JAERI-M
90-060

原子力知能化システム技術の研究
(人間動作シミュレーション・プログラム：HASP)
—平成元年度作業報告—

1990年3月

浅井　清・藤井　実・上中　淳二*・神林　奨
樋口　健二・久米　悦雄・大谷　孝之・秋元　正幸

日本原子力研究所
Japan Atomic Energy Research Institute

JAERI-Mレポートは，日本原子力研究所が不定期に公刊している研究報告書です。
入手の問合わせは，日本原子力研究所技術情報部情報資料課（〒319-11茨城県那珂郡東海村）あて，お申しこしください。なお，このほかに財団法人原子力弘済会資料センター（〒319-11茨城県那珂郡東海村日本原子力研究所内）で複写による実費頒布をおこなっております。

JAERI-M reports are issued irregularly.
Inquiries about availability of the reports should be addressed to Information Division Department of Technical Information, Japan Atomic Energy Research Institute, Tokai-mura, Naka-gun, Ibaraki-ken 319-11, Japan.

編集兼発行　日本原子力研究所
印　　刷　いばらき印刷㈱

JAERI－M 90－060

原子力知能化システム技術の研究
（人間動作シミュレーション・プログラム：HASP）
－平成元年度作業報告－

日本原子力研究所東海研究所計算センター
浅井　清・藤井　実・上中　淳二*・神林　奨
樋口　健二・久米　悦雄・大谷　孝之・秋元　正幸

（1990年2月23日受理）

日本原子力研究所は，1987年からHASP（Human Acts Simulation Program）と名付けた人工知能とロボティックスに関する研究を10年計画で開始した。これは，知能ロボット，知能化プラントの基盤技術を研究開発するものである。その内容は，自然言語理解，ロボット動作計画，神経ネットワーク手法によるパターン認識，ソリッド・モデルによるプラントの三次元モデル化，二足歩行ロボットの動作シミュレーションと映像化，被曝線量計算，これらに加えて被曝線量計算・ロボット視覚計算の高速化を目的とするモンテカルロ計算装置の設計・試作などである。

本報告は，平成元年度のHASPの作業内容について記述する。

東海研究所：〒319－11　茨城県那珂郡東海村白方字白根2－4
＊　外来研究員，CSK（株）

JAERI-M 90-060

A Study on Intelligent Nuclear Systems (HASP: Human Acts Simulation Program) - Progress Report 1989 -

Kiyoshi ASAI, Minoru FUJII, Junji UENAKA*, Shaw KAMBAYASHI
Kenji HIGUCHI, Etsuo KUME, Takayuki OHTANI and Masayuki AKIMOTO

Computing Center
Tokai Research Establishment
Japan Atomic Energy Research Institute
Tokai-mura, Naka-gun, Ibaraki-ken

(Received February 23, 1990)

The third year progress of the Human Acts Simulation Program, HASP in short, has been presented in this report. The HASP started in 1987 at JAERI as a ten-year research and development program of underlying technologies for intelligent robots, intelligent nuclear plants and so on. It consists of the research and development of technologies of a knowledge-base system, robot vision, robot kinematics/kinetics, plant geometry database, dose evaluation and high speed Monte Carlo machine.

Keywords: Artificial Intelligence, Natural Language Processing, Knowledge-base, Neural Network, Pattern Recognition, Robotics, Graphics, Monte Carlo Method, Supercomputer, Simulation

---

* On leave from CSK Corporation

目　　次

1. はじめに …… 1
2. 命令理解システムの試作 …… 3
2.1 はじめに …… 3
2.2 試作システムで用いる理論及び手法 …… 5
2.3 試作システムの機能 …… 13
2.4 まとめ …… 16
3. 視覚認識の研究 …… 21
3.1 はじめに …… 21
3.2 マーク検出によるロボットナビゲーション …… 25
3.3 Hopfield モデルによる両眼立体視画素対応問題の一解法 …… 39
3.4 今後の方向 …… 48
4. 二足歩行ロボット運動学的シミュレーション …… 49
4.1 はじめに …… 49
4.2 二足歩行ロボットモデル …… 49
4.3 シミュレーションとその動画化 …… 51
4.4 おわりに …… 54
5. 施設形状データベース …… 64
5.1 はじめに …… 64
5.2 動画作成作業 …… 65
5.3 動画作成上の問題点 …… 67
5.4 おわりに …… 68
6. 被曝線量評価 …… 74
6.1 はじめに …… 74
6.2 MCNPコードによる被曝線量計算 …… 74
6.3 被曝線量計算に関するその他の手法について …… 76
6.4 おわりに …… 76
7. モンテカルロ計算装置の仕様 …… 79
7.1 モンテカルロ計算装置の必要性 …… 79
7.2 ベクトル・プロセッサ改造型モンテカルロ計算装置 …… 80
7.3 モンテカルロ計算装置のハードウエア仕様 …… 81
7.4 モンテカルロ計算装置のソフトウエア仕様 …… 86
8. おわりに …… 102
謝　辞 …… 102

Contents

1. Introduction 1
2. A prototype knowledge-base system with natural language interface 3
2.1 Introduction 3
2.2 Natural language processing and planning methods of the prototype system 5
2.3 Functions of the prototype system 13
2.4 Summary 16
3. Image recognition 21
3.1 Introduction 21
3.2 A mark detection method for robot navigation 25
3.3 An application of the Hopfield model for solving the stereo correspondence problem at pexel level 39
3.4 A research plan 48
4. Numerical simulation of biped locomotion robot 49
4.1 Introduction 49
4.2 A mathematical model for biped locomotion 49
4.3 Visualization of simulated results 51
4.4 Concluding remarks 54
5. Plant geometry database 64
5.1 Introduction 64
5.2 Visualization of simulated results 65
5.3 Problems of the visualization 67
5.4 Concluding remarks 68
6. Dose evaluation 74
6.1 Introduction 74
6.2 Dose calculations using MCNP code 74
6.3 A method for dose evaluation using computer tomography data 76
6.4 Concluding remarks 76
7. Monte Carlo machine 79
7.1 Necessity of Monte Carlo machine 79
7.2 Revision of vector processor to Monte Carlo machine 80
7.3 Hardware of Monte Carlo machine 81
7.4 Software of Monte Carlo machine 86
8. Concluding remarks 102
Acknowledgement 102

1. はじめに

昭和62年度に原子力委員会が改訂した原子力開発利用長期計画においては，原子力技術開発の基盤ともなり，かつ他の研究開発分野へも波及効果をもたらす可能性のある技術開発に取り組むことが定められた。それらの分野として，原子力用材料，レーザ，人工知能及び放射線リスク評価・低減技術が挙げられ，この4分野について新しい研究を開始することとなった。

日本原子力研究所（以下，原研）における人工知能技術関連の研究開発としては，

1）原子力プラントに装着された各種のセンサー情報から，プラントの状態を把握するセンサー融合の研究，

2）自律型知能ロボット，プラント設計の基礎技術開発である「人間動作シミュレーション技術の研究」

の2つである。

本報告は，このうちの2）の「人間動作シミュレーション技術の研究」に関する平成元年度の年次報告である。また，この研究は，簡単にはHASP（Human Acts Simulation Program）とも呼んでいる。

「人間動作シミュレーション技術の研究」は，10年を一応の区切りとして開始されたが，平成元年度で3年目になる。初年度，2年度は，研究の道具となるワーク・ステーション，映像生成用並列コンピュータなどのハードウエアの整備，日本語解析ソフト，ロボットの動作空間を与える原子炉施設の形状データベースなどのソフトウエアの整備に追われたが，本年は3年目となり研究の基盤も徐々に整って来た。

原子力分野における従来のロボットは，クローラ型，4つ足型などが採用され，技術も，どちらかと言えばハードウエアから先に開発が行われてきた。HASPにおいては，これらとは逆に，ロボットとして2足歩行型を採用し，先ず知能，視覚，動作などの機能のソフトウエアによるシミュレーション可能性を探ろうとしている。これにより，人間作業の定性的・定量的評価を行う方法論を具体化でき，原子力施設のような複雑人工構造物と人間との役割分担を明確にして，人間の行っている作業の機械化や施設の知能化について有用な知見を得ることが期待される。

HASPの構成要素となる技術は，

(1) 知能ロボットが日本文で与えられた作業命令を理解し，自分の取るべき行動を計画する命令理解の研究（第2章　命令理解システムの試作）

(2) ロボットの視覚装置によって得られる画像から，ロボットが環境を理解する視覚認識の研究（第3章　視覚認識の研究）

(3) 原子力施設におけるロボット動作などの運動学的シミュレーションによるロボット設計の研究（第4章　二足歩行ロボット運動学的シミュレーション）

(4) ロボットの動作環境の設定とシミュレーション結果の映像化を行うための施設形状データベースの開発（第5章　施設形状データベース）

(5) 原子力施設においてロボットが被曝する線量を計算，評価する被曝線量評価手法の開発

（第６章　被曝線量評価）

(6) 被曝線量評価，視覚情報処理などで使用される高速モンテカルロ計算装置（第７章　モンテカルロ計算装置）

である。このうち，モンテカルロ計算装置は本報告書作成の時点では詳細設計を行っている段階であるため，当初予定の仕様のみを報告する。試作する装置の仕様は，これとは異っているところがある。

2. 命令理解システムの試作

HASPでは，日常的な日本語表現を，知能ロボットに対する命令として用いる。命令理解システムは，このような命令の意味を解釈し，知能ロボットがなすべき動作列を生成する。本章では，平成元年度に試作した命令理解プロトタイプシステムの理論的背景及び機能について述べる。

2.1 はじめに

HASPでは，自然言語（日本語）による入力インタフェースを持つ知能ロボットを想定し，知能ロボットが原子力プラント内で保守・点検作業をする際の論理的思考過程及び動作過程のソフトウェアシミュレーションを試行している[1)]。命令理解の研究は，知能ロボットが作業命令を受けてから作業に必要な行動を計画するまでの，論理的思考のシミュレーションを取り扱う。この命令理解に必要な要素技術は，①日本語の意味内容を解釈する日本語解析（natural language processing），②作業目標からそれを達成する行動系列を発生させる問題解決（problem solving），の２つである。本章で記述する命令理解システムは，この２つの技術を計算プログラムの形で実現したものである。

我々が日常接しているコンピュータは，そのままでは自然言語を解釈してくれない[2)]。例えば，ディスク上に存在するディレクトリの内容を知りたい場合，キーボードから「現在のワーキングディレクトリの内容はどうなっているのか？」と入力しても，「コマンドまたはファイル名が違います」といったエラーメッセージが出力されるだけである。コンピュータに何らかの仕事をさせるには，自然言語ではなく，コンピュータごとに定義された言語を利用しなければならない。例えば，ディレクトリの内容表示のために，パーソナルコンピュータのOSとして代表的なMS－DOSではdirコマンドが，UNIXワークステーションではlsコマンドが用意されている。そこで，"現在のワーキングディレクトリの内容はどうなっているのか？"という名前の実行ファイルにdirあるいはlsコマンドを実行するようにプログラムを書き込んでおけば，いかにも日本語を解釈したかのようにディレクトリの内容表示が実行できる。しかし，このような方法では限られた命令のみが受け付け可能となってしまい，コンピュータが自然言語を解釈したとは言い難い。

自然言語を用いて柔軟にコンピュータと対話（音声ではなく，キーボードを用いて）するためには，上に述べた「文字列→コマンド」という方法では限界がある。むしろ，自然言語に対して人工的に意味を表す構造を定義しておき，その構造体からコマンドを生成する方法，つまり，「文字列→意味構造→コマンド」という手法をとるべきである[3)]。文字列から意味構造に変換するには，意味構造を定義するのに用いた文法を用いる。さらに，意味構造からコマンドへ変換するには，上のディレクトリ検索の例に現れたように，言語が用いられた環境（MS－DOS，UNIX環境や現在のディレクトリの位置など）に関する情報を利用して，与えられた問題を解決（dir，lsコマンドを発生）しなければならない。知能ロボットへ日本語を用いて命令を下すこともコンピュータと対話することを基本的に等しいので，HASPで行っている命令理解の研究でも，このよ

うな自然言語処理，問題解決の手法を用いる。命令理解システムの構築の際に必要な技術とこれまでに我々が調査・整備した手法[1,5]をTable 2.1 にまとめる。

現在までにTable 2.1 にあげた①から④の項目について，計算機上で稼働する命令理解システムのひな型（以下，試作システムと呼ぶ）を作成した。試作システムは，簡単な日本語で記述された命令を解釈し，ロボットが動作するのに必要となる概念的な動作列（こまかな関節角などの指示をしていないという意味で）を生成する。システムの記述言語は，Kyoto Common Lisp[4]で，開発環境はSUN3ワークステーションである。また，知能ロボットの動作をシミュレートしている際に，いつでもシステムに対して日本語で質問あるいは新たな命令を入力することができる。試作システムで用いている理論・手法は，Table 2.1 の項目①について高木・伊東の手法を，②はCS－PARSER を，③はフレーム理論を，そして④については第1階述語論理とR.C.Schankらのゴール・プランに関する理論を用いている。以下，2.2節では，試作システムで利用している理論・手法の概要及び特徴をまとめ，2.3節ではシステムを実現する際の技術的な手法・機能について述べる。

Table 2.1 Techniques for the action planning system

技術	調査・整備対象
①自然言語の意味構造の定義方法	格文法，モンタギュー文法，高木・伊東の手法，概念依存性理論
②文字列から意味構造を導出する手法	ＧＲＡＤＥ，ＣＳ－ＰＡＲＳＥＲ
③知能ロボットを含めた環境の人工的定義方法	モンタギュー文法，フレーム理論
④環境に応じた問題解決手法	第1階述語論理，スクリプト理論，ゴール・プラン理論，microTALE-SPIN
⑤計算機による学習手法	記憶組織化パケット理論

2.2　試作システムで用いる理論及び手法

本節では，試作システムで用いている自然言語の意味表現手法，環境の定義方法，そして問題解決手法についてその概要と特徴を述べる。

2.2.1　自然言語の意味表現手法

自然言語の意味内容を記述する試みは，機械翻訳システムの開発を境に言語学そして人工知能の両面から数多くなされてきた。これらの試みは，「自然言語の意味とは何か？」という根本的な大問題をいまだ解決していないが，数多くの実用システム（例えば機械翻訳システムや自然言語によるデータベース検索システムなど）を生み出していることから，あるレベルまで「意味の記述」が達成されていると考えられる。

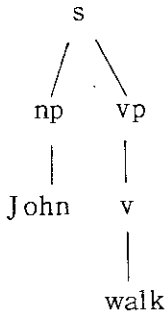
言語の意味を記述する手段として，よく取り上げられるものは述語論理である[6,7]。特に第1階述語論理は，形式的な記述と共にその機械的な証明手続きが明らかなため，現在の計算機を利用して自然言語処理プログラムを開発する際の強力な手段となる[7]。ここで簡単な例として，事象「John walks.（ジョンは歩く。）」を取り上げよう。述語論理式を用いてこの事象を表現すれば，

$$\mathrm{event} = \mathrm{walk(j)} \qquad (2.1)$$

のように記述できるだろう。ここでwalk(x)は"xが歩く"という概念を指し示し，jはある特定の"John"を指し示しているものとする。ここで問題となることは，(i)自然文からどのように式（2.1）を導出するのか，(ii)特定の"John"はどのように定めるのか，そして(iii)walk(x)という概念はどのように決定したら良いのかということである。(i)~(iii)の問題に対して系統的に解答を示したのがモンタギュー文法である[7]。モンタギュー文法を使って例文がどのように式（2.1）に変換されるかを示そう。処理の流れは以下のとおりである。

① 自然文の句構造木を作る。

John walks.



（句構造規則）

s → np vp

vp → v np

np → John

v → walk

② ①で得られた終端記号に対して，内包論理式を対応させる。

〔λP〔P{j}〕〕（^walk'）

（変換規則）

John → λP P{j}

walk → ^walk'

③　②で得られた論理式を展開する。

walk'(j)

この展開の詳細については参考文献(7)を参照されたい。上に述べた展開はきわめて機械的に実行できる。しかしながら，モンタギュー文法を実際の自然言語処理プログラムとして実現する際，次の問題点があげられる。まず，比較的語順の制約が緩い日本語を，句構造規則を用いて解析するのは難しい。処理②に現れる"John"の特定方法が明示されていない（東京のジョンさんか，はたまたニューヨークに住んでいるジョンさんか）。なぜならば，表現 $\lambda P\ P\{j\}$ は単純に，ある可能世界の１つに対して，"John"という表現に対応する"j"が存在するということを述べているだけであり，可能世界の特定方法には言及していない。さらに，動作概念を示す walk' のような表現を単語１つ１つに定義するのは物理的に困難である（動詞の数だけ規則を用意しなければならない）。これらの問題が解決できれば，モンタギュー文法の処理の流れを利用して，機械的に自然文の意味（述語表現）を得ることが可能と考えられる。

日本語は英語と比較して次の特徴を持っている。(i)単語と単語の区切りに空白をいれない記述（べた書き）が普通である。(ii)単語の並びに対する制約が比較的緩い。(iii)句構造規則にしたがって文が成立していると言うよりは，単語間の意味的な接続関係によって文の構造が決定されている。(ii)及び(iii)の理由から，日本語の処理には格文法あるいは格文法を拡張した手法が，機械翻訳システムなど実用プログラムに多用されている[8]。特徴(i)に対しては，文字並びに関する辞書情報を豊富に用意し，文中に出現する単語の切り出しを経験的な規則を用いて実現するのが普通である。ここでは，(ii)および(iii)の特徴を考慮して，上に述べた処理①～③の問題点を解決する方法を考察する。

R.C. Schankらは，格文法の考え方に沿って形式的に意味構造を定義した[9]。彼らの理論は，概念依存性理論（Conceptual Dependency theory ： CD 理論）と呼ばれる。ＣＤ理論を用いると，上にあげた事象「ジョンは歩く」は次のように記述できる。

$$PP1 \Leftrightarrow PTRANS \xleftarrow{O} PP1 \left\langle \begin{array}{l} \xrightarrow{D} ?TO \\ ?FROM \end{array} \right. \xleftarrow{I} \begin{array}{c} PP1 \\ \Updownarrow \\ MOVE \\ \uparrow_{O} \\ PART(PP1) \end{array} \qquad (2.2)$$

$$PP1 \Leftrightarrow name(ジョン) \qquad (2.3)$$

この表現の読み方そしてＣＤ理論の詳細については，参考文献(9)を参照されたい。ＣＤ理論の特徴は，ある事象を表現する際，動的な概念（物理的な動作，抽象概念の移動など）と静的な概念（ものに固有の性質の記述）とを，それぞれ，式（2.2）及び式（2.3）のようにわけて考えるところにある。つまり，式（2.1）で現れた walk(x) という表現に対して式（2.2）を対応させ，特定の"John"を表していた j"という表現に対して式（2.3）を対応させている。さらに，「歩く」という動的概念が，「自分の体の一部を動かして，自分を移動させる」という表現に置き換えられることから，移動を示す基本概念 PTRANS と自分の体の一部を動かすことを示すMOVEとを組み合わせることで「歩く」を表現する。また，格文法の教えるところにより，〔誰〕が，〔なに〕を，〔どこ〕から，〔どこ〕まで，〔どのように〕移動するのかを表現するため，⇔

（主体），$\overset{o}{\leftarrow}$（対象），$\overset{D}{\leftarrow}$（方向），$\overset{I}{\leftarrow}$（道具）という格を導入している。CD表現（2.1）及び（2.2）を述語論理式に置き換えると次のように表現できる。

$$\text{event} = \text{PTRANS}\ (*\text{actor}\ (\text{PP1}), *\text{object}\ (\ ), *\text{from}(\ ), *\text{to}(\ ), *\text{inst}\ (\ldots)) \quad (2.4)$$

$$\text{PP1} = \{\text{name}\ (\text{ジョン}), *\text{other-attribute}\ (\text{value}), \ldots\} \quad (2.5)$$

式（2.4）は，式（2.1）が1つの引数だけをとっていたのに対して，格に相当する5つの引数からなる述語表現となっている。式（2.4）と（2.5）は，モンタギュー文法を用いる際の処理②の問題点に対する解答をあたえる。すなわち，特定の対象物を決定する際に，式（2.5）のように対象物を属性と値の集合で記述しているため，式（2.5）相当の表現とシステムが取り扱っている世界に含まれる複数の実体とを比較することで対象物が特定できる。単語1つ1つに対して述語を定義するのではなく，基本的な動作概念をあらかじめ定義しておくため，有限の述語の集合だけの取り扱いだけですむ。

残された問題は，文字列で表現された自然文から，どのようにして式（2.4），（2.5）のような表現を導出するのかということである。この問題に対しては，CD理論と同様の発想に基づいて，日本語の解析を行うプログラムCS-PARSERを導入することで対処した[1,5]。CS-PARSERは，高木・伊東の意味表現手法[10]にのっとった日本語解析システムで，CD理論を日本語向けにインプリメントしたものと見なすことができる。CS-PARSERは，日本語に特有の助詞（特に格助詞）を式（2.4）に含まれている格に，「丸いボール」といった形容表現を式（2.5）相当のものに対応させることで，日本語に対する意味表現を出力する。

本項をまとめると，我々の自然言語解析手法（意味表現手法）は，モンタギュー文法の意味表現導出方法に倣い，

① 自然文から構文を導出するのにCS-PARSERを用いる。
② 動的概念について格構造に基づいた述語を定義する。
③ 実体を属性と値の集合で定義する。
④ 実体を対象世界と比較することで特定する。
⑤ 動作概念に有限個の述語を用いる。

という手法を用いることで，日本語に適した意味表現を導出するものである。得られた意味構造は，問題解決部に送られロボットの動作列を決定することになる。さて，まだ解決されていない問題がある。上にあげた処理④で，実体を特定するために「対象世界」を用いている。「対象世界」はどのように定義したらよいのだろうか？次項で，この対象世界の定義方法をまとめる。

2.2.2 環境の定義方法

モンタギュー文法では，文中に含まれている実体概念を可能世界への外延として取り扱うことで，具体的な意味を持たせていた。このような操作を行う場合，可能世界をどのように定義したら良いかという問題が持ち上がる。少なくとも，可能世界を定義した構造体は，2.2.1項で示した属性と値による実体の定義と矛盾してはならない。また，入力された命令に対して可能世界を

選択するための情報を用意しなければならない。そこで我々は，以下の環境定義に対するモデルを用いる[9]。

① 環境に含まれる実体を属性と値によって表現する。

② 実体の包含関係に階層関係を持たせる。

③ 知能ロボットに自分のおかれた環境（主に物理的な環境）に関する情報を与える。

入力された命令に対して，実体を示す意味構造が得られた段階で，③で示すロボットの認識情報を頼りに，その実体を特定する。例えば，「自動販売機に行って，コーラを買ってこい」という命令に含まれる"自動販売機"は，「コーラを買う」ことができ，位置的に「ロボットの移動できる範囲にある」もの，として特定する（属性と値の突き合わせ）。

我々は，属性と値を効率よく記述できる手法として，フレーム（あるいはフレーム相当の）手法を用いる。具体的には，Common Lispで定義されている"structure"(構造)を用いている[11]。structureは，関数defstructによって以下のように定義できる[10]。

```
(defstruct ship
    x-position
    y-position
    x-velocity
    y-velocity
    mass)                                                    (2.6)
```

式（2.6）は，2次元世界を飛行する"宇宙船"を定義したstructureである。この定義に基づき，飛行船「ship1」をシンボルship1に対応させることができる：

```
(setq ship1 (make-ship :x-position 0
                       :y-position 0
                       :mass 1000))
#S(ship x-position 0
        y-position 0
        x-velocity nil
        y-velocity nil
        mass 1000)                                           (2.7)
```

式（2.7）は，2.2.1項でみた実体の意味構造と比較可能なものである。我々は，知能ロボットを取り巻く環境を定義するためのstructureとして，次の定義を用いる。

```
(defstruct object
        name      ; 実体の名前
        jittai    ; 実体であるかどうか（t or nil）
        group     ; 保有する実体グループ
        imi       ; 意味（代表名）
        upper     ; 上位概念
        lower     ; 下位概念
        position  ; 物体の位置
```

mark ; 地図上で物体を示すマーク（フラグ）
in ; 物体に"内部"があるとき，それを示すマーク
situation ; 物体の状態（属性と値とのリスト）
function ; 物体の保有する関数
kakatteiru); 入力文に物体を限定するような概念が存在するかどうか

③に示した，ロボットの認識情報は，ロボットを示すシンボルの属性リストとして表現する。

```
(setf (get robot speed ) normal  ;ロボットの移動速度
      (get robot 入力文)  nil     ; 入力文
      (get robot 立場)    nil     ; 命令者に対するロボットの立場
      (get robot 行動指針)nil     ; 立場から導かれるロボットの行動指針
      (get robot front    front   ; ロボットの向き
      (get robot position)(10 10); ロボットの位置
      (get robot p_scene) genken  ; 以前の位置状況
      (get robot scene )  genken  ;現在の位置状況
      (get robot pocket)  nil     ); 現在持っているものの名前と数量
```

上にあげたstructureと属性リストを用いて，物体の特定を行う。例を挙げて説明すると，「1次区画のPCS熱交換器の線量を測定しろ」と命令された場合，「PCS熱交換器」は1次区画にあるものと限定されているため，structureのkakatteiruスロットに「1次区画」が記述されている"PCS熱交換器"であることがわかる。さらに，ロボットの位置状況を頼りにその"1次区画"が「genken」という上位概念を持っているものと特定できる。

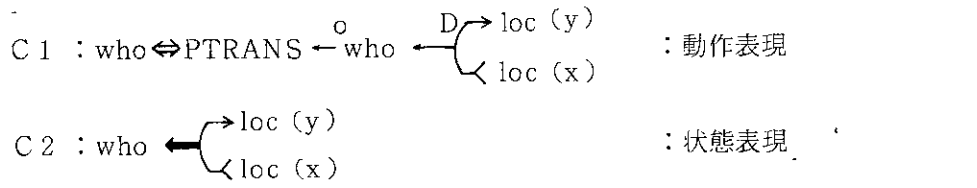
まとめると，試作システムではロボットを取り巻く環境（可能世界）をstructureを用いて定義し，ロボットの認識情報を属性リストで表現する。日本語の意味構造の導出にあたって，2.2.1及び本項で示した手法を用いることで，モンタギュー文法に倣った手法を実現した。

2.2.3 問題解決手法

日本語で記述された命令の意味構造（以下，ゴールと呼ぶ）を導出した後，命令理解システムは命令を達成するために必要な動作列（コマンド列）を生成しなければならない。人工知能の1分野である問題解決は，このようなゴールからコマンド列の生成を行うものである。一般に問題解決の手法は，以下の2つに分類できる[5]。

① ゴールを解決する行為を導く。
② ゴールを満たす状態遷移図を導く。

①の手法は，ロボットの動作（歩行，腕の動きなど）によって，ゴールを解決するための手続きを得るものである。具体例として，FORTRAN言語などで記述されたプログラムがある。他方，②の手法は，作業を対象物の状態遷移として捉え，ゴールを満足する遷移過程を，状態遷移図から選び出し，それに必要なロボットの動作を導き出すものである。具体例として，経路探索があげられる。経路探索は，ロボットの現在位置から目標位置までの，位置について空間・時間的に矛盾のない遷移図を導くことで実行できる。この2つの手法の違いは，CD理論で事象を記述すると良く理解できる。



C1とC2は同時に起こっている事象である。①の手法は，C1→C2への展開を行うのに対し，②の手法は，C2→C1への展開を行っているのである。（CD表現は，矛盾を招かないため，どちらの手法をとっても最終的に得られる動作列は等しいものとなるはずである。）我々は，動作列生成を「ロボット動作言語による自動プログラミング」という観点から取り扱うこととし，①の手法をとることとした。②の手法は，複雑な制約下（時間，距離，被曝線量などの制約）の経路探索に用いることとし，今回の試作システムには取り入れていない。①の手法を表現し直せば，“目的－手段型の知識を利用した問題解決”となる。以下，この目的－手段型の問題解決手法を考察する。

目的－手段型の問題解決用知識は，論理式を用いて表現できる[7]。ここで，項{G1, G2, G3}及び{e1,e2,e3,e4,e5}を考える。さらに，項の間の論理的な関係を以下のように定義する。

$$\begin{aligned} G1 &\leftarrow G2 \wedge e1 \\ G2 &\leftarrow e2 \wedge G3 \\ G3 &\leftarrow e4 \wedge e5 \end{aligned} \qquad (2.8)$$

項G1をゴールと見なすと，導出原理により，

$$G1 \leftarrow e2 \wedge e4 \wedge e5 \wedge e1 \qquad (2.9)$$

という項{e1,e2,e3,e4,e5}のみの論理式を得る[7]。{e1,e2,e3,e4,e5}を知能ロボットの動作要素に対応させると，式（2.9）はゴールを解決するための動作列となる。一般に，項{G1,G2,G3}及び{e1,e2,e3,e4,e5}は，変項を持つ述語項{G1(x1,y1,・・・),・・・}あるいは{e1(x1,y1,・・),・・・}として定義可能である。このとき，述語を固定して論理式の意味論を議論することを，第1階述語論理と呼ぶ[7]。我々は目的-手段型の知識表現として，式（2.8）に変項を含んだ論理式，つまり第1階述語論理式を用いる。変項の特定は，単一化（ユニフィケーション）という操作を用いることで実行できる[7]。試作システムでは，ゴールに含まれている定数項及びロボットの認識情報に含まれている定数項を用いて，論理式に含まれる変数を束縛する（具体的なプログラムの例としては参考文献12に詳しい記述がある）。
式（2.8）は，簡単なAND演算だけで構成されていたが，実際にロボットの動作列を記述する場合には，条件分岐など一般のプログラムに要求される仕様を満たしていることが望ましい。そこで，Table 2.2にあげる9つの知識分類（ルールの種類）を用いることとした。この分類を用いることで，if-then-else，条件分岐構文を完全に記述することができる。その結果，問題解決で得られる動作列は，その要素である基本動作を実行可能にするライブラリと動作列の構文を解読できるインタプリンタとを用意することで，直ちに処理できる。

さて，上に述べた項{G1,G2,G3}及び{e1,e2,e3,e4,e5}を知能ロボットに合わせてどのように定義したらよいのだろうか。R.C. SchankとR.P. Abelsonによると，人間が問題解

決や物事の理解に用いている知識そして行為には，次の階層関係がある[13]。

(i) THEME (BELIEF)

(ii) GOAL

(iii) PLAN

(iv) ACTION

(i)～(iv)までの階層の適応の方法は，次の例で明らかになる[14]。

① 「弱いものいじめすることが生きがいの桃太郎がいた（THEME)。ある日，桃太郎は，物ごいをしている鬼を見つけた。いじめる相手を見つけた桃太郎は，鬼を徹底的に懲らしめようと思った（GOAL)。そこで，犬と猿と雉子をお金で雇い，鬼をいじめることにした（PLAN)。計画通り鬼を退治した（ACTIONの遂行)。」

② 「弱いものを助けることが生きがいの桃太郎がいた（THEME)。ある日，桃太郎は物ごいをしている鬼を見つけた。桃太郎は，鬼に食べ物をたくさんあげようと思った（GOAL)。そこで，心ざしの同じ太と猿と雉子とで食べ物を集め，鬼にあげることにした（PLAN)。鬼に，食べ物をあげたとたん，桃太郎は仲間もろとも鬼の口の中に吸い込まれてしまった（ACTIONの遂行)。」

