブ位置認識ソフトウエアの設計

4.2節の動作設計でも述べた通り，ノブの把持動作を行う前にハンドは，マニピュレータを制御することでノブの付近に位置制御される．したがって，ハンドに搭載されたカメラからは，図7.4-(1)に示すような画像が得られる．そこで，このカメラから得られた画像の処理を行い，その画像中のノブの中心位置を抽出するソフトウエアを，視覚センサモジュール上に実装することにした．

さて，図7.4-(1)からも分かる通り，画像中ノブ以外の物体は存在せず，ノブが存在する場所以外の輝度値の変化は少ない．そこでこの特徴を生かし，微分と重心計算を用いて画像中のノブの位置を求めることにした．この処理の詳細を，図7.4に示す具体例を交えながら以下に説明する．

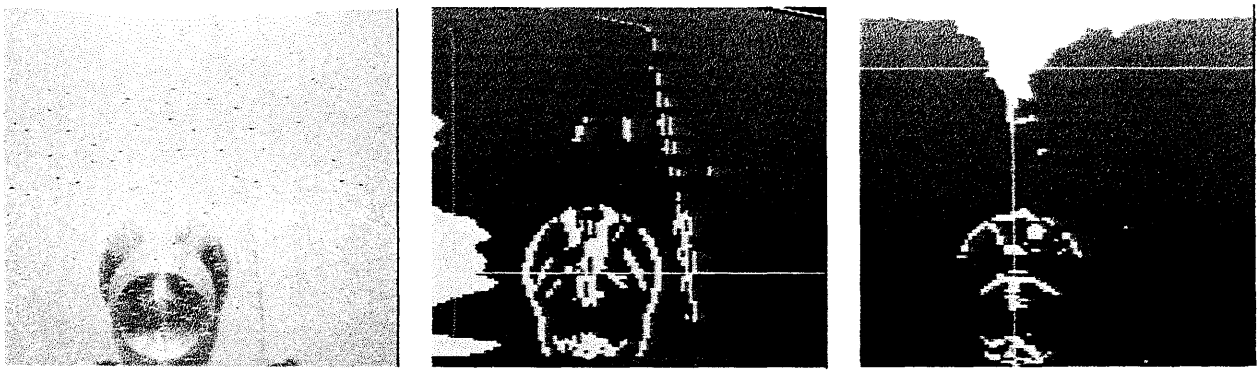

図 7.4: ノブの画像と画像処理結果

まずノブの輪郭を抽出するため，取得した画像を3×3のSobelオペレータを用いて縦方向及び横方向に微分し，適当なスレッショルドを用いて二値化することにした．ただし，処理スピードを上げるため，ここでは一つ飛ばしの画素（画像サイズは128×120となる）について処理を行うことにした．次にそれぞれの方向に輝度値のヒストグラムをとり，これらの重心計算を行うことによって，画像上でのノブの中心位置を求めることにした．ただしノイズによる影響を除去するため，重心計算を行う際，輝度値の合計値の最大値の半分の値をスレッショルドとし，それ以下のものは計算に含めないことにした．

元画像を図7.4-(1)とした場合の，横方向に微分して二値化した結果を図7.4-(2)に示す．

ここで，左側に表示した凹凸が横方向の輝度値の合計値であり，またノイズを除去するため，輝度値の合計値の最大値の半分のところに敷居値を設けている．この処理の結果，ノブの中心が求められていることが判る．また，横方向に関する処理を行った結果を図7.4-(3)に示す．この場合の輝度値の合計値は画像の上側に表示されている．

7.3.2 ノブ位置認識ソフトウエアの実装

以上の設計にしたがって，ノブ位置認識ソフトウエアを視覚センサモジュール上に実装した．

通常本ソフトウエアは，マスタモジュールからのリクエスト待機状態である．画像処理のリクエストが，マスタモジュール上のソフトウエアから双方向メモリを通じて送られた後，画像処理に移行する．画像処理は，カメラから画像をフレームメモリ上に入力した後，このフレームメモリ上で設計した画像処理動作を行う．この処理の結果得られた情報は，画像上のノブの中心位置である．この情報を双方向メモリを通じて，マスタモジュール上のソフトウエアに返すことで，ノブの位置認識の一連の動作が完了し，本ソフトウエアは再びリクエスト待機状態となる．

7.4 力センサソフトウエア

表5.5に示す通り，力センサモジュールが担当する機能は，マニピュレータのコンプライアンス制御機能である．この機能を実現するため，手先にかかる力を処理するソフトウエアを力センサモジュール上に実装した．

5.4.3節に示した通り，力センサのA/Dコンバータユニットからは，力センサが検知した力データが，RS232を用いたシリアル通信で力センサモジュールに送られる．本ソフトウエアは，まずこの力データを受け取り，力データを手先の座標系に合せるための座標変換を行う．ここで得られた力情報は，逐次トランスピュータリンクを通じて，マニピュレータ制御モジュールに送信することにした．

さて，シリアルの通信速度は，本ソフトウエア上の処理スピードと比較すると遥かに遅く，このため力情報の時間遅れはおよそ$20ms$である．仮に力センサモジュール上にA/Dコンバータを搭載し，既存のA/Dコンバータユニットを使用せずに，直接力センサからのデータの処理を行えば，この力情報の遅れを飛躍的に短縮することができるが，これについては今後の課題である．

7.5 既存の機能モジュール上のソフトウエア

表7.1に示した機能モジュール上のソフトウエアのうち,(1) 走行制御ソフトウエア (Spur), (2) 位置推定ソフトウエア (POEM), (4) 測距ソフトウエア, (7) 音声合成ソフトウエアについては，山彦プロジェクトで研究開発された既存の標準ソフトウエアを使用することにした．本節では，これらの各ソフトウエアの仕様の説明を行う．

7.5.1 走行制御ソフトウエア—“Spur”

走行制御モジュール上に搭載された走行制御ソフトウエア “Spur” は，車輪型移動ロボットの走行制御を行うものである．この走行制御方式及び走行制御コマンドの研究開発は，山彦プロジェクトにおいて飯田ら [IY91] によって行われた．現在，山彦プロジェクト内で稼動する山彦ロボットの大部分は，この走行制御ソフトウエアを搭載している．以下に，この走行制御ソフトウエア “Spur” の特徴，構成，及びコマンドシステムについて述べる．

特徴

本走行制御方式の特徴は，駆動輪速度制御器に，移動ロボットの動特性による影響を補償する前置補償器を有する点である．この補償器は，移動ロボットの本体の動特性を解析し，各駆動輪に取り付けられたアクチュエータが発生する駆動力と移動ロボット本体の挙動との間の関係を明らかにし，移動ロボットの逆の特性を持つ前置補償器を構成することで実現される．また，DCモータの制御には，5.3節で述べたPWMスイッチング方式が採用されている．

ソフトウエアの構成

本ソフトウエアは，コマンド解析部とサーボ部から構成される．このソフトウエア構成を図7.5に示す．

コマンド解析部は，双方向メモリを通じてマスタモジュール上のソフトウエアから走行制御コマンドを受け取り，制御パラメータを更新する．このコマンド解析部は，マスタモジュール上のソフトウエアからコマンドが発行された時点で割り込みによって起動される．

一方サーボ部は，ロボットの推定位置をトランスピュータリンクを通じて位置推定モジュールから受け取り，モータに与えるトルクを計算した後，モータテーブルを用いてPWM比

を決定し，これをPWMジェネレータに設定する．このサーボループは，位置推定モジュールからの通信に同期して $5ms$ 毎に実行される．

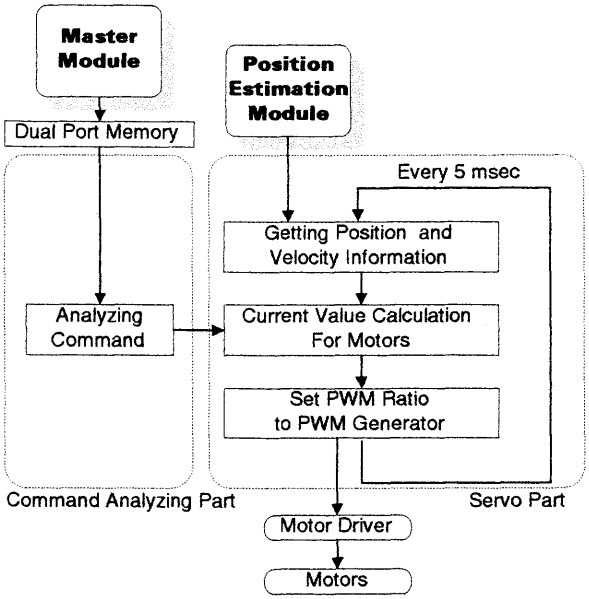

図 7.5: 走行制御ソフトウエアの構成

走行制御コマンドシステム「Spurコマンド」

走行制御ソフトウエアとマスタモジュール上のソフトウエアの通信には，走行制御コマンドシステム「Spurコマンド」が用いられる．このSpurコマンドは，マスタモジュールで実行されるソフトウエア上の，C言語の関数として記述される．

Spur_set_LC_on_LC(x, y, d);	LC座標上で(x, y, d) で示される直線をX軸，(x, y)を原点とした座標系をLC座標系として新たに設定
Spur_line_LC(x, y, d);	LC座標上で (x, y, d) で示される直線に追従走行
Spur_stop_LC(x, y, d);	LC座標上で (x, y, d) で示される直線に対して垂直な直線上に停止
Spur_set_vel_LC(v);	目標速度をvに設定（デフォルトは$300mm/s$）

ただし各変数の型は全て整数型

表 7.3: 本研究で使用するSpurコマンド

このコマンドシステムの特徴は，移動ロボットに目的地を指定するのではなく，目標直

線を指定し，その直線に沿ったロボットの走行制御を行う点である．またこのソフトウエアは，座標系としてグローバル座標（GL），ローカル座標（LC），フロントサイド座標（FS）を持つ（このフロントサイド座標とは，ベースロボットの中心を原点，前方をX軸にとる座標系である）．グローバル座標，及びローカル座標は，環境中任意の位置に設定することができ，Spurコマンドを用いて新たなローカル座標を現在のローカル座標上にマッピングすることもできる．本研究では，ローカル座標のみを使用することにした．

本研究で使用するSpurコマンドとその働きを表7.3に示す．ただしこの表中,「LC座標上で(x, y, d)で示される直線」とは，ローカル座標上において点(x, y)を通りX軸となす角がdで表現される直線のことである．

Spurコマンドがマスタモジュール上のソフトウエアから発行されると，コマンドの種類及び制御パラメータは，双方向メモリを通じて走行制御ソフトウエアに送られる．これをコマンド解析部が解釈して制御パラメータを変更することで，発行したコマンドにしたがったロボットの走行動作が実現される．

7.5.2 位置推定ソフトウエア—“POEM”

位置推定モジュール（POsition Estimation Module—POEM）に搭載された位置推定ソフトウエア“POEM”は，移動ロボットの位置推定を行うものである．この位置推定アルゴリズム及び位置推定モジュールの開発は，山彦プロジェクトにおいて渡辺，上田らによって行われた[WY90][上田92]．この位置推定システムの特徴は，走行制御機能とは独立に移動ロボットの位置及び分散の推定を行う点，及び駆動輪の車輪径も同時に推定する点である．以下に位置推定アルゴリズムの概略，ソフトウエアの動作手順，及び本研究で使用するPOEM関数について説明する．

位置推定アルゴリズム

ここでは，駆動輪の回転数を計数して現在位置を推定するオドメトリを用いて，移動ロボットの現在位置を常時推定する．また，ロボットの走行に伴い，オドメトリによって推定した位置には誤差が生ずるので，ロボットの位置の分散についても同時に推定する．

一方，移動ロボットは，外界センサを用いてランドマーク情報を獲得し，ロボットが有する環境モデルと比較することで位置推定を行うことができる．この位置推定ソフトウエアは，マスタモジュール上のソフトウエアから，この推定位置情報を非同期に受け取ることにより，この推定位置とオドメトリで得られた推定位置を最尤推定法を用いて融合する

ことで，推定した移動ロボットの位置，車輪系，及び分散を更新し，ロボットの推定位置の修正が行われる．

ソフトウエアの動作手順

位置推定ソフトウエアは，駆動輪の回転数をロータリエンコーダから獲得することで位置推定を行うが，この推定は$5ms$毎に行われる．一方，本ソフトウエアは，外界センサを用いてランドマーク情報を獲得することで得られるロボットの推定位置を，マスタモジュール上のソフトウエアから双方向メモリを通じて非同期に獲得し，オドメトリによる推定位置と融合する．こうして得られるベースロボットの推定位置は，トランスピュータリンクを通じて走行制御ソフトウエアに送られ，走行制御に利用される．またこの情報は，双方向メモリ上の指定アドレスにも逐次記入されるため，マスタモジュール上のソフトウエアも，この情報を参照することができる．このソフトウエアの動作を図7.6に示す．

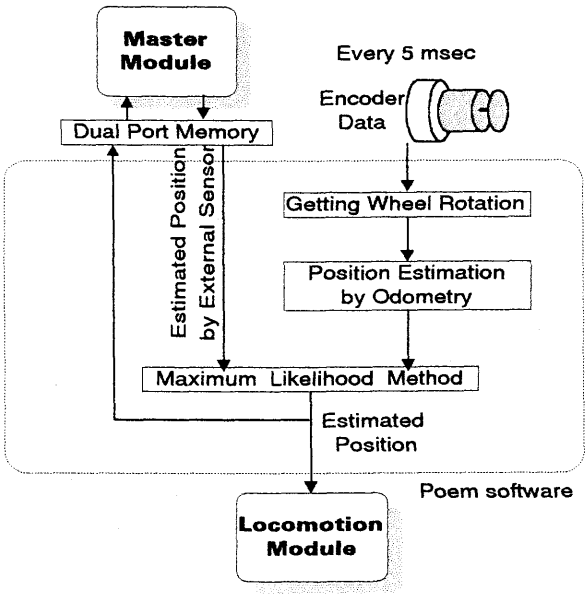

図 7.6: 位置推定ソフトウエアの構成

POEM関数

位置推定ソフトウエアとマスタモジュール上のソフトウエアの通信には，上田ら[上田92]によって開発されたPOEM関数が使用される．このPOEM関数は，マスタモジュールで

実行されるソフトウエア上のC言語の関数として記述される．本研究で使用するマスタモジュール上のPOEM関数とその働きを表7.4に示す．

外界センサによる推定位置を融合するPOEM関数（図7.4中POEM_fuse_y_LC）は，外界センサによって推定した位置情報及びセンシング誤差の偏差を，双方向メモリを通じて位置推定ソフトウエアに送る．位置推定ソフトウエアは，これを解釈することで推定位置の融合を行う．一方，ロボットの推定位置を獲得するためのPOEM関数（図7.4中POEM_get_pos_LC）は，位置推定モジュール上の双方向メモリを参照することで，最新のロボットの推定位置情報を獲得する．

POEM_get_pos_LC(&x, &y, &d)	ロボットのLC座標上の推定位置 (x,y) 及び方位 d を得る
POEM_fuse_y_LC(&y, &d)	LC座標上で外界センサによるy座標の値とその誤差の偏差を与えて，オドメトリとのデータ融合を行う

ただし各変数の型は全て整数型

表 7.4: 本研究で使用するPOEM関数

7.5.3 測距ソフトウエア

超音波距離センサモジュール上に搭載された測距ソフトウエアは，ロボットから超音波を反射する対象物までの距離を測定するものである．この測距システムは，山彦プロジェクトにおいて宋ら[SY89a]，及び大野ら[OOY95]によって研究・開発が行われた．以下にこのシステムによる測距ソフトウエアの動作，及び本研究で使用する測距関数について説明する．

測距ソフトウエアの動作

この測距ソフトウエアは，超音波距離センサ HiSonic の超音波送受信回路(以下送受信回路と呼ぶ）に送信信号を送ったと同時に，各送受信回路に対応したタイマを始動させる．対象物に反射した超音波を各送受信回路が受信し，超音波距離センサモジュールがその受信信号を受けた瞬間に対応するタイマを停止させ，そのタイマの情報と音速から各方向の対象物までの距離を算出する．

測距関数

超音波を送信する間隔は $60ms$ なので，測距情報は$60ms$ 毎に更新される．この情報は双方向メモリ上に記入され，マスタモジュールから参照できる．マスタモジュール上のソフトウエアは，C言語の関数として実装された測距関数 us_dist を使用することで，各送受信回路の方向に対応した対象物までの距離情報が参照できる．この関数の使用例を以下に示す．

$$\text{(int) d} = \text{us\_dist(US\_RIGHT);}$$

ただし，この関数の引数である US_RIGHT は，右方向の送受信回路に対応する値のマクロである．送受信回路は，5.4.1節で述べたように，ベースロボットの前方4箇所，斜め前方2箇所および左右の2箇所に搭載されているので，その方向に対応したマクロをこの関数の引数とすることで，その方向の対象物までの距離情報が得られる．また，この関数の返り値は整数型であり，測距情報はセンチメートル単位で得られる．

7.5.4 音声合成ソフトウエア

音声合成モジュールに搭載された音声合成ソフトウエアは，音声を用いてロボットの内部状態を人間に伝えるものである．このシステムは，山彦プロジェクトにおいて前山らによって研究開発が行われた[前山94]．

特徴

このソフトウエアの特徴は，音素辺を用いた音声合成方式と，PCM方式で録音した音声データの再生という2種類の方式で音声が生成される点である．前者の方式は，音素辺を組み合わせることで様々な音声を合成することができ，またメモリもあまり必要としないが，音質が悪いという欠点を持つ．一方後者の方式は，ほぼ肉声に近い音声を発声することができるという利点を持つが，多くのメモリを必要とし，また録音された音データしか再生できないという欠点を持つ．本研究では，音声情報をデバッグ時に使用するため，音質にこだわる必要はない．そこで，主に音素辺を用いた音声合成方式を使用した．

音声コマンド

音声合成ソフトウエアとマスタモジュール上のソフトウエアの通信には，音声コマンドが用いられる．この音声コマンドは，マスタモジュール上で実行されるソフトウエア上の，

C言語の関数として記述される．本研究で使用する音声コマンドを表7.5に示す．

音声コマンド	動作内容
sayw("akasatana");	音素辺を用いた音声合成手法を用いて， 文字列をローマ字読みで発声
sayd(int i);	音素辺を用いた音声合成手法を用いて， 整数を日本語の数字の読みで発声
sayp("name");	音声合成モジュール上にある"name"という ラベルのPCM録音データを再生

表 7.5: 音声コマンドの例

このコマンドがマスタモジュール上のソフトウエアから発行されると，音声パラメータは双方向メモリを通じて音声合成ソフトウエアに送られる．このパラメータにしたがって，音声合成ソフトウエアはスピーカを通じて音声情報を発声する．

7.6 まとめ

第6章でも示した通り，ロボットシステムのソフトウエアは，機能分散コンセプトにしたがって，マスタモジュール上のソフトウエアと機能モジュール上のソフトウエアの2階層で構成した．このうち本章では，機能モジュール上のソフトウエアについて扱った．

この中でも，マニピュレータ制御ソフトウエア，ノブ位置認識ソフトウエア，力センサソフトウエアについては，本研究で設計及び実装を行った．一方，その他の機能モジュール上のソフトウエアである，走行制御ソフトウエア，位置推定ソフトウエア，測距ソフトウエア，音声合成ソフトウエアについては，山彦プロジェクトで開発された既存のソフトウエアを使用することにした．

各機能モジュール上のソフトウエアとマスタモジュール上のソフトウエア間の通信は，マスタモジュール上に実装した通信用のコマンドや関数によって実現することにした．このコマンドや関数を用いて，各機能を統括しロボットの動作を実現するマスタモジュール上のソフトウエアの詳細について，第8章，第9章で述べる．

第 8章

アクションプリミティブの実現

8.1 はじめに

ロボットの動作全体を制御するマスタモジュール上のソフトウエアは，第6章で述べた通り，各部分動作を実現するための「アクションプリミティブ」というソフトウエア群と，このアクションプリミティブのシーケンスを作成し順次実行する「統括制御ソフトウエア」で構成した．ここで部分動作とは，目標動作を時間の経過に沿って動作のまとまり毎に分割したものである．本章では，この各部分動作を実現するためのソフトウエア「アクションプリミティブに」焦点を当てる（統括制御ソフトウエアについては第9章で詳しく述べる）．

さて，目標タスク「ドアの通り抜けを含む自律ナビゲーション」は，表3.1に示した通り「経路追従走行」,「ノブの把持動作」,「ノブのリリース」,「ドアの開閉動作」の4つの部分動作に分割できる．そこで本研究では，これらの各部分動作を実行する4つのアクションプリミティブを実装した．本章では，各アクションプリミティブに共通する設計，及び実装方法について述べ，8. 3節から8.6節では，個々のアクションプリミティブの設計，実装，及び性能を評価するための動作実験について述べる．