①，②のどちらも桃太郎を題材にした創作であるが，桃太郎の信念（THEME）の違いで，異なった結果を導くことがわかる。知能ロボットに命令を与える場合でも同様のことが考えられる。いま，２つのロボットを考えよう。１つは，見回り専門のロボット，他方は見回りと修理のできるロボットとする。「配電盤の異常を知らせろ」という命令が与えられた場合，見回り専門ロボットはメータの値を読み正常値と比較して異常の有無を伝えるだけである。しかし，もう一方のロボットは，異常を見つけたら機転をきかせて，異常箇所の修理に向かうだろう。

我々はSchankらの知識・行為の階層関係に基づき，目的－手段型の問題解決で用いる知識（ルール）に対して，以下の階層関係を定義する。

① THEME

② GOAL

③ THEMEに従ったGOALに対するPLAN

④ PLAN中に存在するACTION

入力命令は，②のGOALに対応する。THEMEは，知能ロボットに対して予め与えておく。問題解決は，まず，ロボットのTHEMEを調べ，それにしたがった行動指針をたてる。行動指針に従い，GOALを解決するPLAN（動作列）を生成する。この知識・行為の階層関係を用いることで，さまざまな用途のロボットを，同じ問題解決のアルゴリズムを使って制御することが可能となる。この問題解決の手法は，第１階述語論理式を用いて次のように書き下すことができる：

$$\mathrm{GOAL} \leftarrow [\lambda T\,(\mathrm{PLAN}\,(\mathrm{GOAL}\,;T))]\,(\mathrm{THEME}\,(\mathrm{robot})) \qquad (2.10)$$

式（2.10）に含まれるPLAN（g;T）は，テーマTにしたがった（Tという制約下で）ゴールgに対するプラン（動作列）を示す。そこで，①～④の知識はすべて式（2.8）の相当の表現が可能となる。問題解決のアルゴリズムは，単純な導出原理となる。第１階述語論理を用いて，意図あるいは信念を記述しようとする試みとしては，P.R. CohenとH.J. Levesqueの実例がある[15]。

ここで，上に述べてきた問題解決手法の問題点をあげる。式（2.9）を導出する際に，項{e1, e2,e3,e4,e5 } の真偽について述べなかった。我々は，ロボットの基本動作は如何なる場合においても「実行可能」と考え，項{e1, e2, e3, e4, e5 }を恒真の項として取り扱う。そのため，導出された式（2.9）は常に真となる。そのため，ゴールから導出された論理式に基本動作を示す項以外のものが含まれていた場合，この手法は問題解決をできないまま終了してしまう。つまり，知識を入力する際に，人間が注意深くルールの整合性をチェックしなければいけない（知識作成の難しさ）。さらに，恒真と考えた基本動作が，環境によって実行不可能（偽）となった場合，導出によって得られたプランは偽となり，ゴールは解決されない（環境に依存した動作）。前者の問題点に関しては，今のところ解決策を見出していない。後者に関しては，ゴールが達成されない場合，その原因（動作の前提条件の不満足）を動作列から検出し，ゴールを達成するような新規の動作をとることで解決しようと考えている。

まとめとして，我々は入力命令に対する問題解決に第1階述語論理を用い，そこで使う問題解決知識にTHEME，GOAL，PLAN，ACTIONという階層関係を定めた。これにより，単目的ロボット，複目的ロボットを問わない問題解決の手法を提案した。

Table 2.2 Types of planning rules. (G: Goal, PC: Precondition, BA: Basic Action, CA: Confirmative Action, t: truth)

if-then type	**until-do type**
(1) G←{PC's}∧{BA's}	(7) G←[until(CA),do({BA's})]
(2) G←{PC's}∧{G's}	**conditional type**
(3) G←t∧{BA's}	(8) G←cond[(CA∧{BA's}),...]
(4) G←t∧{G's}	**or type**
if-then-else type	(9) G←(CA's∧{BA's})∨.....
(5) G←{PC's}∧[CA∨{BA's}]	
(6) G←t∧[CA∨{BA's}]	

2.3 試作システムの機能

試作システムは，2.2節で述べた自然言語処理及び問題解決手法を用いて，いくつかの簡単な命令を解釈し，ロボットの概念レベルの動作列を生成する。本節では，試作システムが取り扱うことのできる環境，機能，そしてシステムの構成について述べる。

2.3.1 試作システムが取り扱う環境

試作システムが取り扱う世界は，現在，日本原子力研究所で改造中の研究炉JRR-3の1階と地下1階の一部分を簡略化したものである（Fig.2.1）。この世界は，システム上にstructureのデータ構造で記述され（2.2.2項参照），システム内に仮想的に配置されたロボット（実はシステム自身である）がその中で行動する。いまのところ，この仮想世界の作成には，5章で述べる施設形状データベースを用いていない。その理由は，施設形状データベースの内容が実体の性質の一部である幾何学的性質に偏っているので，経路探索や問題解決などに適していないためである。十分に整備されている施設形状データベースを活かした環境設定には，2.2.2項で述べた仮想世界の定義手法に倣って，データを変換・追加することが必要と考えられる。

現在までに，Table 2.3にあげた日本文をシステムに入力できるよう辞書・知識を整備した。命令文，疑問文，平叙文の区別は，形態素情報を用い，意味構造にそのラグをたてることで実現している。今後，辞書・知識を追加して行くことにより，「喉が乾いた」という状態表明の平叙文の入力を受けて，ロボットがジュースを買いに行くような問題解決を行わせることも可能である。

Table 2.3 Acceptable sentences of the prototype system.

種類	文
命令型単文	「待て」「継続しろ」
	「n階へ行け」「階段を昇れ」「階段を降りろ」
	「エレベータに乗れ」「エレベータを降りろ」
	「一次区画へ行け」「ＰＣＳ熱交換器の近傍へ行け」
	「ＰＣＳ熱交換器の線量を測定しろ」
命令型複文	「エレベータを使ってn階へ行け」（限定の”て”）
	「階段を使ってn階へ行け」（〃）
	「一次区画へ行ってＰＣＳ熱交換器の線量を測定しろ」（順序の”て”）
	（条件の”たら”）
質問文	「一次区画についたら連絡しろ」
	「どこにいるか」「何をしているか」
	「線量計を持っているか」

2.3.2　試作システムの構成及び機能

試作システムは，日本語解析プログラムCS－PARSER（ゴール生成機能を含む），問題解決用知識から動作列を導出するための知識適応規則，シミュレーション世界を定義するstructure群，メイン制御プログラム，システム制御用データ，などから構成されている（Fig. 2.2）。システムの規模は，Table 2.4 に示すとおりである。日本語処理部分が最も規模が大きくなっている。各プログラムは，Kyoto Common Lisp を用いて開発した。また実行環境は，SUN3／260ワークステーションである。

メイン制御プログラムは，単純なループ構造を持ったインタプリタである。Fig. 2.2 中のデータ部分を参照し，それに見合ったプロセスを起動するようになっている。そのため，システム制御データを変更することで，システムを簡単に修正することができる。各プロセスは，メイン制御プログラムの１ループで１つのステップを実行する。そのため，プロセスの途中で新たな入力が加えられても，直ちにその解析を実行することができる（割り込み処理）。割り込み処理を可能とするために，Fig. 2.1 中のデータ部分はスタック構造となっており，新規入力に対する動作が終了すると，古い動作を再開する仕組みである。問題解決によって得られる動作列（フローチャート）は，Table 2.5 に示す基本動作関数から構成されている。二足歩行パターン生成や腕の動きの制御などは，これらの基本動作には含まれていない。

システムを起動すると，Fig. 2.3 に示す画面が，ワークステーション上の１つのウィンドウ上に現れる。ウィンドウには，入力欄，受付文表示欄，プロセス表示欄，応答欄などが配置されている。さらに，問題解決や動作列の実行状況を知るために，別のウィンドウ上に動作列の導出過程，動作列の実行状況を表示することが可能である。入力欄に日本語を用いて命令をタイプすると，システムは入力解析，問題解決，動作実行というプロセスを次々と実行する。また，ロボットの移動状況を知るために，２次元の簡単な画像を表示することが可能である。

Table 2.4　Size of the prototype system

CS-PARSER	7.0 MB
辞書	0.5 MB
ゴール生成部	0.1 MB
知識ベース	0.02MB
（問題解決用知識，システム制御データ）	
オブジェクトファイル群	0.3MB
ユーティリティ	0.01MB
合計	7.93MB

Table 2.5 Basic actions for the action planning

関数名	機能
[一般実行関数]	
answer	引数の評価結果をロボットの応答欄に出力する
check_button	エレベータのボタンが押してあるかどうか確認する
check_imi	２つの物体が同じ意味であるかどうか確認する
check_kakatteiru	ある物体を限定するような何かが係っているかどうか確認する
check_reference	ある物体を限定するようなデータがないかどうかを確認する
check_scene	ロボットの現在位置の認識
check_tree	ある問題解決木から特定のゴールに関する部分を抜き出す
cmove_1	移動関数
joutai_yomu	入力文から状態を読み取る（知識検索）
search_map	引数で与えられた座標に何があるかを返す
operate_object	物体を操作する（物体の所持する関数を起動する）
push_button_1	エレベータのボタンを押す
put_in	物体に何かを入れる
scan_situation	物体の状態を認識する
search_base	知識ベースの検索
search_have	ロボットの持ち物を確認する
search_object	物体が（指定距離に）あるかどうか確認する
search_position	物体の位置を認識する
search_situation	物体の存在を確認する
stop_this_rule	問題解決を中断する
take_up	物体を持ち上げる
wait	”待つ”関数
do_nothing	”待つ”を解除するかどうかの確認関数
exit_nothing	”待つ”を解除する関数
[メタ関数]	
clear_rule	引数で与えられたゴールについて問題解決を行い，フローチャートにして実行する
search_rule	与えられたゴールについて問題解決を行う（実行はしない）
jikkou	引数で与えられた問題解決木をフローチャートになおし実行する

2.4 まとめ

日本語命令を受け付け，命令を達成するための動作列を生成する命令理解システムの概要と試作システムについて述べてきた。システムは，取り扱う日本文や仮想世界の大きさから，きわめて規模の小さなものである。しかし，システムの基本的な制御構造は，知識の拡充にも十分対処できるものと思われる。しかも，知識ベースやシステム制御データなどには処理手続きを記述していないため，単純なデータとして取り扱える。このため，きわめてモジュール化したシステムの構成を得ることができた。

最後に，今後の課題として次の４点をあげて本節をしめくくりたい。

① 現在の知識ベースは，対象世界が小さいために小規模であるが，今後複雑なシミュレーション世界を考えた場合，知識ベースがきわめて大きくなる可能性がある。これは知識ベースを検索する際の大きなネックとなる。そこで，作業対象物ごとに，知識を分散させて配置する方法が必要となる。このような知識ベースの構成を今後検討していき，現在の試作システムに反映していきたい。

② 試作システムは，ロボットの作業経験を活かした推論を行っていない。そこで，作業終了時にいままで行った動作系列を環境に沿って組織化し，次の作業に反映させる手法が必要である。このためには，計算機による学習手法をインプリメントしなければならない。現在，R.C. Schank らの記憶組織化パケット理論[16]を参考に，その実現を目指している。

③ 2.2節で述べたように，動作列を導出する際に，時間に依存した制約を取り込んでいない。特に，経路探索を行う場合このような時間制約は必要になる。述語論理の世界でも時間制約を取り扱う試み[17]がなされており，その手法を調査していきたい。

④ 今回のシステムは，あくまでプロトタイプとして位置づけているので，システムの処理速度に関する考慮が余りなされていない。実用システムとするには，細部のプログラミングを注意深く吟味し，処理速度向上を達成する必要がある。現在，日本語解析部をLispからC言語に変換して，高速処理を実現することを検討している。

参考文献

1) 浅井 他：JAERI－M 89－023（1989）
2) R.C. Schank and P. Childers, "The Cognitive Computer" (Addison-Wesley, California, 1984)
3) 辻井：情報処理，30－10，1142（1989）
4) T. Yuasa and M. Hagiya, "Kyoto Common Lisp Report" (TEIKOKU INSATSU, 1984)
5) 神林，上中：JAERI－M 89－218（1989）
6) 土屋，白井，鈴木，川森：言語，19－1，84（1990）
7) 長尾，淵：論理と意味”（岩波書店，1983）
8) 野村：情報処理，30－10，1161（1989）

9) R.C. Schank, "Conceptual Information Processing", 2nd ed. (North-Holland, Amsterdam, 1984)
10) 高木, 伊東：自然言語の処理（丸善, 1987）
11) G.L. Steel Jr. et al., 後藤 監訳： Common LISP（共立出版, 1986）
12) R.C. Schank and C.K. Riesbeck, 石崎 監訳：自然言語処理入門（総研出版, 1986）
13) R.C. Schank and R.P. Abelson, "Scripts, Plans, Goals and Understanding" (Lawrence Erlbaum Associates, New Jersey, 1977);
R.C. Schank, "Explanation Patterns" (Lawrence Erlbaum Associates, New Jersey, 1986)
14) 中村：言語, 18-2, 10 (1989)
15) P.R. Cohen, H.J. Levesque, "Persistence, Intention and Commitment", AI Center and CSLI (1987)；解説は, 向井：人工知能学会誌, 3-3, 289 (1988)
16) R.C. Schank, "Dynamic Memory" (Cambridge University Press, 1982)
17) D. McDermott, Cognitive Science 6, 101 (1982)

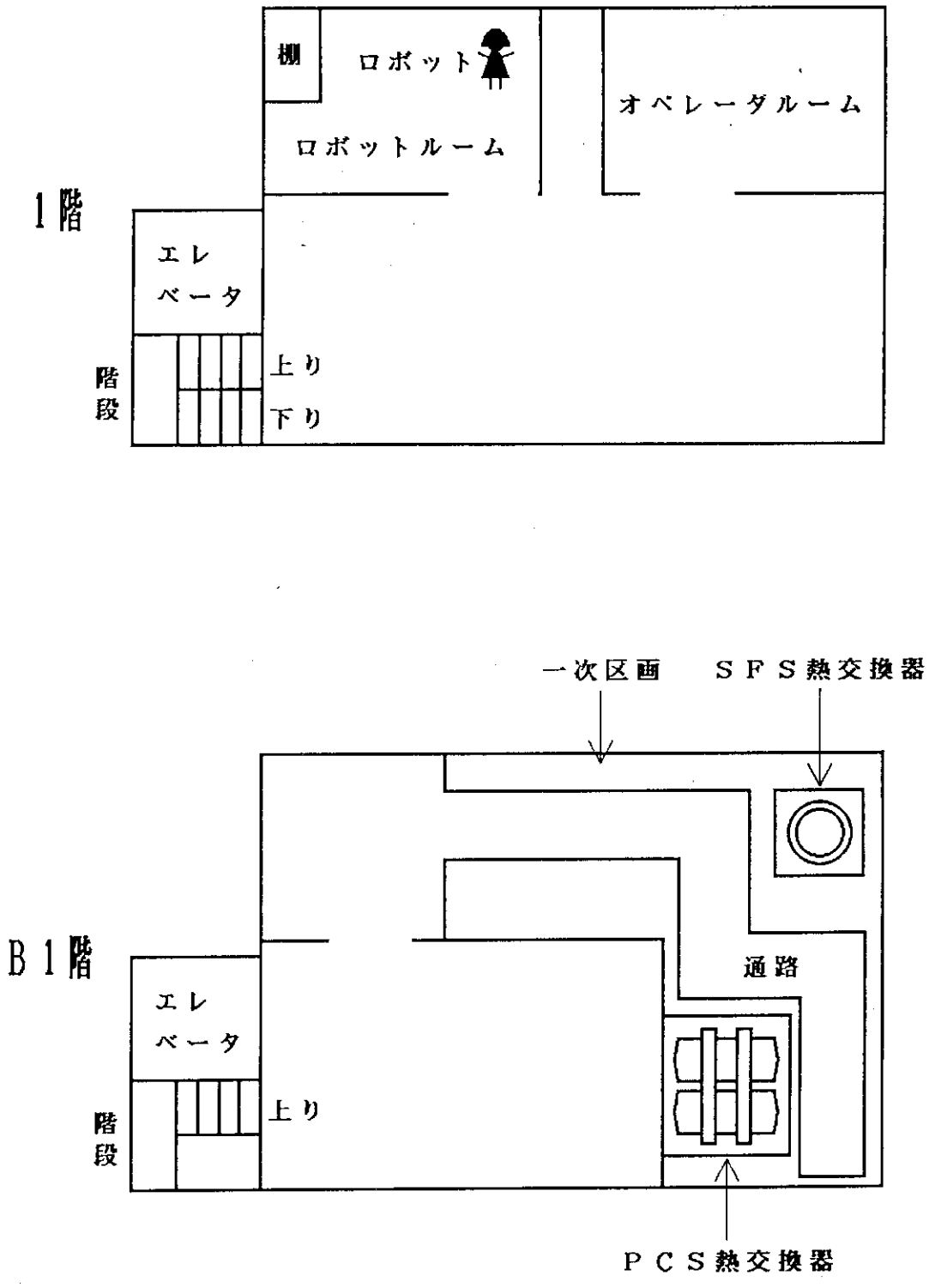


Fig. 2.1 A prototype world model for the intelligent robot

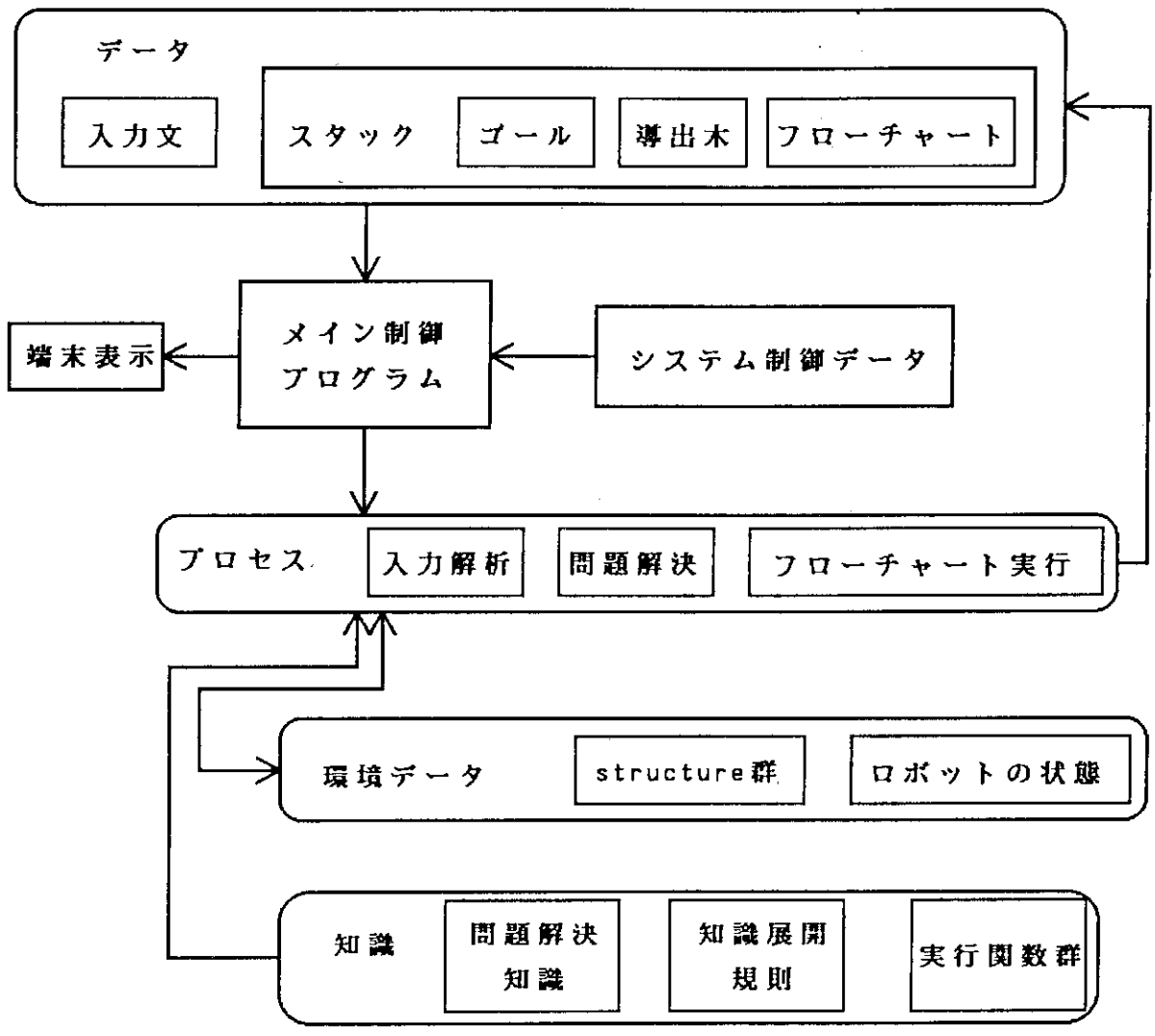


Fig. 2.2 Overview of the prototype system

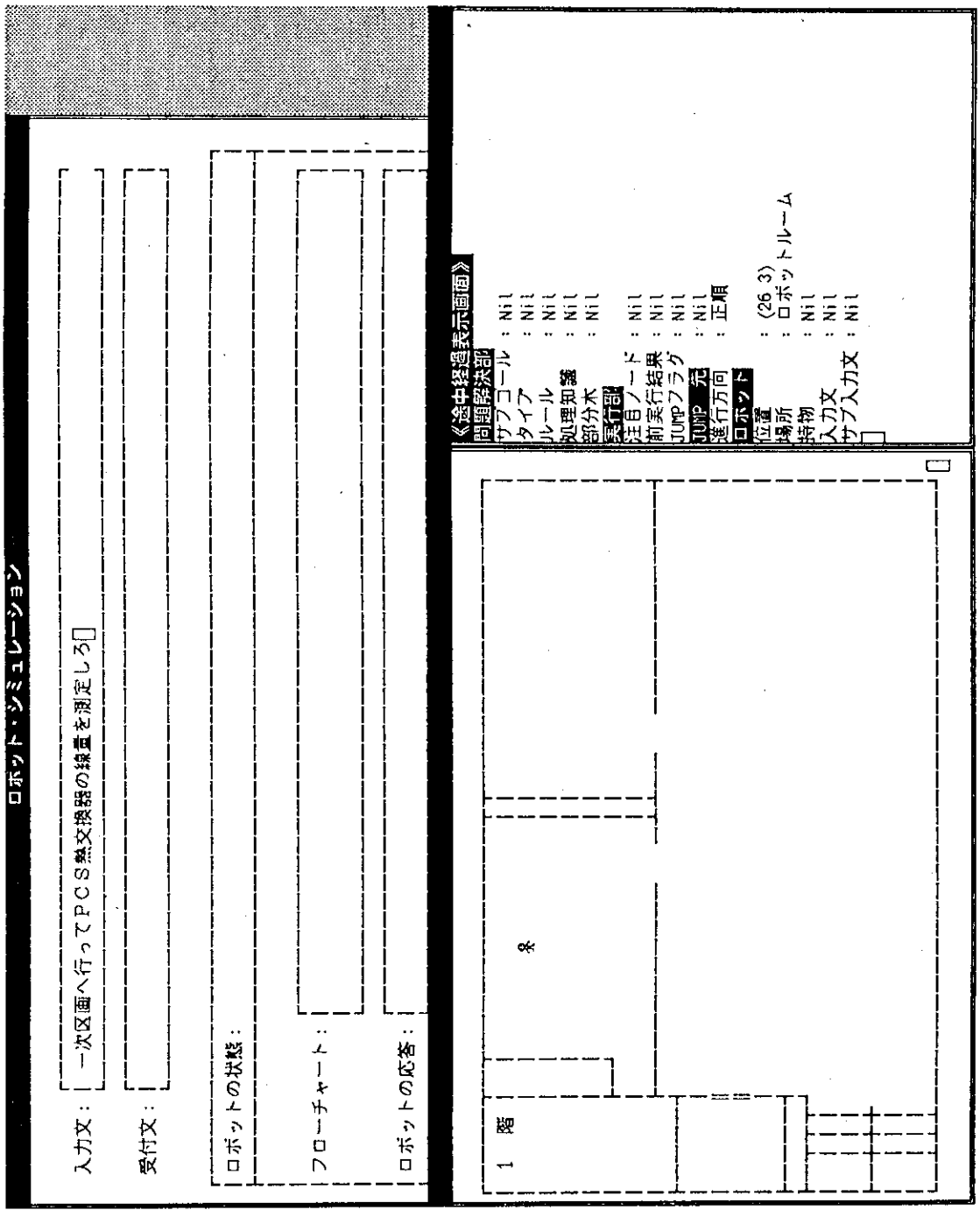


Fig. 2.3 Display image of the prototype system

3. 視覚認識の研究

3.1 はじめに

3.1.1 経　緯

原子炉施設で作業するロボットは，施設内の状況や作業対象物を視覚的に認識する能力を持つ必要がある。

人間動作シミュレーション技術の研究（HASP）では，ロボットの視覚装置（ハードウェア）で収集，生成された画像情報からロボットの見ているシーンを理解，認識する過程について昭和63年秋より研究を開始した。

しかし，筆者らは，視覚認識の分野の知識，経験に乏しかったため，最初は，神経回路網理論（ニューロ手法）による画像認識の可能性を追求した。

このため，昨年度は，次の２つの認識実験を行った。

(1) バックプロパゲーション（BP）モデルによる手書き数字の学習実験[1)]

(2) ネオコグニトロン・モデル[2)]による手書き数字及び簡易画像の認識実験[1)]

両実験ともドットイメージの入力データを使用した。この結果，BPモデルは極めて簡単に作成でき，手書き数字程度なら数字の大きさと向きをある程度正規化しておけば認識できることがわかった。ネオコグニトロン・モデルは入力する文字の大きさや提示位置のずれなどを吸収するニューロ・モデルとして有名であるが，モデル作成にかなりのノウハウを必要とする。このため，文献2）で示された手書き数字に関しては，この文献の処理アルゴリズムを参考にしてこれらの数字をほぼ認識するモデルを作成できたが，簡易画像を認識するモデルは，うまく作成することができなかった。

また，これらのニューロ手法は文字認識及び単純図形の認識には適用可能であるが，知能ロボットの視覚装置からドットイメージで入力される一般画像（多数の物体が一枚の画像中に内包されている画像）の理解に直接適用することは非常に難しいことがその後の文献調査でもわかった。

このため，一般画像の理解には既存の画像処理（エッジ処理，領域分割等）手法を適用して，まず画像中の物体抽出と抽出した物体の特徴抽出を行い，ニューロ手法は抽出されたそれぞれの物体の解釈や画像中の文字の認識に使用することにした。

そこで，今年度は既存の画像処理手法の適用と一般画像に対する視覚認識方法の検討を行った。

HASPにおける視覚認識研究は，現在まだその最も初期的な段階にあり，視覚認識研究分野の現状を調査・学習する段階を脱してはいない。平成元年度前期は，既存の視覚認識分野の研究成果について調査し，その一部，特に画像処理技術について実際にプログラムを整備し，その実用性・汎用性を評価した。後期は，その結果を踏まえ既存の視覚認識手法の一部をHASPに適用するとともに，両眼立体視の画素対応問題をニューロ・モデルの一種であるHopfieldモデル[3)]によって解くことを試みた。

HASPの視覚認識研究においては，その研究段階を既知環境下における視覚認識，半既知環境

下における視覚認識，そして未知環境下における視覚認識という三段階を設定し，今年度は，既知環境下と半既知環境下の視覚認識方法を検討した。これを3.1.2及び3.1.3項に示す。

3.1.2　既知環境下における視覚認識方法（Fig.3.1）

既知環境下では，施設（装置等も含む）に関する詳細なデータベース（施設ＤＢ）が完備されており，ロボットはその施設ＤＢにアクセスできること，即ちロボットは施設内のいかなる装置についてもその名称，形状，設置されている位置等を知ることができることを仮定している。この環境下における認識方法は以下のように行う。

まず，ロボットは施設内に設置された位置認識用マークをマーク検出用テンプレートを使用して画像内から検出し，マークの中心点及びマークのサイズから自己の位置を算出する。次に算出されたロボットの位置に立って，ロボットが見ている方向の映像を施設形状ＤＢを使って生成し，ロボットが実際に見ている画像と対比することで，見ている画像の画素レベルの内容（その画素が施設形状DBのどの物体に属するかなど)まで理解することが可能である。精度の良い位置検出法及び大きさやずれのある２枚の画像の高速な対比方法が主な研究内容である。

今年度は，このマーク検出を利用したロボット・ナビゲーション方法について検討した。この詳細は3.2節に記述する。

3.1.3　半既知環境下における認識方法（Fig.3.2）

半既知環境下では，施設及び装置等の大まかなデータベース（視覚認識ＤＢ）が完備されており，ロボットは装置等の特徴や輪郭画像を使ってこの視覚知識DBをアクセスできることを仮定している。この環境下における認識方法は以下のように行う。

まず，ロボットは両眼視画像から距離画像を生成し，Hough変換[4]等によって物体の面を検出し，画像中の物体を分離，抽出する。次に，各物体の特徴量及び正規化した輪郭画像を使って視覚認識ＤＢをアクセスし，その物体が何かを理解する。両眼立体視の画素対応法，視覚知識ＤＢの構造と検索法，物体抽出法，各種ニューロ手法の適用等が主な研究内容である。

ここで使用する両眼立体視は，画像から立体形状を抽出する作業が，三角測量という幾何学的原理のみに基づいており，いわゆる" 対象領域に依存した知識 " などの助けを必要としない明解な処理で済むという非常に有用な特徴を持つ，また，ここで得られる距離画像は，シーン中の物体を分離・抽出する最も有効なデータとなる。しかし，この三角測量を行うためには，その前に，両眼立体視（両眼視画像）の画素対応を行っておく必要がある。しかし，この画素対応問題の有効な解法はまだ存在しないというのが現状である。

そこで，HASPでは，この未解決な両眼立体視の画素対応問題をニューロ手法の一種であるHopfieldモデルによって解くことを試みている。3.3節にこの概要を示す。

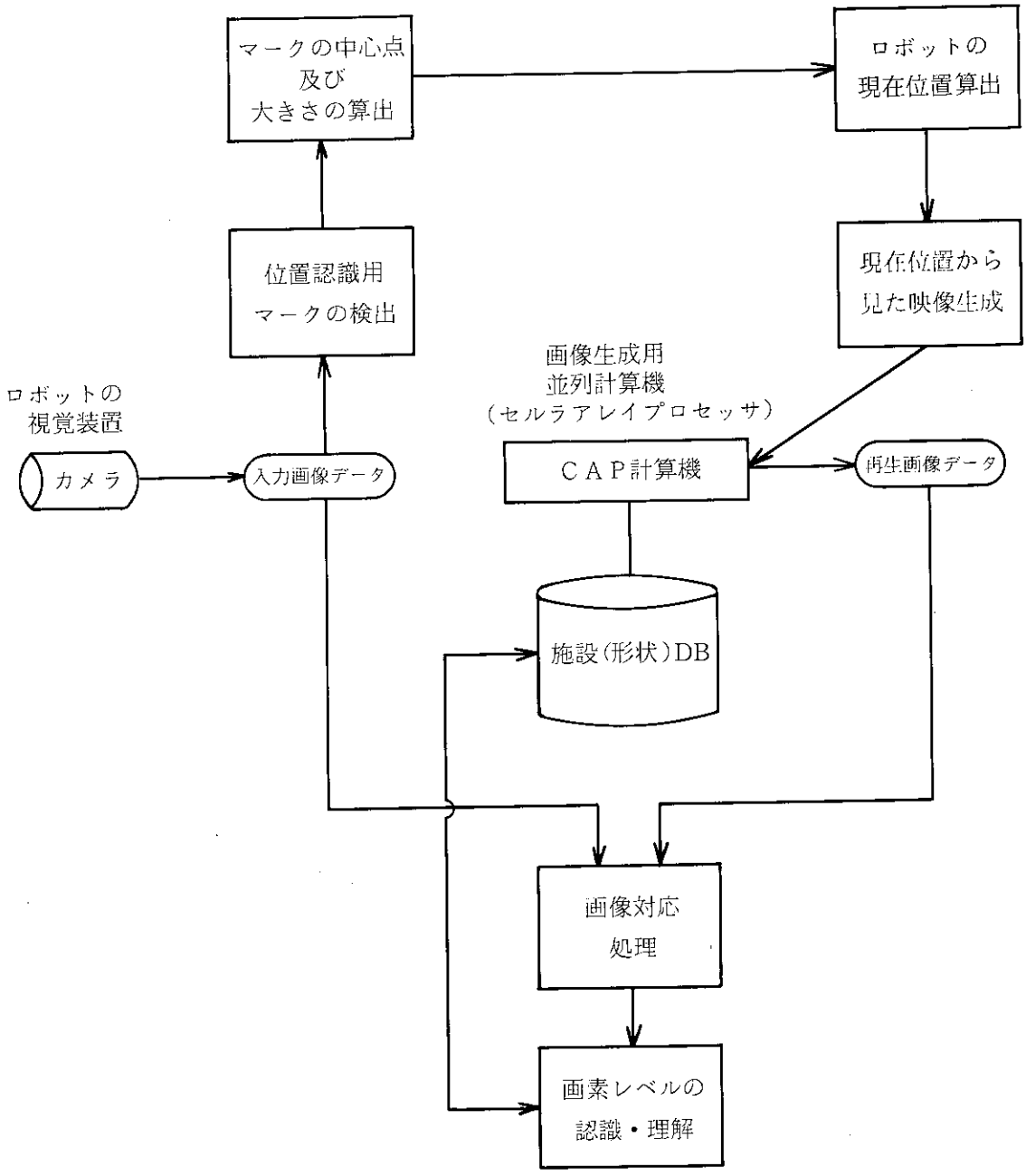


Fig. 3.1 A recognition process of the vision in a known environment

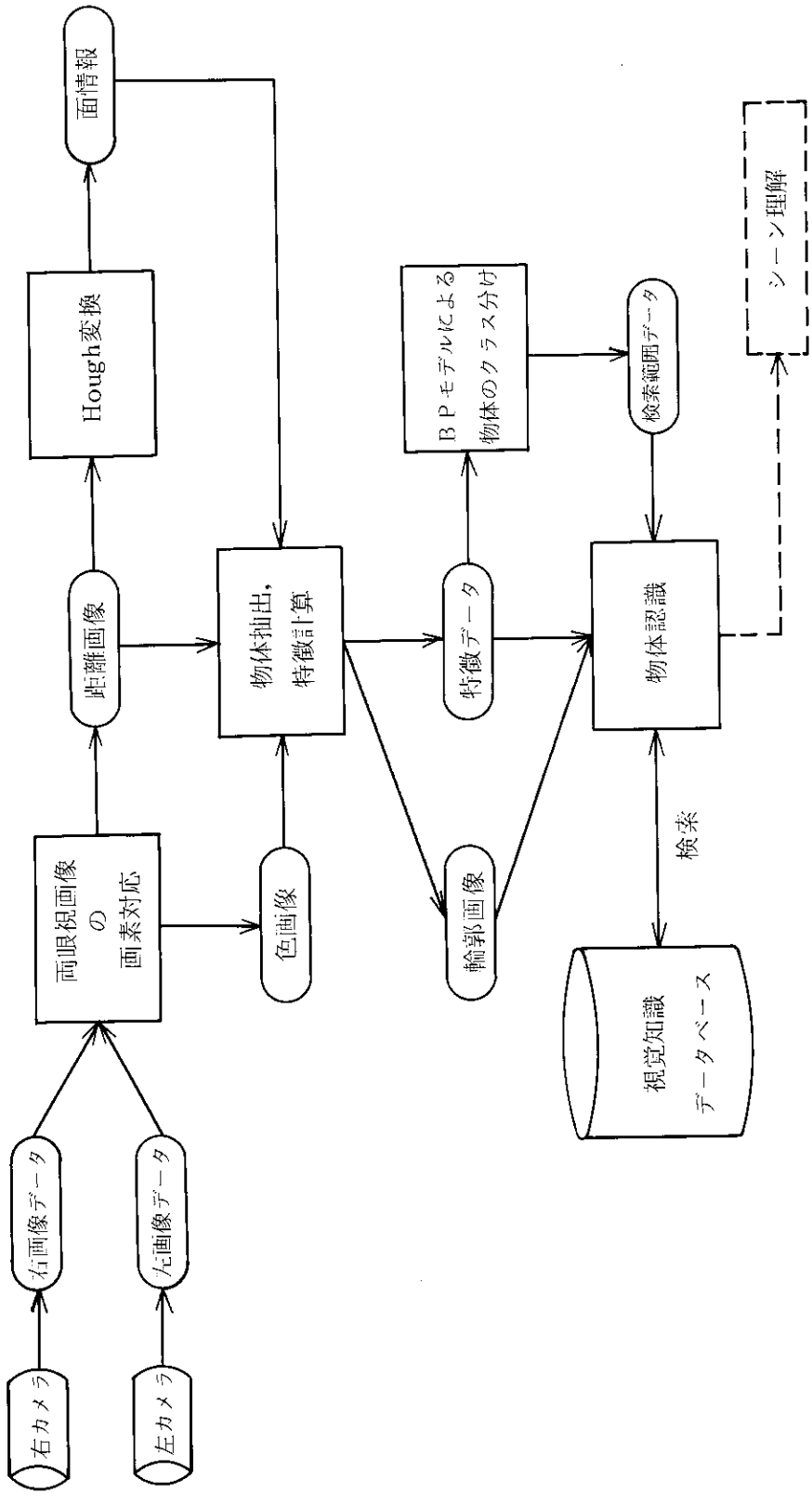


Fig. 3.2 A recognition process of the vision in a semi-known environment

3.2 マーク検出によるロボットナビゲーション

ここでは，既知環境下における視覚認識研究のHASPへの初歩的な応用として，既知環境内を移動する知能ロボットのナビゲーション手法について研究を行った結果を報告する。

3.2.1 ロボットナビゲーション

ロボットのナビゲーションは，主に自己位置の同定と障害物の回避という2つの具体的内容から成っている。移動ロボットにとって自己の位置を知ることはロボット自身が地図を持ち，その地図上での位置を知ることである。通常移動ロボットは移動の履歴から概略の現在位置を算出する。例えば車輪の回転の累積によりロボットの自己位置を知るデッドレコニングは移動ロボットに不可欠な機能である。HASPにおけるシミュレーションでは，知能ロボットは二足歩行によって移動するため車輪などは有していないが，歩数などを用いて何らかのデッドレコニングを行う必要はある。しかし，デッドレコニングには移動に伴い推定位置の確度が低下していくという問題がある。また，何らかの理由でロボットが自己の位置を見失った時や電源投入時に移動の履歴無しに自己の位置を知る必要があり，このような自己位置同定問題は興味深いものである。一方で，ロボットの目前に何らかの障害物があった場合，これを回避することも移動ロボットにとっては重要なことである。

ロボットナビゲーションでは，ロボットが視覚センサを持ち，その画像情報を基にナビゲーションを行うことが多い。この手法は，得られた画像情報の中から基準あるいは手がかりとなる物体を探して自己位置の同定や障害物の回避を行う。画像中の手がかりとしては，FAにおける運搬ロボットなどのようにマークやライン[5]を使うもの，既知環境内の物体を使うもの[6]，移動施設内の平行線を使うものなどがある。いずれにせよロボットのナビゲーションには移動に伴う実時間性，検出確度，システムの小型軽量化及び予定外の環境の変化への適応性などが必要とされるが，これらをすべて満足するシステムは確立されておらず，現在視覚以外のセンサとの融合などの研究が行われている。

本研究では，移動ロボットは原研が導入したCAP映像作成システム[7]によりモデリングされ，同様に計算機上に構築された原研3号炉（JRR-3）の模擬空間内を移動する。CAPシステム及びJRR-3のモデリングについては本報告の第5章を参照するものとし，ここでは解説を省略する。今年度は，既知環境下における視覚認識の研究として，Fig. 3.3に示すディジタル画像処理の組み合わせによるロボットのナビゲーションアルゴリズムを構成した。ここで記述するロボットナビゲーションの特徴は，ロボットの環境内にナビゲーションの手がかりとなる標識，すなわち自己位置認識マークを設置していることである。次節では，この特徴的なマークの機能について述べる。

3.2.2 位置認識マーク

本研究において自己位置認識のための標識として採用したマークの形状をFig. 3.4に示す。縦400 mm横300 mmの白地のプレートに100 mm×100 mmの黒い正方形が描かれている。なお，これ以後の説明を明確にするためにマークの中心，すなわち2つの正方形の接点をC点とする。

C点を基準として黒い部分の左端をL点，右端をR点，上端をU点，下端をD点とする。また，これ以後の説明における座標系をFig.3.4のように定義する。

このマークは施設内において，ロボットの目（視覚センサ）と同一の高さに位置することを仮定し，その仮定の下で位置認識の基準標識として機能するように設計されている。もし，マークがロボットの目と同じ高さにあれば，ロボットの視覚画像内におけるマークの画像は上下方向の幾何学的歪を含まない。すなわち，図中のｙ軸方向の像のゆがみは無視し得る。したがって，視覚システムの撮像系のパラメータ及び視覚スクリーンへの縮尺が一定であれば，ロボットの視覚画像におけるマークのｙ軸方向の大きさ，ＵＤ間の距離はロボットとマークとの距離のみに依存することになる。マークのｙ軸方向の画像情報からロボット・マーク間の距離が算出されたとすると，次にマークのｘ軸方向の画像情報に対し，前述のロボット・マーク間距離によるマーク像の大きさを補正すれば，ロボットとマークとのｘ軸方向の変位の情報が得られる。つまり，ロボットの視覚像内のマークのｘ軸方向とｙ軸方向の比，すなわちLR間とUD間の距離を比較することにより，ロボットがマークの正面方向に対してどれほどの角度の方向に位置しているかが算出できる。このようにFig.3.4に示した位置認識マークは２つの黒色正方形によってロボットの自己位置算出の基準として機能する。マークの色採は，このような機能を発揮するために，一般的な撮像ハードウェアを意識して，最もコントラストが明確になるよう，白と黒の組み合わせを選んだ。

3.2.3　マーク検出

ここではFig.3.3に示したマーク検出処理について具体的に記述する。図からもわかるようにマーク検出処理は，本研究のロボットナビゲーションにおける最も重要な過程の１つである。Fig.3.5にマーク検出のアルゴリズムを示す。CAPシステムにより生成された視覚画像は，まず，色相判別を行ってマークの可能性のある候補画像を限定している。これによってマーク検出に不必要なエッジの検出が防止でき，エッジ画像の二値化やテンプレートマッチングといった処理を容易にしている。さらに，色相判別以後の処理過程において視覚画像内の全画素に対して処理を行う必要はなくなり，画像メモリ容量の減少と計算時間の短縮が可能である。ただし，本研究ではメモリ容量，計算時間共にさほど深刻な状況にはないためにこのようなメモリや時間の節約は行っていない。しかし，将来，より大きな画像あるいは複数の画像の処理を行う際や現実の知能ロボットに組み込まれた小型計算機によって視覚処理を行う際には，このような手法が有効となると考えている。

次にエッジ検出を行い，原画像を二値画像に変換している。これはテンプレートマッチング処理の際に対称物の位置を正確に求めるために行われる。通常の画像では自己相関が比較的高く，テンプレートマッチングを行うと，テンプレートと画像の類似度は対称物の存在する画素を中心になだらかなピークを形成する。すなわち，テンプレートが対称物（本研究ではマーク）の真の位置から少し離れてもかなり高い類似度を示す。ところが，対称物の位置を正確に求めるためには類似度の分布ができるだけ鋭い方が望ましい。そのための１つの方法としてここでは濃淡画像によるマッチングではなく，これに比べて鋭い類似度の分布を示す輪郭画像のマッチングを行っている。

さらに細線化処理を施して，二値化された輪郭画像の線幅を1（画素）としている。これは，テンプレートマッチングのためのテンプレートを用意する上で必要不可欠な処理である。もし，マッチングを行う輪郭画像の線幅が決定していなければ，画像の線幅に応じて複数のテンプレートを用意しなければならない。また，前述のような鋭い類似度分布のためにも，輪郭画像の線幅は1であることが理想的である。

以上のような前処理の後にテンプレートマッチングを行うことで，マッチングの精度，信頼性の向上を図っている。テンプレートマッチングの結果，視覚画像内のマークの有無，マークの位置及びU，D，L，Rの各点の位置が求められ，次の自己位置算出過程に渡される。

（色相判別）

Fig.3.5に示した色相判別[8]処理では，RGB表色系で表された視覚画像をRGB座標上から次式を用いて座標変換し，色彩を評価している。

$$\begin{Bmatrix} I \\ Q \end{Bmatrix} = \begin{Bmatrix} 0.596 & -0.274 & -0.322 \\ 0.211 & -0.523 & 0.312 \end{Bmatrix} \begin{Bmatrix} R \\ G \\ B \end{Bmatrix} \qquad (3.1)$$

ここでI，Qは変換後の座標軸である。この座標変換によってマークの色彩（白と黒）はI－Q座標の原点に投影される。I－Q座標系ではFig.3.6に示すように原点のまわりに色相が並び，原点から遠ざかるに伴って各色相の彩度（色の鮮明度，純粋度）が上昇する。したがって，このときI－Q座標上の原点付近の点に投影されるような色彩はマークに相当している可能性が高い。言い換えれば，その他の点に投影される色彩を持つ画素はマーク検出の対称外とすることができる。

（エッジ検出）

エッジ検出に関しては，いくつかの手法についてプログラムを整備し，視覚画像を実際に処理して評価した。ここで取り上げた手法はいずれも空間微分法によるエッジ検出であったが，評価の結果Sobelオペレータ[9]によるエッジ検出を採用した。二次元画像の濃度関数f（x，y）の一次微分は，

$$d = \frac{\partial f(x, y)}{\partial x} i + \frac{\partial f(x, y)}{\partial y} j \qquad (3.2)$$

という形のベクトル量であり，これは離散系では

$$d = \{f(x+1, y) - f(x, y)\} i + \{f(x, y+1) - f(x, y)\} j \qquad (3.3)$$

という差分演算となる。ここでiはx方向，jはy方向の単位ベクトルを表す。今回のエッジ検出においては，濃度変化の方向に係わらず，エッジ検出するので，濃度変化の強度が求まればよい。したがって，

$$S = \sqrt{\left\{\frac{\partial f(x, y)}{\partial x}\right\}^2 + \left\{\frac{\partial f(x, y)}{\partial y}\right\}^2} \qquad (3.4)$$

Sobelオペレータでは，これをFig.3.7のような3×3近傍で計算しており，離散系では次式が得られる。

$$\frac{\partial f(x, y)}{\partial x} = (X_2 + \alpha X_1 + X_8) - (X_4 + \alpha X_5 + X_6)$$
$$\frac{\partial f(x, y)}{\partial y} = (X_6 + \alpha X_7 + X_8) - (X_4 + \alpha X_3 + X_2) \qquad (3.5)$$

ここでαは定数で，$\alpha=2$としたのがSobelオペレータである。さらに濃度変化強度の計算を次のように簡単化している。

$$S=|X_2+2X_1+X_8-X_4-2X_5-X_6|+|X_6+2X_7+X_8-X_4-2X_3-X_2| \tag{3.6}$$

本研究では，この3×3マスクオペレータを用いてエッジ強調を行い，判別分析法[10]によってしきい値を選択し，画像の二値化を行っている。

（しきい値選択）

判別分析法は濃度値の集合を2つのクラスに分けたときのクラス間の分散を考え，これが最大になるようにしきい値を選択する手法である。通常，分散σ^2は，

$$\sigma^2=\sum_{i=1}^{k}(X_i-\mu)^2 f(X_i) \tag{3.7}$$

で表される。ここでμは平均値，$f(X_i)$はX_iの生起確率である。いま，入力画像の濃度レベルを0～255とし，濃度レベルiの頻度をn_iとすると，全画素数N及び各レベルでの確率P_iは以下のようになる。

$$N=\sum_{i=1}^{255}n_i$$

$$P_i=n_i/N$$

濃度レベルtをしきい値として$C_0=\{0\sim t\}$，$C_1=\{t+1\sim 255\}$の2つのクラスに分割することを考えるとき，C_0の生起確率$W_0=W(t)$

さらに，

$$C_0\text{の平均値}\quad \mu_0=\sum_{i=1}^{t}\frac{iP_i}{W_0}=\frac{\mu(t)}{W(t)}$$

$$C_1\text{の平均値}\quad \mu_1=\sum_{i=t+1}^{255}\frac{iP_i}{W_1}=\frac{\mu-\mu(t)}{1-W(t)} \tag{3.8}$$

ただし，

$$\mu=\sum_{i=0}^{255}iP_i,\quad \mu(t)=\sum_{i=0}^{t}iP_i$$

である。したがって，2つのクラス間の分散$\sigma^2(t)$は

$$\sigma^2(t)=W_0(\mu_0-\mu)^2+W_1(\mu_1-\mu)^2$$

$$=\frac{\{\mu\cdot W(t)-\mu(t)\}^2}{W(t)\{1-W(t)\}} \tag{3.9}$$

この式を用いて分散$\sigma^2(t)$が最大となるようなtを求め，これをしきい値として二値化を行った。こうして得られた二値画像にさらに細線化を施す。細線化とは線画像の本質を変えることなく，その線幅が1になるように画素を削除する処理である。次に細線化について記述する。

（細線化処理）

ある1画素とその周囲8近傍の画素値X_0～X_8をFig. 3.7のようにとったとき，画素X_0における連結数[11] N_Cを次のように定義する。

$$N_C=\sum_{k\in S}(\overline{Xk}-\overline{Xk}\cdot\overline{Xk+1}\cdot\overline{Xk+2}) \tag{3.10}$$

ここで，S＝{1，3，5，7}，$\overline{Xk}=1-Xk$，$X_9=X_1$とする。
連結数N_Cは0～4の値をとる。この連結数N_Cは直観的には画素が結合している連結成分の個数を表すものとして捉えられる。すなわち，連結数$N_C=1$となる画素X_0が削除可能な画素である。ここで採用した細線化処理は，この連結数に基づく削除可能性を評価して画面全体の処に処理を反復しながら画素を削っていき，これ以上削れなくなったところで，残された1画素の集合を最終結果とする。

このような前処理の結果，マークの中心部分はFig.3.8のような二値画像になっていると考えられる。したがって，このような形状の線画を検出するためのテンプレートとして本研究では，Fig.3.9に示す6×6テンプレートを提案した。Fig.3.8と3.9と比較すれば，このテンプレートの機能が理解できる。すなわち，Fig.3.8の画像の黒点を1，白点を0としてFig.3.9のテンプレートに重ね合わせる。このとき各画素において画像の数値（0又は1）とテンプレートの数値（－2，－1，0，1）との乗算を行い，テンプレート内全画素における和を計算する。これを評価値とすれば，テンプレートがマーク中心部に位置したとき，評価値は著しく大きな値をとる。したがって，このテンプレートを全面画内においてサーチすればマークの検出が行える。このようにしてマークの中心を検出した後に，上下左右に画素値1（黒）の点を走査してU，D，L，Rの各点を検出した。

3.2.4　自己位置の算出

すでに述べた如く視覚画像内のマークのy軸方向の大きさはロボットとマークとの距離のみに依存する。前節のマーク検出の結果得られたUD間距離を基にロボット・マーク間距離を算出する。y軸方向の端点UD間の画素数は実世界の200 mmの長さに相当していることからFig.3.10を参照して次式が得られる。

$$d=\frac{512\times\alpha}{N\tan\theta} \tag{3.11}$$

ここでαはマーク上の黒色正方形の一辺の長さ100 mmを示し，NはUD間の画素数，θはロボットの撮像装置の視野角を示している。CAPシステムの仮想視覚スクリーンは焦点から距離1のところにあり，その画素数は512個である。したがって，視野角θの範囲の画像がスクリーン上では512画素に投影される。今，ロボット・マーク間距離をdとすると，512画素に相当する実世界における長さは$d\tan\theta$である。これに対し，実世界で100 mmであるマーク上の正方形の一辺αは，CAPシステムのスクリーン上にはN画素に投影されている。したがって，この両者の比を考えれば，式（3.11）が得られ，α，N，θから距離dが算出される。

次にx軸方向の端点（L，R）間画素数から，いま求めたdを用いてマークから見たマーク正面方向に対するロボットの方向角を求める。Fig.3.11はマークとロボットの位置関係を鉛直上方から見たものである。この図から幾何学的に次の式が導出される。式（3.12），（3.13），（3.14）はそれぞれ図中の①，②，③に相当する。

$$z=\frac{d\cos\phi}{d\sin\phi+\alpha}x+\frac{\alpha d\cos\phi}{d\sin\phi+\alpha} \tag{3.12}$$

$$z = \frac{d\cos\phi}{d\sin\phi - \alpha}x - \frac{\alpha d\cos\phi}{d\sin\phi - \alpha} \tag{3.13}$$

$$z = \frac{d\cos\phi}{d\sin\phi}x \tag{3.14}$$

ここでd及びαは前述のとおり，ϕはマークの正面方向（z方向）に対するロボットの方向角である。③と直交する直線④を考えると，④の方程式は

$$z = -\frac{d\sin\phi}{d\cos\phi}x \tag{3.15}$$

と求まる。④はロボットの視覚スクリーンに平行な直線である。式(3.12)，(3.15)より点R'の座標は，

$$\left\{\frac{-\alpha d\cos^2\phi}{d+\alpha\sin\phi}, \frac{\alpha d\sin\phi\cos\phi}{d+\alpha\sin\phi}\right\}$$

と求まる。同様に点L'の座標は，

$$\left\{\frac{\alpha d\cos^2\phi}{d-\alpha\sin\phi}, \frac{-\alpha d\sin\phi\cos\phi}{d-\alpha\sin\phi}\right\}$$

したがって線分R'L'の長さℓは，

$$\ell = \frac{2\alpha d^2\cos\phi}{d^2-\alpha^2\sin^2\phi} \tag{3.16}$$

となる。つまり，マークのx軸方向の端点間距離RLはマークとロボットの位置関係，すなわち方向角ϕによって画像に幾何学的歪を生じ，その見かけの長さR'L'が式(3.16)に相当する。さらに，これがFig.3.10に示した視覚スクリーンに投影されるから，スクリーン上，つまり視覚画像内ではマークのx軸方向端点間距離は次式に示すN個の画素に相当する。

$$N = \mathrm{int}\left\{\frac{256\ell}{d\tan\theta}\right\} \tag{3.17}$$

ただし，intは整数化を意味し，θは撮像系の視野角を示す。本研究においては，$\theta = 20$ deg.としてシミュレーションを行った。Nが求まることにより，式(3.16)，(3.17)よりマークとロボットの間の方向角ϕを算出した。

3.2.5 まとめ

Fig.3.12にマーク検出処理の各過程における出力像を示す。色相判別，エッジ検出などの各処理過程の役割とそれを順調にこなしている様子がうかがえる。マーク検出処理における結論を以下にまとめる。

① 色相判別によるマーク候補の限定の結果は極めて良好で，これにより，不必要なエッジの検出が未然に防止できた。しかし，本研究ではCAPシステムによる模擬画像を入力としたために特に好ましい結果に至った可能性があり，実用を考える際には撮像系ハードウェアの特性を考慮する必要がある。

② エッジ検出に関しては，Laplacianオペレータ，Prewittオペレータなどの数種の手法について検討した結果Sobelオペレータを採用した。

③ 細線化に関して，線画像の端の部分が著しく縮退するという問題が生じた。この原因を調査した結果，図形の端点保存性を吟味する必要があることがわかった。そこで，削除可能画素に対して，その画素が線図形の端点である場合にはこれを削除しない，という処理を加えることによりこの問題を解決した。

④ 細線化の結果，図形の交差部で線のずれが生じることがある。ずれの大きさは最大で1画素（512×384画素中）である。

⑤ テンプレートマッチングに関しては，本研究の特徴でもある位置認識マークに適用可能なテンプレートを考案し，良好な結果を得た（Fig.3.12参照）。

次に自己位置算出過程における結論を示す。前述の自己位置算出手法に基づきプログラムを整備し，CAPシステムによりモデリングされたJRR－3内で自己位置算出を試みた結果⑥，⑦の結論を得た。

⑥ ロボット・マーク間距離の算出では，ロボットとマークの真の距離が10m以内の範囲における最大誤差は，真の距離の5％程度であった。

⑦ マークの正面方向に対するロボットの方向角の算出においては，その最大誤差は前述の範囲で7deg.であった。

⑥，⑦に示した結果を踏まえ，ロボットの撮像系がズーム機能（最大4倍ズーム）を有することを仮定し，これによって精度の向上を図った。その結果⑧の結論を得た。

⑧ 最終的に既知環境下における自己位置同定手法としてFig.3.13に示すアルゴリズムを得た。ズーム機能に関しては，通常倍率，2倍，3倍，4倍ズーム時の4枚の画像を取り込むことを想定して自己位置算出を行った。これにより，Table 3.1に示す精度を得た。

⑨ 以上のように視覚情報によってロボットは自己の位置を確認できることがわかった。よってこの位置情報を基にして施設形状データベースを参照することにより，ロボットの持つ環境情報に無い物体，すなわち予想外の障害物を認識し，これを回避できる可能性が見い出された。

Table 3.1 Errors of the calculation of the distance

マーク・ロボット 間距離(mm)	最大誤差 (%)	
	ズーム無し	ズーム使用
1,000	0.3	0.1
2,000	0.6	0.1
3,000	0.7	0.2
4,000	0.7	0.2
5,000	0.7	0.2
6,000	1.9	0.2
7,000	1.9	0.3
8,000	2.3	0.3