8.2 アクションプリミティブの設計と実装

第6章でも述べた通り，目標動作を時間の経過に沿って動作のまとまり毎に分割し，各部分動作を実現するソフトウエアの単位を「アクションプリミティブ」と名付けた．これにより，ロボット全体の動作は，アクションプリミティブのシーケンスとして表現される．本節では，各アクションプリミティブに共通する設計及び実装法について述べる．

アクションプリミティブに共通の設計

各アクションプリミティブは，統括制御ソフトウエアから動作パラメータを受け取ることで，動作を開始することにした．また，アクションプリミティブは動作開始後，受け取った動作パラメータと各アクションプリミティブの制御アルゴリズムにしたがって，機能モジュール上の双方向メモリを通じて機能モジュールと通信を行うことで，担当する部分動作を実現することにした．

さて，3.4節の分析でも触れたように，移動マニピュレータを制御する際の大きな問題は，オドメトリに含まれるベースロボットの推定位置誤差である．この誤差のため，ハンドの位置にも誤差が生じ，これが対象物体の把持や操作を行う上で大きな問題となる．よって，外界センサなどを用いて可能な限りこれらの位置誤差を削減することが，移動マニピュレータの制御には必要不可欠である．ただし，この外界センサを用いた位置推定アルゴリズムは，動作環境やロボットが行う動作に大きく依存する．そこで，各アクションプリミティブ毎に，外界センサを用いたベースロボットやハンドの位置推定アルゴリズムを搭載することにした．外界センサを使用した位置推定機能を含むアクションプリミティブの動作の概略を図8.1に示す．

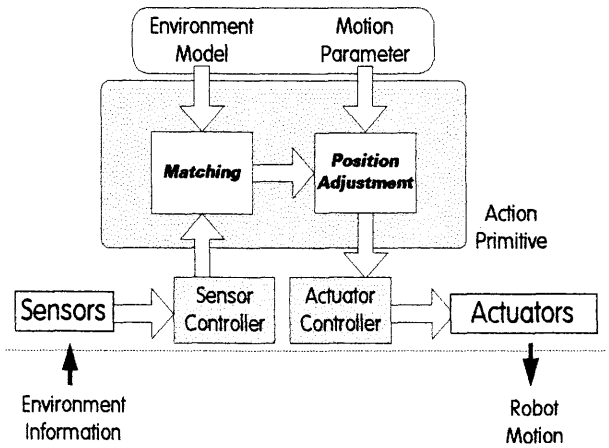

図 8.1: アクションプリミティブに共通する動作

動作中のアクションプリミティブは，センサ情報を用いて，各動作の終了条件が満たされたと判断したとき，または動作の継続が不可能であると判断したとき，統括制御ソフトウエアにアクノレッジを返し，動作を終了することにした．

アクションプリミティブの実装

統括制御ソフトウエア，及び各アクションプリミティブは，マスタモジュール上のプロセスとして実装した．また，アクションプリミティブのプロセスと統括制御プロセスとの通信は，メッセージ通信で実現した．

具体的には，まずアクションプリミティブを実現する各プロセスは，ロボットの動作開始時に生成し，全てのプロセスをメッセージ待機状態にして動作を止めておく．各プロセスは，統括制御プロセスから動作パラメータや環境情報を受け取ることで動作を開始する．また，動作中のアクションプリミティブのプロセスと，各機能モジュール上のソフトウエアとの通信は，バスを通じて機能モジュール上に搭載された双方向メモリ上の指定アドレスを読み書きすることで行う．また，センサ情報より終了条件が満たされたと判断すると，このプロセスは統括制御プロセスに動作終了を表すメッセージを送信し，再びメッセージ待機状態となる．

次節より，各々のアクションプリミティブの具体的な説明を行う．

8.3 アクションプリミティブ「経路追従走行」の実現

マスタモジュール上に実装したアクションプリミティブ「経路追従走行」は，ベースロボットを目標経路に追従走行せるためのソフトウエアである．4.2.2節でも述べた通り，この軌跡追従には，基本的にはオドメトリによる推定位置を使用するが，この推定位置には誤差が含まれるという問題が生ずる．したがって，できる限り正確にベースロボットを目標経路に追従させることが，移動マニピュレータの制御にとって必要不可欠である．そこで，本アクションプリミティブ「経路追従走行」では，超音波距離センサで環境中に存在する平らな壁を検知し，ロボットが有する壁の位置情報と比較することで，ベースロボットの推定位置を修正する手法を採用することにした．本節では，マスタモジュール上に搭載したこのアクションプリミティブ「経路追従走行」の設計，実装，及び走行実験について述べる．

8.3.1 アクションプリミティブ「経路追従走行」の設計

経路地図の表現

ベースロボットの目標走行経路は2次元平面上の線分列で記述することにした．またこの線分列については，直進走行距離と曲がる角度で表現した．さて，ベースロボットの位

置修正を行うためのランドマークとして，ここでは走行経路に沿った平らな壁を使用する．そこで，この壁に関する情報も，目標走行経路情報に付加することにした．位置修正を行うためのランドマークとして必要な壁の情報には，(1) 壁が始まる位置，(2) 壁が続く長さ，(3) 走行経路から見た壁の方向，(4) 走行経路から壁までの距離，が挙げられる．この壁の情報が付加された目標走行経路情報を**経路地図**と呼ぶことにした．

例として，図8.2に示すベースロボットの走行経路及び経路に沿った壁に関する情報を，ツリー構造の経路地図で表現したものを図8.3に示す．

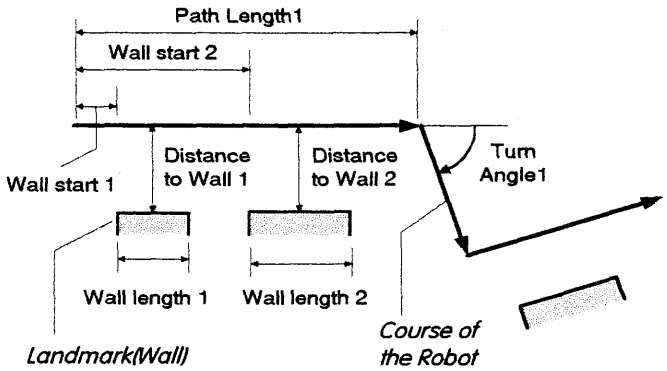

図 8.2: ロボットの走行経路及び壁の情報

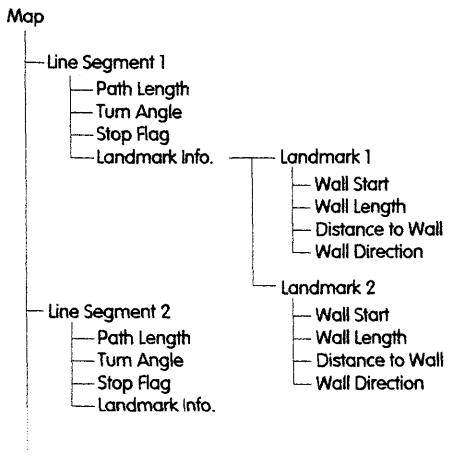

図 8.3: ツリー構造で表現した図8.2の経路地図

経路追従走行

目標走行経路は図8.2に示すように経路地図によって線分列で表現したため，ここではベースロボットを，各線分毎に追従させるように制御することにした．またベースロボットは，この線分上を走行中，現在位置の監視を常に行い，線分の終端付近に到達した地点で，目標線分を次の線分に切り替えることで，経路の乗り換え動作を行うことにした．

ベースロボットの推定位置の修正アルゴリズム

オドメトリによって生ずる推定位置の誤差を減少させるため，アクションプリミティブ「経路追従走行」では，走行経路に沿って壁が存在する場所において，表8.1に示す手順でベースロボットの位置修正を行うことにした．ただし，(3)の推定位置を融合する部分については，位置推定モジュール上で動作する既存の位置推定ソフトウエア“POEM”（7.5.2節参照）を使用し，また(4)の走行制御については，走行制御モジュール上で動作する既存の走行制御ソフトウエア“Spur”（7.5.1節参照）を使用することにした．

(1)	超音波距離センサを用いて壁までの距離を測定
(2)	測距距離と経路地図上の壁までの距離を比較し壁方向のベースロボットの位置を推定
(3)	超音波距離センサによる推定位置とオドメトリによる推定位置を最尤推定を用いて融合
(4)	融合した推定位置を用いたベースロボットの走行制御

表 8.1: 「経路追従走行」におけるベースロボットの位置修正アルゴリズム

8.3.2 アクションプリミティブ「経路追従走行」の実装

アクションプリミティブ「経路追従走行」は，マスタモジュール上の1プロセスとして実装した．以下に，実装したこのプロセスの動作の詳細について述べる．

統括制御ソフトウエアとのインタフェース

「経路追従走行」を実現するこのプロセスは，通常メッセージ待ちの状態で休止しており，統括制御ソフトウエアからメッセージを受けることで実行される．ただし，このメッセージには，図8.3に示すツリー構造の経路地図を付加する必要がある．

経路追従走行

動作を開始した本プロセスは，ベースロボットの走行制御を行うためのSpurコマンド（表7.3）及びベースロボットの位置推定を行うためのPOEM関数（表7.4）を用いて，以下に示す手順で経路追従走行を実現した．

まず，(1) Spurコマンドの座標設定コマンドを発行し，走行開始地点を原点，進行方向をX軸とするローカル座標（LC座標）を設定する．次に，(2) LC座標上におけるX軸に追従するSpurコマンドの直線追従コマンドを発行することで，ベースロボットのLC座標上のX軸に沿った走行が行われる．走行中，本プロセスは (3) ベースロボットの推定位置情報をPOEM関数を用いて逐次獲得する．ベースロボットの走行距離が「LEN（線分の長さ）$-\alpha$」を超えたとき，(4) Spurコマンドの座標設定コマンドを用いて線分の乗り換え点を原点，曲がる角度がX軸となるようなLC座標を新たに設定する．(5) この新たに設定したLC座標上のX軸に追従するSpurコマンドの直線追従コマンドを発行することで，ベースロボットの走行経路の乗り換えが行われる．

以上の手順を繰り返すことで，線分で表現した経路地図に沿ったベースロボットの経路追従走行が実現される．この手順を，単純な経路を例にとり，使用するSpurコマンド，並びにPOEM関数を併せて図8.4に示す．ただしこの図中の括弧内の数字は，上記の手順の括弧内の数字に対応する．

ベースロボットの推定位置の修正動作

本プロセスは，ベースロボットが経路地図に記載された壁に沿った経路を走行する際，表8.1に示した推定位置の修正アルゴリズムにしたがって，超音波距離センサを用いた測距関数(7.5.3節)及びPOEM関数（表7.4）を用いて，以下の手順で推定位置の修正を行う．

まず (1) 測距関数*us_dist*を用いて，超音波距離センサで測距した壁までの距離情報を獲得する．次に (2) この距離情報と経路地図に記載された壁までの距離情報の差をとることで，壁方向のベースロボットの位置を推定する．(3) この外界センサで推定した推定位置を，POEM関数の融合関数を用いて，位置推定ソフトウエアに送る．これにより，位置推定ソフトウエアの内部で，外界センサを用いたベースロボットの推定位置とオドメトリによる推定位置は最尤推定法によって融合され，ベースロボットの推定位置が修正される．この修正された推定位置を用いてベースロボットが経路追従を行うことで，ベースロボットの位置修正が行われる．

この修正動作は，走行経路に沿って壁が存在する経路上で，ベースロボットが5*cm*進む

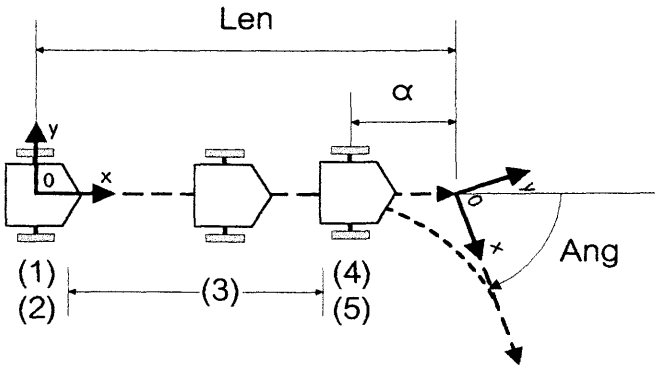

図 8.4: 経路追従走行の手順

ごとに行われる．この融合を繰り返すことで，位置推定ソフトウエア内のベースロボットの推定位置と方向の相関から，ベースロボットの方向の誤差修正も同時に行われる [WY90].

モジュール間通信

以上に述べた通り，アクションプリミティブ「経路追従走行」を実現するプロセスは，動作中に，Spur コマンド，POEM 関数，及び超音波距離センサによる測距関数を用いて，走行制御ソフトウエア，位置推定ソフトウエア，及び測距ソフトウエアとの間で情報交換を行う必要がある．この通信には，5.5.3節でも述べた通り山彦バスと双方向メモリを用いて実現した．このプロセスが動作するマスタモジュールと各機能モジュール間の情報の流れを図8.5に示す.

8.3.3 経路追従走行実験

実装したアクションプリミティブ「経路追従走行」の性能評価を行うため，このアクションプリミティブを用いてベースロボットを目標経路に追従走行させる実験を行った.

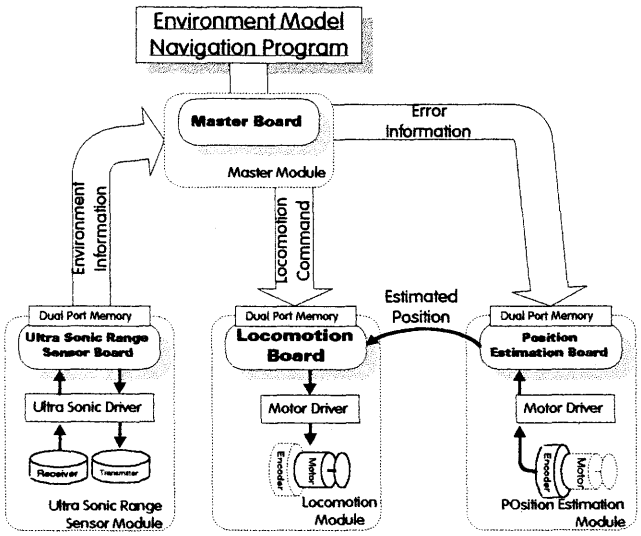

図 8.5: アクションプリミティブ「経路追従走行」におけるモジュール間通信

実験環境及び走行経路

実験環境は2.5節で設定した通り，筑波大学の研究室及びその廊下とした．この環境におけるベースロボットの走行経路は，ドアを出て廊下を壁沿いに走行し，もう片方のドアから部屋の中に入る経路とした．ただし双方のドアは片側が開いており，ドアの開き角度は，壁に対してほぼ直角となるように設定した．この経路上で最も幅の狭い場所は，ドアを通り抜ける部分となり，その幅は約90cmである．実験環境及び目標走行経路の概略を図8.6に示す．

経路地図

図8.6の走行経路及び壁の情報を表す経路地図は，表8.2に示すC言語の構造体で表記した．本実験では，テストプログラムがこの表現の経路地図を図8.3に示すツリー構造表現（C言語のポインタ表現を使用）に変換し，アクションプリミティブ「経路追従走行」のプロセスにこの経路地図を付加したメッセージを送ることで，ベースロボットの走行動作が行われた．

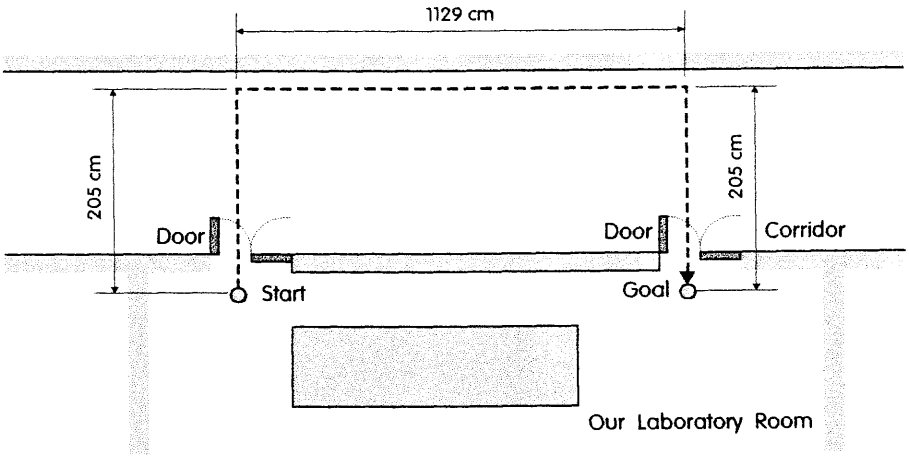

図 8.6: 経路追従走行の実験環境及びベースロボットの走行経路

```
/*****************************************************************
 *   Include file of Trajectory_data
 *****************************************************************/

static int trajectory_data[] = {

    0,              /* [map->number]   path number              */
    205,            /* [map->length]   path length (cm)         */
    -90,            /* [map->angle]    path next angle(degree) */
    0,              /* [map->stop]     stop flag                */
    1,              /* number of landmarks                      */
          80,       /* [mark->start]   start landmark (cm)        */
          80,       /* [mark->length] length landmark (cm)        */
          -1,       /* [mark->dir]     1:right  -1:left  landmark */
          39,       /* [mark->dist]    distance from landmark(cm) */

    1, 1129, -90, 0, 2,
        10, 130, -1, 68,
       260, 600, -1, 68,

    2, 205, 1, 1,
10, 80, 1, 39,

    -99             /* means data end */
};
```

表 8.2: 走行実験用経路地図（C言語の構造体で表現）

実験結果及び考察

走行結果は，ベースロボットが壁を用いた位置修正を行った実験では，ベースロボットは目的地に到達し，誤差は目測で2cm以下であった．またベースロボットの方向についても，走行終了後，目で見てわかるほどの誤差は無かった．一方，壁を用いた位置修正を行わない実験では，ほとんどの場合ベースロボットは環境中の物体に接触し，走行動作は失敗した．

この走行実験から，屋内環境であれば，環境中の壁と超音波距離センサを用いた位置修正アルゴリズム，及びオドメトリによって，比較的長距離の走行においても，精度の良い経路追従走行が行えることが分かった．ただし，超音波距離センサの測距誤差などにより，このシステムでは1～2cmの位置誤差は不可避であることも，この実験から明らかとなった．

8.4 アクションプリミティブ「ノブの把持動作」の実現

マスタモジュール上に実装したアクションプリミティブ「ノブの把持動作」は，マニピュレータを制御して，ノブにハンドをアプローチし，このノブを把持する動作を行うソフトウエアである．ただし，この動作の前提として，ベースロボットは,ほぼドアの前の規定の位置に停止しており，ノブの位置も既知とする．

第3.5節の分析でも述べた通り，仮にベースロボットが傾くことなく，正確に規定の位置に停止すれば，ハンドがノブを把持するためのマニピュレータの姿勢は計算することができる．しかし現実的には，ベースロボットの位置誤差や傾きが原因で，ハンドには位置誤差が生ずるため，マニピュレータの姿勢制御のみでノブを把持することは困難である．そこで本アクションプリミティブ「ノブの把持動作」では，ハンドに搭載した視覚センサ及び手先に搭載した力センサを用いてノブの位置を認識し，この情報を基にマニピュレータを制御し，ハンドのアプローチを行うことにした．

8.4.1 アクションプリミティブ「ノブの把持動作」の設計

このアクションプリミティブ「ノブの把持動作」では，ノブを把持するまでの一連の動作を，マニピュレータの初期姿勢の制御，視覚センサを用いたアプローチ，ドアへの接触動作，ノブの把持 の4段階で行うことにした．以下にこの動作設計を各段階毎に説明する．

ハンド座標系の設定

アクションプリミティブ「ノブの把持動作」の設計を行うにあたり，まず，ハンドを基準とした座標系を設定した．以後この座標系をハンド座標系と呼ぶことにする．このハンド座標系は，ハンドの先端を原点とし，手先方向をZ_{hand}軸，ハンドが開閉する向きをY_{hand}軸，この両方の軸に垂直な軸をX_{hand}軸とした（図8.7参照）[P.P81].