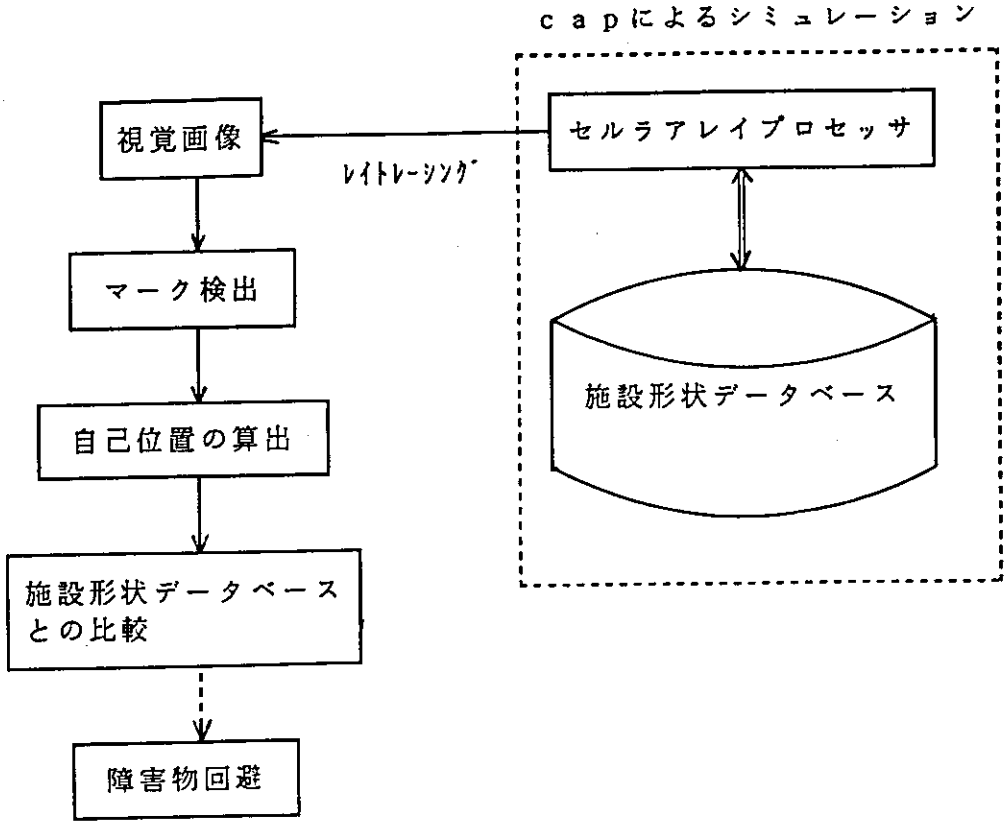


Fig. 3.3 The navigation method of the robot by using guide marks

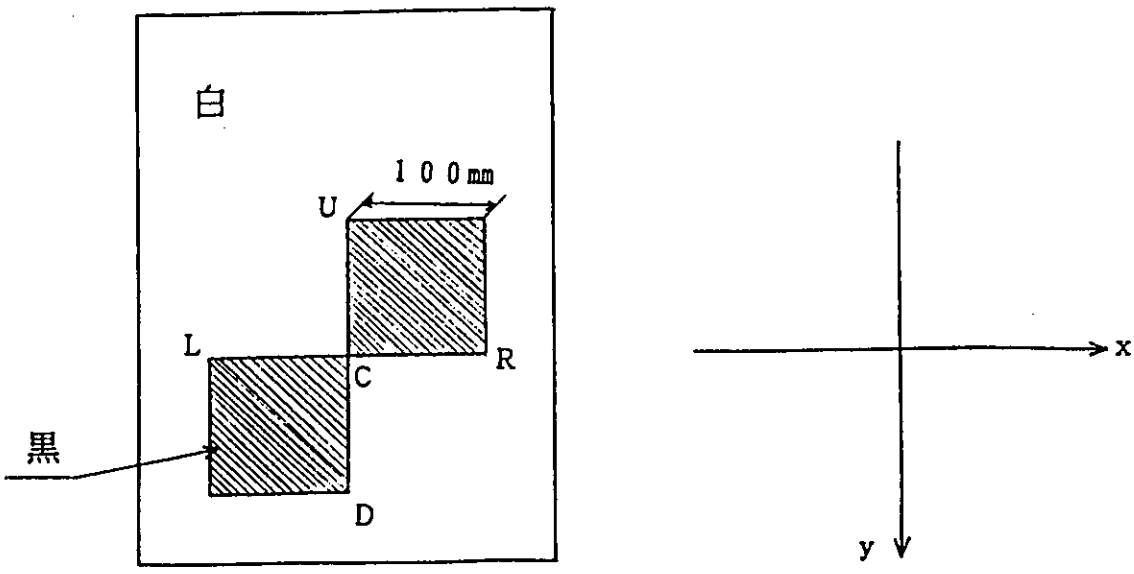


Fig. 3.4 The guide mark

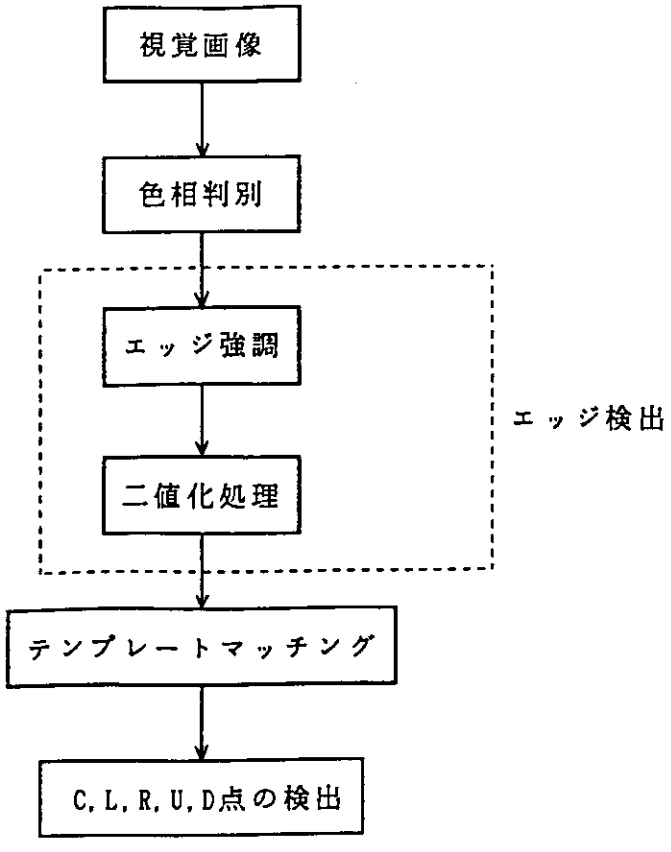


Fig. 3.5 The algorithm of the mark detecting process

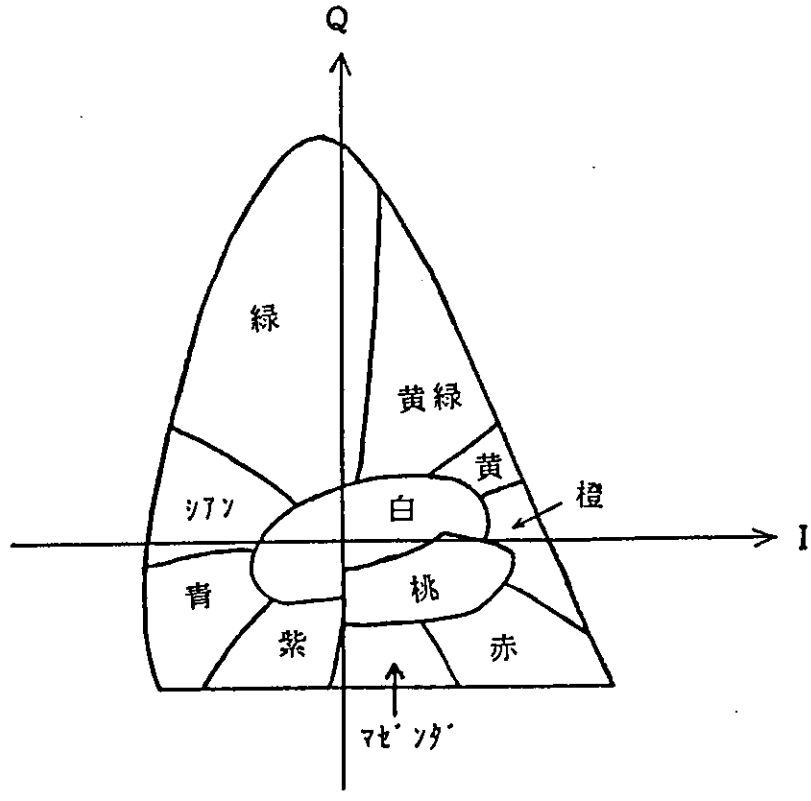


Fig. 3.6 Colors on the I-Q chromaticity coordinates

X 4	X 3	X 2
X 5	X 0	X 1
X 6	X 7	X 8

Fig. 3.7 The definition of the 3*3-pixels

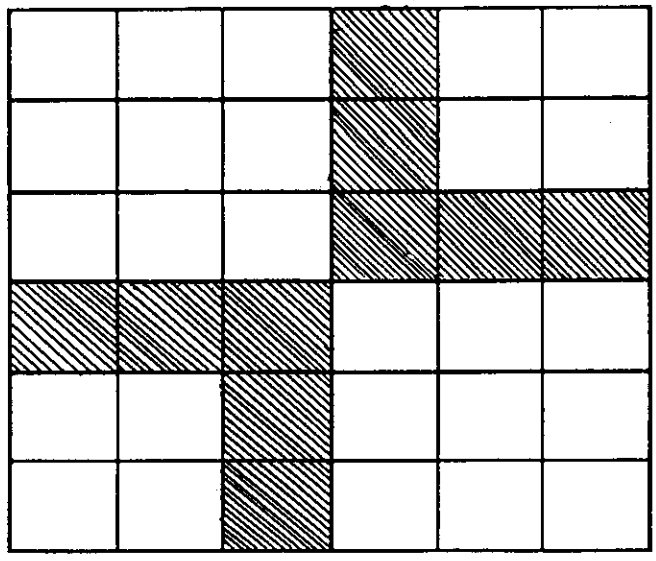


Fig. 3.8 The output of the thinning process

− 2	− 1	0	1	− 1	− 2
− 1	− 1	0	1	− 1	− 1
0	0	0	1	1	1
1	1	1	0	0	0
− 1	− 1	1	0	− 1	− 1
− 2	− 1	1	0	− 1	− 2

Fig. 3.9 The template for the mark detection

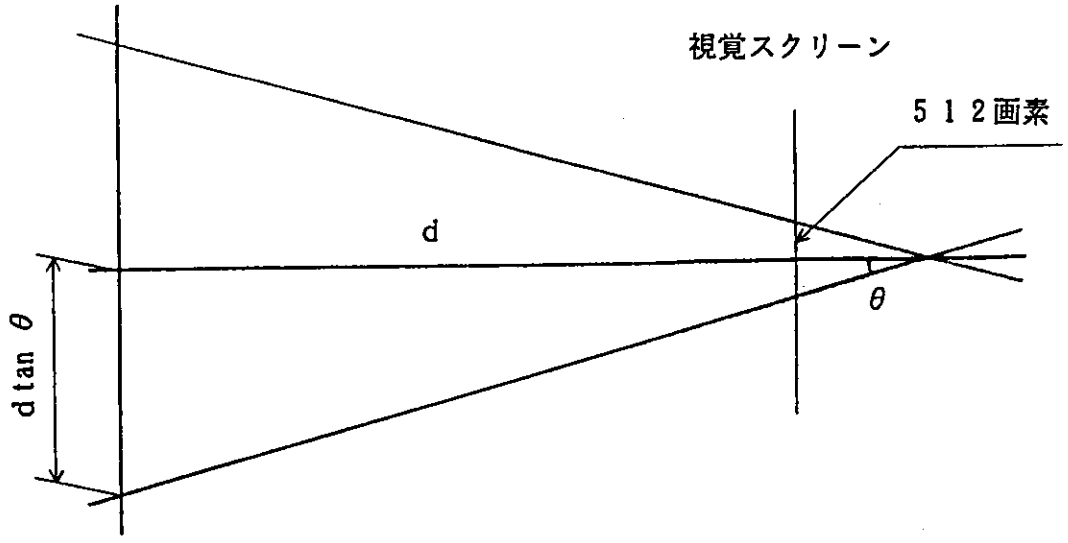


Fig. 3.10 The transfer function of the image screen

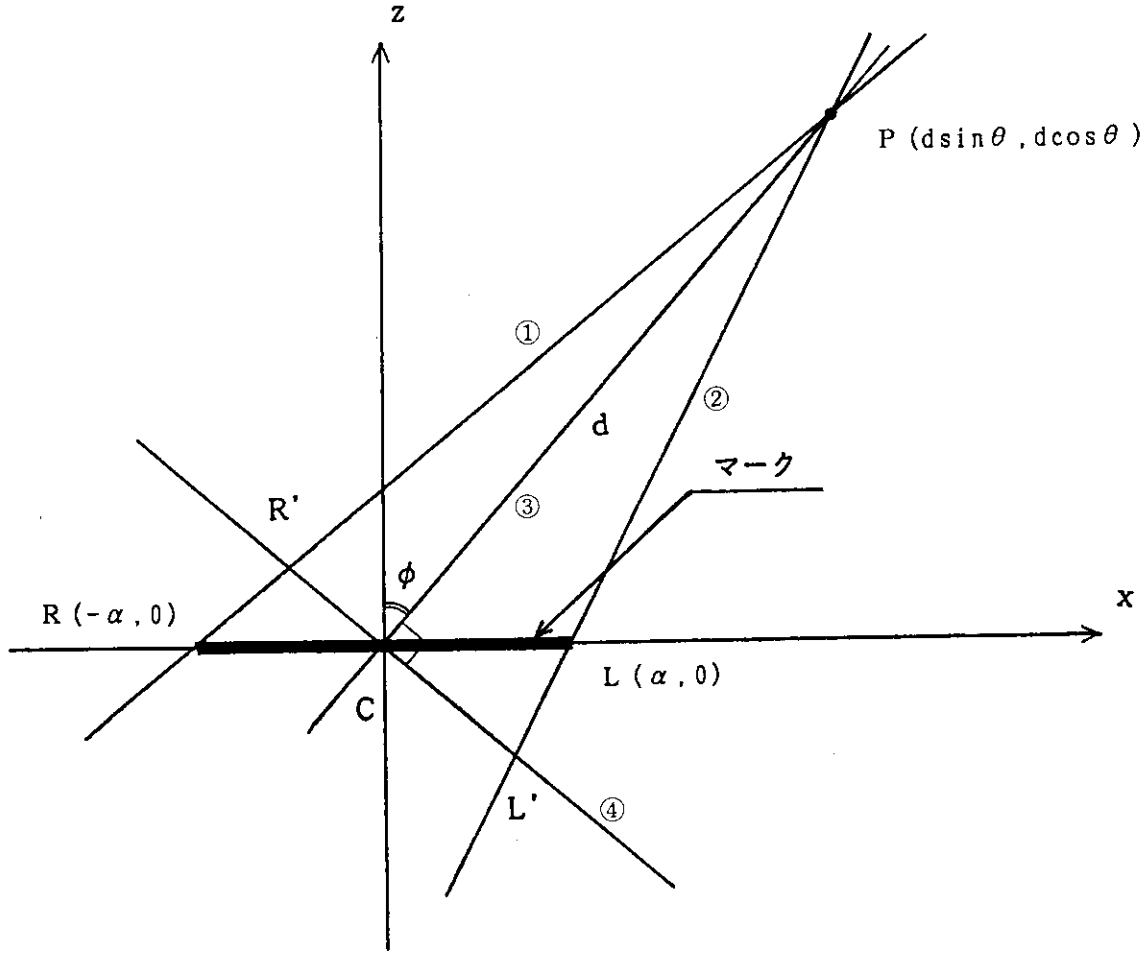
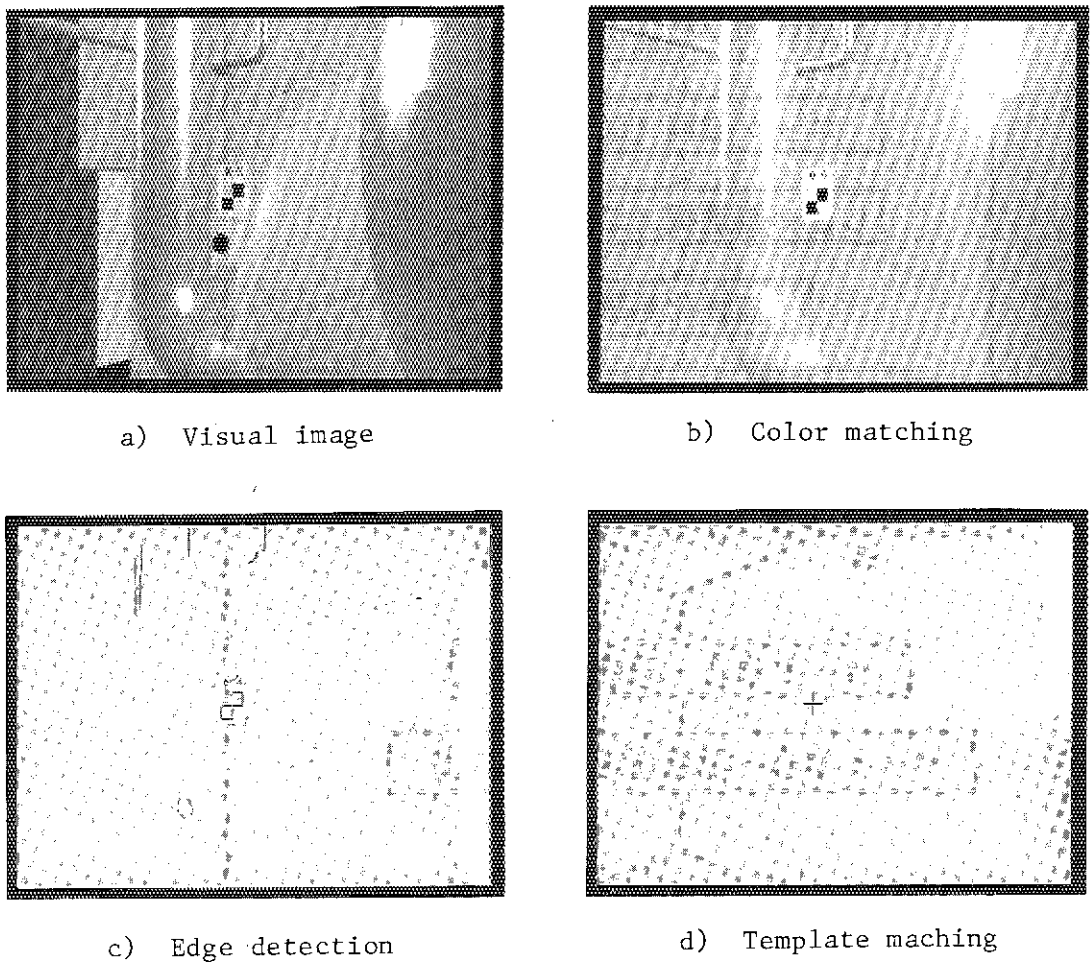


Fig. 3.11 The local relation between the mark and the robot



a) Visual image

b) Color matching

c) Edge detection

d) Template maching

Fig. 3.12 Results of each process of the mark detection

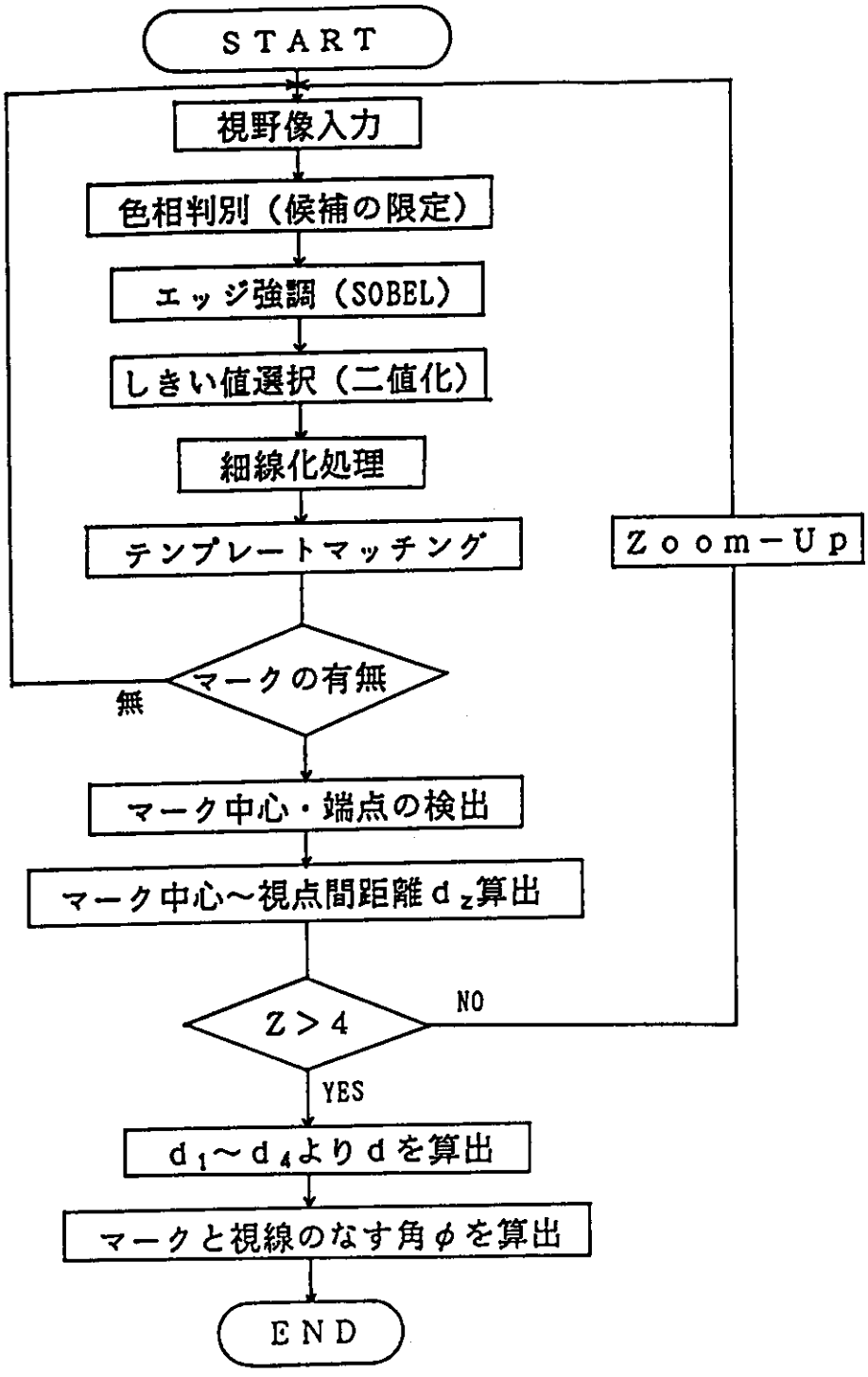


Fig. 3.13 The navigation method of the robot utilize zooming function

3.3 Hopfieldモデルによる両眼立体視画素対応問題の一解法

3.3.1 はじめに

両眼立体視では，左右画像間の対応づけ（対応問題）が非常に困難な課題である[12]。

両眼立体視では，Fig.3.14に示すように，左右のカメラのレンズの中心を結ぶ直線とシーン中の任意の一点で決定される平面（エピポーラ平面）が，左右の画像面と作る交線はエピポーラ線と呼ばれ，エピポーラ面上にのっているシーン点の左右画像上の像は，必ず，左右一対のエピポーラ線上に存在する性質がある。したがって，立体視における左右画像間の対応問題は，左右エピポーラ線上の画素間の対応関係を求める問題に分解できる[13]。

しかし，この問題も1エピポーラ線上の画素数が多くなると，結局，多対多の組合せ最適化問題となり，膨大な計算量が発生する。したがって，この計算量の削減手法が立体視における重要な研究課題になっている。

これに対して従来，左右画像からそれぞれのエッジを検出し，検出されたエピポーラ線上のエッジ群を動的計画法によって対応づける方法，2個のエッジで区切られたエピポーラ線上の区間毎に画素を動的計画法で対応づける方法[14]などが提案されている。

しかし，動的計画法によるこれらの対応づけは，

- 両端の対応づけは，既になされていることを仮定している
  （両端の対応づけは，別な方法で行っておく必要がある。），
- 画素レベルの対応づけは，計算時間がかかりすぎる，
- 少し複雑な画像を処理すると，誤った対応づけを行う確率が高い

などの問題点があり，両眼立体視の画素対応問題を解く実用的手法にはなっていない。

本稿では，この両眼立体視の対応問題を

- エッジを抽出せずに直接画素レベルで対応づける。
- 一度，整数2次計画問題として記述し，それをニューロ手法の一種で最適解探索に利用できるHopfield型モデルに変換して解く。

これにより，並列計算機を使えば瞬時処理が可能になる道を開くことを試みる。また，この方法は，動的計画法では誤った対応づけを招きやすかった右目で見えて，左目で見えない部分や対応関係のないエッジが生ずる画像であっても対応関係がない部分であることを解として出力できる。

3.3.2 モデル化の概要

ロボットの左右のカメラから得られる静止映像を左画像，右画像と呼び，それぞれの画像の1エピポーラ線上の画素数を2Nとし，以下，右画像の左からi番目の画素を右画素i，左画像の左からj番目の画素を左画素jと略称する。

左右画像の対応づけは，左画像の左半分の領域（左目優位領域と呼ぶ），と右画像の右半分の領域（右目優位領域と呼ぶ）を固定し，右画像の左半分を左画像に，左画像の右半分を右画像に対応させることで行う。

Marrは，両眼立体視における対応点が満たすべきヒューリスティックな条件として以下の3個の制約をあげている[15]。

(i) 適合性(compatibility): 対応する点は類似の性質をもつ。

(ii) 一意性（uniqueness）：一方の画像上の点は，他方の画像上の２点以上と対応しない。

(iii) 連続性（continuity）：視差はなめらかに変化する。

本稿では，これらの条件を満たす画素対応問題をまず数理計画法[16]を使用して整数２次計画問題として記述し，次にこの整数２次計画問題をニューロ手法のHopfield型モデルに変換し，Hopfieldモデルの最適解探索アルゴリズムを使用して高速に解く方法を検討した。以下にその概要を示す。

3.3.3 両眼立体視画素対応問題の整数２次計画モデル

(1) 変　数

$2N$：左画像，右画像の一行の画素数

x_{ij}：左画素 j と右画素 i の対応関係を示す変数

$$x_{ij} = \begin{cases} 1 & \text{：対応関係有} \\ 0 & \text{：〃　無} \end{cases}$$

y_{ij}：右画素 i がすべての左画素 j と対応関係が無い場合（例えば，右目で見えて左目で見えない部分などの場合），

その右画素 i が左画素 j の位置で対応がとれていないことを示す変数

$$y_{ij} = \begin{cases} 1 & \text{：右画素 i が左画素 j の位置で対応がとれていないことを示す} \\ 0 & \text{：関係なし} \end{cases}$$

z_{ij}：左画素 j がすべての右画素 i と対応関係が無い場合（例えば，左目で見えて右目で見えない部分などの場合），

その左画素 j が右画素 i の位置で対応がとれていないことを示す変数

$$z_{ij} = \begin{cases} 1 & \text{：左画素 j が右画素 i の位置で対応がとれていないことを示す} \\ 0 & \text{：関係なし} \end{cases}$$

これらの変数の定義域をFig. 3.15に示す。

(2) 制約条件

① 画素対応の一意性

(i) 右画像の左半分の画素は，左画像の１画素と対応するかまたは対応しないかのいずれかである。

$$\sum_{j=i}^{2N} (x_{ij} + y_{ij}) = 1 \quad (i = 1, \cdots N) \tag{3.18}$$

(ii) 左画像の右半分の画素は，右画像の１画素と対応するか又は対応しないかのいずれかである。

$$\sum_{i=1}^{j} (x_{ij} + y_{ij}) = 1 \quad (j = N+1, \cdots 2N) \tag{3.19}$$

② 画素対応の連続性

Fig. 3.16に示す左右エピポーラ線上で要素の順序が入れ替わる例を除いて，左右エピポーラ線上で互いに対応する部分要素の順序は入れ替わらない。

(i) 右画素の左半分の画素の連続性

$$\sum_{k=i}^{2N} k\,(x_{i,k}+y_{i,k}) \leqq \sum_{k=i+1}^{2N} (x_{i+1,k}+y_{i+1,k}) \quad (i=1,\ \cdot\cdot N-1) \tag{3.20}$$

(ii) 左画像の右半分の画素の連続性

$$\sum_{k=1}^{j} (2N-k+1)(x_{k,j}+z_{k,j}) \leqq \sum_{k=1}^{j-1} (2N-k+1)(x_{k,j-1}+z_{k,j-1})$$

$$(j=N+2,\ \cdot,\ 2N) \tag{3.21}$$

動的計画法では，左右エピポーラ線上で互いに対応する部分要素の順序は替わらないという仮定を設ける必要があり，Fig.3.16に示す左右エピポーラ線上で対応する要素の順序が入れ替わる対象は正しく処理することができない[13]。これに対し，本稿の式（3.20），（3.21）の制約は，Fig.3.16のケースも正しく処理する（図中のcとfのどちらか一方は，対応関係がないという解がでる）ように定議されている。

(3) 評価関数

評価関数Gは，以下の3つのサブ評価関数g_1, g_2, g_3の和として定義され，両眼立体視の画素対応問題はGを最小化する変数x_{ij}, y_{ij}, z_{ij}を求める問題になる。

$$G = g_1 + g_2 + g_3 \tag{3.22}$$

① 画素対応の適合性コスト

左右画像の対応する画素は類似の色をもつ。この左右画像の画素間の色の違いの度合いをコスト（cost）と考え，全画素の対応コストの総和g_1を最小にすることで最適な画素対応を図る。

$$g_1 = \sum_{i=1}^{N} \sum_{j=i}^{2N} \left\{ (r_i^R - r_j^L)^2 + (g_i^R - g_j^L)^2 + (b_i^R - b_j^L)^2 \right\} x_{ij}$$

$$+ \sum_{j=N+1}^{2N} \sum_{i=1}^{j} \left\{ (r_j^L - r_i^R)^2 + (g_j^L - g_i^R)^2 + (b_j^L - b_i^R)^2 \right\} x_{ij} \tag{3.23}$$

ここで，r_i^R, g_i^R, b_i^R, は右（Right）画素iのそれぞれ赤（red），緑（green），青（blue）の輝度値，r_j^L, g_j^L, b_j^L, は左（Left）画素jのそれぞれ赤，緑，青の輝度値である。

② 対応画素のない場合のペナルティ・コスト

右目で見えて左目で見えない箇所やFig.3.16のように対応する要素の順序が入れ変わる対象がある場合などには，左右画像に対応のない画素を生じさせる（最適解としてy_{ij}やz_{ij}が1の値をとる）ことが必要である。しかし，対応のない画素はg_1のコストが0であり，対応のある画素のコストより小さくなり，不都合が生じる。このため対応のない画素が生じた場合は，対応のある画素のコスト以上のペナルティ・コストを与える必要がある。このペナルティ・コストの総和をg_2で表す。

$$g_2 = \sum_{i=1}^{N} \sum_{j=i}^{2N} p \cdot y_{ij} + \sum_{j=N+1}^{2N} \sum_{i=1}^{j} p \cdot z_{ij} \tag{3.24}$$

ここで，pは対応画素のない場合のペナルティ・コストである。

③ 画素対応の連続性コスト

制約条件の式（3.20）と（3.21）は，対応する部分要素の順序は一部の例外を除いて入れ

替わらないことを単に示したもので，視差はなめらかに変化することを示したものではない。

ここでは，視差の変化の大きさをコストとしてとらえ，その総和をg_3で表す。

$$g_3 = M \cdot \sum_{i=1}^{N-1} \{ \sum_{k=i+1}^{2N} k(x_{i+1,k} + y_{i+1,k}) - \sum_{k=i}^{2N} k(x_{i,k} + y_{i,k}) - 1 \}^2$$

$$+ M \cdot \sum_{j=N+2}^{2N} \{ \sum_{k=1}^{j-1} (2N-k+1)(x_{k,j-1} + z_{k,j-1})$$

$$- \sum_{k=1}^{j} (2N-k+1)(x_{k,j} + z_{k,j}) - 1 \}^2 \qquad (3.25)$$

ここで，Mは連続性に対する重み係数である。

3.3.4　両眼立体視画素対応問題のHopfieldモデル

前項で両眼立体視画素対応問題を整数2次計画問題に定式化した。これをHopfieldモデル（0又は1の状態をとるユニット群の相互結合型ネットワークとそのエネルギー関数を定義することによって，ネットワークのエネルギーが安定な状態に移行する性質を最適化問題の解法に応用したモデル）に変換するためには，

(i)　不等式制約式をスラック変数を導入して等式制約式にする，

(ii)　すべての変数（ここでは導入したスラック変数）を0－1整数変数にする

ことが必要である。

また，数式記述を簡単化するため，前項で示した変数x_{ij}, y_{ij}, z_{ij}の定義域をFig. 3.17のように拡張する。

これらの修正を前項の整数2次計画問題に施すと以下のようになる。

修正後の整数2次計画問題

(1)　制約条件

$$\sum_{j=1}^{2N} (x_{ij} + y_{ij}) = 1 \qquad (i = 1, \cdots, N) \qquad (3.26)$$

$$\sum_{i=1}^{2N} (x_{ij} + z_{ij}) = 1 \qquad (j = N+1, \cdots, 2N) \qquad (3.27)$$

$$\sum_{k=1}^{2N} k(x_{i+1,k} + y_{i+1,k}) - \sum_{k=1}^{2N} k(x_{i,k} + y_{i,k}) = \sum_{l=1}^{Q} u_{il} \quad (i = 1, \cdots N-1) \qquad (3.28)$$

$$\sum_{k=1}^{2N} (2N-k+1)(x_{k,j-1} + z_{k,j-1}) - \sum_{k=1}^{2N} (2N-k+1)(x_{k,j} + z_{k,j})$$

$$= \sum_{l=1}^{Q} u_{jl} \qquad (j = N+2, \cdots, 2N) \qquad (3.29)$$

ここで，$u_{i,l}$, $u_{j,l}$は不等式制約式を等式制約式にするために導入したスラック変数で，0－1整数変数である。Qは，視差の変化の最大許容画素数を示す。

(2)　評価関数

$$G = g_1 + g_2 + g_3 \qquad (3.30)$$

$$g_1 = \sum_{i=1}^{N} \sum_{j=1}^{2N} \left\{ (r_i^R - r_j^L)^2 + (g_i^R - g_j^L)^2 + (b_i^R - b_j^L)^2 \right\} x_{ij}$$

$$+ \sum_{j=N+1}^{2N} \sum_{i=1}^{2N} \left\{ (r_j^L - r_i^R)^2 + (g_j^L - g_i^R)^2 + (b_j^L - b_i^R)^2 \right\} x_{ij} \qquad (3.31)$$

$$g_2 = \sum_{i=1}^{N} \sum_{j=1}^{2N} p \cdot y_{ij} + \sum_{j=N+1}^{2N} \sum_{i=1}^{2N} p \cdot z_{ij} \qquad (3.32)$$

$$g_3 = M \left\{ \sum_{i=1}^{N-1} \left(\sum_{l=1}^{Q} u_{il} - 1 \right)^2 + \sum_{j=N+2}^{2N} \left(\sum_{l=1}^{Q} u_{j,l} - 1 \right)^2 \right. \qquad (3.33)$$

Hopfieldモデルへの変換

次に，修正された整数２次計画問題をHopfieldモデルに変換する。Hopfieldモデルのエネルギー関数は次の形式である。

$$E = -\frac{1}{2} \sum_i \sum_j T_{ij} V_i V_j - \sum_i I_i V_i \qquad (3.34)$$

ここで，

V：変数ベクトル

T：２次項の係数行列

I：１次項の係数ベクトル

である。

式（3.26）～（3.33）で定式化された整数２次計画問題を式（3.34）のエネルギー関数に変換する。このため，変数に付けられている制約及び式（3.26）～（3.29）の制約条件を式（3.17）の形の評価関数に直す必要がある。

変数はすべて０－１整数変数，制約条件式はすべて等式制約式に修正しているので，これらの評価関数は以下に示すような制約の違反量ϕ_1とϕ_2に定義できる。

i）整数変数のペナルティ関数：ϕ_1

$$\phi_1 = -\frac{A}{2} \left\{ \sum_S (1 - 2x_{ij})^2 + \sum_{S_R} (1 - 2y_{ij})^2 + \sum_{S_L} (1 - 2z_{ij})^2 \right.$$

$$\left. + \sum_{S_{RQ}} (1 - 2u_{il})^2 + \sum_{S_{LQ}} (1 - 2u_{jl})^2 \right\} \qquad (3.35)$$

ここで，$\{x_{ij} \in S\}$，$\{y_{ij} \in S_R\}$，$\{z_{ij} \in S_L\}$，$\{u_{il} \in S_{RQ}\}$，$\{u_{jl} \in S_{LQ}\}$，

である。

このペナルティ関数は，各変数が０又は１の値をとるときに最小となる。

ii）等式制約のペナルティ関数：ϕ_2

$$\phi_2 = -\frac{B}{2} \left[\sum_{i=1}^{N} \left\{ 1 - \sum_{j=1}^{2N} (x_{ij} + y_{ij}) \right\}^2 + \sum_{j=N+1}^{2N} \left\{ 1 - \sum_{i=1}^{2N} (x_{ij} + z_{ij}) \right\}^2 \right]$$

$$- \frac{C}{2} \left[\sum_{i=1}^{N-1} \left\{ \sum_{k=1}^{2N} k (x_{i+1,k} + y_{i+1,k} - x_{ik} - y_{ik}) - \sum_{l=1}^{Q} u_{il} \right\}^2 \right.$$

$$+ \sum_{j=N+2}^{2N} \left\{ \sum_{k=1}^{2N} (2N - k + 1)(x_{k,j-1} + z_{k,j-1} - x_{k,j} - z_{k,j}) \right.$$

$$- \sum_{l=1}^{Q} u_{jl} \}^2] \qquad (3.36)$$

このペナルティ関数は各等式条件を満足するとき最小となる。

Hopfield モデルの評価関数 ϕ は，整数２次計画問題の評価関数 G に上記のペナルティ関数 ϕ_1，ϕ_2 を加えたものにする。

$$\phi = \phi_1 + \phi_2 + G \qquad (3.37)$$

Hopfield モデルの解法

最初に，式（3.34）と式（3.37）を対応させ，式（3.34）のシナプシス荷重行列Tと各ユニットのバイアス・ベクトルI を求める。

ここで，

$$V = (x, y, z, u)^t \qquad (3.38)$$

である。

次に，入力データとVの初期値を与え，式（3.39），（3.40）の iteration により，収束解を得る。

$$U_i(n) = \sum_j T_{ij} \cdot V_i(n) \cdot V_j(n) + I_i \qquad (3.39)$$

$$V_i(n+1) = \frac{1}{2}\left\{1 + \tanh \frac{U_i(n)}{x_0}\right\} \qquad (3.40)$$

こうして，得られた解は，式（3.34）の極小解となるが，両眼立体視の画素対応問題の場合，最適解の存在する範囲が限定されているので，初期値を最適解の近くにとることができ，最適解が得られる確率は高いと思われる。

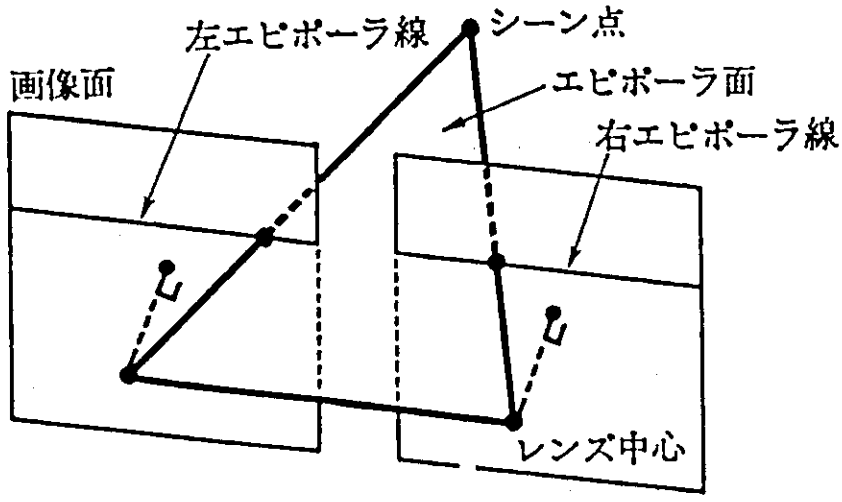


Fig. 3.14 Stereo vision[13)]

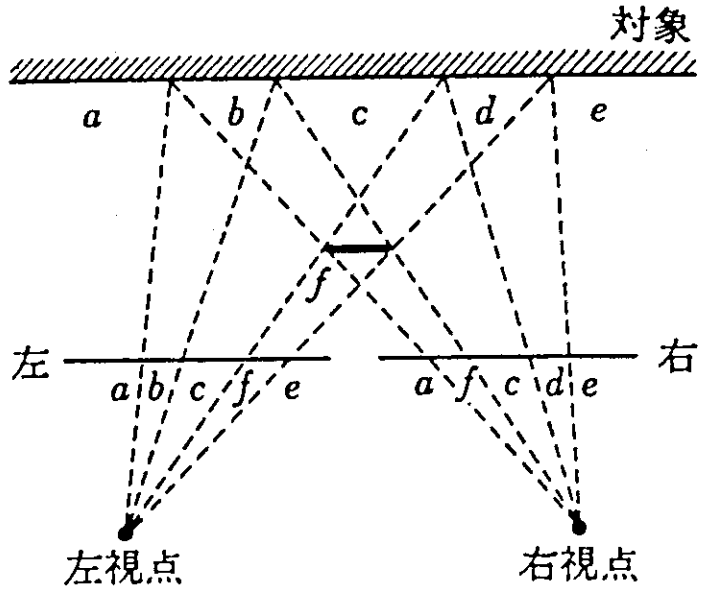


Fig. 3.16 An example of the different orders of elements between on the left and right epipolar lines[13)]

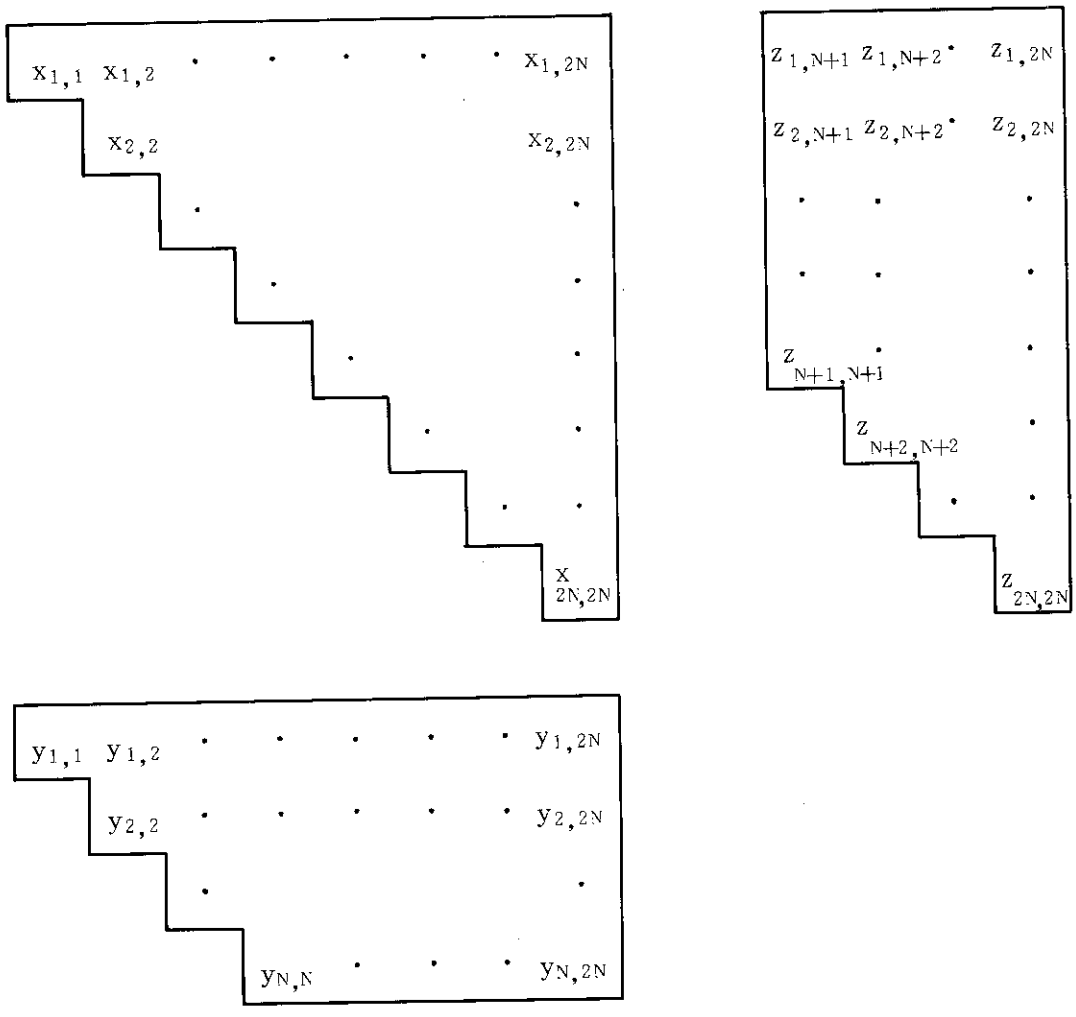


Fig. 3.15 The definition domains of x, y, z variables for our quadratic integer programming model

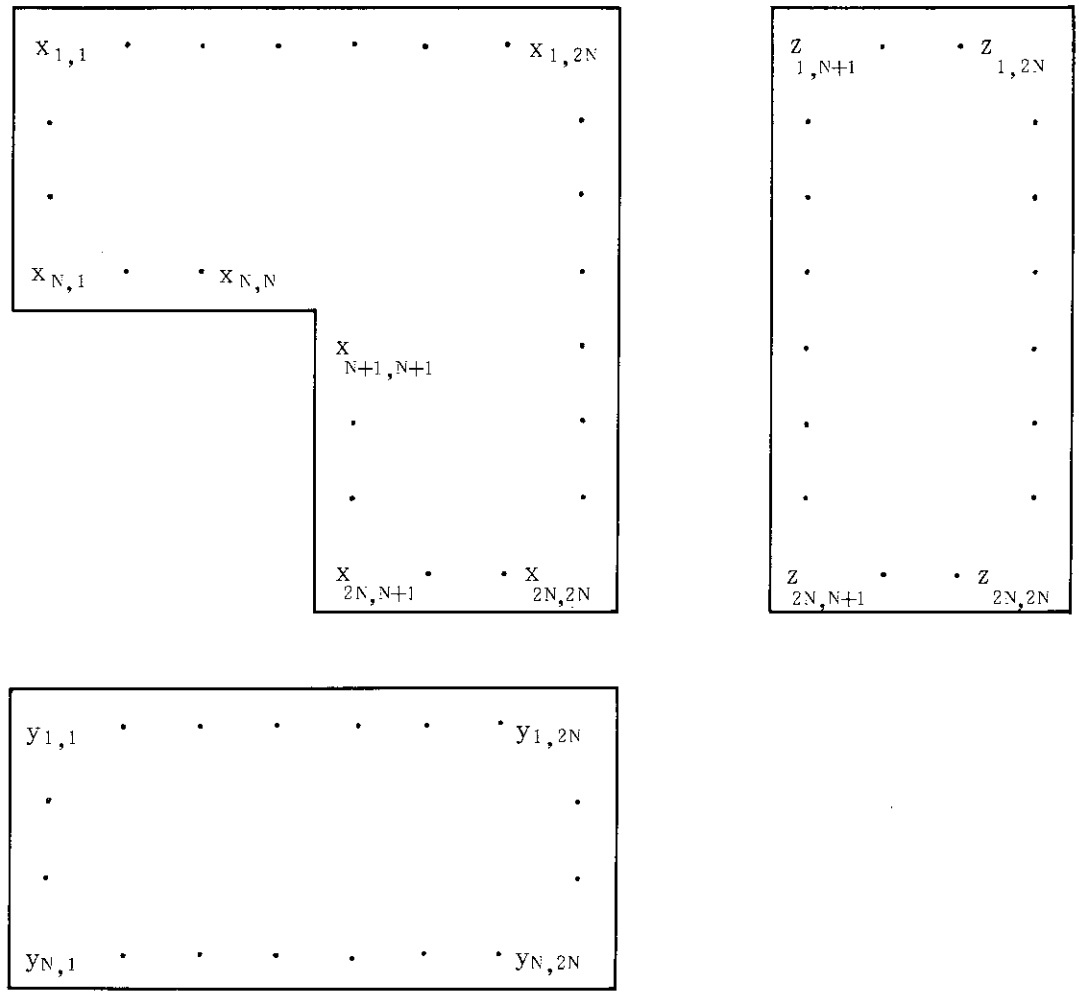


Fig. 3.17 The definition domains of x, y, z variables for our Hopfield-typed model

3.4 今後の方向

視覚認識は，今年度から本格的に研究を開始し，実用的手法としてマーク検出によるロボット・ナビゲーション方法を，基盤的技術としてHopfieldモデルによる両眼立体視画像対応問題の解法を検討した。前者は，HASPで来年度からハードウェア化の検討を開始するマイクロ二足歩行ロボットに搭載する視覚認識システムの検討へとさらに発展させ，後者は今年度検討した解法を実際の視覚データに適用して，本当に正しい画素対応ができるのかどうかを検証して行く予定である。その後は，3.1.2，3.1.3項に示した既知環境下の視覚認識方法と半既知環境下の視覚認識方法の中で示した検討課題に進むつもりである。

参考文献

1) 藤井：ニューロ手法と画像認識，富士通，サイエンティフィック・システム研究会 Newsletter, No.50, pp.7-62 (1990)
2) Fukushima, K.: A Hierarchical neural network capable of visual pattern recognition, Neural Networks, Vol.1, No.2, pp.119-130 (1988)
3) Hopfield, J.J. and Tank, D.W.: Neural computation of decisions in optimization problems, Biol. Cybern., Vol.52, pp.141-152 (1985)
4) Nevatia,R.(南敏監訳)：画像認識と画像理解，啓学出版（1986）.
5) Thorpe, C. et al.: Vision and navigation for the Carnegie-Mellon Navlab, IEEE Trans. Vol.10, No.3, pp.362-373 (1988)
6) 星野，他：原子力プラント用ロボット視覚，産業における画像センシング技術シンポジウム予稿集，pp. 291－296（1986）.
7) 藤崎，久米：セルラアレイプロセッサCAPによる動画作成，私信（1989）.
8) Pratt, W.K.: Digital image processing, Wiley (1978)
9) 尾上，他編：画像処理ハンドブック，昭晃堂（1987）.
10) 長尾：画像認識論，コロナ社（1982）.
11) 長谷川，他：画像処理の基本技法，技術評論社（1986）.
12) 鳥脇，白井：手法から見たコンピュータビジョンの動向，情報処理，Vol. 30，No. 9，pp. 1027－1034（1989）.
13) 大田，山田：動的計画法によるパターンマッチング，情報処理，Vol. 30，No. 9，pp. 1058－1066（1989）.
14) 大田，他：動的計画法によるステレオ画像の区間対応法，信学誌，Vol. J68－D，No. 4，pp. 554－561（1985）.
15) Marr, D.（乾，安藤訳）：ビジョン，産業図書，p 126（1988）.
16) 前田，他（監修）：コンピュータ・マネジメント・サイエンス・ハンドブック，オーム社，pp. 1－159（1971）.
17) 武田：神経回路網と組合せ最適化問題，数理科学，No. 289，pp. 14－21（1987）.

4. 二足歩行ロボット運動学的シミュレーション

4.1 はじめに

人間動作シミュレーションプログラム（HASP）では，プラント内における人間の作業や判断を機械に分担させる手法の確立を目的とし，そのための人間と人工環境との相互作用のモデル化を行っている。HASPにおける本研究の役割は機械的人間モデルによる動作シミュレーションを行うことにある。なぜ人間型なのか，なぜ二足歩行なのか，これはこのテーマにおける最も重要な問題でありかつ最後までついてまわる課題でもある。

プラント内で作業を行わせるロボットを考えるとき，限られた作業を行わせるのであれば個々の機能のみを充実させた最適なロボットが存在する。しかし，作業を限定しない本研究では，現在のプラントが多くの場合人間の存在を前提に作られていることを考慮して，ロボットは人間型であることが望ましいと考えている。また，移動機構に関しても現在では車輪やクローラ型が主流であるが，車輪やクローラ型よりも足のほうが遥かに地形の変化に対する適応性が高いことは明らかである。そのため，機械的人間モデルによる動作シミュレーションを行っている。つまり，人間型知能ロボットの設計の基盤技術を開発するためには，人間と施設とのメカニカルな関わり合いを分析することが必要なのである。

始めに，人間の基本動作の一つである二足歩行を取り上げた。本報告は，二足歩行の運動学的シミュレーションの元年度における作業報告である。

元年度の作業を要約すると次のようになる。

(1) 各歩容の映像化
(2) 歩容の修正
(3) オリジナルプログラム修正，ZMP入力方法の変更
(4) 腕可動型モデルの開発
(5) 映像化用プログラム作成
(6) アニメーション作成用プログラム作成
(7) 元年度版HASP動画作成
(8) 委託調査：両腕作業，二足歩行〔発進，停止〕

これらの内容について以下に報告する。

4.2 二足歩行ロボットモデル

作業報告の前に本研究で用いている二足歩行のロボットモデルについての概略を説明する。本研究のモデルはユーゴスラビアのヴコブラトビッチ（M.Vukobratović）のモデル[1]を基本にした人間型ロボットである。63年度までは腕を固定したモデル[2),3)]（12ボディ，7関節，8自由度）を用いて歩行のシミュレーションを行ってきたが，今年度に腕可動型のモデルの開発を行い

（4.3.2項参照）Fig.4.1に示すように12ボディ，11関節を持つモデルに変更を行った。関節2，5，7，8，10はY軸まわり回転の1自由度を持ち，1，3，4，6，9，11はX軸とY軸まわり回転の2自由度を持っている。但し，下肢のX軸まわりの回転は両足ともに等しく（Y－Z平面内の両足の傾きはともにθで常に等しいと仮定），骨盤は回転しないため全体の自由度は下肢7自由度，腰1自由度，片腕3自由度の合計14自由度である。ただし，現在は前腕の2自由度，足の1自由度は固定している（後述）。

剛体からなるロボットの運動学方程式は一般に式（4.1）で表される。

$$\vec{M}_p = \sum_{i=1}^{n} \varepsilon_{ip} \vec{M}_i + \sum_{i=1}^{n} \vec{M}_{ip} + \sum_{i=1}^{n} m_i [X] \vec{G} \qquad (4.1)$$

n　：ボディの数

ε_{ip}：ボディiが関節pのモーメントに寄与する場合は1，そうでない場合は0を与える。

$\vec{M}_i$：ボディiの回転運動の関節pへの寄与部分

$\vec{M}_{ip}$：ボディiに線加速度を与えるのに必要な関節pでのモーメント

m_i：ボディiの質量

$$[X] = \begin{pmatrix} 0 & -z_i & y_i \\ z_i & 0 & -x_i \\ -y_i & x_i & 0 \end{pmatrix}$$

(x_i, y_i, z_i)：ボディiの絶対座標系での重心の座標

$\vec{G} = (0, 0, g)$，　g：重力加速度

右辺第一項はボディiの回転運動が関節pへ寄与するモーメント，第二項はボディiに線加速度を与えるのに必要な関節pでのモーメント，第三項はボディiの重力による関節pへのモーメントである。式（4.1）においてロボットに作用する床反力の合成力の作用点を考えるとき，この点も一つの関節と考えればこの点におけるX軸とY軸まわりのモーメントの和，すなわち式（4.1）のX成分とY成分が零になるという力学的拘束条件の下に運動学方程式を定式化することができる。我々は，この点をZero Moment Point（ZMP）と呼び歩行中の安定な状態の指標として用いている。つまり，ZMPがロボットの支持脚の足底面内に存在すれば[4]歩行中ロボットはZMPで地面に固定されていると考えるのである。(ただし，現段階では定常歩行のみを取り上げており，単脚支持の繰り返しで両脚支持期を持たないという条件の下でシミュレーションを行っているため，ZMPがロボットの支持脚の足底面内に存在するが，両脚支持の場合にはZMPは必ずしも足底面内に存在するとは限らない）。なお，本文中足はFootを，脚はLegを意味する。

歩行時間は人間の歩行を前提に1歩1秒とし，1秒を30に分割して（t1～t31：動画作成用VTRの1秒の駒割数に依存）各時間メッシュにおいて運動学方程式を解き12ボディの重心の位置を計算した。しかし，運動学方程式に対する拘束条件が前述のように2つしかないため，本来動作から決定される下肢の関節角と腕の振り角及びZMPは基本歩行パターンとしてデータで

入力し，腰の曲がる角度ϕと両脚の左右の振れθを解として求めた。運動学方程式は非線形干渉2階微分方程式であり，基本歩行パターンの他にθ，ϕの角度と角速度を入力しθ，ϕの角加速度を計算するという形になる。そのため，基本歩行パターンの他にθとϕの初期角度と初期角速度（t＝0秒）を入力し，ルンゲ・クッタ，アダムス・モールトン法[5]を用いて，次のθとϕの角度と角速度を数値計算しながら1歩分（1秒間）の角加速度の計算を行っている。

ここでは，定常歩行のみを取り上げており歩行動作の周期性と1歩ずれた左右対称性のため，1歩の始めと終わりでθ，ϕの角度と角速度が一致するという境界条件を設定している。このため，1歩分の計算の後この境界条件のチェックを行い，境界条件が満足されなければ傾斜法やヤコビ法を用いて入力初期値の修正をし，境界条件を満足する初期値が見つかるまでこの処理を繰り返し行っている。この1歩分の計算結果を用いて歩容の周期性及び対称性を利用して，これを反復させることで種々の歩容を作成している。

4.3 シミュレーションとその動画化

4.3.1 歩容の映像化

63年度にCAP（Cellular Array Processor）を用いて作成した平地歩行の動画[6]は一つの歩容を繰り返し用いたもので全て同じ歩行パターンであった。しかし，この歩容以外にこれとは異なった歩行パターンを持つ歩容（平地歩行及び階段歩行）のシミュレーションも行ってきた。これらの結果はSOLVER[7]（A Solid Modeler for Vision Engineering Research）と呼ばれる画像生成ソフトウェアを用いてFig. 4.2に示すように駒撮り風に図形化しているが，この画像だけでは実際の動きがよくわからないという欠点がある。そのため，63年度の動画作成用プログラムに修正を加えCAPを用いてこれらの動画の作成を行った。これはFig. 4.3に示すようにロボットを2体使って正面からと側面からの両方の動作チェックができるようにしている。ただし，階段等のシミュレーションも行うためロボット初期状態は，足が地面に接地しないように設定している。この図は平地歩行のシミュレーション結果をCAPで動画化した一画面のハードコピーである。このロボット軀体モデリングソフトウェアFUSIONについては文献6）を参照されたい。なお，シミュレーションプログラムにミスがあり，ロボットの骨盤のY軸方向の長さが誤って入力されていたため，これを修正した。

このシミュレーションの入力データは全てヴコブラトビッチのデータに基づいたものであるが，動画化すると昨年の動画と同様に上体が非常にギクシャクとした動きをしているうえ歩容自体にも不自然なものが見られた。そこで，この動画を基に歩容の修正を試みた。つまり，入力データである基本歩行パターンを人工的に作成しロボットの動作を修正していくのである。歩容の修正には下肢の関節角の変更とZMPの軌道の変更が考えられるが，ZMPの軌道の変更に関しては昨年までのシミュレーション結果よりZMPを前にシフトすればそれに伴って上体は前に傾く事が分かっている。また，下肢の関節角の変更では冗長自由度を利用して腰の位置を制御する[8]ことにより上体の動きを修正することができる。足の軌道は骨盤が回転しないためX－Z平面の二次元平面内での動きとして考えられる。一般に二次元平面内で任意の位置に移動させるためには2自由度必要であるが，下肢はX－Z平面で5自由度を持っている。そのため，この冗長自由度

を利用して足の位置を変えずに腰の位置だけ変更することができるのである。この場合，腰を前に出せば上体は後ろに傾く。これは倒立振子の制御と同じ方法である。これらを考慮して歩容の修正を行い比較的滑らかな平地歩行と階段歩行の歩容を作成した。Fig.4.4 に歩容の修正前と修正後のシミュレーション結果を示す。図からも明らかに上体の動きが滑らかになっている様子が分かる。なお，現段階では蹴りを考慮していないため足は常に地面に水平な歩容とした。

また，ＺＭＰの軌跡については支持脚の足底面内のＸ軸上のみを動くように仮定しているが，これまでは重心からのＸ軸方向の距離ΔＸ（Fig. 4.1参照）を半周期（１歩）を３つの部分に分けた階段関数で与えていた。しかし，実際はもっと複雑な動きをしており入力データとしては望ましいものではなかった。そのため，ＺＭＰの任意な軌道を入力できるようにプログラムの修正を行った。これと同時に入力したＺＭＰと補償動作が決定されてから逆算したＺＭＰとの比較を行い，一部で数cmの誤差が見られたが他はほぼ一致していることが分かった。しかし，この誤差が出来るだけ小さくなるように反復計算をしている例[9]もあり，今後この方式を本研究に取り入れていくことを検討している。

この歩行動作の動画は次節のものも含めて，１パターン４秒程度のものを約８分作成した。

4.3.2　腕可動型モデルの開発

4.2節でも述べたように６３年度までは腕を固定したモデルを用いてきたが，今年度腕を自由にしたモデルの開発を行った。当初は腕の動きも計算で求めるタイプの開発を行ってきた。これは，腕を振子のように動くと考え両肩において腕のＸ軸とＹ軸まわりの回転を計算するもので，左右対称に動くように境界条件を設定し腕固定の場合と同様に，肩のジョイントにおいてＸ軸とＹ軸まわりのモーメントの和が零になるとして式（4.1）を解くものである。しかし，パラメータが６個となるため4.2節でも述べたように，各々の初期位置と初期速度の合計１２個の入力初期値について，境界条件を満たすものをサーベイしなければならず大変な計算量となるうえに，運動方程式のオイラー角表現モデルへの変更によるプログラムのバグ等により思うように開発が進んでいない。プログラムのバグを発見するために，腕の動きも計算で求めるタイプの腕の計算部分を削除し，腕固定型のオイラー角表現モデルのシミュレーションプログラムを作成した。これにより若干のバグを発見し，その修正をすることでこの腕固定型のオイラー角表現モデルのシミュレーション結果と昨年度までの結果はほぼ一致した。そのため，この方法でもモデル化が可能であることが分かるが，計算機のＣＰＵ時間がこれまでの10～30倍程度かかり，シミュレーションプログラムの改善が必要である。これは，汎用性を考慮して全てのボディの動きをオイラー角表現としたために，どの関節にも３自由度存在し回転していない角度についても０度回転として計算を行っているためである。オイラー角表現モデルはヴコブラトビッチの文献１)に数式化されているために取り上げたが，この様にこのモデルに変更するメリットは少ない。このため，オイラー角表現の腕の動きも計算で求めるタイプの開発を一時中断し，これまでのプログラムを基に腕の動きをデータとして入力するタイプの開発を行った。このモデルは，腕の振り角$\beta_{4L,R}$，$A_{11L,R}$は一定とした（Fig. 4.1参照）。これに伴い動画作成用プログラムも腕が動く様に修正を行った（4.3.3項参照）。

腕の振りのデータは左右対称になるように作成しているが歩行動作とのタイミングについては

人間の振りを模倣している。当初支持脚が切り替わる瞬間に腕の振りが最大になると考えていたが，作成した動画を見るとわずかながらタイミングがずれている様子が分かる。ここで，もう一度人間の歩行動作の観察を行った。通常の歩行では腕の振りは支持脚が切り替わる瞬間に最大になるのではなく，支持脚でない方（遊脚）の大腿の角度（β_{1R}；Fig. 4.1参照）が最大となるとき支持脚側の腕の振り角（β_{4L}）が最大となっている様である。このシミュレーション結果をFig. 4.5に示す。本来これらのシミュレーション結果は動画として表示しているが，本原稿のため動画のシーンの一部をハードコピーして作成したものである。

腕振り動作の追加による補償動作への影響は少なく，例えばTable 4.1は平地歩行のシミュレーション結果の一例であるが，両脚の左右の振れθに対する影響はほとんどなく，上体の傾きψにおいても僅か0.03 rad. 程度の変化であった。これは，θに関して言えば腕を矢状面（X－Z平面）内でしか動かしていないため，前頭面（Y－Z平面）内の動きにはあまり影響してこないものと考えられる（一般に矢状面と前頭面の両面間の運動の干渉は無視できる[10]ものとする場合が多い）。また，ψに関しては左右対称に動かしているため互いのモーメントが打ち消し合っている様に思われるが，腕の振りを前にシフトした場合や左右同じ（同時に前後に振る）とした場合のシミュレーション結果からも始めの場合と同様に上体の補償動作への影響は少なかった。つまり，両腕の慣性モーメントが胴体に比較して小さいため影響が少なかったものと思われる。両腕の慣性モーメントを変化させた場合のシミュレーションは，本研究のモデルがハードウェアを意識したものではないため構造部材やアクチュエータに関するデータ等がなく，行っていない。

なお，骨盤のY軸方向の長さの調整で両脚の左右の振れθを，骨盤のZ方向の長さの調整で腰の曲がる角度ψを変化させることができることもシミュレーションより分かった。この場合も慣性モーメントの変更が必要になってくるがこのモデルは骨盤が回転しないため慣性モーメントの値は無関係である。このように，モデルの各部の寸法等の変更で補償動作を変えることができるが，前述のように本研究のモデルがハードウェアを意識したものではなく，現段階ではハードウェアに依存するデータがないため正確なシミュレーションは行えない。

また，同時に脚の補償動作を支持脚のみとし遊脚のθを常に零とした場合のシミュレーションも行ったが，支持脚のθの値に大きな変化はなく，逆に両脚の干渉の問題も生じてくるため，この場合のシミュレーションは数例について行っただけに留め，動画化は行わなかった。

4.3.3　動画作成用プログラム作成

歩行動作のシミュレーションはM－780上で1歩分の計算を行い計算結果をA－50を介してSUN 3/260上にファイル転送する。SUN上ではFUSIONと呼ばれる3次元ソリッドモデラを用いて映像を作成するが，このモデリングソフトとのインターフェース及び歩行を反復させるためのプログラムを作成した。

また，現在のシミュレーションは停止や発進を除いた定常歩行しか行っていないが，元年度版HASP動画作成のためには，停止や発進といった動作は必要である。そこで，今年度は取り敢えず停止や発進の1歩の動作をアニメーションで作成することにし，このため　アニメーション作成用プログラムを作成した。このプログラムは，入力データをそのまま映像化するものでデータさえあればどのような動きでも作成することができる。ただし，データ作成の際，床や階段等

との干渉のチェックは必要である。

4.3.4　元年度版HASP動画作成

63年度の動画化にひき続き元年度版の動画を作成した。動画のシナリオをTable 4.2に示す。各シーンの間には3～4秒程度の静止画を挿入している。また，動画作成方法等については文献6)を参照されたい。歩行動作の動画化を行う環境は現在モデリング中の施設形状（5章参照）のJRR－3である。今回はJRR－3の地下1階を歩行する様子を階段歩行も含めて動画化した（Fig. 4.6，Fig. 4.7）。今回動画化した歩容は，平地歩行が歩幅362.8 mm，階段歩行が歩幅197.5 mm，高さ190.0 mmの場合で，これらのデータは施設形状に合わせて作成した。また，施設内を歩行するロボットのモデルの改良も行った。しかし，この施設形状のプリミティブの増加に伴い画像生成時間が大幅に伸び，わずか2分数秒の動画作成だけで1カ月強を費やした。このように画像生成に多大な計算時間がかかるうえ，モデリングソフトウェアにバグがあり，まだ使いやすいソフトウェアとは言えない。

4.4　おわりに

二足歩行のロボット運動学的シミュレーションは，これまでヴコブラトビッチの人間型モデルを基に研究を進めて来た。しかし，現在のモデルは以下のような問題点がある。

(1)　常に単脚支持で左右対称な歩容しかシミュレーションできない。そのため両脚支持状態である停止・発進動作及び方向転換等についてはまだシミュレーションを行っていない。

(2)　現在のロボットモデルは人間の大きさや質量を基にしたもので，これらの基本データはヴコラトビッチのモデルのものを引用している。そのため，このロボットモデルに対するシミュレーションしか行えず汎用性に欠ける。

(3)　現在のロボットモデルは実際のハードウェアを意識したものではなく，運動学方程式を解くために必要なハードウェアに依存する機械的データや実験データ等が無い。

これらの問題点解決のため，実際のハードウェアを考慮したモデルへの変更を行う必要がある。そこで実際のハードウェア作成を前提とし，今後人間型ロボットのハードウェア化を目的としたモデル（ソフトウェア及びハードウェア）を開発していく計画である。これらの実現のため本研究では，マイクロロボットの作成を検討している。

二足歩行ロボットの基本設計としてはFig. 4.8に示すように，動作として平地や階段等の定常歩行及び停止・発進，方向転換などを対象とし，その動作に対する判断や知識は外部のコンピュータに依存する。但し，関節駆動及び制御のためのマイクロコンピュータ（マイクロプロセッサ）等の搭載は考える。また，駆動源についてはマイクロロボットのためロボットに搭載する場合と外部設置の場合の両面を考慮し，エネルギー源や動作命令及び情報は光ファイバーケーブル等で外部から伝達する。

2年度の計画としては，歩合動作については元年度の調査結果をもとに停止・発進動作のシミュレーション及び脚の逆運動学計算[11]を含め，冗長自由度を利用して脚の軌道－ZMPの軌道－

腰の位置－上体の動き，これら全てを考慮したトータルなシステムとしての歩行パターン作成支援システム[12]を開発する。また，両腕作業については，これの運動学・逆運動学計算のプログラム化を行う。ただし，両腕作業に対する補償動作や，そのときの脚の動きについては別に検討する必要がある。さらに，前述のようなマイクロロボットを作成するため二足歩行ロボットの機械モデル（構造部材，アクチュエータ，ジョイント，駆動源等）の設計に関する調査を行う。

参考文献

1）M. Vukobratović （加藤一郎・山下忠 訳）：歩行ロボットと人工の足，日刊工業新聞社，1975.