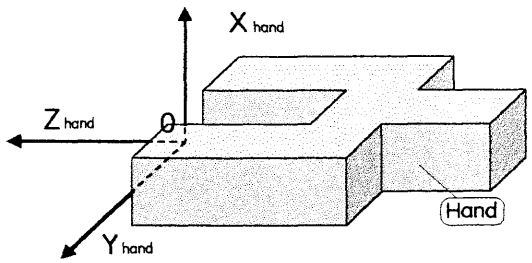

図 8.7: ハンドを基準とした座標系の設定（ハンド座標系）

第1段階：マニピュレータの初期姿勢の制御

まず，ハンドに搭載した視覚センサによって，ノブの画像が獲得できるように，マニピュレータの初期姿勢をマニピュレータにとらせることにした．このマニピュレータの初期姿勢は，ベースロボットの推定位置誤差が無い理想状態でハンドがノブを把持した時のマニピュレータの姿勢（以後マニピュレータの理想把持姿勢と呼ぶ）から，手先をZ_{hand}軸のマイナス方向にアプローチ距離分だけスライドし，さらにX_{hand}軸のマイナス方向に$5[cm]$スライドさせた姿勢とした（図8.8参照）．

第2段階：視覚センサを用いたハンドのアプローチ

ここでは，ハンドに搭載した視覚センサからのノブの位置情報をフィードバックしてマニピュレータを制御し，ハンドのノブへのアプローチを行うことにした．まず，画像上のノブの位置認識を画像モジュール上のノブ位置認識ソフトウエアが行い，この情報から，ハンドに対する空間中のノブの相対位置を計算することにした．その後，ハンドの推定位置誤差が小さくなるようにマニピュレータを制御しつつ，ノブの方向へハンドをアプローチすることにした．

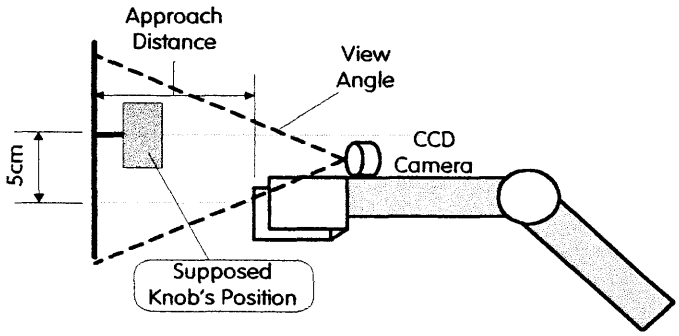

図 8.8: ノブの把持動作におけるマニピュレータの初期姿勢

以上のノブの位置認識とハンドの位置修正及びアプローチを，ハンドがノブの近傍に来るまで繰り返すことで，視覚センサを用いたハンドのアプローチ動作を行うことにした．この動作アルゴリズムを図8.9に示す．

第3段階：ハンドのドアへの接触動作

次に，ハンドとドアとの距離を正確に求めるため，ハンドがドアに接触するまでアプローチを行うことにした．

視覚センサを用いたハンドの位置誤差の修正は，ノブの軸に垂直な平面上の誤差については容易に実現できるが，ノブの軸方向の誤差については困難である．このため，視覚センサを用いたハンドのアプローチによって，ノブの位置は，ハンド座標系上で，誤差αを含む$(5,0,3\pm\alpha)$となる．このαを確定するため，ハンドがノブに十分に近づいたこの段階では，視覚フィードバックを行わずに，ハンドをドアに接触させてこの誤差を減少させることにした．また，ドアへの接触は，力センサによって検出することにした．この動作アルゴリズムを図8.10に示す．

ノブの軸に垂直な平面上のハンドの位置誤差は第2段階で既に修正されているため，この接触動作によって，ハンド座標系上におけるノブの位置は，ほぼ$(5,0,0)$となる（図8.11参照）．

第4段階：ノブの把持

ハンドに対するノブの相対位置は，この時点でほぼ$(5,0,0)$なので，ハンド座標上でハンドを$(5,0,-2)$だけ移動させることで，把持中心とノブの回転軸を合わせることができる．

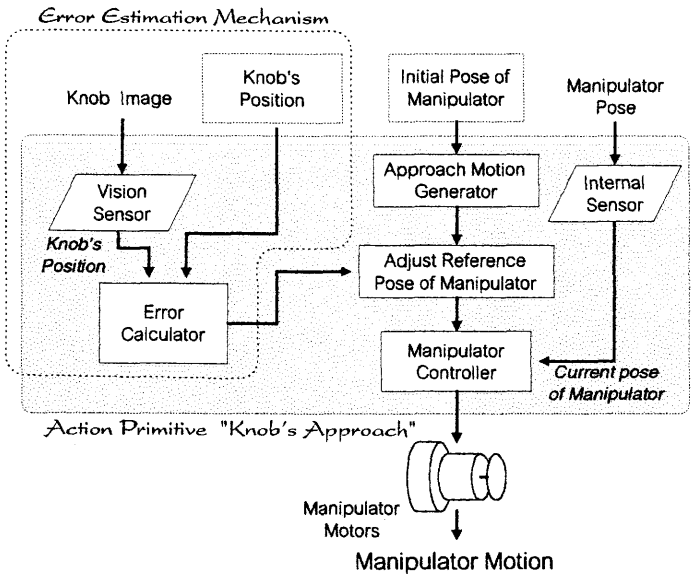

図 8.9: 視覚センサを用いたハンドのアプローチの動作アルゴリズム

ハンドの移動後，スライド式のハンドの指を閉めることによってノブの把持を行うのだが，ハンドに生ずる若干の位置誤差に対処するため，この段階ではマニピュレータのコンプライアンス制御も行うことにした．

ディファレンシャルモーション

設計の最後に，ハンド座標系上でハンドをある方向に移動させるための，各関節の必要回転角を計算するディファレンシャルモーションについて説明する [P.P81]．

まず，ハンド座標系においてハンドの移動量を，$[{}^H d_x \ {}^H d_y \ {}^H d_z]^T$ とする．ただし，本研究では，ハンドの平行移動さえ行えば良いので，回転に関する変数はここでは考慮しない．この時の各関節の必要回転角 $[dq_1 \ dq_2 \ \cdots; dq_6]^T$ を求めることが，ここでの目的である．ただし，使用するマニピュレータは7関節なので，ここでは第5関節を固定し6関節マニピュレータとして冗長自由度問題を回避した．

さて，マニピュレータの姿勢から決まるヤコビ行列を計算すると，ハンドの移動量と各関節の必要回転角との間には，(8.1) 式が成り立つ．

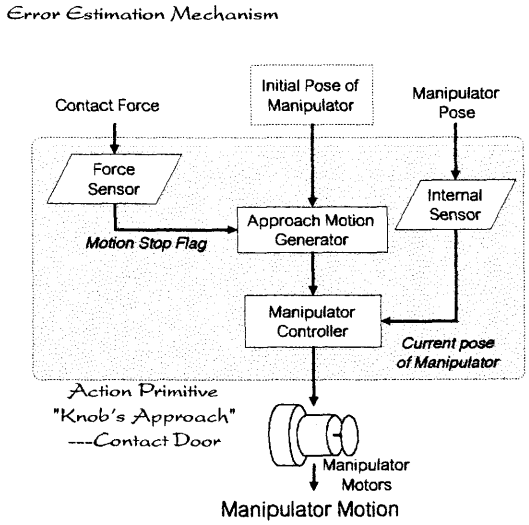

図 8.10: ハンドのドアへの接触動作の設計

$$\begin{bmatrix} {}^{H}d_x \\ {}^{H}d_y \\ {}^{H}d_z \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} j_{11} & j_{12} & \cdots & j_{16} \\ j_{21} & j_{22} & \cdots & j_{26} \\ \cdots & & & \\ j_{61} & & & j_{66} \end{bmatrix} \begin{bmatrix} dq_1 \\ dq_2 \\ \vdots \\ dq_6 \end{bmatrix} \tag{8.1}$$

ただし，j_{mn}はヤコビ行列のm行n列の要素である．よって，この連立方程式を解くことで，ハンド座標系上の移動量に対する各関節の必要回転角が求まる．ただし，${}^{H}d_x, {}^{H}d_y, {}^{H}d_z$は十分に小さい必要があるため，ハンドを大きく移動させる場合は，この移動量を小さく設定し，ヤコビ行列の計算及び連立方程式の計算を繰りし行う必要がある．

8.4.2 アクションプリミティブ「ノブの把持動作」の実装

アクションプリミティブ「ノブの把持動作」は，マスタモジュール上の1プロセスとして実装した．以下に実装したこのプロセスの動作の詳細について述べる．

統括制御ソフトウエアとのインタフェース

「ノブの把持動作」を実現する本プロセスは，通常メッセージ待ちの状態で休止しており，統括制御ソフトウエアからメッセージを受けることで実行される．ただし，このメッ

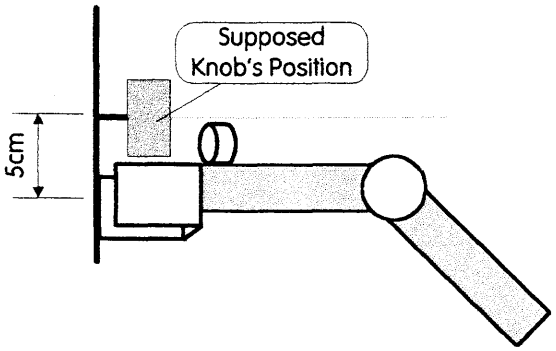

図 8.11: ハンドがドアに接触した時のマニピュレータの姿勢

セージには，マニピュレータの理想把持姿勢における各関節の目標角，及びマニピュレータの初期姿勢におけるハンドのドアからの距離（アプローチ距離）が必要となる．

第1段階：マニピュレータの初期姿勢の制御

動作を開始した本プロセスは，まずマニピュレータを設計した初期姿勢（図8.8）となるようにマニピュレータの姿勢制御を行う．ここで，Z_{hand}軸方向，及びX_{hand}軸方向へハンドを移動するための各関節の必要回転角の計算は，ディファレンシャルモーションを用いて行う．ここで計算したマニピュレータの各関節の必要回転角を，マニピュレータの理想把持姿勢における各関節角に加えて，姿勢制御コマンド（表7.2）を用いてマニピュレータ制御ソフトウエアに送ることで，マニピュレータは図8.8に示す初期姿勢をとる．

以上の，ディファレンシャルモーションを用いて必要回転角を計算し，姿勢制御コマンドを用いてマニピュレータの先端を移動させる一連の手順を，以後**ディファレンシャルモーションによるスライド法** と呼ぶことにする．

第2段階：視覚センサを用いたハンドのアプローチ

次に，本プロセスは設計の第2段階で示した通り，視覚センサを用いたハンドのアプローチ動作を行う．

まず，画像処理のリクエストを，双方向メモリを通じて視覚センサモジュール上のノブ位置認識ソフトウエアに送る．ノブ位置認識ソフトウエアは，画像処理を行った後，画像上のノブの中心位置を本プロセスに返す．そこで，画像上のノブの目標中心位置，及び1

画素あたりの実空間の距離を，ハンドとノブの間の距離に応じ予めテーブルとして準備することで，空間中のハンドの位置誤差を推定し，この位置誤差を減少させる方向へハンドの移動を行う．またドア方向には，Z_{hand}軸方向に$1cm$だけ移動する．これらの移動には，ディファレンシャルモーションによるスライド法を使用する．ただし，ハンドとノブの間の距離については，ベースロボットの位置誤差が無いと仮定した．

以上に述べた通り，アクションプリミティブ「ノブの把持動作」の第2段階「視覚センサを用いたハンドのアプローチ」を実現するプロセスは，動作中に，ノブ位置認識ソフトウエア，及びマニピュレータ制御ソフトウエアとの間で情報交換を行う．この通信は，山彦バスと双方向メモリを用いて実現した．このプロセスが動作するマスタモジュールと各機能モジュール間の情報の流れを図8.12に示す．

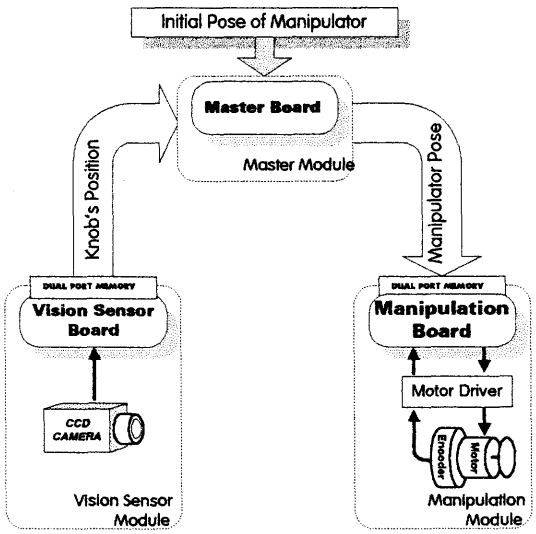

図 8.12: 視覚センサを用いたハンドのアプローチにおけるモジュール間通信

第3段階：ハンドのドアへの接触動作

次に本プロセスは，設計の第3段階で示した通り，ハンドのドアへの接触動作を行う．

ハンド座標系における Z_{hand} 方向にハンドを移動させるための手法は，ディファレンシャルモーションによるスライド法を使用する．一方，力情報の監視は，力情報参照コマンド（表7.2参照）によって行う．ハンドがドアに接触すれば，Z_{hand} 方向の力に変化が起きるため，この力情報を監視することで接触が検知できる．

以上に述べた通り，この第3段階を実現するプロセスは，動作中に，マニピュレータ制御ソフトウエアと情報交換を行う．ただし，力センサモジュールが得た力センサ情報については，トランスピュータリンクを通じて，力センサモジュールからマニピュレータ制御モジュールが得た情報を，マニピュレータ制御ソフトウエアより獲得する．このマスタモジュールと各機能モジュール間の情報の流れを図8.13に示す．

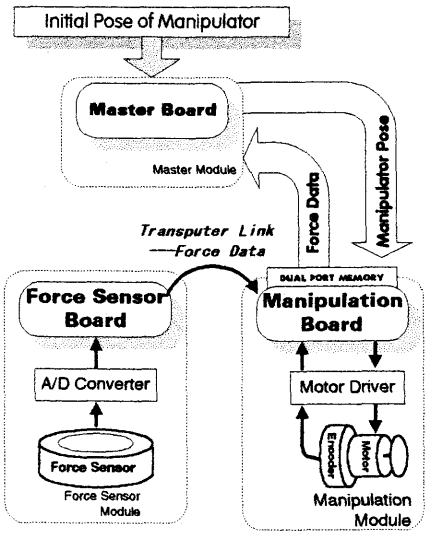

図 8.13: 力センサを用いたドアの接触の動作におけるモジュール間通信

第4段階:ノブの把持

最後に第4段階において，ハンドはノブの把持を行う．

設計でも述べたように，ノブを把持する位置へハンドを移動させるために，ここでもディファレンシャルモーションによるスライド法を使用する．

また，ノブの把持については，ハンドの指の幅を設定するコマンド（表7.2参照）によって実現する．また，センシング誤差などによるハンドの微少な位置誤差を吸収するため，ノブを把持するハンドの先端のコンプライアンス制御を行うことで実現する．

8.4.3 ノブの把持動作実験

実装したアクションプリミティブ「ノブの把持動作」の性能評価を行うため，このアクションプリミティブを実現するプロセスを用いて，ノブの把持動作実験を行った．ただし，

実験には第2章で設定した環境のドアノブを使用する.

実験を行う前に,ベースロボットをドアの前より80cm手前の位置に目測で設置した.このため,ベースロボットの位置には若干の誤差が含まれる.一方マニピュレータの理想把持姿勢については,ワークステーション上で各関節角の計算を予め行った.

動作結果は,ベースロボットを意図的にずらして設置した場合でも,ノブの位置に対するハンドの修正はハンド座標系上で行われるため,ベースロボットの位置誤差が数センチ程度ならば,マニピュレータを制御することでノブの把持動作はほぼ成功した.この結果より,アクションプリミティブ「ノブの把持動作」は,ロバストな把持動作を行うことが分かった.

8.5 アクションプリミティブ「ノブのリリース」の実現

マスタモジュールに実装したアクションプリミティブ「ノブのリリース」は,マニピュレータとハンドを制御して,ハンドがノブを掴んでいる状態からノブをリリースし,マニピュレータがベースロボットの走行の邪魔にならない姿勢(図5.1の写真でマニピュレータがとっている姿勢:以後この姿勢を待機姿勢と呼ぶ)となる動作を行うソフトウエアである.

8.5.1 アクションプリミティブ「ノブのリリース」の設計

ハンドがノブを把持した状態からマニピュレータの待機姿勢に移行するため,まずハンドの指を開きノブをリリースすることにした.この状態からマニピュレータを待機姿勢に移行するために大きく動かすと,ハンドがノブに接触し,動作が遂行できないことが起こる.そこで,まずハンドを図8.7に示すハンド座標系のZ_{Hand}軸のマイナス方向,及びX_{Hand}軸のマイナス方向に$5cm$ずつ移動した後,マニピュレータを待機姿勢となるように姿勢制御することにした.これにより,ハンドがノブに接触することなく,マニピュレータを待機姿勢に戻すことができる.

8.5.2 アクションプリミティブ「ノブのリリース」の実装

アクションプリミティブ「ノブのリリース」は,マスタモジュール上の1プロセスとして実装した.

まず本プロセスは,ハンドの指の幅を設定するコマンド(表7.2参照)によって指の幅を大きく設定し,ノブを離す動作を行う.続いて,ノブのリリースの設計にしたがい,ハン

ドを移動させるが，この動作もディファレンシャルモーションによるスライド法を使用する．この動作が終了した後，姿勢制御コマンドを用いて待機姿勢をマニピュレータ制御ソフトウエアに送ることで，マニピュレータは待機姿勢をとり，ノブのリリース動作は完了する．

8.6 アクションプリミティブ「ドアの開閉動作」の実現

マスタモジュール上に実装したアクションプリミティブ「ドアの開閉動作」は，マニピュレータとベースロボットを制御してハンドで把持したドアの開閉動作を行い，ドアを通り抜ける動作を実現するためのソフトウエアである．

3.7節の分析でも述べた通り，搭載型のマニピュレータの長さには制限があるため，このロボットによるドアの開閉動作を実現するためには，ベースロボットは走行しつつマニピュレータがドアを操作するといった，走行制御とマニピュレータ制御の協調動作を行う必要がある．本研究では，予めオフラインでベースロボットとマニピュレータの動作を計画し，その計画にしたがって動作の同期を取りつつ各制御ソフトウエアにコマンドを送ることで，この協調動作を実現することにした．

一方，走行系とマニピュレータの協調によってドアを開ける際，ベースロボットの推定位置の誤差によって，ドアノブを把持しているハンドに大きな力がかかるという恐れがある．そこで，このベースロボットの位置誤差を減少させるため，ハンドがノブを把持した際のマニピュレータの姿勢から，ベースロボットの位置推定を行うことにした．また，ここで推定しきれないベースロボットの位置誤差によって，ハンドに生ずる力を吸収するための，マニピュレータのコンプライアンス制御も同時に行うことにした．

一方，ドアに鍵がかかっている場合，ドアの通り抜け動作を続行することはできない．このような状況に対処するため，ドア開け動作を行う前には，鍵がかかっているかどうかを確かめる鍵のチェック動作を行うことにした．

本節では，以上の動作を実現するアクションプリミティブ「ドアの開閉動作」の詳細を，動作計画，動作設計及び実装方法といった項目に分けて説明し，最後にドアの押し開け動作実験によるこのアクションプリミティブの性能評価について述べる．

8.6.1 マニピュレータとベースロボットの動作計画

まず，マニピュレータとベースロボットの協調動作を実現するため，この協調動作をオフラインで計画した．この動作計画では，まずロボットがドアを通り抜ける時のドアの開き角度に対するベースロボットの位置をヒューリスティックに決定した．次に，ベースロボットの各位置におけるマニピュレータの姿勢を，各ノブの位置とベースロボットの位置から計算した．ただし，ベースロボットの走行位置やドアを開けるスピードによって，マニピュレータがドアに衝突したり，対応する姿勢をマニピュレータがとれなくなることも起こり得る．そこで，ドアの開き角度に対するベースロボットの位置のパラメータを試行錯誤的に変更し，ベースロボットとマニピュレータの動作シミュレーションを繰り返し行うことで動作計画を行った．以下に，ドアの押し開け動作，及びドアの引き開け動作に関する動作計画の手順，並びに協調動作の表現法について述べる．

ドアの開き角に対するベースロボットの位置の計画

まず，ドアの押し開け動作について，ドアの開き角度に対するベースロボットの位置を，以下の式で表現することにした．

$$\text{Position} = \mathit{Line\_Len} \times \sin\left(\mathit{Door\_Angle} \cdot \frac{\pi}{180}\right) \tag{8.2}$$

この式を用いた場合の，ドアの開き角度とベースロボットの位置の関係は，図8.14のようになる．

この押し開け動作では (8.2) 式中のsin関数によって，ドアが開くにつれてベースロボットの移動量が小さくなる．このsin関数を用いた理由は，ドアの開き角度が大きくなるにつれて，ドアが開く方向とベースロボットの移動方向がなす角が大きくなり，このためにノブに力が生ずるのを防ぐためである．

一方ドアの引き開け動作については，ドアの開き角度とベースロボットの位置の関係は図8.15 に示すものとした．この計画では，ベースロボットはドアの角度が45度となるまではドアの開き角に沿った円弧追従を行い，ドアが45 度より開いた後は直線的に後退する動作を行う．

また，ドアを通り抜けた後のドアの押し閉め動作については，ドアの引き開け動作を逆に辿ることで実現できるため，動作計画は省略した．同様に，ドアの引き閉め動作についても，ドアの押し開け動作を逆に辿ることによって実現できるため，動作計画は省略した．

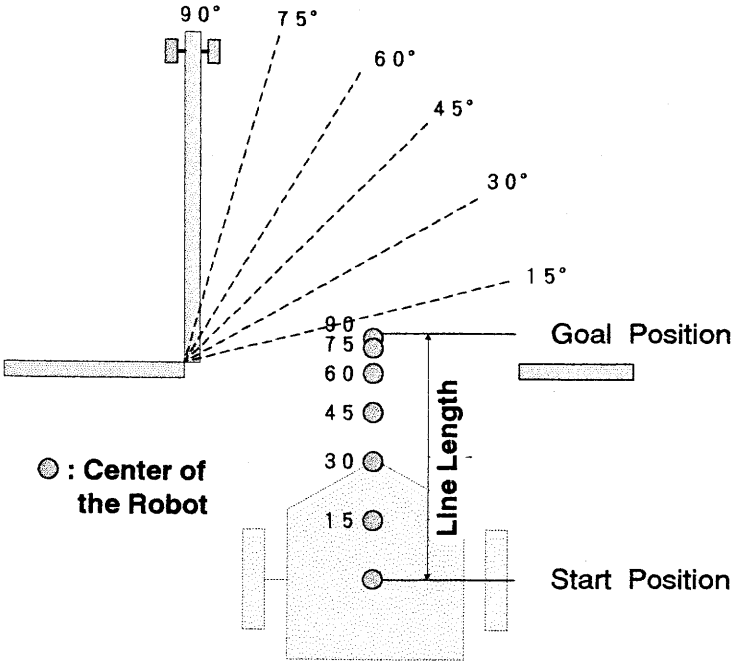

図 8.14: ドアの押し開け動作の計画

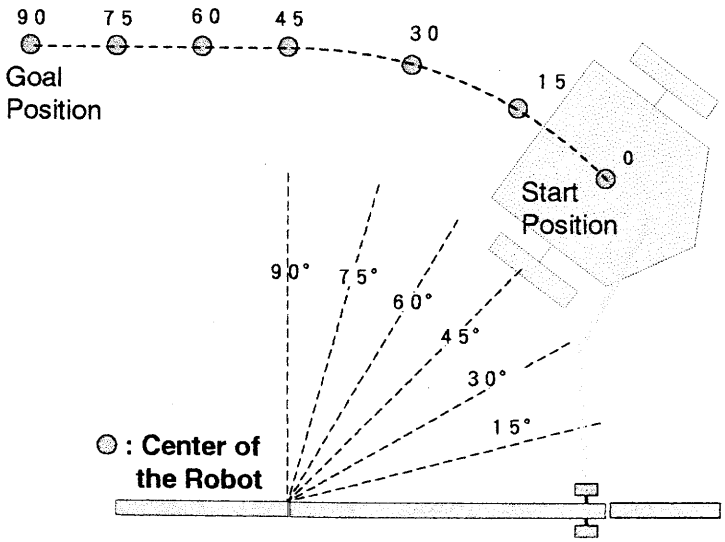

図 8.15: ドアの引き開け動作の計画

マニピュレータの姿勢の計画

次に，ベースロボットの各位置におけるマニピュレータの姿勢の計算を行う．

ドアの開き角度が決まると，自動的に空間中のノブの位置及びノブの回転軸の方向が決定する．ハンドは開閉動作中，このノブを把持し続ける必要があるので，このノブの位置と方向，及びベースロボットの位置と方向から，マニピュレータの姿勢は一意に決まる．ただしここでは，マニピュレータは第5関節を固定し，6関節のマニピュレータとして扱うことにした．このインバースキネマティクスについては，ディファレンシャルモーションの計算を繰り返し行うことで計算した．このように，ベースロボットの位置・姿勢，及び把持対象の位置・姿勢を予め計画することで，移動マニピュレータの冗長自由度の問題を回避した．

シミュレータを用いたベースロボットとマニピュレータの動作計画

試行錯誤的にベースロボットとマニピュレータの動作計画を行うため，本研究では環境に対するマニピュレータの衝突判定，及びベースロボットの衝突判定を行うことができる，移動マニピュレータシミュレータを制作した．このシミュレータは，ドアの開き角に対するベースロボットの位置からマニピュレータの姿勢を計算し，画面に表示するものである．したがって，シミュレータへの入力は，ベースロボットの初期位置，及びドアの開き角に対するベースロボットの位置の関数である．これらを代入すると，このシミュレータは，ドア，ベースロボット，及びマニピュレータの幾何的な動作シミュレーションを行い，その結果より動作が実現可能かどうかが判断できる．よって，入力するベースロボットの初期位置，及びドアの開き角に対するベースロボットの位置の関数を変更することで，試行錯誤による動作計画を行う．

この手法で計画したドアの押し開け動作，及びドアの引き開け動作のシミュレーションの結果をそれぞれ図8.16，図8.17に示す．

ただし，このシミュレーションは，ベースロボットの位置とドアの開き角度に対するマニピュレータの姿勢の計算を行うもので，ベースロボットの走行誤差やダイナミクスは考慮していない．したがって，このシミュレーション結果は，計画した動作によって実際にドアが開けられるということまでは保証しない．本研究では，実際のロボットを動かすことで，計画の妥当性を検証することにした．

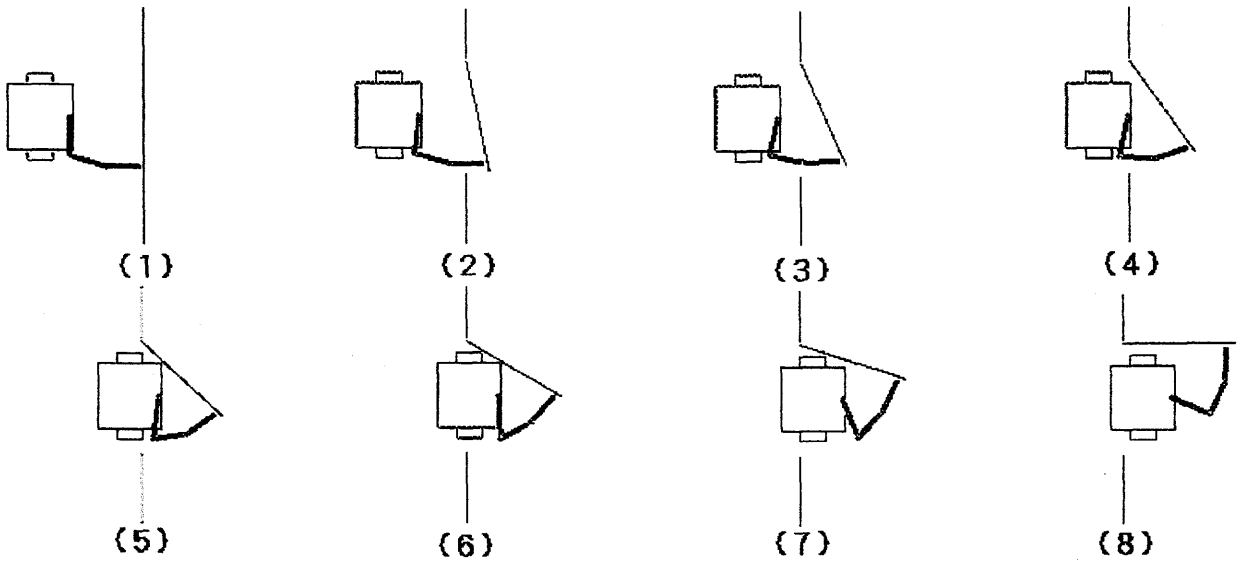

図 8.16: シミュレーション：ドアの押し開け

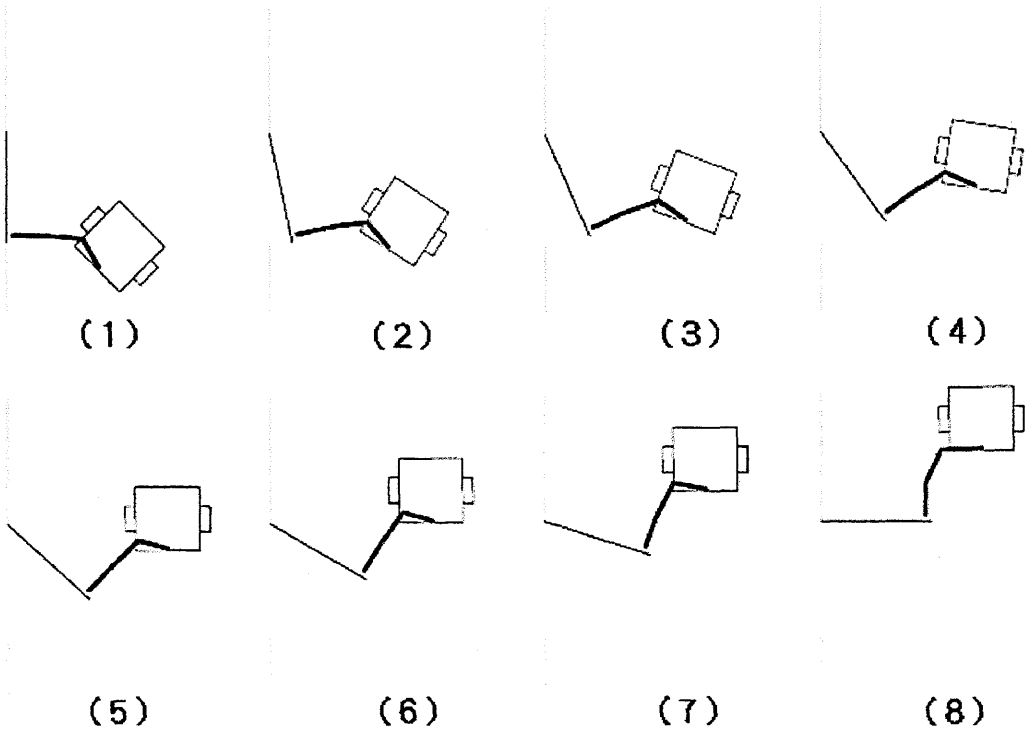

図 8.17: シミュレーション：ドアの引き開け

ベースロボットとマニピュレータの動作表現

動作計画の最後に，計画したベースロボットとマニピュレータの協調動作の表現について説明する．

計画したベースロボットとマニピュレータの協調動作は，ドアの開き角度をパラメータとした，ベースロボットの位置に対するマニピュレータの各関節角の組で表現することにした．この組の数が多いほど，マニピュレータの姿勢は補完され，動きはスムーズになるが，必要とするメモリが大きくなるという問題がある．ここでは，ドアを90度開けるための動作を，ドアの開き角度を1.4度刻みとし，65組のベースロボットの走行距離に対するマニピュレータの各関節の目標角度で表現した．この動作表現の例を表8.3に示す．

ドアの開き	ベースロボットの	マニピュレータの 各関節角度 $[deg]$			
角度 $[deg]$	走行距離 $[cm]$	θ_1	θ_2	...	θ_6
0.0	6.6	51.4	57.9	...	46.7
1.4	33.2	51.1	58.4	...	46.7
2.8	59.7	50.8	58.8	...	46.6
⋮	⋮	⋮	⋮		⋮

表 8.3: マニピュレータとベースロボットの協調動作表現

8.6.2 アクションプリミティブ「ドアの開閉動作」の設計

マニピュレータとベースロボットの協調動作によるドアの開閉動作は，基本的には8.6.1節で計画した協調動作にしたがって双方の動作を行うことで実現される．しかし，実環境で動作する際には，ベースロボットの推定位置誤差などの問題に対処する必要がある．そこで本アクションプリミティブでは，「マニピュレータとベースロボットの協調動作」だけでなく，ベースロボットの推定位置誤差に対処するための「マニピュレータのコンプライアンス制御」，及び「ドアノブの把持姿勢によるベースロボットの位置誤差の推定」を実現することにした．また，ベースロボットの推定位置誤差によってハンドに生ずる力に対処するための「ドアの開閉時におけるベースロボットの位置修正」，及び鍵がかかっているという予定外の状況に対処するための「鍵のチェック動作」についても実現することにした．

マニピュレータとベースロボットの協調動作の設計

ドアの開閉動作中，走行制御ソフトウエアによって，ベースロボットは計画した経路に沿って低速で走行することにした．一方，ベースロボットの推定位置と計画した協調動作よりマニピュレータの姿勢を決定し，この姿勢となるようにマニピュレータを制御することで，マニピュレータとベースロボットの協調動作を実現することにした．

マニピュレータのコンプライアンス制御

3.7節で述べた通り，ドアは片側が拘束されているため，マニピュレータがノブを把持した状態でベースロボットの位置推定に誤差が生ずると，ハンドには力が発生する．この力を吸収するため，ドアの開閉動作を行う際には，マニピュレータの手先のコンプライアンス制御を行うことにした．ただし，ドアからの反力でハンドが戻されないように，ノブの軸に垂直な平面内に拘束されたコンプライアンス動作を行う．

ドアノブの把持姿勢によるベースロボットの位置誤差の推定

ドアの前に停止したベースロボットの推定位置には，オドメトリによって生ずる位置誤差が存在する．この誤差によって生ずるハンドの位置誤差を吸収するため，視覚センサ情報を用いてマニピュレータの理想把持姿勢を変更することで，ノブの把持動作を行う（8.2節参照）．この時，ノブの位置は空間中に固定されているため，ノブを把持したマニピュレータの姿勢には，ベースロボットの位置誤差情報が含まれる．そこでドアの通り抜け動作を行う前に，このマニピュレータのノブの把持姿勢から，ベースロボットの位置誤差の推定を行うことにした．

さて，ノブを把持した状態におけるベースロボットの推定位置の候補は，マニピュレータの姿勢が一定でも，中心がノブの位置，半径がノブからベースロボットまでの距離とした円の円周上に無数に存在し，特定することができない（図8.18）．そこで，ベースロボットの前後方向の誤差に関しては，ロボットの前面に搭載した超音波距離センサを用いて正確に規定の位置に停止できるという前提で，この位置誤差の推定を行うことにした．以下に，設計した誤差推定アルゴリズムの詳細を示す．

まず，マニピュレータの土台を原点としたグローバル座標系を図8.19に示す通り設定し，ベースロボットの位置及び方向の誤差を $[\Delta X, \Delta Y, \Delta\theta]^T$，マニピュレータの理想把持姿勢からノブの把持姿勢となるまでに動かしたハンドの移動量を $[\delta x, \delta z]$ と表現する．ここで，未知パラメータが $[\Delta X, \Delta Y, \Delta\theta]^T$ の3つであるのに対して，既知パラメータは $[\delta x, \delta z]$ の2

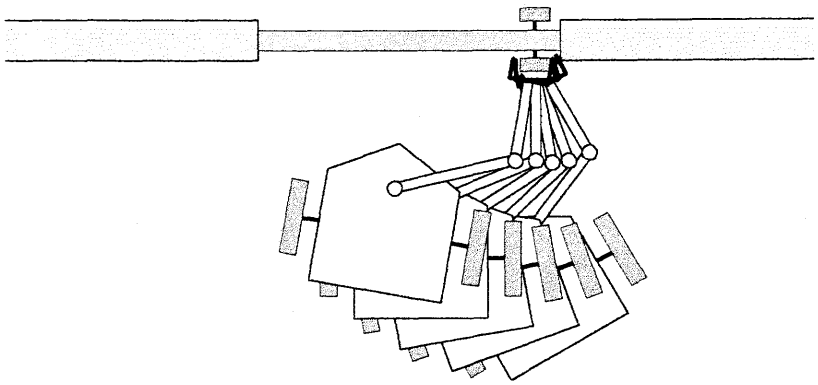

図 8.18: ノブを把持したベースロボットの推定位置の候補

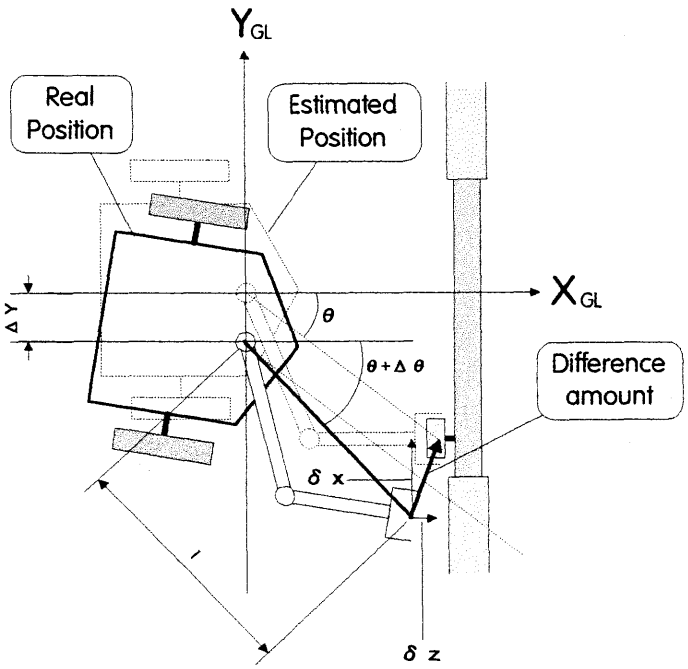

図 8.19: ドアノブの把持姿勢によるベースロボットの位置誤差の推定

つなので，このままではベースロボットの位置誤差を計算することができない．そこでベースロボットの前後方向の誤差は，ドアの前にベースロボットが停止する際に超音波距離センサを用いて修正することにする．これにより，$\Delta X = 0$ となるので未知パラメータは $\Delta Y, \Delta\theta$ の2つとなり，図8.19に示す幾何的拘束より，$\Delta\theta$ が小さい場合，$\Delta Y, \Delta\theta$ は，(8.3)式，(8.4)式によって一意に決定する．

$$\Delta\theta \simeq \sin\Delta\theta = \frac{\delta z - l\sin\theta}{\delta z\tan\theta + l\cos\theta} \tag{8.3}$$

$$\Delta Y = \delta x - l\cos\theta\sin\Delta\theta \tag{8.4}$$

ドアの開閉時におけるベースロボットの位置修正

ノブの把持によってベースロボットの位置誤差が推定できるので，ドアの通り抜け動作中にこの誤差を修正することにした．ただし，ベースロボットの位置に対するマニピュレータの姿勢は予め計画されており，ハンドは常にノブを把持しているため，ベースロボットの位置や方向に対応して，マニピュレータの姿勢を変更することにした．以下に設計した修正法の詳細を述べる．

ベースロボットは，走行制御系，及び位置推定系が独立に，誤差を減少させるように経路修正を行いつつ走行する．この間，ベースロボットの位置，方向，及び計画した協調動作におけるドアノブの位置から，その地点におけるノブを把持するためのマニピュレータの姿勢をディファレンシャルモーションを用いて計算する．この姿勢となるようにマニピュレータを制御することで，ベースロボットが修正経路を走行する間，ハンドはノブを把持したまま開閉動作を続けることができる．

この修正法の有効性を検証するために，簡単なドアの押し開けシミュレーションを行った．このシミュレーション結果の一例を図8.20に示す．ただしこの例では，ベースの初期位置の誤差を，

$$\Delta Y = 2cm, \quad \Delta\theta = 1deg$$

とした．

この例より，ハンドがノブを把持したまま，ベースロボットが位置修正を行う経路を走行することが可能であることが分かった．ただし，このシミュレーションにおけるベースロボットの位置修正を行う経路については，山彦プロジェクトで開発された移動ロボットシミュレータ [KY92] によってジェネレートされた軌跡を用いた．

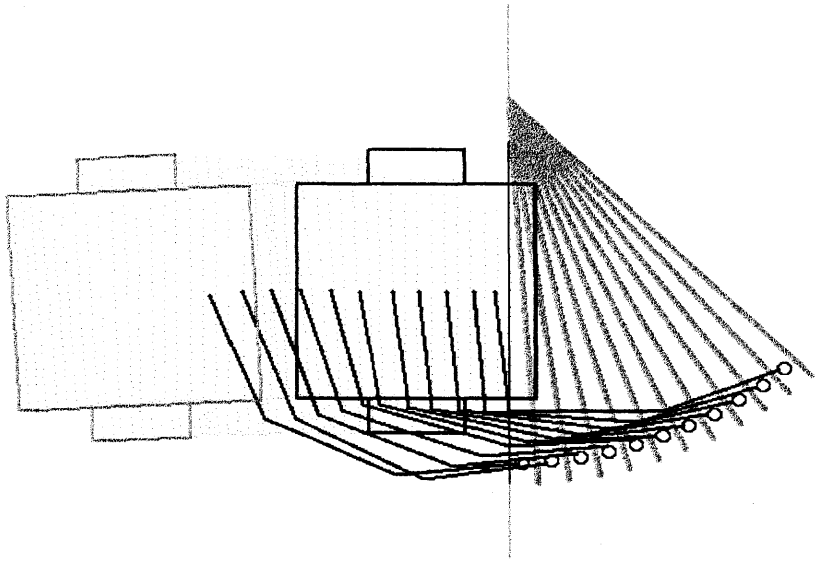

図 8.20: シミュレーション：ベースロボットの位置誤差の修正

本修正法は，計算にディファレンシャルモーションを使用しているため，この修正法が適用できるのは，ハンドを移動させる量が十分に小さい範囲内である．よって，ベースロボットの位置や方向が大きく変化するような場合には対応できない．

鍵のチェック動作

ドアに鍵がかかっている場合，ロボットが鍵を開ける機能を持たない限り，当然ドアを開けることはできない．仮に鍵がかかった状態でドア開け動作を行うと，ノブには大きな力が加わり，ノブまたはロボット自身を破壊する可能性もある．そこで，ドアの開閉動作を実行する前に，人間が行う動作を参考にした鍵のチェック動作（鍵がかかっているかどうかの確認の動作）を行うことにした．通常人間は，ノブを回した状態で少し引き（または押し），ドアが開くかどうかを確かめることで，鍵がかかっているかどうかを判断する．本アクションプリミティブでは，これと同様の動作をロボットに行わせることで，鍵のチェックを行うことにした．具体的には，まずハンドユニットがノブを把持した状態で手先にコンプライアンスを発生させ，ベースロボットをドアが開く方向に少し移動させる．このとき，鍵がかかっていれば，ハンドの位置は動かないため，コンプライアンス制御によってマニピュレータの手先が伸びる (図 8.21-(A))．一方，鍵が開いていれば，ドアが少し開き

ハンドはこれにならう(図8.21-(B)). この時の, ベースロボットに対するハンドの位置を検知することで, 鍵のチェックを行うことができる. 仮に鍵がかかっていた場合, ロボットは, この後のドアの開閉動作を全てキャンセルして, 別の動作を実行することにした.