2）石黒美佐子・藤崎正英：二足歩行ロボット運動学的シミュレーション，JAERI－M 88－080，1988.

3）浅井清・久米悦雄 他：原子力知能化システム技術の研究 －昭和63年度作業報告－，JAERI－M 89－023，PP. 70－88，1989.

4）高西淳夫，加藤一郎 他：2足歩行WL－10RDによる動歩行の実現，日本ロボット学会誌3巻4号，PP. 67－78，1985.

5）伊理正夫・藤野和建：数値計算の常識，共立出版株式会社，1985.

6）藤崎正英・久米悦雄：セルラアレイプロセッサCAPによる動画作成，私信，1989.

7）越川和忠：SOLVER，私信，1985。

8）高西淳夫・加藤一郎 他：上体と腰の協調動作による2足歩行制御，第27回計測自動制御学会学術講演会，PP. 643－644，1988.

9）高西淳夫・加藤一郎 他：上体補償機構を有する2足歩行ロボットWL－12の開発，第5回日本ロボット学会学術講演会，PP. 579－582，1987.

10）高西淳夫：2足歩行ロボットによる準動歩行，日本ロボット学会誌1巻3号，PP. 36－43，1983.

11）川路茂保 他：逆運動学を用いた二足歩行ロボットの目標軌道生成アルゴリズム，第6回日本ロボット学会学術講演会，PP. 123－124，1988.

12）例えば，若原龍雄・高西淳夫・加藤一郎 他：2足歩行パターン作成支援システム「WALK MASTER－4」の開発，第5回日本ロボット学会学術講演会，PP. 575－578，1978.

Table 4.1 Comparison of simulation results

時間	θ [rad.]		ψ [rad.]	
	腕固定	腕振り	腕固定	腕振り
0.0	0.0022	0.0021	0.1783	0.1721
0.0333	0.0244	0.0236	0.2138	0.2018
0.0667	0.0441	0.0427	0.2321	0.2181
0.1000	0.0615	0.0596	0.2440	0.2312
0.1333	0.0770	0.0746	0.2550	0.2414
0.1667	0.0905	0.0878	0.2631	0.2477
0.2000	0.1023	0.0993	0.2630	0.2486
0.2333	0.1125	0.1092	0.2562	0.2426
0.2667	0.1212	0.1177	0.2445	0.2316
0.3000	0.1285	0.1248	0.2306	0.2203
0.3333	0.1345	0.1307	0.2170	0.2090
0.3667	0.1393	0.1355	0.2040	0.1977
0.4000	0.1430	0.1391	0.1918	0.1874
0.4333	0.1455	0.1417	0.1816	0.1789
0.4667	0.1470	0.1432	0.1730	0.1716
0.5000	0.1473	0.1436	0.1605	0.1611
0.5333	0.1466	0.1430	0.1394	0.1435
0.5667	0.1449	0.1413	0.1156	0.1238
0.6000	0.1420	0.1385	0.0871	0.0996
0.6333	0.1381	0.1347	0.0576	0.0737
0.6667	0.1333	0.1299	0.0282	0.0471
0.7000	0.1273	0.1240	-0.0025	0.0195
0.7333	0.1200	0.1169	-0.0371	-0.0103
0.7667	0.1113	0.1083	-0.0675	-0.0400
0.8000	0.1009	0.0982	-0.0759	-0.0525
0.8333	0.0885	0.0861	-0.0548	-0.0316
0.8667	0.0742	0.0721	-0.0117	0.0099
0.9000	0.0582	0.0565	0.0385	0.0507
0.9333	0.0402	0.0391	0.0852	0.0897
0.9667	0.0202	0.0196	0.1308	0.1311
1.0000	-0.0022	-0.0021	0.1783	0.1721

Table 4.2 Scenario

SCENE No.	シナリオ	時間（秒）	カメラ視点	ターゲット
0	原子炉全体を鳥瞰しながらカメラが地下１階のエレベータの前まで降りてくる	１１	移動	移動
1	エレベータの扉が開く	2	固定	固定
2	ロボットがエレベータから出てくる	7	〃	〃
3	ロボットがエレベータ前の階段を降りる	6	〃	〃
4	ロボットが一次区画の入口に向かって歩く	１１	〃	〃
5	遮蔽扉中央を見てから扉左側の位置認識マークを見る（ロボット停止）	3	ロボットの目	移動
6	遮蔽扉が開く（ロボット停止）	5	固定	固定
7	ロボットがＳＦＳ熱交換機に向かって歩く	8	ロボットの目	固定
8	〃	１０	固定	固定
9	〃	4	ロボットの目	固定
１０	ＳＦＳ熱交換機の前で左右を見回す（ロボット停止）	１０	ロボットの目	移動
１１	ＰＣＳ熱交換機に向かって歩く	7	固定	固定
１２	〃	１０	〃	〃
１３	歩きながらＰＣＳ熱交換機を見る	8	ロボットの目	移動

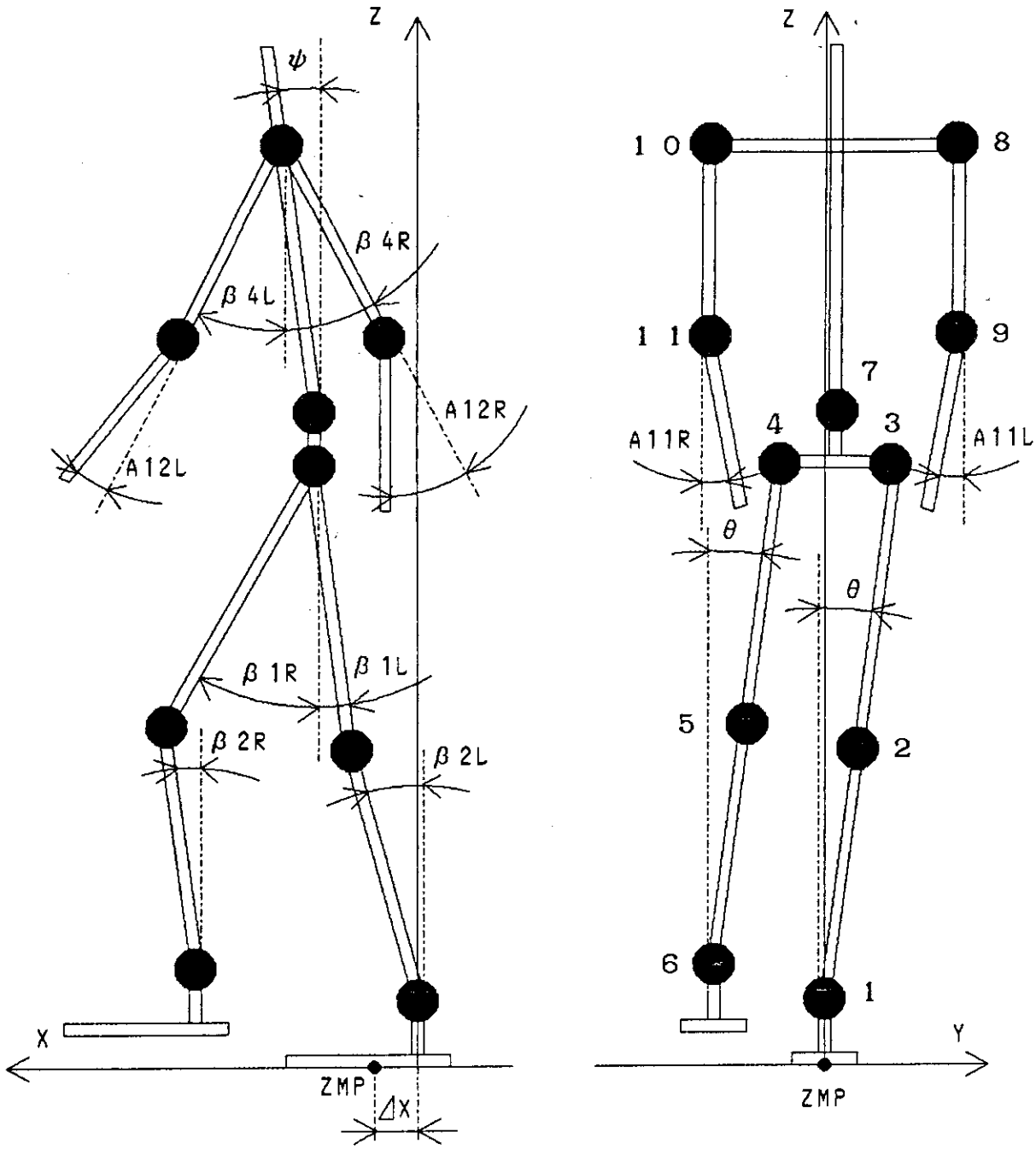


Fig. 4.1 Configuration of human biped locomotion model

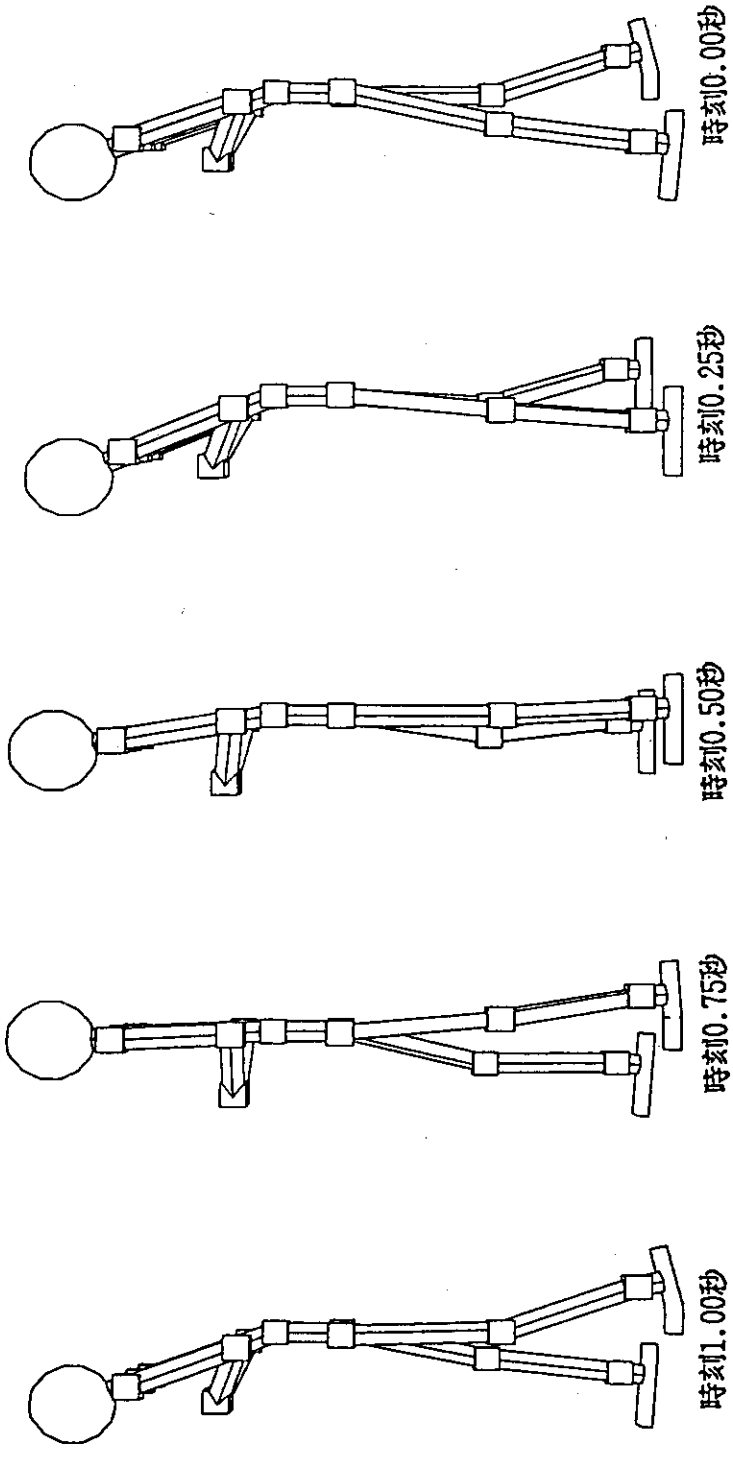


Fig. 4.2 Walking shot sequence during one step. (Using "SOLVER")

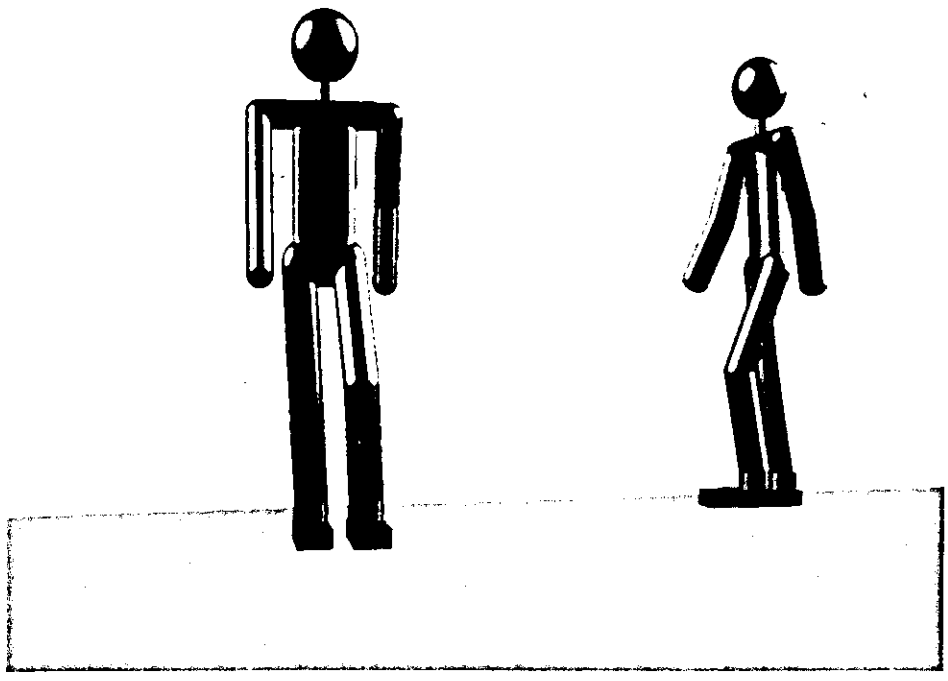


Fig. 4.3 Test scene of flat floor walking

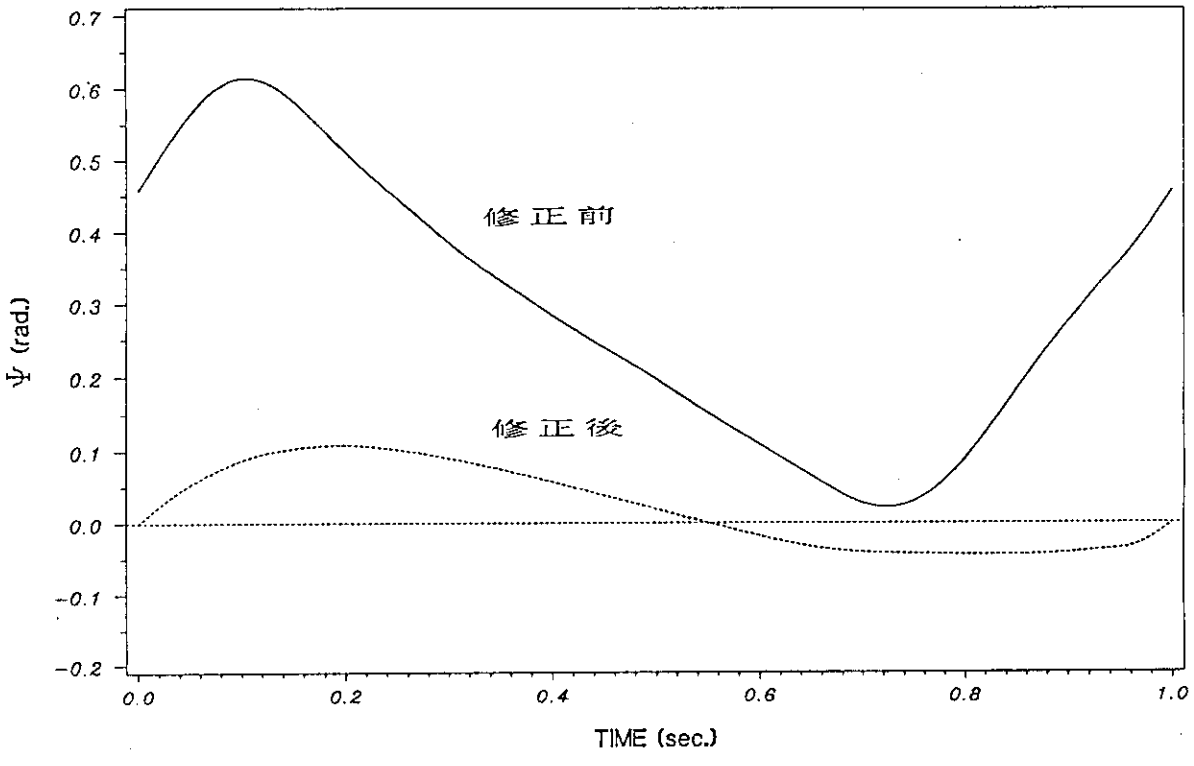


Fig. 4.4 Comparison of simulation results

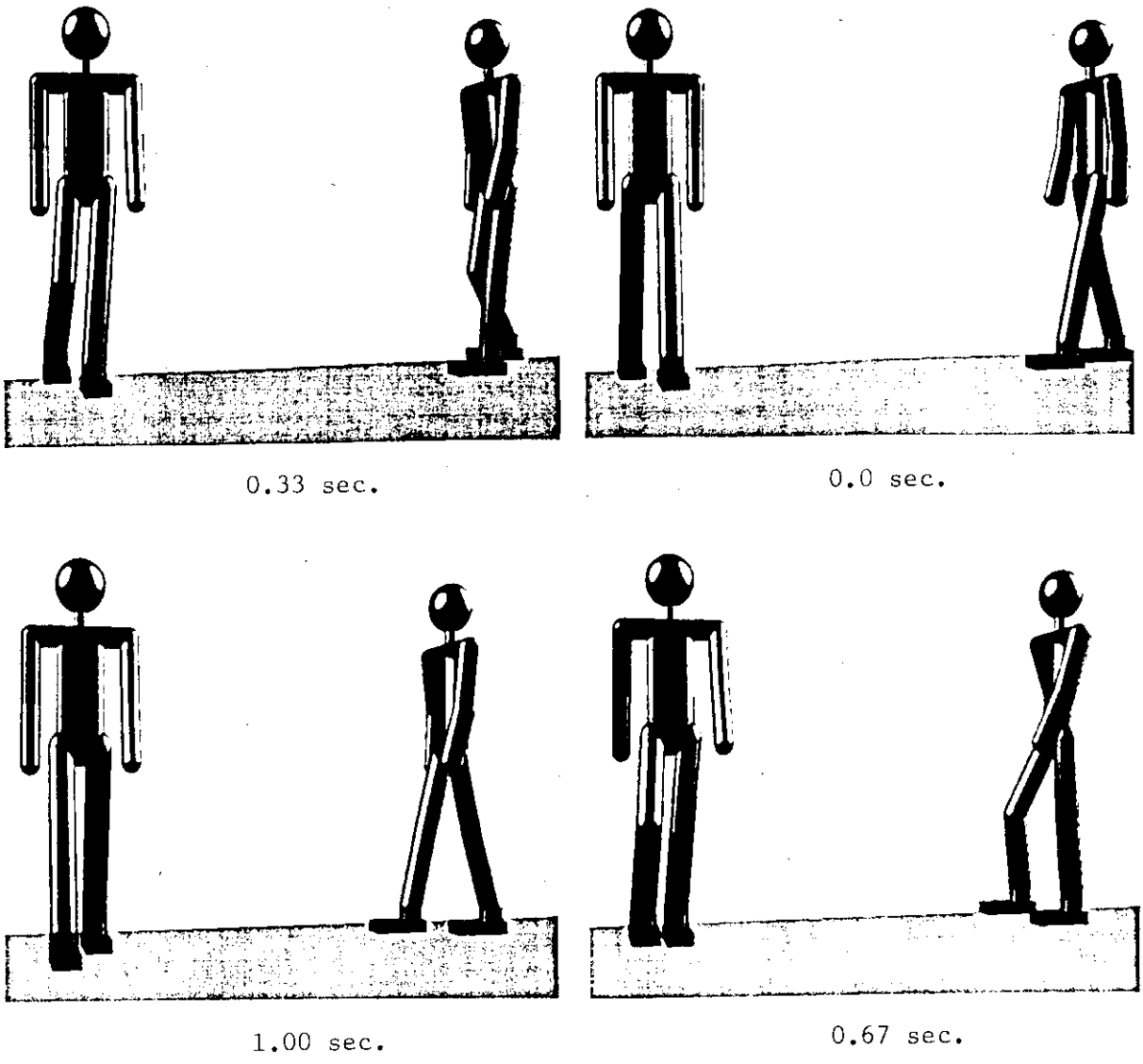


Fig. 4.5 Flat floor walking

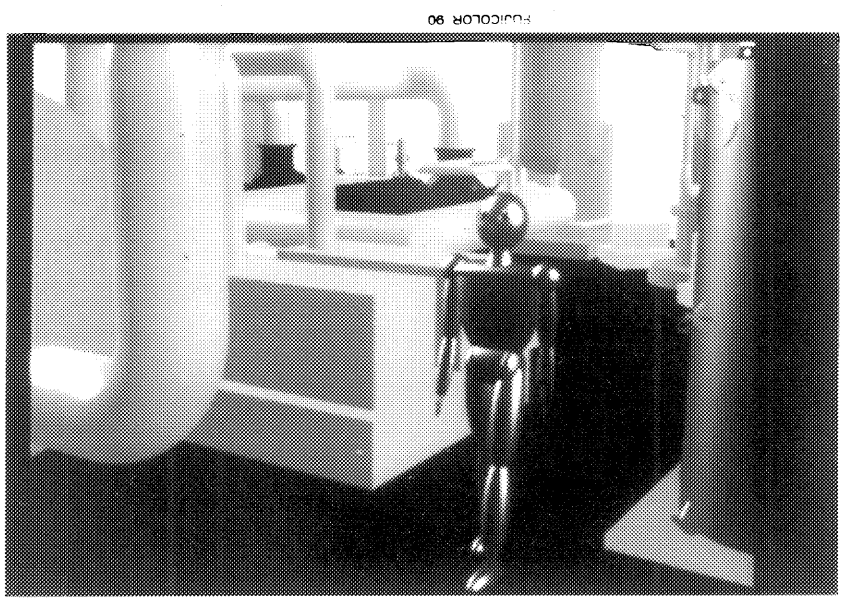


Fig. 4.6 A simulation of a biped locomotion robot in a nuclear plant JRR-3. (Flat floor walking)

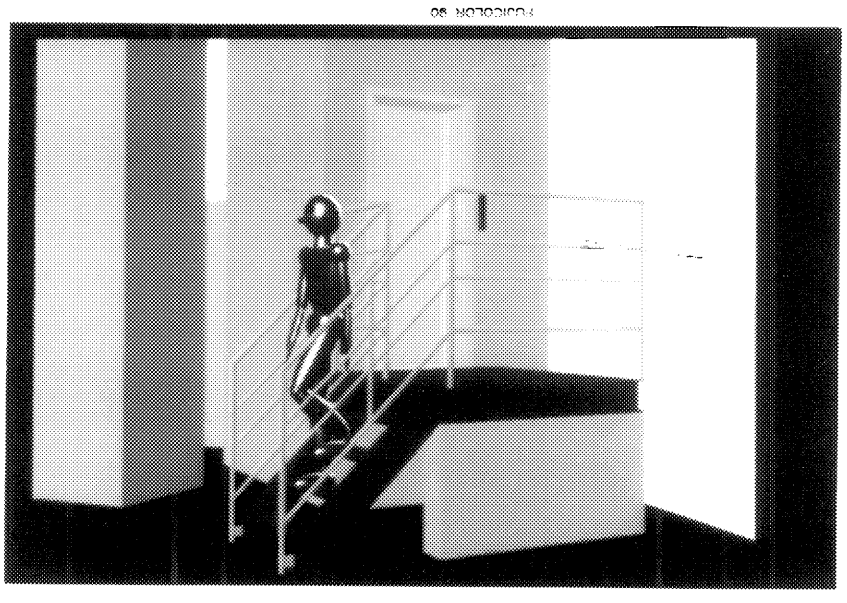


Fig. 4.7 A simulation of a biped locomotion robot in a nuclear plant JRR-3. (Go down the stairs)

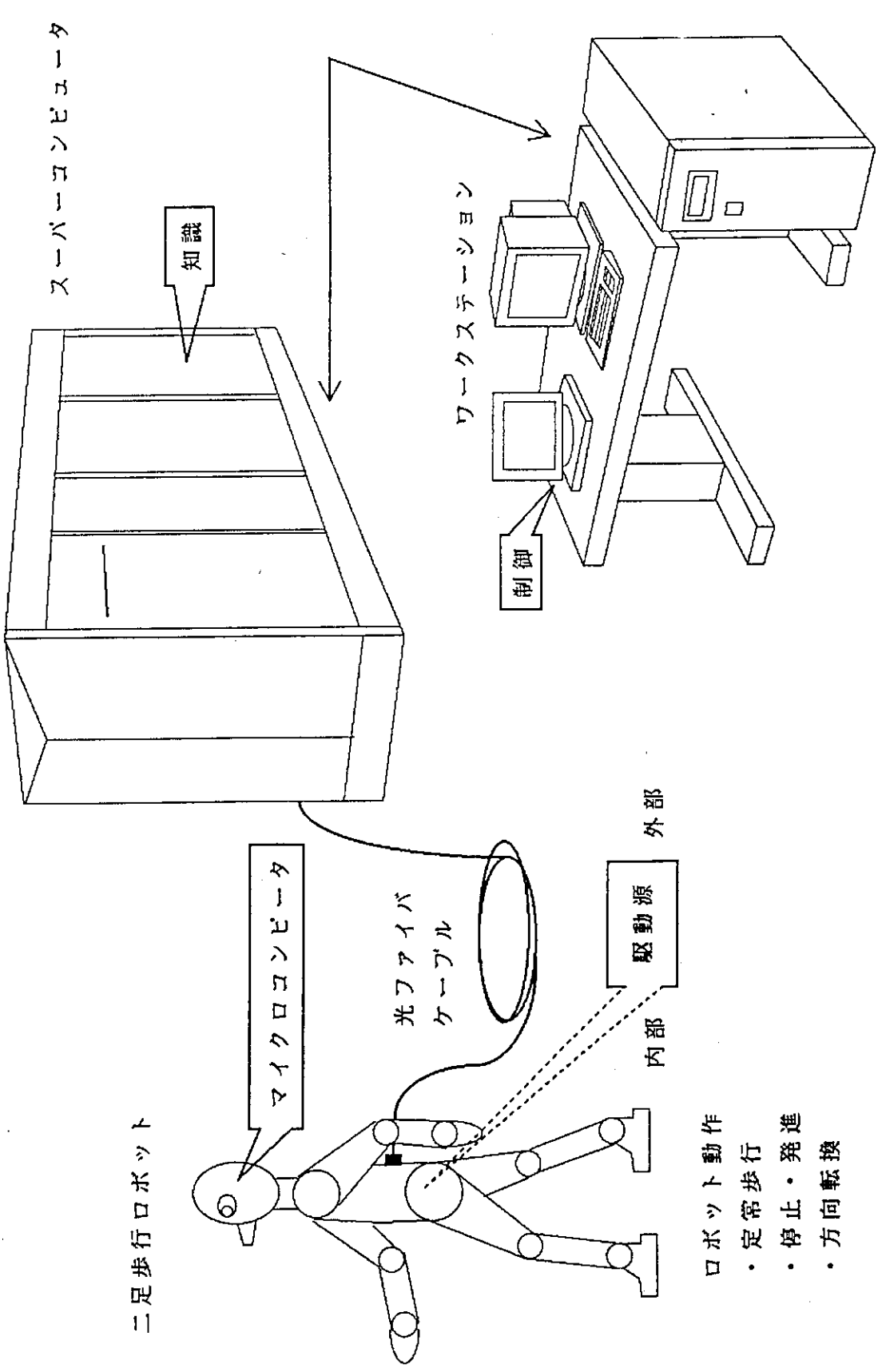


Fig. 4.8 Conception of a micro-robot

5. 施設形状データベース

5.1 はじめに

本研究では，原子力施設における知能ロボットの作業を対象に，さまざまなシミュレーションが行われる。例えば，知能ロボットが与えられた命令を理解し，それを実行するための行動シミュレーション，知能ロボットが詳細な動作を行うために環境を理解する視覚シミュレーション，作業中の知能ロボットに対する被曝線量計算等である。これらのシミュレーションでは

① 施設に対する形状データの管理

② シミュレーション結果の映像表示

を施設形状データベースを使用して行う。

本研究においては，昭和62年度より施設形状データベースの試作を開始した。この試作においては，知能ロボットの動作空間としてJRR－3施設を想定し，建屋及び機器類の入力を行ってきた。この際，3次元物体の入力はソリッド・モデルの集合演算によって物体を定義するCSG (Constructive Solid Geometry)手法により行った。シミュレーション結果の映像化については，生成された画像をVTR装置によってコマどりすることでこれを行う。この時，画像生成については，アルゴリズムとしてレイ・トレーシング法を採用し，並列計算機によって高速処理している。昭和63年度に映像生成用並列計算機，形状データ定義用ワークステーション及びVTR装置等の機器の導入整備を行い，施設形状データベースを使用する作業が実行可能となった。以下の(1)，(2)にソフトウェア及びハードウェアの説明を行う。各手法については，本研究に関するいくつかの報告書，例えば文献1)等に詳しいので，本章では簡単に述べる。

(1) 映像生成用ソフトウェア

施設形状データベースにおいて使用されているソフトウェアは，3次元物体モデリング用ソフトウェアFUSION [2] と画像生成用プログラムである。

FUSIONでは，プリミティブと呼ばれる素立体の集合演算により3次元物体を定義する。言語型モデラと異なり，会話形式で入力値を確認しながら物体定義を行うことができる。また，UNIX系計算機の特長であるマルチウインドウ・システムを用いて，その操作を行うことができる。

画像生成用プログラムのアルゴリズムは，レイ・トレーシング法である。レイ・トレーシング・アルゴリズムは，鮮明な画像を得られる反面，画像を生成する際に多大の計算時間を要する，という欠点がある。この欠点を克服するために，本研究で使用されているプログラムでは，3次元DDA (Digital Differential Analyzer) アルゴリズム [3] により計算時間の短縮を図っている。このアルゴリズムでは，2のべき乗個に分割されたサイコロ状の空間を設定し，各空間内に存在する物体とのみ交差テストを行う工夫がなされている。また，後述する映像生成用並列計算機CAPを使用してより有効な並列計算を行うために，各プロセッサに割り当てられた計算量が均等化する工夫がなされている。

(2) 映像生成用ハードウェア

昭和63年度に，JRR-3施設地下1階におけるロボットの二足歩行シミュレーション結果を動画化した。この際使用したハードウェアをFig.5.1に示す。まず，計算センターに設置された大型計算機M780上で二足歩行シミュレーションのための計算を行う。M780はA-50とDSLINKにより結合されており，高速ファイル転送が可能である。さらに，A-50とSUN3/260CはETHER NETで結合されており，やはり高速ファイル転送が可能である。SUN3/260Cは，3次元物体の形状データ入力に使用される。前述のソフトウェアFUSIONを用いて入力作業を行う。各時刻における二足歩行シミュレーション結果は大型計算機よりSUN3/260Cへ転送され，ロボットの動きに応じたロボット躯体の形状データがここで自動的に生成される。この形状データを使用して動画を作成するが，画像生成には，映像生成用並列計算機CAP(Cellular Array Processor)[4]を使用する。CAPは128台のプロセッサを持った16 bitの並列計算機であり，各プロセッサは，2MBのローカル・メモリを持ち，マイクロプロセッサにはi80186を，浮動小数点演算にi8087を使用している。この並列計算機CAPで生成された画像を，VTR装置（1/2 β-CAM）において1枚ずつコマどりすることで動画を構成している。

5.2 動画作成作業

平成元年度に，施設形状データベースを用いてJRR-3施設における知能ロボットの二足歩行シミュレーション結果の映像化を行った。この動画は約1,300枚程の静止画像で構成される。使用された1枚の画像をPhoto.5.1に示す。また，使用された形状データ数，映像生成時間等をTable5.1に示す。各項目の意味については以下の通りである。

(1) プリミティブ数

映像生成を行う際，ロボットの位置，視点及び視方向によって画像生成時の計算対象となる物体を特定することができる。そこで，画像生成のための計算時間短縮のために，動画中の情景ごとに表示対象物体を限定したクラス（後述）を設定し計算を行っている。クラスを構成する物体の集合演算に使用された素立体の数がプリティブ数である。

(2) 形状データ展開時間

本研究において使用されたモデリング用ソフトウェアFUSIONでは，会話形式で行われる入力作業をできる限り簡略化するために，物体の大きさや詳細な形状に依存しない抽象化された概念として「クラス」と呼ばれるものを定義することができる。入力作業者は，この「クラス」の各パラメータに具体的な数値を与えることにより具象物を定義することができる。また，入力作業者が任意の値を「クラス」の各パラメータに対して与えることにより，同じ構造を持ったさまざまな3次元物体を定義することができる。画像を生成する際，FUSIONは各パラメータの値を参照し，画像生成用プログラムに形状データを渡す。形状データ展開時間とは，このためのLisp言語による処理時間である。

(3) 映像生成時間

通常，動画を作成する際，1秒間に15枚または30枚の画像を連続表示することでこれを実

現するが，映像生成時間とは，１枚の画像生成用並列計算機で生成した際の経過時間である。この映像生成時間は，形状データ展開時間とは異なり，プリミティブ数に比例していない。この理由は，映像生成用プログラムで採用されているDDAアルゴリズムの処理内容に帰因する。レイ・トレーシング法により３次元画像を生成する場合，視点を始点としスクリーン中のある画像を通る直線（視線）が体系中のどの物体に当たるか，ということを先ず調べる。この時，「ある視線が計算対象となっている体系内のどの物体と最初に衝突するか」という計算は，通常，体系内のすべての物体とその視線との交差テストを行い，その距離を比較することによりなされる。本研究において使用されている画像生成プログラムでは，多大の計算時間を要するこの交差テストの回数を減らすために次のような計算を行う。

ⅰ）予め体系内の空間をサイコロ状に分割し，各サイコロ（VOXELと呼ばれる）内に含まれる物体（プリミティブ）を登録しておく。VOXELの数は通常16×16×16である。

ⅱ）視線と物体（プリミティブ）との交差テストは直線が通過するVOXELに含まれる物体（プリミティブ）とのみ行う。

ⅲ）視線が進入するVOXELの番号は直線の傾きから逐次的に求める。

形状データに対して以上の前処理及び交差テストを行うことにより，視線と体系中のすべての物体との交差テストは不必要となり，計算時間は大幅に削減される。特に，各VOXELに平均して物体が存在する体系において，このアルゴリズムは良い結果を示す。しかし，いくつかの特定のVOXELに多くの物体が偏って存在する体系においては良い結果が望めない。通常のレイ・トレーシング・アルゴリズムによる画像生成のための計算時間は，体系中に存在するプリティブの数に比例するが，本研究における画像生成時間は，以上の理由により，体系に存在する物体の分布状態に依存する。

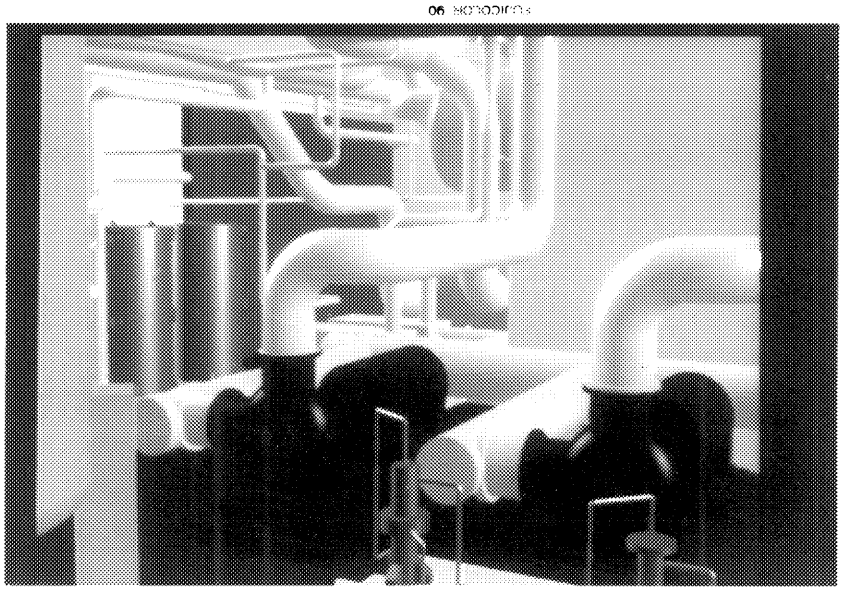


Photo. 5.1 The visualization of JRR-3 by using Plant Geometry Data Base

5.3 動画作成上の問題点

前節で述べたように，平成二年度にIRR-3施設地下1階を動作空間として二足歩行を行うロボット動作の動画化を行った。この作業を通じて，今後の施設形状データベースの使用に関する問題点が明らかになってきた。これらは，昭和62年度より開始した入力作業が進み，物体を表現するプリミティブの数が増加するに伴い生じてきた問題点である。

この問題点とは，現在施設形状データベースに登録された物体表現のためのプリミティブ数が増大し，画像生成時間が大幅に増加したことである。現在登録済の形状データを使用して画像を生成した時，前節で述べたように一枚数十分の時間を要する。動画を作成する際，これは非常に不便であり，また，将来シミュレーション結果の実時間表示を行うためにも，画像生成時間の短縮が必要である。このための問題解決を行うために検討すべき手法は以下の3点である。

① 物体記述モデル及び画像生成アルゴリズムの変更

② 高性能な映像生成用計算機の使用

③ 画像生成処理に適した言語による物体記述

各項目に対する考案結果を以下に示す。

(1) 物体記述モデル及び画像生成アルゴリズムの変更

1980年代後半に，より高性能な3次元画像表示専用のワークステーション（Graphic Workstation，以降GWSと呼ぶ）が商品化され，シミュレーション結果の映像化に利用されるようになってきた。これらのGWSでは，複数のポリゴンによって表現された3次元物体をスキャンライン・アルゴリズムやZバッファ・アルゴリズム等の従来の3次元表示アルゴリズムによって画像生成を行う。さらに，これらのGWSでは，アフィン変換（2次元投影）のためのマトリクス計算を行うハードウェア（グラフィック・エンジン，レンダリング・プロセッサ等と呼ばれる）を使用して高速に画像生成を行う。3次元表示のためのシェーディング処理を行った際，その表示能力は，6万～20万ポリゴン／秒である。

一方，本研究では，球や直方体等の素立体の組合せによって表現された3次元物体をレイ・トレーシング・アルゴリズムにより画像生成している。したがって，これらのGWSを用いて，3次元画像生成を行うためには，素立体の組合せによって記述された形状データを複数のポリゴンを使って3次元物体を表現する形状データへ変換しなければならない。ところが，現在施設形状データベースに登録されている物体を表現するために3,400個程の素立体が使用されており，これをポリゴン・データに展開した場合，その10倍～100倍のポリゴンつまり34,000～340,000ポリゴンのデータ量となる。GWSの表示能力を考慮すると，これほど大量の形状データに対して高速に処理を行う実時間表示は難しい。また，複数のポリゴンを使って3次元物体を表現する形状データは，行動計画・被曝線量評価に必要なある直線と物体との交差テストを行うには不向きである。したがって，素立体の組合せによって表現された形状データがやはり必要となる。さらに，複数のポリゴンによって表現された形状データを素立体の組合せによる形状データへ変換することは不可能であるが，逆のデータ変換は随時可能であり，必要に応じてこれを行う。

(2) 高性能な映像生成用計算機の使用

現在，本研究で映像生成用に使用している計算機は，128台のプロセッサを持つ並列計算機である。この並列計算機において使用されている個々のプロセッサについては，その20倍～50倍の処理能力を持つプロセッサが現在の技術水準で開発可能である。さらに，現在128であるプロセッサの台数を8倍に増やすことにより，160倍～400倍の処理能力を持った映像生成用計算機が実現できる。

(3) 画像生成処理に適した言語による物体記述

現在，物体に関する形状データの定義用プログラム（モデラ）は，Lisp言語で書かれている。これは，

① 入力作業者とワークステーション間のマンマシンインタフェースを考慮した機能を実現するために望ましい，

② 施設内の機器に関して，幾何学的情報のみでなく，装置名，物質名，色等の非幾何学的情報を追加する際，各データの追加が簡単に行える柔軟なデータ構造が望ましい。

との理由からである。しかし，映像表示のみに目的を限定すれば，C言語のような画像処理に適した言語によるモデラ及び画像生成プログラムが望ましい。

5.4 おわりに

施設形状データベースの試作においては，知能ロボットの動作空間をJRR－3と想定し，昭和62年度より入力作業を行ってきた。現在入力済みの部分は，JRR－3建屋，階段，エレベータ，一次冷却系配管，地下1階1次区画内の機器，タンク類等である。平成二年度は，地下1階の計器類，1階の各装置類，バルブ類を入力する予定である。

また，今後知識ベースを使った行動計画，二足歩行シミュレーション，被曝線量評価プログラム等とのインターフェースを考慮し，現在SUN3／260ワークステーション上にある形状データを大型計算機上で取り扱うことを検討する。現在登録済みの形状データは，Lisp言語によるリスト構造を取っているが，大型計算機上のデータは，

① FORTRAN言語で読み込み可能なデータ形式

② モンテカルロ・コード等にみられるような，簡潔に領域表現を行っているデータ形式

が望ましい。現在，FUSIONによって入力済みの形状データとモンテカルロ・コードにおけるデータのそれぞれの内部表現をFig. 5.2に示す。両者のデータは，配管の曲部という簡単な形状に対するデータである。FUSIONは，マンマシンインタフェースを考慮した会話形式による入力作業を重点に設計されているために，その内部表現は冗長であり複雑である。FUSIONによって表現されたデータの大型計算機上への移行プログラムについては，昭和63年度に作成済みである。今後は，この形状データをFig. 5.2(b)に示される形式のモンテカルロ・コードにおいて使用されている簡潔な表現に書き換えるプログラムの開発が必要となる。

参考文献

1） 浅井，他：原子力知能化システム技術の研究（人間動作シミュレーション・プログラム：HASP）昭和63年度作業報告，JAERI－M 89－023，日本原子力研究所，1989年3月

2） 村上：“（言語＋対話）型ＣＧモデラ：FUSION／MODELER”，情報処理学会グラフィクスとCAD研究会　第1回集中研究会，Aug.，1987

3） 村上：“セルラアレイプロセッサCAPによるレイ・トレーシング”，情報処理学会グラフィクスとCAD研究会，Jul.，1986

4） 佐藤：“高並列計算機CAPとその応用”，FUJITSU，Vol．38，No．3，1987

Table 5.1 The Computing costs in making the animation

クラス	プリミティブ数	形状データ展開時間	1フレームの映像生成時間（ＡＮＴＩ　ＬＥＶＥＬ：１）
0	１８０５	１時間１５分	１６分
1	２５３	２０分	９分
3	８６０	３０分	６分
4	１４００	５０分	４８分
5	１７４６	１時間　５分	５０分
7	１７８９	１時間　５分	５０分
8	１７５６	１時間　５分	５０分

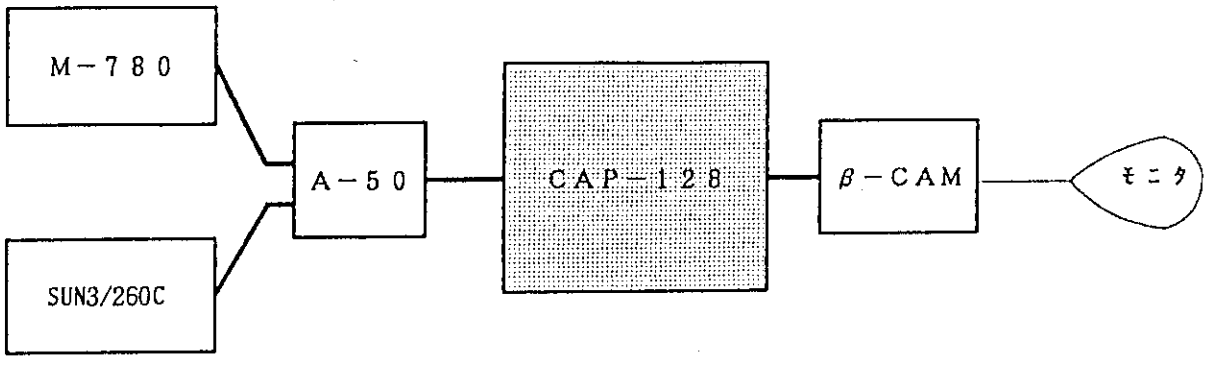


Fig. 5.1 Hardware configuration used in the Plant Geometry Database

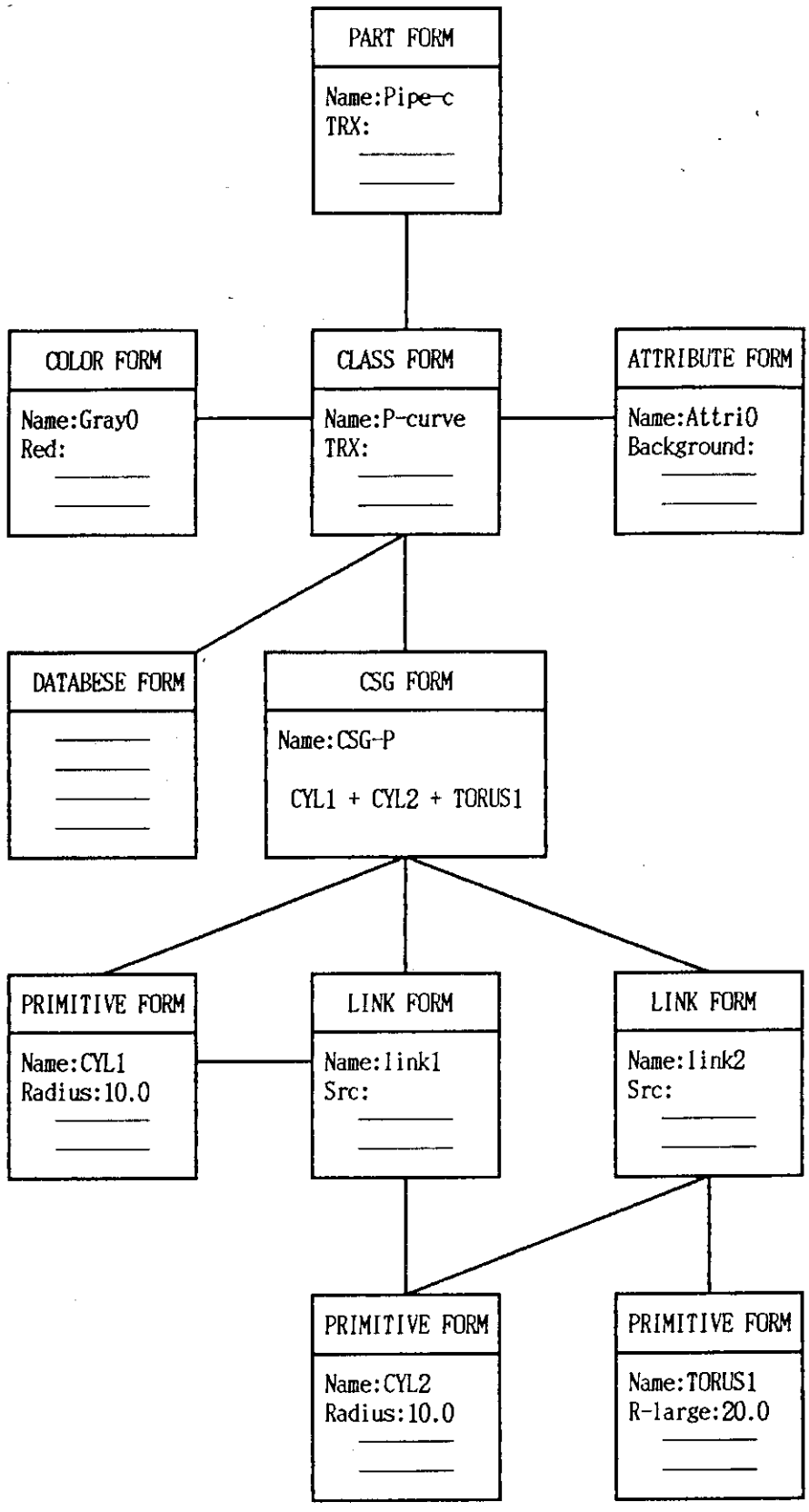


Fig. 5.2(a) Internal representation of geometric data in FUSION

Body Number	ITYPE	VALUE
1	1	0.0 0.0 0.0 1.0 1.0 1.0 10.0
2	1	10.0 10.0 10.0 0.0 0.0 0.0 10.0
3	2	0.0 0.0 0.0 10.0 20.0

ZONE NUMBER	
1	(1 2 3)

Fig. 5.2(b) Internal representation of geometric data in Monte Carlo code

6. 被曝線量評価

6.1 はじめに

本研究においては，原子力施設における人的作業の定性的定量的評価を行うシステムの開発が目的の１つとなっている。そこで，原子力施設において作業を行う人体またはロボットに対する被曝線量評価を行う。この時，人体については各臓器に対する放射線被曝を，ロボットに対してはロボット駆体に内蔵された各回路に対する放射線による損傷の度合いを計算する必要がある。

本研究においては，知能ロボットの動作空間としてJRR－3施設を想定しており，こういった複雑な形状によって計算領域が表現される体系においては，モンテカルロ・コードによる線量計算が必要となってくる。

本研究では，JRR－3施設において作業を行う人体または知能ロボットに対する被曝線量計算を行うシステムの開発を目指している。この計算には，現在施設形状データベースを使って入力中のJRR－3施設に関する形状データを用いることができる。また，計算センタで開発中のモンテカルロ計算装置を使用した高速計算も可能である。平成元年度には，最初のステップとして，米国のオークリッジ研究所において開発されたCristy phantom[1]を用いて，モンテカルロ・コードMCNP[2]による被曝線量計算を試みた。

6.2 MCNPコードによる被曝線量計算

平成元年度に，モンテカルロ・コードMCNPを用いて光子被曝についての線量計算を行った。以下に，使用した人体モデル，MCNPコードにおける形状データの作成方法，作業において判明した問題点について述べる。

(1) 人体モデル

今回計算に使用した人体モデルは，米国のオークリッジ研究所において開発されたCristy phantomである。このCristy phantomは，Snyderらによって開発された人体モデル[3]に，顔面の骨と乳房をつけ加え，更に肺や心臓等のいくつかの器官に改良を加えた人体モデルである。各臓器は，画像処理におけるCSG(Constructive Solid Geometry)法とほぼ同じ手法でその形状が表現されている。つまり，各臓器は，与えられた数式で表現される曲面の組み合わせにより記述されている。この手法により，例えば胃壁及び胃の内部は以下の２つの式によって表現される。

$$f(x,y,z) = \left\{\frac{x-x_0}{a}\right\}^2 + \left\{\frac{y-y_0}{b}\right\}^2 + \left\{\frac{z-z_0}{c}\right\}^2 - 1$$

$$g(x,y,z) = \left\{\frac{x-x_0}{a-d}\right\}^2 + \left\{\frac{y-y_0}{y-d}\right\}^2 + \left\{\frac{z-z_0}{c-d}\right\}^2 - 1$$

ここで，a＝4.0，b＝3.0，c＝8.0，d＝0.613，x_0＝8.0，y_0＝4.0，z_0＝35.0とする。この値は成人男子に対するデータである。胃壁は，f（x，y，z）で表される楕円体の負の部分及びg（x，y，z）で表される楕円体の正の部分の共通部分として定義される。また，胃の内部はg（x，y，z）の負の部分として定義される。計算に使用されたその他の各臓器に関する数学的表現については文献1）に詳しい。この人体モデルの断面図をFig. 6.1に示す。

被曝線量計算に使用される人体モデルは，Cristy phantomの他，MIRD（Medical Internal Radiation Dose Committe）によるMIRD phantom，西独のGSFにおいて開発されたADAMとEVA[4]等がある。これらの人体モデルでは，いずれもCristy phantomと同様，数式で定義された曲面体を組み合わせて各臓器や骨等を表現している。しかし，各臓器の位置や形状がやや正確でないという指摘と共にコンピュータ・トモグラフィーからのデータを使って被曝線量評価を行うといった手法[5]が提案されている。この手法については，6.3節で述べる。

(2) MCNP用形状データの作成

今回，被曝線量計算に使用したMCNPコードは，ロスアラモス国立研究所において1970年代の始めに開発されたMCNコードに光子輸送問題も扱えるように改良を加えられたモンテカルロ・コードである。遮蔽計算及び臨界計算が可能，ソース分布の種類が豊富，種々の分散低減法が用意されている等の汎用性の高いコードである。MCNP以外の多くのモンテカルロ・コードでは，球や直方体等の簡単な幾何形状の集合演算によって3次元の物体を定義するのに対し，MCNPでは平面や任意の2次曲面の組み合わせによって3次元の領域を定義する。

この作業において，Cristy phantomを構成する各臓器や種々の骨を表現する形状データをMCNPコード用の形状データによって表現するためのプログラムを作成した。このプログラムによって読み込まれたCristy phantomの形状データをMCNPコード用の形状データに変換する。このデータ変換処理における主な注意点は以下の通りである。

① Cristy phantomを表現する際に多用されている楕円球や楕円錐は任意の2次曲面として定義した。

② MCNPコードでは，形状表現に使用できる面として2次曲面のみを使用する。そこで，Cristy phantomを表現する際用いられている4次曲面については，それをZ方向（人体の背骨に平行な方向）に分割し複数の2次曲面で近似した。

このデータは変換作業において，以上の注意点の他に以下のような問題点がわかった。

(3) データ変換作業における問題点

MCNPコードにおいては，体系を記述するために使用できる領域（セルと呼ばれる）は，150セル以内であり，また，各セルは100 word以内のデータで表現されなければならないという制限がある。一方，さまざまな臓器や骨が重なり合う人体を表現した形状は複雑であり，ある領域を記述する際，そのデータ量が100 wordを越えてしまうことがある。そこで，多数の臓器及び骨が存在する胴体を人体の背骨に垂直な平面でいくつかに分割し，各領域における形状データの量を分散した。しかし，現在の所，Cristy phantomの一部をMCNPコード用形状データに変換したに過ぎず，完全な人体モデルについてのMCNP用形状データを作成する場合にこの制約は大きな困難となる。さらに，人体だけでなく，将来JRR－3施設における被曝線量計算を行おうとした時，この形状データの記述に対する制約によってそれが不可能に

なることは明らかである。

MCNPコードの150セル，100 word／セルという制約は，プログラムの容量だけでなく領域の複雑さによって増大する計算量をも考慮して制限しようとしたものである，と思われる。しかし，現在では主記憶及び磁気ディスクの大量な使用も可能となり，計算機の計算速度も飛躍的に速くなった。そこで，より複雑な形状の入力を可能とするため，MCNPコードの拡張を行う予定である。

6.3 被曝線量計算に関するその他の手法について

6.2節で述べたように，「Cristy phantomやその他の人体モデルにおいては，各臓器の位置や形状がやや不正確であり，特に骨随の表面が正確でない」との指摘と共に，従来とは全く違った人体ファントムが西独のGSFにおいて提案された[5]。この手法では，人体の各臓器，骨及び骨随を人間が数式を用いて定義するといういわば主観的表現に依らず，コンピュータ・トモグラフィから得られた形状データをそのままファイルにとり込むことにより，被曝線量計算を行おうという試みがなされている。この手法には，各臓器や組織の位置，形状に関して改善がなされたという利点の他に，骨と骨随の比率を正確にとらえて被曝線量計算を行えるという利点がある。これは，Co^{60}を使ったγ線照射に対する被曝線量を計算する際，骨を構成するカルシウムから出る2次電子の影響を正確に評価できるという意味を持つ。

形状の表現は，すべて同じ寸法の直方体（VOXELと呼ばれ，大きさは$0.85\times0.85\times4.00\ mm^3$または$1.54\times1.54\times8.00\ mm^3$）の積み重ねで行われている。これは，さまざまな幾何形状との交差テストを行うことが効率の良いベクトル処理を妨げているMCNP等のモンテカルロ・コードと異なり，良いベクトル処理結果を導く要因と考えられる。また，複雑な体系及び広域な領域における被曝線量計算を行うためのコードの拡張も容易であると考えられる。

6.4 おわりに

平成元年度は，モンテカルロ・コードを用いて人体に対する被曝線量計算を試みた。この際，モンテカルロ・コードとして汎用性の高いMCNPコードを用いた。また人体モデルとして，形状表現がMCNPコードと同等の手法を採用しているCristy phantomを用いた。平成2年度は，MCNPコードについて形状データを取り扱う部分の拡張を行い，Cristy phantomのMCNPコード用形状データを作成する。また，現在の多くの人体モデルでは直立状態のみを扱っているため，例えば歩行動作に対応する人体モデルの開発を行う。その他の手法についても原研の各関連分野と連絡をとりつつ検討していく予定である。

参考文献

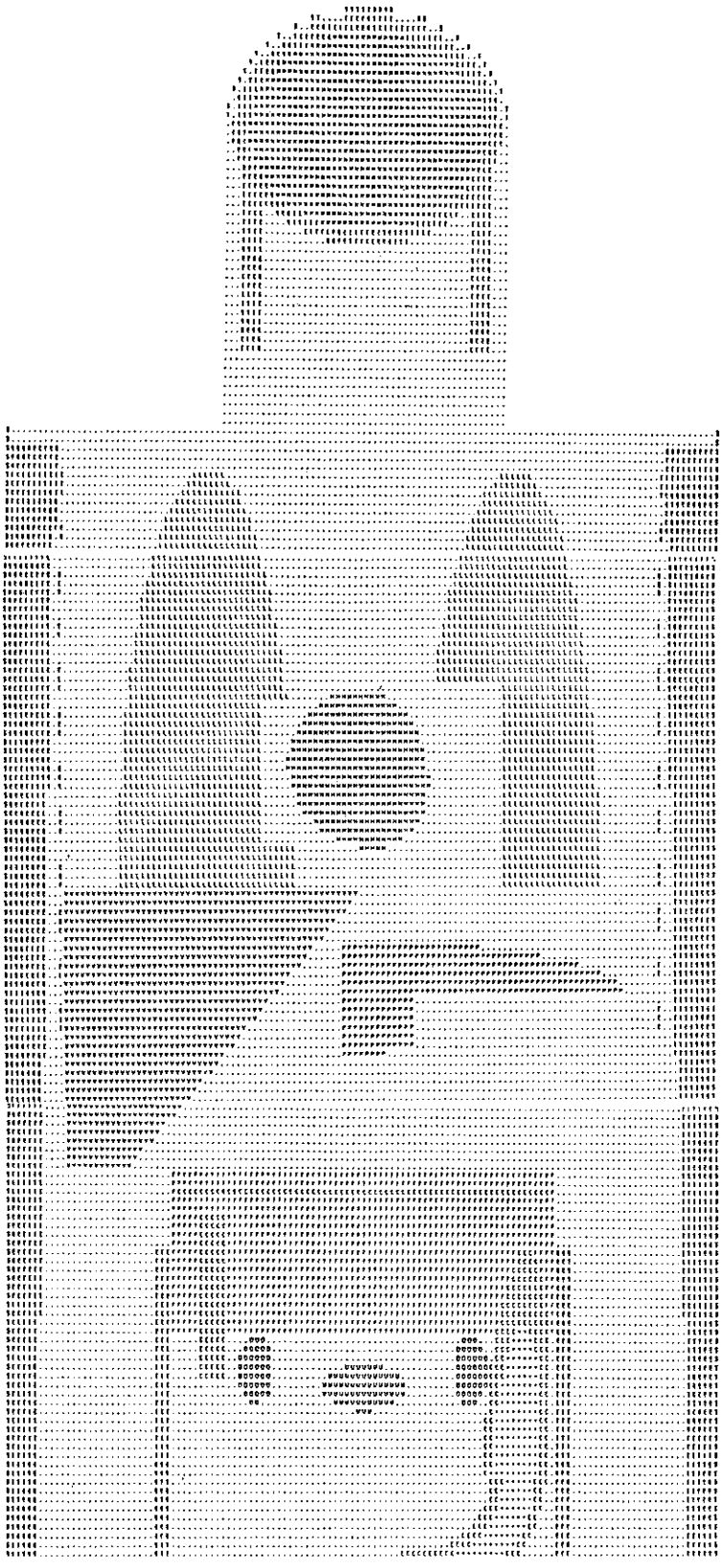
1) Cristy, M.: Specific Absorbed Fractions of Energy at Various Ages from Internal Photon Sources., ORNL-TM-8381, 1987