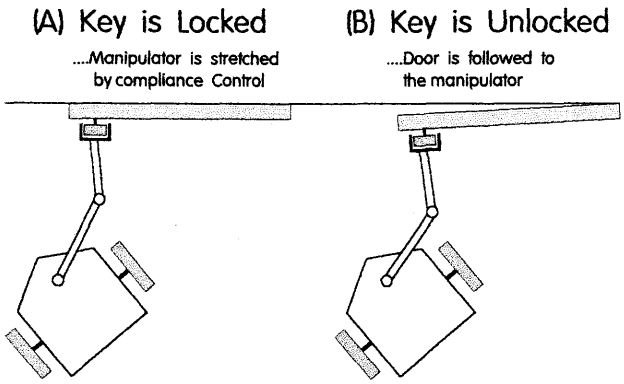

図 8.21: 鍵のチェック動作

8.6.3 アクションプリミティブ「ドアの開閉動作」の実装

8.6.2節で述べた設計に基づいて, アクションプリミティブ「ドアの開閉動作」を, マスタモジュール上の1プロセスとして実装した. 以下に, 実装したこのプロセスの動作の詳細を, 動作開始から時間の経過に沿って述べる.

ドアノブの把持姿勢によるベースロボットの位置誤差の推定法の実装

本プロセスは, 動作開始後, まずノブを把持した状態のマニピュレータの姿勢から, 8.6.2節のベースロボットの位置誤差の推定法の設計にしたがって, ベースロボットの位置誤差を計算する. ただし, 本研究で使用するマニピュレータは自作のため, 5.2.2節でも述べたように, 各関節のギアのバックラッシュが大きく, またマニピュレータのイニシャライズ機能が不十分でもあるため, ハンド座標系上の Z_{Hand} 軸方向の移動量 δz が正しく求まらない. 仮にマニピュレータの先端の位置精度が良ければ, (8.3) 式を用いて$\Delta\theta$ を精度良く推定することができるが, δz に含まれるギアのガタの誤差が $\Delta\theta$ に大きく影響を与えるため, 本実装では $\Delta\theta$の推定を行わないことにした.

鍵のチェック動作の実装

ドア開け動作の場合，動作の前に本プロセスは鍵のチェック動作を実行する．手先のコンプライアンス制御を行うため，コンプライアンス行列を$diag(c_1, c_2, c_3, 0, 0, 0)$と設定し，表7.2のコンプライアンス行列設定コマンドを発行することで，コンプライアンス制御を起動する．次に引き開け動作の場合は，Spurコマンド（表7.3参照）の停止コマンドを発行し，ベースロボットを$5cm$後退させ（押し開け動作の場合は前進），マニピュレータの姿勢情報獲得コマンド（表7.2参照）によってマニピュレータの姿勢を獲得する．この姿勢から，マニピュレータのキネマティクスを計算することで，ベースロボットに対するハンドの位置を計算する．仮に鍵が掛かっていれば，ハンドの位置は通常の位置から大きく移動するため（図8.21（A）参照），この移動量から鍵のチェックを行うことができる．この後，再びSpurコマンド（表7.3参照）の停止コマンドを用いてベースロボットを元の位置に戻し，鍵のチェック動作を完了する．仮に鍵が閉まっていると判断した場合，本プロセスは，統括制御ソフトウエアのプロセスに異常終了のメッセージを送り，動作を終了する．

ドアの開閉動作の実装

次にドアの開閉動作の実装について説明する．まず本プロセスは，ノブの軸に垂直な平面に拘束されたコンプライアンス制御を行うため，コンプライアンス行列を$diag(c_1, c_2, 0, 0, 0, 0)$と設定し，コンプライアンス行列設定コマンド（表7.2参照）を発行することで，マニピュレータのコンプライアンス制御を行う．次に，Spurコマンド（表7.3参照）の速度設定コマンドを用いてベースロボットの速度を$5cm/s$に設定し，走行制御コマンドを用いてベースロボットの目標経路を設定する．これにより，走行制御ソフトウエアは，本プロセスとは独立に，秒速$5cm$というゆっくりとしたスピードで，ベースロボットを目標経路に沿うように走行させる．一方，本プロセスは，POEM関数（表7.4）の推定位置獲得関数を用いて，位置推定モジュールから常時ベースロボットの推定位置を獲得する．ここで得たベースロボットの推定位置とオフラインで計画した協調動作より，ベースロボットに位置誤差が無い場合のマニピュレータの目標関節角度を決定する．また走行中，ベースロボットはノブを把持することで推定した位置誤差の修正を行いつつ走行する．そこで，この修正中にハンドがノブを把持し続けるように，8.6.2節で述べた手法を用いてマニピュレータの姿勢を変更する．

以上に述べた通り，アクションプリミティブ「ドアの開閉動作」を実現するプロセスは，動作中，走行制御ソフトウエア，位置推定ソフトウエア，マニピュレータ制御ソフトウエア

との間で情報交換を行う．この通信は，山彦バスと双方向メモリを用いて実現した．このプロセスが動作するマスタモジュールと各機能モジュール間の情報の流れを図8.22に示す．

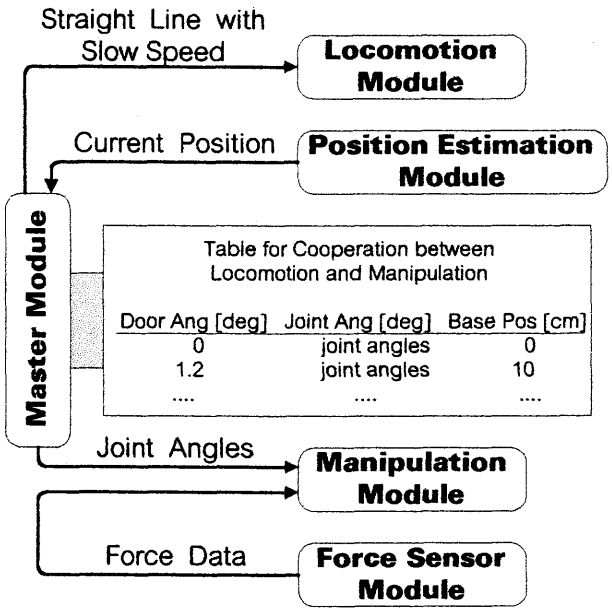

図 8.22: アクションプリミティブ「ドアの開閉動作」におけるモジュール間通信

8.6.4 ドアの押し開け動作実験

実装したアクションプリミティブ「ドアの開閉動作」の性能評価を行うため，移動マニピュレータによるドアの押し開け動作実験を行った．ただし，実験環境は第2章で設定した環境中のドアとした．また，ベースロボットとマニピュレータの動作計画は，図8.16に示したシミュレーションで計画したものを使用した．このロボットによるドアの押し開け動作の様子を図8.23に示す．

この実験では，ベースロボットの後部に目印を付け，ロボットが目標軌跡より$2cm$程度ずれた位置から動作を開始した（図8.23 - (1)において，ロボットから出ている突起がロボットの中心，また地面に示した点線がベースロボットの目標走行経路）．まず，アクションプリミティブ「ノブの把持動作」を実行し，ノブの把持を行った．次に，アクションプリミティブ「ドアの開閉動作」を実行した．ここでロボットは，マニピュレータの姿勢からベースロボットの位置誤差を推定した後，ベースロボットの位置修正を行いつつドアの押し開け動作を行った（図8.23 - (2)～(5)）．

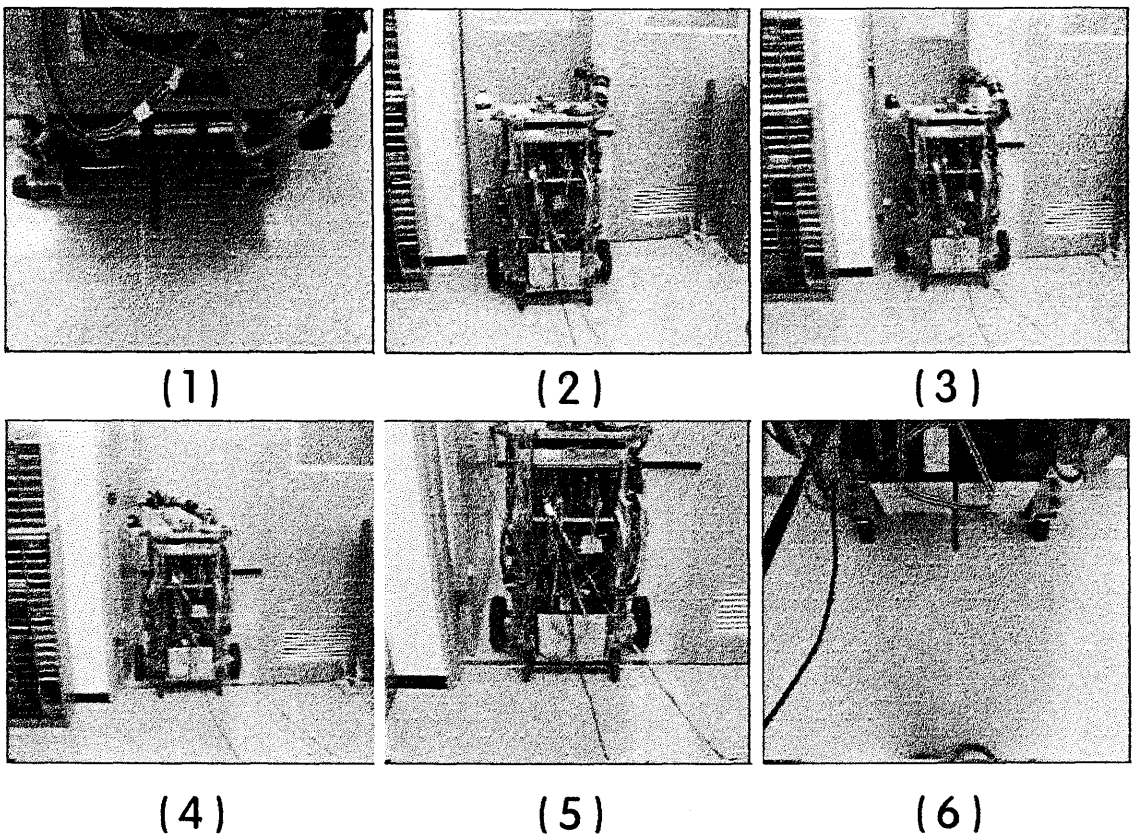

図 8.23: ベースロボットの位置修正動作を伴ったドアの押し開け動作

この図からも分かるとおり，移動マニピュレータによるドアの開閉動作実験は成功した．またドアの押し開け時のベースロボットの位置修正に関しては，この程度の誤差は修正できることが分かった（図8.23 - (6)）．ただし，ベースロボットの位置誤差が大きい場合には，走行制御ソフトウエアの機能によってベースロボットの方向が位置修正を行うために大きく外れ，動作がうまくゆかない実験例もあった．もっとも本研究では，ドアの幅が狭い環境におけるドアの通り抜けを対象としているので，ベースロボットの誤差が数cm以内となるようにドアの前に停止できなければ，もともとドアの通り抜け動作は不可能である．したがって，経路追従走行の時点でベースロボットの大きな誤差は修正する必要があり，この修正がなされた後の，ドアの開閉動作における位置修正法としては，本方式は有用であると考えられる．

8.7 まとめ

本章では，各部分動作を実現するためのソフトウエアの単位であるアクションプリミティブの動作概略について紹介し，実装したアクションプリミティブ「経路追従走行」,「ノブの把持動作」,「ノブのリリース動作」,「ドアの開閉動作」の詳細を説明した．

アクションプリミティブ「経路追従走行」は，ベースロボットを目標経路に追従させるためのソフトウエアである．ここでは，ベースロボットの位置推定にオドメトリを使用するが，ここで生ずる誤差を減少させるため，超音波センサを用いた位置修正を行うことにした．また，アクションプリミティブ「ノブの把持動作」は，ハンドでノブを把持する動作を行うためのソフトウエアである．ここでは，視覚センサ情報をフィードバックしてマニピュレータを制御することで，ノブの把持動作を実現した．また，アクションプリミティブ「ノブのリリース」は，把持したノブをリリースし，マニピュレータを走行の邪魔にならない待機姿勢に戻す動作を行うものである．ここでは，マニピュレータの制御のみで実現した．また，アクションプリミティブ「ドアの開閉動作」は，把持したノブを操作してドアを開閉する動作を行うものである．この動作は，ノブの把持によるベースロボットの位置推定を行った後，マニピュレータの手先のコンプライアンス制御を行いつつ，ベースロボットとマニピュレータの協調動作を用いて実現した．各アクションプリミティブは，マスタモジュール上の1プロセスとして実装を行った．また実装した各アクションプリミティブ毎に動作実験を行い，それぞれのアクションプリミティブが期待通り動作することを確認した．

第 9章

統括制御ソフトウエアの実現

9.1 はじめに

ロボット全体の動作を制御するマスタモジュール上のソフトウエアは，既に第6章でも述べた通り，各部分動作を実現する複数のアクションプリミティブというソフトウエア群と，このアクションプリミティブを統括する統括制御ソフトウエアの2階層で構成した．この内アクションプリミティブについては，第8章で詳しく述べた．本章ではアクションプリミティブを統括するための統括制御ソフトウエアについて詳しく述べる．

さて，ロボットの動作は，部分動作を実行するアクションプリミティブのシーケンスで表現できる．そこで，このアクションプリミティブのシーケンスを計画し，これを順次実行する統括制御ソフトウエアをマスタモジュール上に実装した．これにより，目標タスク「ドアの通り抜けを含む自律ナビゲーション」が実現する．本章では，この統括制御ソフトウエアの動作設計及び実装方法について説明する．

9.2 統括制御ソフトウエアの設計

アクションプリミティブのシーケンスを計画するため，まずロボットの部分動作を実現するアクションプリミティブを単位として，対象とする環境でロボットがとり得る動作全体を有向グラフで表し，これをアクションネットワークと名付けた．ただし，ノードにはロボットの状態を対応させ，またアークにはアクションプリミティブを対応させることにした．このアクションネットワーク上で，目的地までの最適経路をグラフサーチによって求めることで，アクションプリミティブのシーケンスを計画するのだが，この計画はロボットが動作を開始する前にオフラインで行うことにした．統括制御ソフトウエアの内，この

アクションプリミティブのシーケンスの計画を担当する部分を，ここではオフライン動作プランナと呼ぶ．

アクションプリミティブのシーケンスの計画が終了後，計画したシーケンスにしたがって，アクションプリミティブを順次実行することにした．このアクションプリミティブを実行する部分を，ここではオンラインアクションプリミティブ実行部と呼ぶ．実質的なロボットの動作は，各アクションプリミティブが行うため，アクションプリミティブを実行した後，統括制御ソフトウエアはアクノレッジが返ってくるまで動作を休止することにした．

さて，実行したアクションプリミティブの動作が終了すると，統括制御ソフトウエアはアクションプリミティブからアクノレッジを受け取り再び動作を開始する．このアクノレッジが正常終了であれば，オンラインアクションプリミティブ実行部は，計画したアクションプリミティブのシーケンスにしたがって，次のアクションプリミティブを実行する．これを繰り返すことで，計画したシーケンスに沿って順次アクションプリミティブが実行され，目標動作が実現される．

一方，アクノレッジが異常終了であれば，計画したアクションプリミティブのシーケンスの実行は不可能である．そこで，問題となったアクションプリミティブに相当するアークのパラメータを変更し，このアークの始点のノード（現在のロボットの状態に相当）をスタートノードとして，再びグラフサーチを行うことで，アクションプリミティブのシーケンスを再計画することにした．この後，新たに計画したこのシーケンスに沿って，アクションプリミティブの実行を繰り返し行うことで，目標動作が実現される．

以上の統括制御ソフトウエアの動作手順とソフトウエアの構成を図9.1に示す．本節では，このソフトウエアの構成要素であるアクションネットワーク，オフライン動作プランナ，及びオンラインアクションプリミティブ実行部の詳細について以下に述べる．

9.2.1 アクションネットワーク

前述した通り，ロボットの動作を実現するために，アクションプリミティブを単位として，対象とする環境中でロボットがとり得る動作全体を有向グラフで表現し，これをアクションネットワークと呼ぶことにした．したがって，このグラフにおける各ノードは，ロボットの動作の分岐点におけるロボットの状態を表し，また各アークはアクションプリミティブを表す．また各アークには，アクションプリミティブを実行するために必要な動作パラメータおよび動作コストといった属性を付加することにした．

ここでは，このアクションネットワークを，具体例を挙げて説明する．閉じているドア

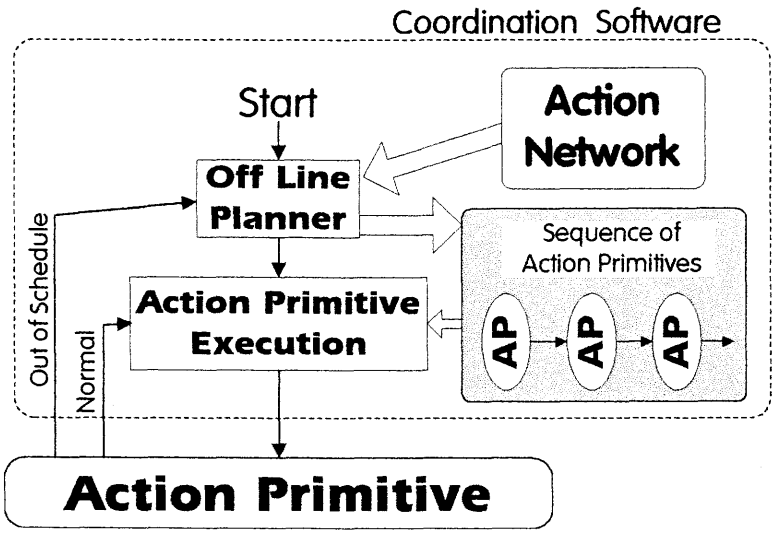

図 9.1: 統括制御ソフトウエアの構成と動作手順

のノブをハンドが把持した状態（これを状態1とする）からは，ロボットが行う動作には，ドアを開ける動作とノブをリリースする動作の２つの動作があり得る．ドアを開ける動作を行った場合，ロボットは開いたドアのノブをハンドが把持した状態（状態2）となる．一方，ノブのリリース動作を行った場合，ベースロボットは動かずにマニピュレータは待機状態（状態3）となる．これをアクションネットワークで表現すると，各状態（状態1-状態3）はそれぞれノードとなる．また，状態1から状態2に向かうアークは，アクションプリミティブ「ドアの開閉動作」を表す．このアークには，ドア開けを行うための動作パラメータ，及び動作コストといった属性を付加する．一方，状態1から状態3に向かうアークは，アクションプリミティブ「ノブのリリース」を表し，属性として動作コストを付加する．以上の要領で表現した，ドアの通り抜け動作に関するアクションネットワークを図9.2に示す．

さて，各アークに属性として付加する動作コストは，一般的に決まらない．そこでこの動作コストを，(1) アクションプリミティブが「経路追従走行」であれば走行距離に伴って増加，(2) ランドマークを用いてベースロボットの自己位置修正を行うことができる「経路追従走行」の動作コストは小，(3)「ノブの把持動作」や「ドアの開閉動作」は困難なのでコストは大，といったヒューリスティックを用いて決定した．

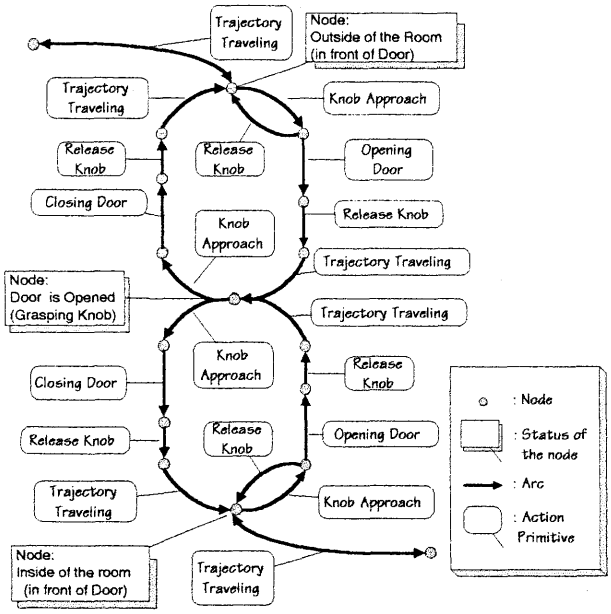

図 9.2: アクションネットワーク

9.2.2 オフライン動作プランナ

オフライン動作プランナは，統括制御ソフトウエアの中で，ロボットの動作を始める前にアクションプリミティブのシーケンスを計画する部分である．アクションネットワークは，9.2.1節で述べた通りアクションプリミティブを表すアークの接続で構成されるため，アクションプリミティブのシーケンスの計画は，アクションネットワーク上の経路計画を行うことで実現した．この経路計画の手法を以下に示す．

まず，ベースロボットの現在位置をスタートノード，目的地をゴールノードとしてアクションネットワークに追加する．次に，これらのノードと付近のノード（ただし，マニピュレータが待機状態のもの）を，アクションプリミティブ「経路追従走行」を表すアークで接続し，アクションネットワークを拡張する．この拡張したネットワーク上で，スタートノードを始点としたグラフ探索を行い，スタートノードからゴールノードまでの動作コストの合計が最小となる経路を探索する．この経路探索には，ダイクストラのアルゴリズム[E.W59]を使用した．ここで計画されたアクションネットワーク上の経路が，ロボットの動作を表すアクションプリミティブのシーケンスである．

9.2.3 オンラインアクションプリミティブ実行部

オンラインアクションプリミティブ実行部（以下オンラインAP実行部と表記する）は，統括制御ソフトウエアの中で，計画したアクションプリミティブのシーケンスにしたがってアクションプリミティブを順次実行する部分である．このオンラインAP実行部の動作イメージを図9.3に示し，以下にこの動作について説明する．

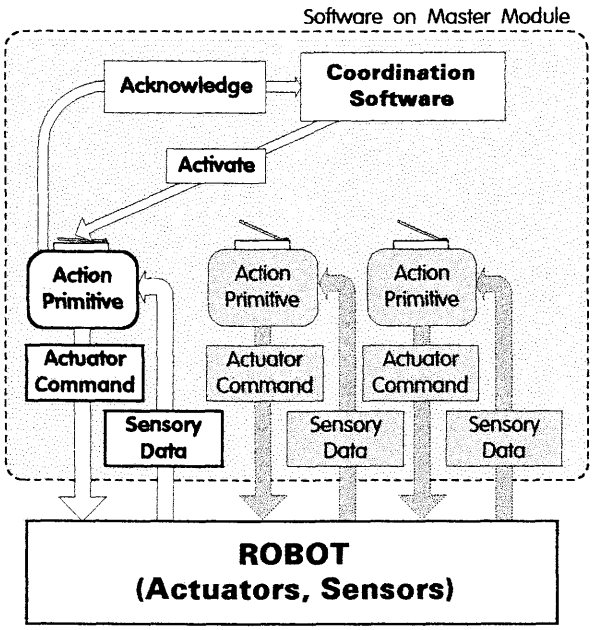

図 9.3: オンラインアクションプリミティブ実行部の動作イメージ

アクションプリミティブの実行後，アクションプリミティブが終了しアクノレッジが返ってくるまで，このオンラインAP実行部は待機する．アクションプリミティブから返ってきたアクノレッジが正常終了であれば，統括制御ソフトウエアは再びアクションネットワーク上の計画した経路を辿り，順次アクションプリミティブを実行する．一方，アクノレッジが異常終了の場合，まず対応するアクションネットワーク上のアークのコストを変更する．例えばドアに鍵がかかっていることを検知した場合，このアークの動作コストを十分に大きくし，アクションプリミティブ「ドアの開閉動作」を表すそのアークを抹消する．ここで，アークの基点となるノードは，ロボットの現在の状態を表しているので，このノードをスタートノードとし，再びグラフサーチを行う．これにより，動作コストを変更したアークを通らない，新たなアクションネットワーク上の経路が計画される．この経路にし

たがって，順次アクションプリミティブの実行を行う．

9.3 統括制御ソフトウエアの実装

設計した統括制御ソフトウエアは，マスタモジュール上の1プロセスとして実装した．以下に実装した本プロセスの動作の詳細について述べる．

アクションネットワークには，予めスタートノードとゴールノードが加えられた，拡張されたものを使用し，これをC言語のポインタ接続で表現した（ノード及びアークを付け加えるアルゴリズムは現在実装していない）．この表現を図9.4に示す．このため，現在はロボットの動作開始位置及び目的地が，このアクションネットワークを搭載した時点で決定される．

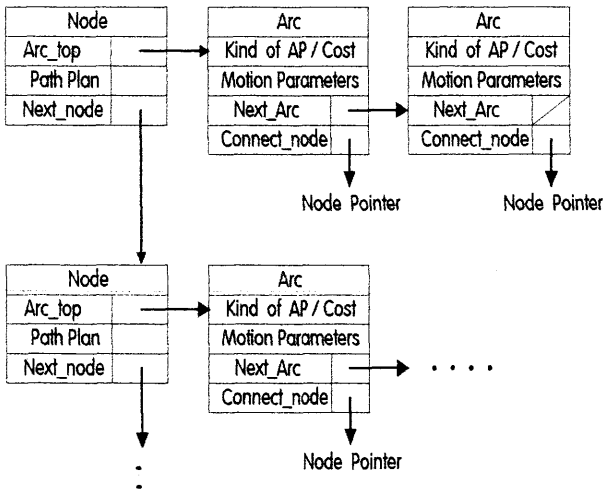

図 9.4: アクションネットワークのポインタ表現

統括制御ソフトウエアのプロセス(以後統括制御プロセスと表記する）は，動作開始後，アクションネットワーク上で動作コストの合計値が最小となる経路を探索し，その経路情報をアクションネットワーク上に記入する（図9.4の Path Plan の項目に通過するアークのポインタを記入する）．次に，アクションネットワーク上のスタートノードから計画した経路を辿ってゆき，通過したアークに対応するアクションプリミティブを実行する．この実行は，各アクションプリミティブに対応するプロセスに，アークの属性である動作パラ

メータを付加したメッセージを送信することで実現する．この後，統括制御プロセスをメッセージ待機状態にすることで，動作主体をアクションプリミティブのプロセスに移す．

実行されたアクションプリミティブのプロセスは，担当する動作が終了後，統括制御プロセスに終了メッセージを送る．メッセージを受けた統括制御プロセスは，メッセージが正常終了か異常終了かを判断し，正常であれば次のアクションプリミティブの実行，異常であれば動作の再計画のため，オフライン動作プランナを実行する．

以上の動作によって，予定外の状況にも対応した移動マニピュレータのドアの通り抜けを含む自律ナビゲーションが実現される．

9.4 まとめ

本章では，統括制御ソフトウエアの設計及び実装について述べた．この統括制御ソフトウエアは，本研究で考案したアクションネットワークを利用してアクションプリミティブのシーケンスを計画し，このシーケンスにしたがってアクションプリミティブを順次実行するものである．このシーケンスの計画は動作前にオフラインで行い，アクションプリミティブの実行は順次オンラインで行うことにした．また動作中に問題が生じた場合でも，ロボットの動作を一旦中止し，アクションネットワーク上の動作コストを書き替え，オフラインでアクションプリミティブのシーケンスを再計画するといった，予定外の状況にも柔軟に対応できる動作を行うソフトウエアを実現した．

以上，第5章から第9章によって目標タスクを実現するために構築したロボットシステムの詳細を述べた．第10章では，このロボットシステムを用いて，実環境内で行ったナビゲーション実験について報告する．

第 10 章

ナビゲーション実験

10.1 はじめに

第4章から第9章では，目標タスク「ドアの通り抜け動作を含む自律ナビゲーション」を実現するための，ロボットシステムの構築について述べた．これらのハードウエア及びソフトウエアを全て5.2.1節で述べたベースロボット「山彦てん」上に搭載し，ドアの通り抜けを含むナビゲーション実験を，経路を幾つか変更して行った．ただし，これらの実験は，全て2.5節で設定した我々の研究室（筑波大学3L402号室）及びこの部屋に接続する廊下という実環境を使用した．本章では，数多く行った実験の中から2つの経路について結果を報告し，実験の検討を行う．

10.2 ナビゲーション実験

10.2.1 実験1：ドアの通り抜けを含む自律ナビゲーション実験

まず，ドアの開閉動作及びベースロボットの走行実験を行った．この実験の具体的なタスクは，ロボットが，部屋の中よりドアを開閉して廊下に出て，廊下を走行し，部屋のもう片方のドアを開閉して再び部屋の中へ入るという一連の動作を実現することである．

アクションプリミティブのシーケンスの作成

この実験では，ロボットの動作開始位置は，研究室内の片方のドアの入口，目的地はもう片方のドアの入口となる．また走行経路を廊下に限定するため，この実験におけるアクションネットワークを，図10.1に示すように単経路のネットワークとした．このため，統

括制御ソフトウエア上で計画されるアクションプリミティブのシーケンスも，図10.1に示されるアクションネットワークと同一となる．ただし，この図中，ノブのリリース，及び右側のドアにおけるノブのアプローチに関しては表示を省略した．

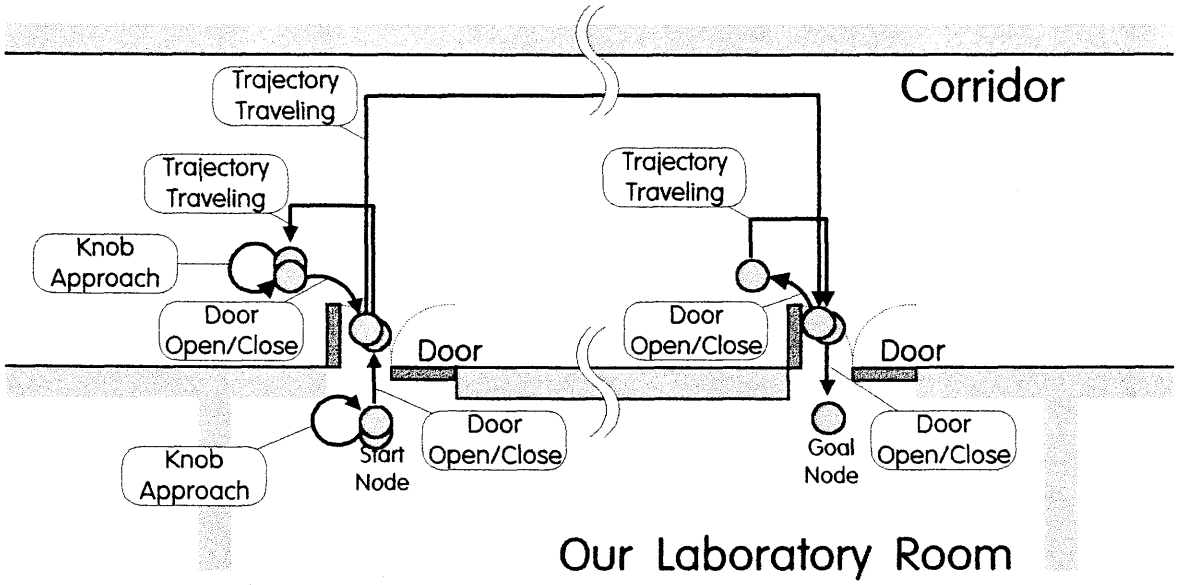

図 10.1: 実験1の実験環境及び使用するアクションネットワーク

実験結果

このアクションネットワークを用いて統括制御ソフトウエアを起動したところ，アクションプリミティブは図10.1に示す順に動作し，これにしたがってロボットの部分動作は順次実現され，ロボットは目的地である反対側のドアの内側に到達した．この実験の模様を連続写真で図10.2に示す．ただし，アクションネットワークが単経路のネットワークであるため，ドアに鍵がかかっている状況では，ロボットは動作変更ができずに，その場で動作を終了した．

10.2.2 実験2：動作の分岐が存在する自律ナビゲーション実験

次に，ロボットの動作に分岐が存在するアクションネットワークを使用したナビゲーション実験を行った．この実験の具体的なタスクは，幾つか走行可能経路がある中で，ロボットが廊下中のある地点から，部屋の中のある地点に到達することである．

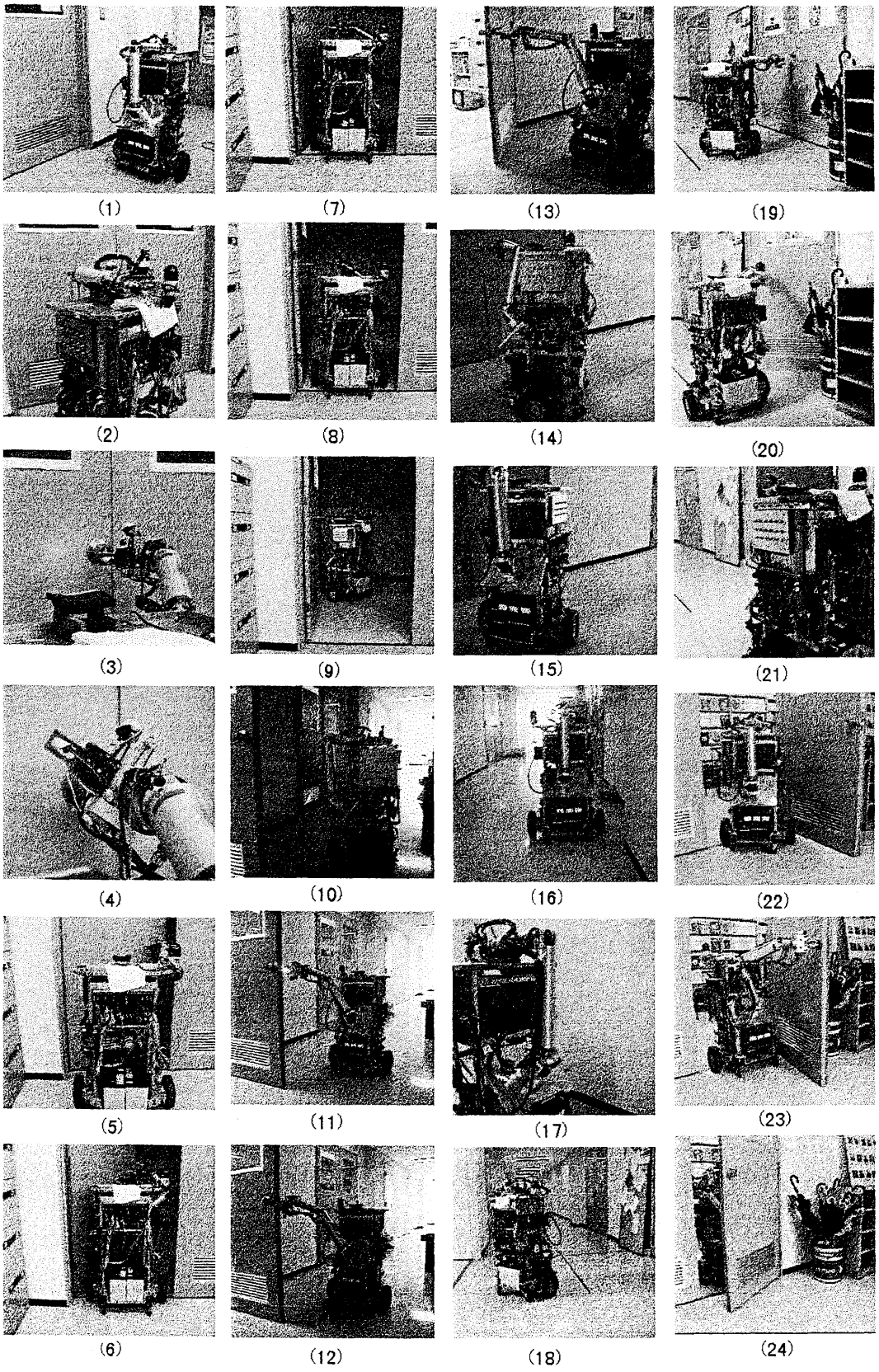

図 10.2: 実験1のドアの通り抜けを含むナビゲーション動作

この実験では，廊下中のある位置をロボットの動作開始地点，部屋の内側のある位置を目的地と設定した．この状況でアクションネットワークを構成し，ロボットの走行経路と動作の計画及びナビゲーション実験を行った．

アクションネットワーク

本実験では，アクションネットワークを図10.3に示す通り構成した．ただし，ドアを通り抜ける際のノード及びアークについては，表示が煩雑になるのを避けるため，図10.3中には主要な環境中のベースロボットの位置を示すノードのみ表示した．

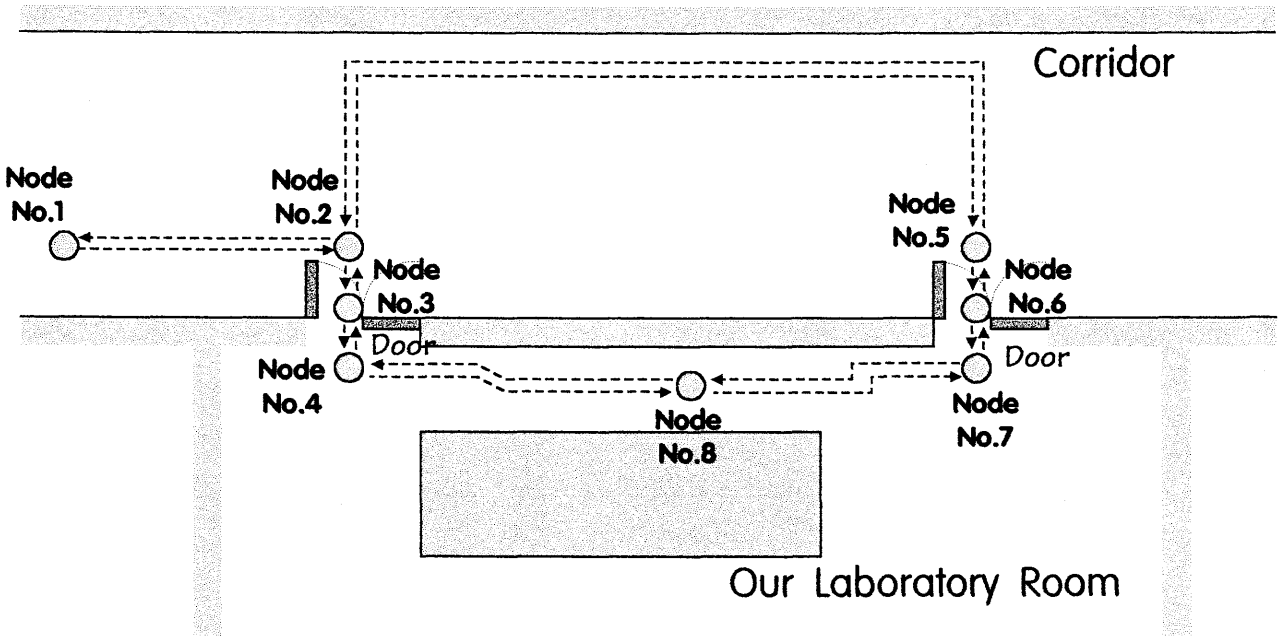

図 10.3: 実験2の実験環境および使用するアクションネットワーク

アクションプリミティブのシーケンスの作成

図10.3に示すアクションネットワークを用いて統括制御ソフトウエアを実行したところ，表10.1に示すアクションプリミティブのシーケンスが計画された．また，図10.3中のノード番号の順では，“1-2-3-4-8” という，ゴールに近い経路が計画された．

図10.3中の ノード番号		アクション プリミティブ			
ノードNo.1	→	経路追従走行	→		
ノードNo.2	→	ノブの把持	→	ドアの開閉（引き開け）	→
	→	ノブのリリース	→	経路追従走行	→
ノードNo.3	→	ノブの把持	→	ドアの開閉（引き閉め）	→
	→	ノブのリリース	→		
ノードNo.4	→	経路追従走行	→		
ノードNo.8					

表 10.1: 実験2において計画されたアクションプリミティブのシーケンス

実験結果

計画したアクションプリミティブのシーケンスにしたがって，統括制御ソフトウエアは順次アクションプリミティブを実行し，ロボットは目的地であるノードNo.8の位置に到達した．

一方，図中のノードNo.2においてドアに鍵がかかっていた場合，ドアの開閉動作（引き開け）を行う時点で鍵のチェック動作を行って状況を検知し，このドアを開けることを中止した．この後，ノードNo.2をスタートノードとしたアクションプリミティブのシーケンスの再計画を行った．この結果，表10.2に示すアクションプリミティブのシーケンスが計画された．また，図10.3中のノード番号の順では，新たに “2-5-6-7-8” となった．このアクションプリミティブのシーケンスにしたがって，再び統括制御ソフトウエアは，順次アクションプリミティブを実行し，ロボットはもう片方のドアを通り抜けて，目的地に到達できた．

また，もう片方のドアにも鍵がかかっていた場合，ロボットが目的地に到達する経路が存在しないため，ロボットはその時点で動作を終了した．

10.2.3 実験に対する検討

以上に示した実験を繰り返し行った結果，ドアの通り抜けを含むナビゲーションの成功率は8割程度であった．このナビゲーション動作は，以下に示す状況においても成功した．

1. ベースロボットの初期位置の誤差

図10.3中の ノード番号		アクション プリミティブ			
ノードNo.2	→	ノブのリリース	→	経路追従走行	→
ノードNo.5	→	ノブの把持	→	ドアの開閉（引き開け）	→
	→	ノブのリリース	→	経路追従走行	→
ノードNo.6	→	ノブの把持	→	ドアの開閉（引き閉め）	→
	→	ノブのリリース	→		
ノードNo.7	→	経路追従走行	→		
ノードNo.8					

表 10.2: 実験2において再計画されたアクションプリミティブのシーケンス

ベースロボットの初期位置のずれが数センチメートルある場合でも，走行中に環境中の平らな壁を検知し，ロボットが有するランドマークモデルと比較することで位置修正を行い，ドアを通り抜けて目的地に到達した．

2. マニピュレータのイニシャル姿勢の誤差

ベベルギアのガタのために，マニピュレータの初期姿勢にも誤差が生じ，このためハンドの先端の位置にも誤差が生じてしまった場合でも，ハンド上に搭載した視覚センサによってノブとハンドの相対位置を認識することで，ノブを把持することができた．

3. 環境の明るさの変動

ハンドにライトを搭載することで，昼間，夜間を問わず，ノブを把持することができた．

4. 予定外の障害物

ロボットの走行経路上に障害物が存在する場合，ロボットは超音波距離センサを用いてこれを認識し，走行動作を中断することで，障害物との衝突を回避することができた．

また，動作中問題が生じたとき，例えばノブをしっかり把持していないときなどに，把持の手助けをするなどのサポートを人間が行うことで，9割以上の成功率で目標動作は達成された．

さて，人間のサポートがない場合，本ナビゲーションは2割程度目標位置に到達できずに失敗したが，この原因は，そのほとんどがベースロボットの推定位置の誤差に因るものであった．以下に，このナビゲーションが失敗したときの考えられる原因とその状況を列挙する．

1. ベースロボットの推定位置の誤差—その1

   ベースロボットが長距離走行した後に生ずる推定位置の誤差のため，ノブを把持する際にノブが視覚センサの視野外となり，ハンドユニットはノブを把持することができなかった．

2. ベースロボットの推定位置の誤差—その2

   環境中の平らな壁を用いてベースロボットの位置修正を行うため，壁際に人や予定外の物体などがある場合，位置修正が失敗し，ベースロボットは目標経路から逸れた．

3. ベースロボットの推定位置の誤差—その3

   ドアを開ける際に駆動輪がスリップしてベースロボットの方向が変わり，このためドア開け動作が成功した後の経路追従走行に失敗した．

4. ベースロボットの推定位置の誤差—その4

   ドア開け動作が終了した際，ベースロボットの推定位置の誤差のためにドアの開き角度が90度に満たなくなり，ノブをリリースした直後にバネの力によって開けたドアが閉まった．

5. ノブの把持の問題—その1

   ドアの押し開け動作中，ハンドユニットがノブをしっかり把持していない場合に，ハンドユニットがドアの端に接触し，ロボットはドアの押し開け動作に失敗した．

6. ノブの把持の問題—その2

   ドアの引き開け動作中，ハンドがノブをしっかり把持していないときに，ハンドがノブから離れ，ドアを引き開けている最中にドアが閉まった．

7. ノブの把持の問題—その3

   ノブを回転する際にハンドの内側が滑り，ドアの支えが外れなかったため，ドアに鍵がかかっていると誤判断をして，ロボットは動作を終了した．

8. マニピュレータの動作計画の問題

   ノブのリリース時にマニピュレータの先端がノブやドアに接触し，動作の続行が不可能となった．これは，ノブのリリースに関するマニピュレータの動作計画が練られていない点に原因がある．

9. 自作のマニピュレータ固有の問題

   自作のマニピュレータは，自動イニシャライズ機能が無いため，マニピュレータの初期姿勢は，現在人間が目測で設定している．また，ベベルギアを使用したマニピュレータの関節のガタも大きい．これらが原因で，マニピュレータが計画通りに動作しないことがあり，このためノブの把持動作やドアの開閉動作が失敗した．

10. 経路地図の誤差の問題

    ベースロボットが走行するための経路地図は，人間の手によって制作されるため，ここには誤差が存在する．この誤差が大きい場合，ベースロボットはスリップや地図の凸凹が無くても，期待通りの経路を走行することができず，結果としてベースロボットの推定位置の誤差を生じ，様々な問題の原因となった．

ナビゲーションが失敗するときは，以上に挙げた状況が重なった場合に起こることが多い．

10.3 タスクの達成度の評価

前節の「実験に対する検討」で示した通り，本研究で実現したナビゲーション動作は，まだ幾つかの問題点及び改良すべき点を含むが，成功率8割という数字は十分に高いと考えられる．特に，ベースロボットの位置やマニピュレータの関節角度に誤差が存在する場合も，センサを用いてこの誤差を推定・修正を行い，ロボット自身の力で目的地に到達することができた．

また，ロボットはドアを開け通り抜けた後にそのドアを閉めることで，目標タスクの要求事項である環境状態を変えないという点をクリアした．

また，走行中前方を常に監視し予定外の障害物が存在した場合にベースロボットが緊急停止すること，鍵がかかっている場合には走行経路を変更することなど，予定外の状況に対処するといった目標タスクの要求事項も満たした．

以上より，目標タスク「ドアの開閉動作を含む自律ナビゲーション」は，本ロボットシステムによって，ほぼ達成した（9割5分程度）と筆者らは考えている．

また，実験で用いたアクションネットワークは比較的小さいため，ロボットが移動できる範囲は狭く，現在の動作のバリエーションも少ない．しかし，このアクションネットワークを広げてゆくことで，ドアを含む屋内環境の広いエリアを網羅したナビゲーションが可能となる．また，他のアクションプリミティブを追加することで，動作のバリエーションも広がる．

10.4 まとめ

構築したロボットシステムを用いて，実環境内でのロボットのナビゲーション実験を行った．この実験を行った結果，動作の成功率は8割程度であり，目標タスクはほぼ達成できたと考えられる．また，実験に対する検討を行った結果，ナビゲーション動作が失敗する際の最大の原因はベースロボットの推定位置誤差であることが分かった．

第 11 章

本研究に対する検討と評価

11.1 はじめに

ナビゲーション実験を行った結果，目標タスク「ドアの通り抜けを含む自律ナビゲーション」をほぼ達成したことにより，タスクオリエンテッドアプローチの下で行われた本研究は一旦終了した．本章では，タスクオリエンテッドアプローチで行った本研究を，もう少し広い立場から見ることで，移動マニピュレータやロボットの自律動作に関する問題点や得られた知見に関する検討を行うことにする．

11.2 移動マニピュレータの自律動作に関する問題

まず，目標タスクを実現する過程で確認された移動マニピュレータに関する問題点を以下に列挙する．

- **ベースロボットの推定位置誤差の問題**

  移動ロボットのナビゲーションを行う際に問題となるのが，走行に伴って生ずる推定位置の誤差である．特に，移動マニピュレータという土台を物理的に固定しないロボットで物体を扱う場合，この推定位置の誤差によって様々な問題が生ずる．

  この問題に対処するため，本研究では，超音波距離センサを用いてランドマークである壁の情報を獲得し，ロボットが有する壁の位置情報と比較することで，オドメトリで推定した位置誤差を減少させることにした．また，マニピュレータでドアノブを把持する際には，ハンドに取り付けた視覚センサを用いてノブの位置を認識することで，ハンドの位置誤差を減少させることにした．

ここに挙げた誤差の修正方針を用いたナビゲーション実験では，ベースロボットのオドメトリの精度が比較的良く，また超音波距離センサを用いた位置修正も有効に働いたため，ベースロボットは期待通りの誤差の範囲内（$2[cm] - 3[cm]$ 以内）でドアの前に停止することができた．また，ノブの把持動作については，ベースロボットの停止位置の誤差が多少大きくても，ノブが視覚センサの視野に入っている限り，視覚センサを用いた把持動作によって，ほぼ間違いなくノブを把持できた．

しかし，この実験を通じ明らかとなった移動マニピュレータの最大の問題は，やはりベースロボットの位置誤差であり，ナビゲーション動作が失敗する場合は，その大部分がベースロボットの位置誤差に因るものであった．特に小さな誤差が他の誤差を生み出し，結果として失敗するという例が目立った．例えば，ハンドがノブを把持した状態で，ベースロボットの位置に生じた小さな誤差が，マニピュレータのコンプライアンス制御を行っていない方向に力をかける場合がある．この場合，この力によって駆動輪がスリップして更に誤差が大きくなり，その後のナビゲーションが失敗した実験例もあった．この他のナビゲーション実験における失敗例は，10.3節の「実験に対する検討」に示したが，これらの失敗例からも分かる通り，移動マニピュレータのロバストな制御には，より正確な位置推定が不可欠であることが明らかとなった．極論すれば，移動マニピュレータの問題は，**ベースロボットの位置誤差の問題だけである**という印象さえ受けた．

- **物体を操作する際に生ずる力の問題**

移動マニピュレータで拘束された対象物を操作する際には，ベースロボットの推定位置の誤差のため，把持した対象物に大きな力が生ずるという問題がある（この問題は，ベースロボットの位置誤差に関する問題の一部であるとも言える）．本研究では，拘束されたノブを操作する際，力センサを用いたアクティブなコンプライアンス制御を行うことで，この問題を解決した．ただし実験中，このコンプライアンス動作によってマニピュレータの姿勢が変化した場合に，マニピュレータやハンドが，環境中の別の物体（例えばドアを押し開ける際のもう片側のドア）に衝突する場合もあった．このような例は，反射レベルのロボットの動作が動作計画レベルに影響する問題なので解決が難しく，本研究では考慮することができなかった．今後，ロボットが狭い環境で動作する場合，反射動作も考慮したロボットの動作計画法を考案する必要があると考えられる．

- **ハンドによる把持の問題**

  10.4節の実験に対する考察でも述べた通り，ハンドがノブをしっかり把持していないときに指の内側でノブが滑り，ドア開けの動作が失敗することがあった．本研究では，この指の内側に滑りにくいゴムを使用するなどの工夫をして，この問題に対処したが，今後確実な把持動作を目指すためには，そのためのセンサを用いて把持の確認を行うなど，全ての動作が予定通り進行していることを確認しつつ動作を進めてゆく必要がある．

- **画像処理時の輝度の問題**

  ロボットが，照明の明るい場所で動作を行う場合には考慮する必要はないが，廊下など比較的照明の暗い場所で動作を行う場合（特に夜間），視覚センサから得られる画像は全体的に暗くなり，輝度値の差が小さくなるため，画像処理を行う上で問題となる．これが原因で，実験の初期の段階では，ノブの把持動作に失敗することが多かった．この問題を解決するため，本研究では，視覚センサの上部に搭載したライトを点灯し，画像上の輝度値の差を大きくする工夫を行うことにした．これにより，照明の暗い環境でも，ロボットはノブの把持を行うことができるようになった．このことからも，動作範囲の広い移動マニピュレータには，視覚センサを活用するため，ロボット本体に照明用ライトを搭載することが有用であることが明らかとなった．

11.3 ハードウエアの構成に関して得られた知見

次に，本タスクを実現する過程で得られた，移動マニピュレータのハードウエアの構成に関する知見について以下にまとめる．

- 小型マニピュレータの重要性

  本研究で開発した小型マニピュレータには，視覚センサ，及び力センサを搭載し，各センサ情報をフィードバックすることで，ベースロボットやハンドの位置誤差を吸収し，ノブの把持動作やドアの開閉動作を実現した．

  この研究を通じて，移動マニピュレータを使用する研究には，画像処理機能，コンプライアンス制御機能を有する小型マニピュレータが必要不可欠と考えられる．今後，視覚センサ，及び力センサを搭載した小型マニピュレータを，研究のプラットフォームとして開発することは，非常に意義のあることと考えられる．

- マニピュレータの関節の構成法

  ロボットに搭載する小型マニピュレータは，その動作範囲を広げるため，できるだけスリムに設計することが望ましい．そこで本研究では，第3–第6関節のDCモータは，マニピュレータのリンクの向きに縦に配置した．このうち，モータ軸と関節の回転軸が同軸上にある関節では，ハーモニックギアを用いてモータの回転を減速することにした．一方，モータ軸と関節の回転軸が直交している関節（人間の肘に相当する関節など）では，モータの回転をプラネタリギアを用いて減速し，スパイラルベベルギアを用いてこの回転方向を変えることにした．これによりスリムなマニピュレータを製作することができたが，一方でベベルギアの噛み合わせにおいて大きなバックラッシュが発生し，マニピュレータの先端の位置精度が著しく低下した．しかも動作を行うにつれて各パーツは摩耗することが予想されるため，このバックラッシュは今後増大すると考えられる．これに対し，減速の最終段にハーモニックギアを用いた関節には問題が起きていない．

  以上より，マニピュレータを設計する上で，減速の最終段をベベルギアで組むことには，大きな問題があるという点が明らかとなった．今後マニピュレータを製作する際には，全ての関節において減速の最終段にハーモニックギアを使用することが望ましいと考えられる．

- 小型ハンドの必要性

  ドアの押し開け動作時に，ハンドはノブを把持する必要があるが，このような人間が操作する対象物体は，人間の手のサイズに合せて設計されている．このため，実環境で作業するためのロボットのハンドのサイズは，できるだけ人間の手に近いことが望まれる．本研究では，製作したハンドがDCモータで駆動されるために，ハンドの全長は長くなり，このため動作計画を行う際に非常に苦労した．よって，小型ハンドの設計・製作も，移動マニピュレータにおける今後の研究課題の一つである．

- 機能分散コントローラの有用性

  本研究で使用する移動マニピュレータのコントローラには，山彦プロジェクトで提案された機能分散コントローラを採用した．これにより，既に山彦プロジェクトで研究開発が行われた移動ロボットの走行システムやセンサシステムなどの各機能をそのまま利用することができ，システム全体の開発時間を大幅に短縮することができた．また，新たに設計及び製作を行ったマニピュレータ制御を実現するためのコントローラ

についても，機能分散コントローラによって，他の機能とは独立に動作チェック及びデバッグを行うことができた．

以上より，本研究の様に多くの機能を有するロボットを開発する場合，研究開発を逐次的に行うために，ロボットのコントローラを機能分散で構成することが有効であり，この機能分散コントローラの有用性が本研究でも確認された．

11.4 ロボットの動作設計に関する知見

次に，本タスクを実現する過程で得られた，移動マニピュレータの動作設計に関する知見について以下にまとめる．

- ソフトウエアを時間の経過に沿った部分動作に分割して設計開発を行うことの有効性

  ロボットの複雑な動作を，複数の動作に分割して整理し，個々の動作を実現することで全体の動作を目指す方針は一般的である．しかし，本研究にも示したマニピュレータとベースロボットが協調して動作を行うようなシステムの場合，それぞれの機能毎に動作を分割することは不可能である．そこで本研究では，目標動作を時間の経過に沿った部分動作に分割し，これをアクションプリミティブと名付けて動作ソフトウエアの単位とした．これにより，各アクションプリミティブ内の構成は多少複雑となるが，ロボットの動作全体はアクションプリミティブのシーケンスという単純な構造で表現でき，動作のプランニングやリプランニングが容易に実現できる．また，新たなアクションプリミティブの設計及び実装を行うことで，現在のロボットの動作に新たな動作を加えることもできる．このように本研究では，時間の経過に沿った複雑な動作の分割が，動作計画を行うために，極めて有効に働くことが確認された．

- 動作計画と実行を分けることの利点

  反射を用いたリアクティブな動作などと比較すると，本研究で採用したモデルベースで動作計画を行い，これを実行するという手法は，動作設計を行う際に非常に手間がかかる．しかし，実環境においてロボットが状況を把握しつつ逐次動作を行うためには，現在のセンサの性能及びコンピューティングパワーでは足りない．したがって，本タスクに代表されるように実環境で複雑な動作を実現するためには，やはりモデルベースで動作計画を行って実行するという手法をとらざるを得ず，できるだけ丁寧な環境の記述がここでは重要である．ただし，動作のロバスト性を向上させるため，コ

ンプライアンス制御などのローカルな部分におけるリアクティブ性は，ロボットの動作には不可欠である．

- 移動マニピュレータの動作設計に関する印象

  目標タスクを実現する上で最も苦労したのは，実環境において各基本動作を正確に，またロバストに動作させることであった．一方，ドアの開閉時のマニピュレータとベースロボットの協調動作の計画については，計画を行うにあたり試行錯誤を必要としたが，一度オフラインで動作計画を行った後は，常にその計画を使用することができた．このため，このような複雑な動作を実現する場合，経路や動作の計画問題より，各部分動作を如何にロバストに，また正確に実現できるかどうかが，動作成功の鍵であるという印象を受けた．

11.5 研究方針について

最後に，本研究で用いた研究方針の利点について考察する．

1. タスクオリエンテッドアプローチによる研究の有用性

   タスクを明確に設定して研究を行った結果,「ドアの通り抜けを含む自律ナビゲーション」という自律的な動作を比較的短期間のうちに実現することができた．またこの研究を通じて，移動マニピュレータによる自律動作を実現する上での問題点の発見や，考案したアルゴリズムの有用性の検証，今後このような自律動作を実現するための課題の提案などを行うことができた．

   また，本研究で設定したような複雑なタスクは，プリミティブな動作を少しずつ積み上げてゆき，そこから自然と実現されてゆくものではない．むしろ目標タスクを設定し，これを実現する動作を分割して，個々の部分動作を丁寧に実現することによって初めて複雑な動作が可能となる．このような点からも，タスクオリエンテッドアプローチで研究を進める有用性が明らかとなった．

2. ドアの通り抜けを含む自律ナビゲーションの新規性

   最後に，実現したドアの通り抜けを含む自律ナビゲーションの新規性について述べる．第1章の従来研究でも述べた通り，ドアの通り抜け動作は，自律ロボットの研究における1つの標準問題であり，この動作の実現を目指して過去にも幾つかの試みが

なされてきた．これらの研究では，ドアの通り抜け動作に問題を絞り，ロボットのために設定した環境内でのみ，この動作は実現された．

これに対し本研究では，タスクオリエンテッドアプローチに沿って，実環境におけるナビゲーション機能としてのドアの通り抜けを目指したことにより，実際にロボットが屋内で使用されるときに生ずる様々な問題を明らかにした．また，実ロボットを用いて実環境で実験を行うことで，人間が普通に生活する屋内環境における移動マニピュレータの自律動作を実現した．このような，実環境で動作する，移動マニピュレータによるドアの通り抜けを含むナビゲーションは，現在世界的に見ても他に例を見ないロボットの自律動作であると筆者らは確信している．

11.6 まとめ

本章では，タスクオリエンテッドアプローチ，及び実験ロボット工学に沿って行った本研究によって得られた，移動マニピュレータの自律動作に関する知見や問題点についてまとめた．

る．また，タスクの実て仕切られる環境でき来することができの第2の成果である．

第 12章

結論

本研究では,「移動マニピュレータによるドアの通り抜けを含む自律ナビゲーション」という具体的なタスクを設定し，このタスクの実現を目指すタスクオリエンテッドアプローチで研究を進めてきた．また，筆者らが属する自律移動ロボットの研究プロジェクトである山彦プロジェクト[YY87][YSI91]において研究開発を行った山彦ロボットに，自作のマニピュレータを搭載し，実際に実環境内で動作させることで，問題点を発見・克服しつつ研究を進めた．その結果，ドアで区切られた屋内環境を行き来することができる，ロボットの自律ナビゲーションを実現した．本論文は，このタスクオリエンテッドアプローチに沿って行われた研究の報告である．

本論文では，まず目標タスクの解析を行い，これを実現するために設計した動作について述べた．次に，この動作を実現するためのロボットのハードウエアの設計・製作，及びソフトウエアの設計・実装について述べ，このロボットを用いた実験結果を報告した．最後に，この研究を通じて明確となったロボットの自律動作に関する問題点や得られた知見について述べた．

本研究で明らかとなった移動マニピュレータの最大の問題は，ベースロボットの推定位置の誤差である．この誤差を如何に正確に推定し減少させるかが，本研究で目指したような複雑な自律動作を実現させるための鍵である．本研究では，ベースロボットの位置誤差に対処するため，屋内環境に自然に存在する平らな壁をランドマークとして，超音波距離センサでこれを認識することによって推定位置の修正を行い，またハンドの位置誤差に対処するため，視覚センサを用いたハンドの位置修正を行った．また，ドアを開ける際には，ベースロボットの推定位置の誤差がハンドに生ずる力に対処するため，手先のコンプライアンス動作を実現した．

このように，設定したタスクを実現する過程において，実環境下における移動マニピュ

レータに関する問題点を明確にし，その問題を一つ一つ解決することで本研究は進められた．本研究で得られた移動マニピュレータの自律動作に関する様々な知見は，今後このような研究を行う場合に十分参考になると考えられる．

一方，自律ロボットによるドアの通り抜け動作は，自律ロボットに関する標準課題のひとつである．この動作を，ナビゲーション動作と共に実環境内で実現したこと自体に新規性があり，本研究は世界的に見ても意義深いものであると筆者らは確信している．

謝辞

本研究は，工学博士 油田信一 筑波大学 電子情報工学系 教授 のご指導のもとで行ったものである．同教授には本研究を進めるにあたり，適切かつ熱心なご指導と多くの貴重な御意見を頂いた．また，論文をまとめるにあたり，適切なご指導を頂いた．

工学博士 太田道男 筑波大学 構造工学系 教授，工学博士 谷江和雄工業技術院 機械技術研究所 バイオロボティクス研究室長 (筑波大学 構造工学系 連携大学院 教授)，工学博士 寅市和男 筑波大学電子情報工学系 教授，工学博士 大田友一 筑波大学 電子情報工学系 教授には，本論文をまとめるにあたり適切なご指導と御助言を頂いた．

Prof. Arthur Sanderson (Rensselear Polytechnic Institute), Dr. Howie Choset (Carnegie Mellon Univ.), Dr. David Wettergreen (NASA AMES), Prof. Oussama Khatib (Stanford Univ.), Prof. Ray Jarvis (Monash Univ.), Prof. Phillip McKerrow (Wollongong Univ.), Dr. Alexander Zelinsky (Australian National Univ.) には，本研究に深い興味を示して頂き，本研究に関する多くの御助言を頂いた．

筑波大学 知能ロボット研究室の諸氏には，ロボットのハードウエア，ソフトウエアに関する貴重な技術的アドバイスを頂き，また実験への御協力をして頂いたのみならず，筆者が研究が行き詰まった時に多くの技術的，精神的アドバイスを頂いた．特に，工学博士 大矢晃久 筑波大学 電子情報工学系 講師には，研究に対する姿勢からプライベートに関することまで，多くのご意見，ご指導を頂いた．工学博士 坪内孝司 筑波大学 電子情報工学系 講師 には，研究に関する技術的な相談から論文をまとめる段階まで，多くの助言を頂いた．前山祥一 筑波大学 工学研究科 には，ロボットシステムに関する相談に乗って頂き，特にハードウエアに関する多くの助言を頂いた．また安藤吉伸 筑波大学 工学研究科， 矢田晃子 筑波大学 理工学研究科 には，本論文締め切りの直前でご尽力を頂いた．

他大学のロボットの研究者諸氏には，学会等で本研究に関するディスカッションに参加して頂いた．特に 工学博士 細田耕 大阪大学電子制御機械工学科 助手，工学博士 鈴木昭二 大阪大学 電子制御機械工学科 助手，松本吉央 東京大学 大学院工学系研究科，吉岡孝

大阪電気通信大学大学院 には，学会等で多くの技術的な助言を頂いた．

綜合警備保障株式会社の技術研究室の方々には，初期のマニピュレータを提供して頂いた他，ディスカッションにも参加して頂いた．

小柳栄次 横須賀工業高校 機械科 教諭には，ロボットに搭載するマニピュレータ及びハンドを製作する上で多大なご尽力を頂いた．彼の協力無しには，本研究のハードウエア部分に関する研究は行きづまっていたであろう．

居酒屋「甚八」の経営者 広田美津子 には，研究方針やプライベートに関するディスカッション，また研究の区切りにおける打ち上げなどを行う上で，心地よい場を提供して頂いた．

弟 永谷真之 には，ロボットの動作イメージをコンピュータグラフィックスで作成して頂くなど，彼の専門分野から多くの協力をして頂いた．

父 永谷英治，母 永谷智寿子 には，研究を継続することに深い理解を示して頂き，経済的また精神的に多大なご支援を頂いた．両親の支援無しには，本研究を継続することはできなかったであろう．

近い将来妻となる最愛の 紲谷美香 には，本研究の最後の一年間，絶えず筆者を激励し，精神的に支えて頂いた．この心の支えなしには，本研究をここまで仕上げることができなかったであろう．

以上の皆様の御指導，御協力，応援や支えがなければ，本研究はここまで為し得ることができず，本論文を執筆することができなかったであろうと確信し，以上の皆様に対し深く感謝の意を表す．

参考文献

[AS94] Kianoush Azarm and Gunther Schmidt. Integrated mobile robot motion planning and execution in changing indoor environments. In *Proc. of IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, pp. 298–303, 1994.

[Bro86] Rodney A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, No. 1, pp. 14–23, 1986.

[CB96] Howie Choset and Joel Burdick. Sensor based planning for planar rod robot. In *Proc. of IEEE International Conf. on Robotics and Automation*, pp. 3584–3591, 1996.

[C.E89] C.E.Thorpe. Outdoor visual navigation for autonomous robots. In *Proc. of Intelligent Autonomous Systems 2(IAS2)*, pp. 530–544, 1989.

[DT87] Steven Dubowsky and Albert B. Tanner. A study of the dynamics and control of mobile manipulators subjected to vehicle disturbances. In *Proc. of Int. Conf. on Robotics Research 4th*, pp. 111–117, 1987.

[E.W59] E.W.Dijkstra. A note on two problems in connection with graph. *Numerische Mathematik*, Vol. 1, pp. 269–271, 1959.

[HSK94] Qiang HUANG, Shigeki Sugano, and Ichiro Kato. Stability control for a mobile manipulator using a potential method. In *Proc. of IEEE/RSJ International Conf. on Intelligent Robots and Systems*, pp. 839–846, 1994.

[inm89] inmos. *The Transputer Databook*. inmos, 1989.

[IY91] Shigeki Iida and Shini'chi Yuta. Vehicle command system and trajectory control for autonomous mobile robots. In *Proceedings of International Workshop on Intelligent Robots and Systems*, pp. 212–217, 1991.

[Kha86] Oussama Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, Vol. 5, No. 1, pp. 90–98, 1986.

[Kha95] Oussama Khatib. Robot planning and control. In *Proc. of International Workshop on Some Critical Issues in Robotics*, pp. 65–80, 1995.

[KOT92] Kiyoshi Komoriya, Eimei Oyama, and Kazuo Tani. Planning of landmark measurement for the navigation of a mobile robot. In *Proc. of IEEE/RSJ International Conf. on Intelligent Robots and Systems*, pp. 1476–1481, 1992.

[KY85] Yutaka Kanayama and Shin'ichi Yuta. Computer architecture for intelligent robots. *Journal of Robotic Systems*, Vol. 2, No. 3, pp. 237–251, 1985.

[KY92] Katsumi Kimoto and Shin'ichi Yuta. A simulator for programming the behavior of an autonomous sensor-based mobile robot. In *Proc. of IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, pp. 1431–1438, 1992.

[LC96] Richard LeGrand and Ren C.Luo. Position estimation of selected targets. In *Proc. of IEEE International Conf. on Robotics and Automation*, pp. 1714–1719, 1996.

[MCR+93] Jonathan M.Cameron, Douglas C.MacKenzie, Keith R.Ward, Ronald C.Arkin, and Wayne J.Book. Reactive control for mobile manipulation. In *Proc. of IEEE International Conf. on Robotics and Automation*, pp. 228–235, 1993.

[MII96] Yoshio Matsumoto, Masayuki Inaba, and Hirochika Inoue. Visual navigation using view-sequenced route representation. In *Proc. of IEEE International Conf. on Robotics and Automation*, pp. 83–88, 1996.

[MOY95] Syoichi Maeyama, Akihisa Ohya, and Shin'ichi Yuta. Non-stop outdoor navigation of a mobile robot – retroactive positioning data fusion with a time con-

suming sensor system –. In *Proc. of IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, pp. 130–135, 1995.

[NM95] Kazuhiro Nishikawa and Hideo Mori. Rotation control and motion estimation of camera for road following. In *Proc. of IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, pp. 1313–1318, 1995.

[NOY95] Toshihiro Nishizawa, Akihisa Ohya, and Shin'ichi Yuta. An implementation of on-board position estimation for a mobile robot. In *Proc. of IEEE International Conf. on Robotics and Automation*, pp. 395–400, 1995.

[NY93] Keiji Nagatani and Shin'ichi Yuta. Path and sensing point planning for mobile robot navigation to minimize the risk of collision. In *Proc. of IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, pp. 2198–2203, 1993.

[OKK+96] O.Khatib, K.Yokoi, K.Chang, D.Ruspini, R.Holmberg, and A.Casal. Coordination and decentralized cooperation of multiple mobile manipulators. *Journal of Robotic Systems*, Vol. 13, No. 11, pp. 755–764, 1996.

[OOY95] Takayuki Ohno, Akihisa Ohya, and Shin'ichi Yuta. An improved sensory circuit of an ultrasonic range finder for mobile robot's obstacle detection. In *Proc. of the 1995 National Conference of the Australian Robot Association*, pp. 178–185, 1995.

[OOY96] Takayuki Ohno, Akihisa Ohya, and Shin'ichi Yuta. Autonomous navigation for mobile robots referring pre-recorded image sequence. In *Proc. of IEEE/RSJ International Conf. on Intelligent Robots and Systems*, pp. 672–679, 1996.

[P.P81] Richard P.Paul. *Robot Manipulators*. The MIT Press, 1981.

[R.C91] R.Chatila. An architecture for task refinement and execution control for intervention robots: Preliminary experiments. In *Preprints of The Second International Symposium of Experimental Robotics (ISER)*, pp. 183–188, 1991.

[SHW94] S.Cooper and H.Durrant-Whyte. A kalman filter for gps navigation of land vehicles. In *Proc. of IEEE/RSJ International Conf. on Intelligent Robots and Systems*, pp. 157–163, 1994.

[SIY93] Sho'ji Suzuki, Jun'ichi Iijima, and Shin'ichi Yuta. Design and implementation of an architecture of autonomous mobile robots for experimental researches. In *Proc. of 6th International Conf. on Advanced Robotics*, pp. 423–428, 1993.

[SY89a] Minho Song and Shin'ichi Yuta. Autonomous mobile robot yamabico and its ultrasonic range finding module. In *Proc. of Korean Automatic Control Conference (KACC'89)*, pp. 711–714, 1989.

[SY89b] Shouji Suzuki and Shin'ichi Yuta. A consideration on the programming method of sensor-driven robot behavior based on the action mode representation. In *Proc. of 20th International Symposium on Industrial Robots*, pp. 127–134, 1989.

[TS92] T.Ikegame and S.Ozono. A new pah planning method for mobile robot applicable to an arbitary environment. In *Proc. of IEEE/RSJ International Conf. on Intelligent Robots and Systems*, pp. 453–460, 1992.

[WY90] Yutaka Watanabe and Shin'ichi Yuta. Position estimation of mobile robots with internal and external sensors using uncertainty evolution technique. In *Proc. of IEEE International Conf. on Robotics and Automation*, pp. 2011–2016, 1990.

[YN95] Shin'ichi Yuta and Keiji Nagatani. Realizing a navigation function on small-sized autonomous robot in in-door environment. In *Proc. of the 1995 National Conference of the Australian Robot Association*, pp. 237–244, 1995.

[YSI91] Shin'ichi Yuta, Shoji Suzuki, and Shigeki Iida. Implementation of a small size experimental self-contained autonomous robot – sensors, vehicle control, and description of sensor based behavior –. In *Preprints of The Second International Symposium of Experimental Robotics (ISER)*, pp. 349–368, 1991.

[YX92] Y.Yamamoto and X.Yun. Coordinating locomotion and manipulation of a mobile manipulator. In *Proc. of the 31st IEEE Conference on Decision and Control*, pp. 1–6, 1992.

[YX93] Y.Yamamoto and X.Yun. Control of mobile manipulators following a moving surface. In *Proc. of IEEE International Conf. on Robotics and Automation*, pp. 1–6, 1993.

[YY87] Y.Kanayama and Shin'ichi Yuta. A self-contained robot 'yamabico'. In *Proc. of 3rd USA-Japan Computer Conference*, pp. 246–250, 1987.

[ZY93] Alexander Zelinksy and Shin'ichi Yuta. Reactive planning for mobile robots using numeric potential fields. In *Proc. of Intelligent Autonomous Systems 3(IAS3)*, pp. 84–93, 1993.

[ニッ95] ニッタ株式会社. 6軸力覚センサシステム 取扱説明書. ニッタ株式会社, 1995.

[岩本88] 岩本太郎, 山本広志, 榊原義宏. 複合センシングによる屋内通路の環境把握と形状可変クローラの自律移動制御実験. 日本ロボット学会誌, Vol. 6, No. 5, pp. 405–, 1988.

[栗栖95] 栗栖正充, 吉川恒夫. 押し作業における対象物の軌道計画. 日本ロボット学会誌, Vol. 13, No. 8, pp. 1115–1121, 1995.

[見浪93] 見浪護, 藤原直史, 拓植広志. 自律移動ロボット搭載型マニピュレータの位置・姿勢制御. 日本ロボット学会誌, Vol. 11, No. 1, pp. 156–164, 1993.

[佐々95] 佐々木裕之, 高橋隆行, 中野栄二. 前方向移動ロボットに搭載されたマニピュレータによるドア開け動作の研究. 第13回日本ロボット学会学術講演会予稿集, pp. 711–712, 1995.

[小笠84] 小笠原司, 井上博允. 知能ロボット・プログラミングシステムcosmos. 日本ロボット学会誌, Vol. 2, No. 6, pp. 3–20, 1984.

[上田92] 上田暁彦, 油田信一. 内外界センサのデータ融合に基づく車輪型移動 ロボットのポジショニング. 第10回日本ロボット学会学術講演会予稿集, pp. 85–88, 1992.

[前山94] 前山祥一, 油田信一. 自律移動ロボットの行動プログラム開発支援の ための音声モニタシステム. 第7回知能移動ロボットシンポジウム予稿集, pp. 154–155, 1994.

研究業績

1. Journal Articles (with review)

   - K.Nagatani and S.Yuta, "Designing a Behavior of Mobile Robot Equipped with a Manipulator to Open and pass through a Door" *Robotics and Autonomous Systems 17*, pp53-64, 1996.
   - K.Nagatani, S.Yuta "Door Opening Behavior of an Autonomous Mobile Manipulator by Sequence of Action Primitives" *Journal of Robotic Systems Vol.13 No.11*, pp709-721, 1996.

2. Journal Articles (accepted and not published yet)

   - K.Nagatani, S.Yuta "Path and Sensing Point Planning for Mobile Robot Navigation to Minimize the Risk of Collision" *Journal of Robotics Society in Japan* (in Japanese).

3. Journal Articles (submitted)

   - S.Yuta, K.Nagatani, S.Maeyama, "Realizing a robust Navigation Function on a Small Size Experimental Autonomous Mobile Robot" *Journal of Robotics and Autonomous Systems, Special issue on "Critical Issues in Robotics".*

4. International Conference Proceedings(with review)

   - K.Nagatani and S.Yuta, "Path and Sensing Point Planning for Mobile Robot Navigation to Minimize the Risk of Collision" *Proc. of IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, pp2198-2203,1993.

- K.Nagatani and S.Yuta, "Designing a Behavior to Open a Door and to Pass through a Door-way using a Mobile Robot Equipped with a Manipulator" *Proc. of IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, pp847-853, 1994.
- K.Nagatani and S.Yuta, "An Experiment on Opening-Door-Behavior by an Autonomous Mobile Robot with a Manipulator" *Proc. of IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, Vol.2, pp45-50, 1995.
- K.Nagatani and S.Yuta, "Realizing a Navigation Function on Small-sized Autonomous Robot in In-door Environment" *Proc. of Robots for Australian Industries*, pp237-244, 1995.
- K.Nagatani and S.Yuta, "An Autonomous Mobile Manipulator with a Functionally Distributed Control System" *Proc. of International Conference on Automation*, pp165-168, 1995.
- K.Nagatani and S.Yuta, "Designing Strategy and Implementation of Mobile Manipulator Control System for Opening Door " *Proc. of International Conference on Robotics and Automation*, vol.3, pp2828-2834, 1996.
- K.Nagatani and S.Yuta, "Opening Door Behavior of an Autonomous Mobile Manipulator by Combination of Action Primitives " *Proc. of International Conference on ISRAM96*, 1996.
- K.Nagatani and S.Yuta, "Autonomous Navigation of Mobile Manipulator in In-Door Environment with Opening / Closing Doors — Architecture of Controller and Experiment —" *Proc. of Fourth International Conference on Control, Automation, Robotics and Vision*, pp267-271, 1996.
- K.Nagatani and S.Yuta, "Realizing a Behavior to Pass through a Door-way by an Autonomous Mobile Manipulator" *Proc. of U.S.-Japan Graduate Student Forum in Robotics*, pp73-76, 1996.

5. Video Proceedings (with review)

- Keiji Nagatani and Shin'ichi Yuta, "An Autonomous Mobile Robot YAMABICO —The path and sensing point planning and navigation in real in door environment—

" *Robotics and Automation Conference VIDEO Proceedings, Mobile Robots-1*, 1993.

6. Domestic Conference Presentations

- Keiji Nagatani and Shin'ichi Yuta, "Path planning for mobile robot with sensing", *Proceedings of 2nd Robot Symposium of Robotics Society of Japan* 1992 (in Japanese).
- Keiji Nagatani and Shin'ichi Yuta, "Path Planning for Mobile Robot with Considering a Risk of Corrision" *10th Lecture meeting Proceedings of Robotics Society of Japan*, 1992 (in Japanese).
- Keiji Nagatani and Shin'ichi Yuta, "Design of Coordination between Locomotion and Manipulation for Door Passing Behavior" *11th Lecture meeting Proceedings of Robotics Society of Japan*, 1993 (in Japanese).
- Yun-Su Ha , Keiji Nagatani and Shin'ichi Yuta, "Indoor Navigation by Self-Contained Mobile Inverse Pendulum", *Proceedings of 4th Robot Symposium of Robotics Society of Japan* 1994 (in Japanese).
- Keiji Nagatani and Shin'ichi Yuta, "Door Opening Behavior by Autonomous Mobile Robot equipped with Manipulator" *12th Lecture meeting Proceedings of Robotics Society of Japan*, 1994 (in Japanese).
- Keiji Nagatani and Shin'ichi Yuta, "Door Opening Behavior by Autonomous Mobile Manipulator –Door Knob Recognition by Hand -Eye system–", *Proceedings of 5th Robot Symposium of Robotics Society of Japan* 1995 (in Japanese).
- Keiji Nagatani and Shin'ichi Yuta, "Door Opening Behavior by Autonomous Mobile Manipulator –Realization of Grasping Door Knob and Door Opening Behavior–" *13th Lecture meeting Proceedings of Robotics Society of Japan*, 1995 (in Japanese).
- Keiji Nagatani and Shin'ichi Yuta, "Door Opening Behavior by Autonomous Mobile Manipulator –Design and Implementation using Action Primitive Concept–" *Proceedings of 1st Robotics Symposia of Robotics Society of Japan*, 1996 (in Japanese).

- Keiji Nagatani and Shin'ichi Yuta, "Realizing a Behavior to Pass through a Door-way by an Autonomous Mobile Manipulator —Motion Planner of the Behavior and Implementation—" *14th Lecture meeting Proceedings of Robotics Society of Japan*, 1996 (in Japanese).