2) Judith, F.: MCNP--A General Monte Carlo Code for Neutron and Photon Transport Version 3A, LA-7396-M, Rev.2 Manual, 1986

3) Snyder, W.S. et al.: A tabulation of Dose Equivalent Per Microcurie-Day for Source and Target Organs for an Adult for Various Radionuclides, ORNL-5000, 1974

4) Kramer R. et al.: The calculation of dose from external photon exposures using reference human phantoms and Monte Carlo methods. Part 1: The male (Adam) and female (Eva) adult mathematical phantoms., GSF-Report S-885, 1982

5) Zankl, M. et al.: The construction of computer tomographic phantoms and their application in radiology and radiation protection, Radiat Environ Biophys Vol.27, pp.153-164, 1988



NOTATION OF ORGAN REGION

1	A	3	ADRENAL
2	G	4	GALL BLADDER
3	G	5	G.B.CONTENTS
4	N	6	BRAIN
5	M	7	STOMACH
6	C	8	UPP.LG.INT.
7	C	9	LOW.LG.INT
8	*	10	G.I.CONTENTS
9	F	11	SM.INT.+CONT.
10	H	12	HEART
11	K	15	KIDNEY
12	V	16	LIVER
13	L	19	LUNG
14	O	63	OVERY
15	P	64	PANCREAS
16	E	75	SKELETON
17	S	79	SKIN
18	X	80	SPLEEN
19	T	83	TESTIS
20	Y	84	THYMUS
21	I	85	THYROID
22	.	89	TISSUE
23	U	90	UTERUS
24	B	104	BREAST
25	R	144	URINARY BLADDER
26	R	145	U.B.CONTENTS

Fig. 6.1 Human phantom used in the calculation

7. モンテカルロ計算装置の仕様

7.1 モンテカルロ計算装置の必要性

(1) 日本原子力研究所におけるスーパーコンピュータの利用

科学技術計算用の特殊機能を有し，その時代の汎用コンピュータに比較して性能に格段の差があるコンピュータをスーパーコンピュータと呼んでいる。

現代のスーパーコンピュータと呼ばれるものの大多数はベクトル演算装置を内蔵するベクトル・プロセッサである。

日本原子力研究所（原研）では，ベクトル・プロセッサ（現在スーパーコンピュータと呼ばれているものは，すべてベクトル・プロセッサである）の原子力コードへの適応性調査を，1977年頃から開始した。この頃から原研における計算需要の伸びが，シリコン回路素子に基盤を置くコンピュータの性能の向上をはるかに上廻ることが明らかとなったからである。ベクトル・プロセッサは，この計算需要の増加とコンピュータ性能向上の鈍化とから起こる問題の有力な解決手段のひとつと予想された。

これはスーパーコンピュータCRAY－1が発表された頃の話である。日本においても当時二台のスーパーコンピュータが製作されていたが，[1]そのうち一台を上述の適応性研究に利用した。1977年から1年間は調査を，1978年からは上述の日本製のスーパーコンピュータを使用して約50本の原子力コードのベクトル化を行ない，その結果ベクトル・プロセッサによって大部分のコードは計算時間の大幅な短縮が可能であること，平均するとコードを表現しているFortranステートメントの15％程度を書き替えればよく，さほどのマンパワーを必要としないことなどを確認した[2,3]。特に核融合研究の流体モデルによるプラズマの不安定性解析には大きな力となった[4]。これらの事実をもとに原研は1985年に当時開発された新しいスーパーコンピュータを，1986年に二台目のスーパーコンピュータを導入した。これらのスーパーコンピュータは，核分裂，核融合の研究開発に大きな力を発揮している[5]。

原子力分野の計算は，核反応のみならず，構造，材料，機械力学など多くの分野の計算を包含しているので，原子力でスーパーコンピュータが有効であれば，ほとんどの科学技術計算においても同様である。日本においては，原子力分野へのスーパーコンピュータの導入をきっかけとして産業一般への普及が始まった。これは，原子力が新技術開発及び普及の牽引力となった一例である。

(2) モンテカルロ計算装置開発の意義

ベクトル・プロセッサに最も高い性能を発揮させるには，ベクトル演算器に途切れなくデータを供給する必要がある。これに適合する計算法の代表例は，有限差分法，有限要素法などであり，原子力分野ではよく使用されている。これらの計算法に対応する物理モデルは，希薄であれ濃密であれ，メッシュで区切られた空間の各セルには連続的に計算対象となる物質が存在すると仮定する連続体モデルである。物質は均質と見做され，物質の量は，セル内の平均値と

して求められる。このようなモデルにおいては，ベクトル演算器を重装備した最近のベクトル・プロセッサが適合する。

原研の調査で明らかとなったベクトル・プロセッサが有効でない計算の例として原子炉の三次元冷却材喪失事故解析，モンテカルロ法による三次元中性子・光子輸送解析がある。これらの解析に対応する計算には条件分岐が多く，データの連続的な流れを途切れさせ，またベクトルの要素数も短く，ベクトル・プロセッサの性能発揮の妨げとなる。モンテカルロ法では，空間に対して離散的な粒子が計算の対象となる。粒子は核種種別，エネルギー・レベルなどの個性を持つ。

最近の情勢からすると，個々の分子，原子を個別に眺めるような研究が新しい現象，物理法則の発見につながるようである。それに対応する計算法はモンテカルロ法である。例えば，原研で研究を進めている，核破砕による消滅処理のシミュレーション計算では，破砕によって生じる原子等を個別に分類して処理するが，この計算にはモンテカルロ法が適合する。

原理の成立性を検証するだけなら大量の計算は必要ないことが多い。核破砕のシミュレーションも原研での例では，ごく単純なモデルについてなら汎用大型コンピュータで1ケースについて10分程度の計算時間である。しかし，工学的な成立性が問題となると，多くの材質，エネルギー・レベルを対象としなければならず，所要計算時間は2桁以上増加するであろう。核反応を利用した新しい研究領域を広げようとすればするほど，原子や反応を個別的に眺めなければならず，シミュレーション手法としてはモンテカルロ法が重要性を増す。そして，原理の成立性から工学的成立性への検証段階では，大量のシミュレーション計算が必要となる。

したがって，連続体モデルの計算向きに作られた従来のスーパーコンピュータとは異なる，モンテカルロ計算向きの計算装置は，今後の原研の効率的な研究遂行に必要なものである。これは，大きなベクトル要素数，条件分岐の少ない均質なデータについて最大瞬間風速性能を誇る従来型のスーパーコンピュータとは異なり，少ない要素数，条件分岐の多い不均質なデータの処理について一定の処理速度を持続させる計算装置である　本仕様書によって設計するモンテカルロ計算装置は，このようなものである[6]。その概略図をFig. 7.1に示す。

7.2　ベクトル・プロセッサ改造型モンテカルロ計算装置

(1)　ベクトル・プロセッサ改造型のモンテカルロ計算装置

昭和63年度に行ったベクトル・プロセッサ改造型モンテカルロ計算装置の概念検討の結果，原研提案の機能を有する改造型ベクトル・プロセッサ1台で，テスト用のベクトル化されたモンテカルロ・コードについて，そのプロセッサのスカラ計算速度の5倍以上の速度向上が得られた。そこで，モンテカルロ計算装置の概念の基礎をベクトル・プロセッサに置くこととする。このタイプの計算装置の利点としては，次の(i)～(iii)が挙げられる。

(i)　従来のソフトウェア遺産の継承

(ii)　広い用途

(iii)　コードの容易な移植性

(2)　付加機能と目標性能

従来のベクトル・プロセッサに付加する機能，目標性能を次のとおりとする。

(a) 本モンテカルロ装置は，従来型のベクトル・プロセッサにFig.7.1の幾何形状分類パイプライン，事象分類パイプライン，領域判定パイプライン，アドレス・プリセット演算器，ロード／ストア・パイプラインの高速化，演算装置の多重化及び付随するソフトウェアの機能拡張によって，粒子モデルの計算に関する現在のベクトル・プロセッサの弱点を解消する。

(b) これらの付加機能によって，1台のモンテカルロ装置ではスカラ計算の2.5倍以上の，演算装置の多重化により，スカラ計算の10倍の処理速度向上を目標とする。

(c) 要求性能の概略はこの章の付録に示すとおり。

(3) 仕様の記述法

次ページ以降のハードウェア及びソフトウェア仕様において

（原研仕様）

のように記述された部分は，本モンテカルロ計算装置のための特注，付加，あるいは改造の仕様を示す。

7.3 モンテカルロ計算装置のハードウェア仕様

モンテカルロ計算を高速実行するための計算装置で，入出力プロセッサ及び制御と演算を行う複数のプロセッサから成る。これらのプロセッサは密結合（TCMP）とする。以下の仕様を満たす設計とする。

7.3.1 設計の範囲

(1) アーキテクチャ

機能分散方式を採用することとし，入出力プロセッサと制御・演算プロセッサとに機能分散を行う方式をハードウェアとして実現すること。

(2) 入出力プロセッサ

モンテカルロ計算装置の運転管理，入出力処理を行う。

(3) 制御プロセッサの機能

ジョブ管理，ファイル管理，通信管理，資源管理，原子力コードのコンパイル・結合編集，主記憶へのロードを行う。これに必要なコンピュータとしての命令と機能を有すること。

(4) 演算プロセッサの機能

スカラ及びベクトル演算による数値計算，また，これら数値計算に係る分岐，論理演算などを実行する。複数の演算プロセッサを使用するパラレル処理を効率よく実行するために演算プロセッサ間の通信機能を有すること。

7.3.2 機器仕様

7.3.2.1 演算プロセッサ

1台の演算プロセッサはベクトル演算器，スカラ演算器，ベクトル・レジスタ，ベクトル・マ

スク・レジスタ，スカラ・レジスタ等から成る。

(1) 演算プロセッサ

① ベクトル演算命令

特に断らない限りは64ビット演算とする。

ベクトル・ロード／ストア命令，ベクトル固定小数点演算命令（32ビット），ベクトル論理演算／シフト命令，ベクトル浮動小数点演算命令，ベクトル・マスク命令，ベクトル断化式命令，ベクトル収集／拡散命令，ベクトル制御命令，制御命令

② スカラ演算命令（原研仕様）

ロード／ストア命令，固定小数点／論理演算命令，シフト命令，浮動小数点演算命令分岐命令，プロセッサ間通信命令（単一及び同報機能を有すること），＊主記憶装置及び通信レジスタに関するテスト・アンド・セット命令＊

③ レジスタ（原研仕様）

スカラ・レジスタ

100個以上のスカラ・レジスタ（64ビット）

ベクトル・レジスタ

32Kバイト以上のベクトル・レジスタ

ベクトル・マスク・レジスタ

ベクトル・レジスタのベクトル要素数に見合ったビット数を有する。

ベクトル長レジスタ

ベクトル・レジスタのベクトル要素数を保存可能なビット数を有する。

＊ベクトル・タイマ・レジスタ＊

ベクトル演算時間を集積するためのレジスタ。ジョブのベクトル演算時間を知るために必要である。代替手段があれば示すこと。

プロセッサ・タイマ・レジスタ

演算プロセッサの演算時間を計測する。

プロセス状態語

演算プロセッサ上で動作するプロセスの状態を保持する（64ビット）。

演算プロセッサ間通信レジスタ

演算プロセッサ間での通信を行うためのレジスタ。マイクロ・タスキングの制御にも使用する。

割り込み状態保存レジスタ

通常の割り込み状態の他に，プロセッサ間通信による割り込み状態保存も可とする。

④ 命令の形式

演算を高速に，またデータのロード／ストアの回数を減らすために，命令はレジスタ上に載っているデータについて行い，結果は可能な限りレジスタ上に残す形式のものとする。このことから命令の形式は，レジスタ・レジスタ間の演算をひとつの命令で行うRR型（32ビット長）とする。また，データのロード／ストア命令は，ベース・レジスタの入れ替えをできるだけ避けるために，命令中のアドレス指定の変位が２＊＊３１まで可能とする。このRX

型命令の語長は 64 ビットとする。

⑤　ベクトル演算パイプライン（原研仕様）

ベクトル演算のパイプラインで，加算，減算，乗算，除算，論理演算，シフト，マージ，マスク生成を実行する。これらの演算を次の４つのカテゴリ，すなわち，浮動小数点／固定小数点加減算，浮動小数点／固定小数点乗算，論理演算／マージ／マスク生成，または，シフト／その他に分類し，それらのカテゴリの演算用に計４本のパイプラインを用意すること。これら４本のベクトル演算パイプラインを１セット以上と，１台の演算プロセッサは１セットのベクトル演算パイプラインを有すること。

⑥　ベクトル・ロード／ストア・パイプライン

原子力分野のモンテカルロ計算においては，１回の加算，または乗算などの演算について，１回以上のロード，またはストアがあるのが普通である。また，リスト・ベクトルを多用することが多く，データ・アクセスも連続番地ではない。モンテカルロ計算装置は，従来のスーパーコンピュータとこれからの点でも異なっている。したがって，各演算プロセッサのロード／ストア・パイプラインは，演算パイプラインの能力に見合ったデータ供給能力を有すること。

⑦　スカラ・パイプライン

文字取り扱い，バイトのゾーン取り扱い，メモリ間のデータ移送などの命令を含まぬＲＩＳＣ仕様の命令体系として，併せてパイプライン演算機能を有すること。命令解続，アドレス計算，データ・アクセスのみならず，固定小数点演算，浮動小数点演算，論理演算，シフト演算についてもパイプライン機能を有すること。

⑧　ベクトル命令とスカラ命令の並行動作

ベクトル命令とスカラ命令は，可能な限り並行動作可能とする。同一アドレス参照などによる並行動作不可能な場合の検出は，可能な限りハードウェアで検出すること。

⑨　データの形式

演算プロセッサの扱うデータの形式は次の３種とする。

- 論理データ
- 単精度，倍精度の２進固定小数点データ（負数は２の補数表示とする）
- 単精度，倍精度の浮動小数点データ（16 進数）

単精度，倍精度データのいずれも語の先頭の１ビットは符号を表すものとする。浮動小数点データでは，符号ビットに続く７ビットは指数を表す。

⑩　アドレス指定の方式及びアドレス変換機構

アドレス指定は通常の，実効アドレス＝空間番号＋ページ番号＋ページ内アドレスの方式とする。計算実行時にアドレス指定のためにベース・レジスタの頻繁な内容入れ替えを避ける必要がある。このためベース・レジスタ，インデックス・レジスタとして使用可能なレジスタを合せて 100 個以上有すること。同様の目的でページ内アドレス指定可能範囲は１Ｍバイト以上とすること。演算プロセッサは実効アドレスから絶対アドレスへの変換のためのアドレス変換機構を有すること。

⑪　ローカル・メモリ（原研仕様）

各演算プロセッサは，16 MB以上のローカル・メモリを有するものとする。モンテカルロ計算実行時には，どのローカル・メモリにも同一のモンテカルロ・コードのコピーがロードされ，各々独立に使用されるものとする（1多重での動作を仮定する）。

FORTRAN／MC言語は，LOCAL，GLOBAL変数の定義を可能とし，ローカル／メイン・メモリ間のデータの転送は代入文によって行うものとする。

このローカル・メモリの存在によって，利用者はモンテカルロ・コードの開発，利用において従来と同様に，単一のコードがあたかもメイン・メモリ上に乗っているかのごとく扱うことが可能となる。

メモリ・コンフリクトの発生を防げ，演算装置を効率的に利用可能となる。特に，将来の多重プログラミングへの機能拡張時に有効である。

ローカル・メモリの存在は，コードの開発・利用・計算速度の点で有効ではあるものの，他方では次のような欠点がある。

(a) メイン・メモリ／ローカル・メモリ間のデータ転送とローカル・メモリ／演算プロセッサ間のデータ転送の効率良い並行動作を保証することは，本質的にはコード依存の問題である。したがって，一般的な制御回路を組み込むことは，必ずしも効果的とは言えない。

(b) ローカル・メモリを有効に使用しようとすれば，利用者が入力データの異なる度にソース・コード中にローカル・メモリとメイン・メモリとの間のデータ転送を指示しなければならず，そのことは利用者にとって大きな負担となる。しかもこれまでの経験によれば，利用者のこのような指示は的を射ていないことが多い。

(c) 原子力分野のモンテカルロ計算においては，演算プロセッサとメモリとの間のデータ転送の量は，連続体モデルの計算ほどは多くない。

ローカル・メモリには，このような得失がある。そこでローカル・メモリに替わるものとして，現在の多重空間・仮想記憶のアドレッシング方式を利用した，単一ジョブの擬似的な多重プログラミング処理が考えられる（メモリ・コンフリクトの発生の問題は依然として残るが）。この方法についてはソフトウェア仕様の項で述べる。

上述のハードウェアによる方法とソフトウェアによる方法について工数，開発期間，費用，効果を比較し，いずれかの方法を選択すること。

ハードウェアとしてのローカル・メモリを装備しない場合には，少なくとも命令の先取りを行なうための命令バッファを有すること。

⑫ アドレス・プリセット演算器（原研仕様）

現在実行している命令群（主として，ひとつのDOループを構成する命令群）とは論理的に無関係なDOループ（利用者がソース・コード中で指定）について，そのループの実行に必要な変数のアドレスを前以って設定するためのアドレス・プリセット演算器を有すること。ベクトル演算のためにスカラ演算器が占有されることもあり，また，コード中の変数の論理的順序関係から，このような演算器の存在が計算の高速化に有効な場合がある。

現在のスーパーコンピュータは，スカラ演算とベクトル演算の並列動作が可能なように設計されている。現状の設計及び命令を変更することでアドレス・プリセット演算の代替が可能であれば，その案と効果，開発工数，開発費等についてプリセット演算器との比較・評価

を行い，代替機能を採用すること。

⑬　幾何形状分類パイプライン等

幾何形状分類，事象分類，粒子領域決定パイプラインについては画像生成，解析等の応用分野も考慮のうえハードウェア化またはソフトウェア化について検討・実現すること。

7.3.2.2　制御プロセッサ

制御プロセッサは，ジョブの入力・出力，ファイル処理，ジョブ・スケジューリング，通信ネットワークの管理を行う。制御プロセッサのアーキテクチャは，演算プロセッサのスカラ演算部分のアーキテクチャと類似のものでもよい。

7.3.2.3　主記憶装置（原研仕様）

512 MB（メガバイト）以上の容量を有し，演算プロセッサの処理能力に見合った転送能力を有すること。演算プロセッサの要求データ処理能力は，1台のプロセッサにつきピーク性能で1 GFLOPS以上である。粒子モデルのモンテカルロ計算では，1演算につき1データ（8バイト以上）が必要と見込まれている。

7.3.2.4　入出力プロセッサ（1台）

周辺装置と主記憶装置の間に在って，制御プロセッサからの指令に従い，入出力処理を行うと共に，OSの初期ロード，モンテカルロ計算装置のシステム初期起動・停止等を行う。

入出力チャネル―― 3 MB（メガバイト）／秒の転送能力。磁気ディスク，磁気テープ，通信制御装置，プリンタ等の接続用。

高速入出力チャネル―― 4.5 MB（メガバイト）／秒の転送能力。

超高速入出力チャネル―― 20 MB（メガバイト）／秒の転送能力。

7.3.2.5　オペレイタ・コンソール

電源投入・切断スイッチ，プリンタ及びカラー・ディスプレイによる状態監視機能を備え，本体から100メートルまで離れた位置から操作可能なこと。

7.3.2.6　自動運転制御装置

自動電源投入・切断機能，システム設置環境の火災，地震，温度，湿度異常の検出報知機能を有する装置であること。

7.3.2.7　冷却装置

空冷型または水冷却装置とする。

7.3.2.8　空調容量

200,000 Kcal／H程度とする。

7.3.2.9　電源

AVR経由の200 V，250 KVA以下の容量とする。この他に，100 V，50 KVA程度の容量の範囲に収めること。

7.3.2.10　設置スペース

本体系で40平方メートル程度，システム全体で100平方メートル程度とする。

7.4　モンテカルロ計算装置のソフトウェア仕様

7.4.1　制御プログラム

モンテカルロ計算装置は特殊目的のものである。汎用的目的の制御プログラム（OS)は，モンテカルロ計算装置の利用の面のみから見ると冗長な部分が多く，使用勝手も良くない。科学技術計算用のコンピュータのOSとして最近多用されているUNIXを利用する。これによって，計算装置の容易な利用が可能となり，また，各種ワークステーション，ミニコン等との接続法，ソフトウェアの共用性が保たれる。

UNIXは，もともとは1960年代の汎用的OSであるMULTICSの考えを簡易に実現することを目標として作られたものであるため，現在の高速コンピュータのOSとしては，次の諸点に弱点があると米国の専門家から指摘されている。

メモリ管理，仮想記憶管理の機能が十分ではない。

ファイル構造が大量データを取り扱うように出来ていない。

イメージ・データを取り扱えず，可視化システム実験が困難である。

パラレル入出力の機能がない。

マルチプロセッサのスケジューリングの機能が弱い。

クラスタ化など複合マシン化の機能がない。

そこで，AT&T　UN X及びCalifornia Univ. Berkeley版UNIXの両方の機能を包含し，かつ上記の機能を付加したモンテカルロ計算装置用の拡張UNIXを新たに開発，乃至は開発済のものがあれば，それを採用することとする。さらに，この拡張UXIX は，ベクトル及びパラレル処理の高速化を制御するためのマクロ及びマイクロ・タスキングの機能，ワークステーション，グラフィック端末等を制御するネットワーク管理の機能及び将来の拡張記憶装置の付加を考慮して高速スワッピングの機能を包含するものとする。

拡張UNIXが保持すべき具体的な機能は次のとおり。

①　メモリ管理（原研仕様）

＊仮想空間管理＊－　システム，プロセス仮想空間の管理など。

ハードウェアとしてのローカル・メモリをソフトウェアで代替する場合には，次の方法によるものとする。

単一ジョブのローカル・データ指定されたデータ領域のコピーが実演算プロセッサの数に対応した数だけメイン・メモリにロードされる。これらのコピーは，それぞれが対応する実演算プロセッサに固有なものである（将来は仮想演算プロセッサにまで

拡張する）。実演算プロセッサがマイクロ・タスキングの実行中は，これらの対応するコピー（擬似，あるいは部分仮想空間）へのアクセスを可能とする。モンテカルロ計算においては，ＤＯループのネストは浅く，通常は１重のループである。したがって，ＤＯループのマイクロ・タスキングはベクトル処理である。これとは別に，各物理事象の処理においては，粒子の独立性を生かしてマイクロ・タスキング処理を行う。その制御に演算プロセッサの１台を当てることとする。この演算プロセッサは，マイクロ・タスキングの制御を行う一方，ＤＯループ処理の対象となっていない事象の計算も行うものとする。具体的には,FORTRAN言語のモンテカルロ計算用PARCASE文のパラレル処理に該当する。このマイクロ・タスキングは，オーバーヘッドを極力減少させるために，演算プロセッサ間の通信によって行う。制御にあたる演算プロセッサからは同報通信によって他の演算プロセッサに通知することもある。したがって，受信するデータには当該プロセッサには関係の無い情報も含まれることがある。通信バッファは，ＯＳカーネルの排他制御，マイクロ・タスクのための同期／スケジューリング，プログラム間での通信のための情報（パラメータ・リストのベース・アドレス等）を格納するための充分な容量を有すること。仮想部分空間のひとつにグローバル変数のための共通領域がある（FORTRANコンパイラ及びリンカ／ローダの項を参照のこと）。

以上の制御方法を実現するために，付加すべき仮想空間管理機能として

＊単一ジョブの仮想部分空間制御＊　－　PARCASE文マイクロ・タスキング処理のための空間制御を行う。

＊データ配置機能＊　－　データ領域拡張，共同メモリの管理など。単一ジョブのローカル・データ領域のコピーをジョブ制御文に指定された実演算プロセッサ数だけメイン・メモリ上に配置する。グローバル変数のための共通領域をメイン・メモリ上に配置する。それらのページ・ベースをマイクロ・タスキング制御用の演算プロセッサの制御用領域にセットする。あるいは，これらと同等の機能の実現を検討すること。

＊高速スワッピング機能＊　－　主記憶，拡張記憶のメモリ入れ替え。

② プロセス管理

・プロセスＩＤ管理，ディスパッチング　－　プロセス認識，実行起動など。

・プロセス動作管理　－　プロセス会計情報，ユーザ管理，実行環境制御など。

③ 実行管理（原研仕様）

・システム運転機能　－　システム初期化，終了など。

・システム運用機能　－　タイマー管理，システム情報，割り込みレベル制御など。

＊タスク処理機能＊　－　マクロ／マイクロ・タスキング，CP／AP 間通信，AP／AP通信など。

④ ファイル管理（原研仕様）

・従来UNIX型ファイル管理機能　－　階層化木構造，周辺装置ファイル化など。

＊大容量ファイル管理＊　－　大容量ファイル作成，高速転送，並行入出力など。

・ネットワーク・ファイル機能　－　ネットワーク透過なファイル・アクセスなど。

ネットワーク管理

・WAN 機能　－　調歩同期無手順，全２重回線のサポートなど。

・LAN機能 － Ethernet，HYPER channelのサポートなど。
・プロトコル機能 － TCP/IP，UDPプロトコルのサポートなど
・構成制御機能 － 構成定義，管理コマンド群，動的構成変更の機能など。

⑤ 運用管理
・利用者管理機能 － 登録，セキュリティ・チェックなど。
・資源保護機能 － ファイル／ディレクトリの保護。
・ファイル領域管理機能 － 利用者ファイル領域の管理
・課金情報管理機能 － ログイン情報，プロセス情報など。
・システム計測機能 － 装置使用率，入出力回数の計測など。
・ファイル・バックアップ機能 － ファイル・バックアップ／回復機能など。
・システム・フリーズ機能 － システム凍結／再開機能など。
・クラッシュ回復機能 － 実行待ちジョブの回復など。

⑥ バッチ処理
・ジョブの登録
・ジョブの実行中断／再開
・ジョブの実行優先度の変更
・出力装置へのファイル出力機能
・分散システム環境でのジョブ転送機能

7.4.2 FORTRAN言語／MC

① 言語仕様
・FORTRAN77準拠であること。

② 拡張仕様（原研仕様）
・4倍精度実数型，倍精度実数型，4倍精度複素数型
・非同期入出力文
・日本語型データの処理
・ベクトル化指示行
・パラレル化指示行
＊拡張ベクトル化指示行＊
＊拡張パラレル化指示行＊

7.4.3 FORTRANコンパイラ／MC

① 最適化機能
・共通式の削除
・不変式のループ外への移動
・外部手続きのインライン展開
・その他

② 自動ベクトル化機能

・対象文　－　代入文，CONTINUE文，IF文，ELSE IF文，END IF文，単純GO TO文。

・拡張ベクトル化機能　－　ループ分割，外部関数の疑似ベクトル関数化，文の入れ換え／作業用ベクトルの導入，ベクトル化指示行，ループ一重化／入れ換え，ループ展開，インライン展開。

③　パラレル化機能（原研仕様）

・マクロ・タスキング　－　FORK／JOIN，POST／WAIT，LOCK／UNLOCKの機能。

・マイクロ・タスキング　－　DOループ対象の自動パラレル化。対象文は，代入文，IF文，ELSE文，GO TO文，DO文，CALL文，ASSIGN文。

指示はPARCASE／CASE／END CASE，SERIAL／END SERIAL及びPARALLEL，NOPARALLELとする。

＊拡張マイクロ・タスキング＊　－　空いているプロセッサに順次タスクを割り当てるダイナミック・スケジューリング方式に加えて，PARCASE文を拡張して利用者がダイナミックにタスク・スケジューリングをプログラム中に行なえる機能を付加する。

本モンテカルロ計算装置では，DOループの自動パラレル化，即ちマイクロ・タスキングは行なわない。モンテカルロ計算においては，DOループのネストは通常は1重であり，またループ内の計算量も多くない。したがって，DOループはベクトル処理を行なうこととし，DOループの幾つかのまとまりをひとつの演算プロセッサに割り当てる方式を採用する。その詳細は別紙5のとおり。あるいは，それと同等または代替となる拡張機能を付加すること。

＊拡張コモン機能＊　－　パラレル処理を実行している多重部分空間上のコードから共通して参照可能なコモン領域を定義可能なこと。

コモン・メモリの代替案に必要となる機能

＊仮想部分空間制御セクションの生成＊　－　モンテカルロ・コードをコンパイルする際に，これが従来のように1個の仮想空間を占有するとしてコンパイルすることに加えて，ローカル・データ領域が仮想部空間を占めるとしてのコントロール・セクションをも作り出すこととする。このコントロール・セクションは，ローダがコードを主記憶（メイン・メモリ）へ多重にロードするために使用する。あるいは，これと同等の機能を実現すること。

7.4.4　FORTRANライブラリ

①　数学サブルーチン・ライブラリ

・スカラ計算用数学ライブラリ　－　行列演算，連立1次方程式，固有値，固有ベクトル，最小2乗法，高速フーリエ変換，数値積分，特殊関数，関数近似，代数方程式，確率分布，検定，回帰分析，乱数など。

②　ベクトル計算用数学ライブラリ　－　同上のベクトル化版

7.4.5　特殊サブルーチン・ライブラリ（原研仕様）

3次元画像生成，解析等への応用を考慮し，ハードウェアとしてのパイプライン化と既存ベクトル命令の組合せによるソフトウェア・ライブラリ化の得失を検討し実現すること。

① ＊幾何形状分類ライブラリ＊　－　モンテカルロ計算において粒子の属する領域の幾何形状をベクトル演算により高速分類するサブルーチン。

② ＊事象分類サブルーチン＊　－　モンテカルロ計算において発生した事象をベクトル演算により高速分類するサブルーチン。

③ ＊粒子領域決定サブルーチン＊－　粒子が正しく定義された領域に存在するかどうかを判定する論理式をベクトル演算により高速計算するサブルーチン。

7.4.6　リンカ／ローダ（原研仕様）

① ＊拡張リンカ／ローダ＊　－　仮想空間及び仮想部分空間用の二種のコントロール・セクションを持つようにコンパイルされているコードを，制御文の指定にしたがって一重，又は多重に主記憶上にロードし，その後制御プロセッサにコントロールを引き渡す機能を有すること。通常の結合・編集機能も有すること。

7.4.7　デバッグ支援ソフトウェア

本モンテカルロ計算装置は，ベクトル演算機能，パラレル演算機能の双方の特長を生かしつつ，高速計算を達成しようとするものである。利用者がこれらの機能を充分に使いこなすことができるような支援ツールが必要である。支援ツールの持つべき機能としては，

プログラム構造の静的解析

ベクトル化率推定

パラレル率推定

プロセッサ利用率推定

などが必要である。これらの解析のためにつぎの解析ソフトウェアを用意すること。

① ANALYZER

サンプル入力データを使用してプログラムを実行し，実行途中の動的情報を翻訳単位で解析し，プログラム中のDOループのベクトル化の可能性，達成可能なベクトル化率などについての情報を出力するソフトウェアである。

以下の情報を出力すること。

〔動的解析情報〕

各実行文の実行回数とコスト，

条件付き分岐文の分岐先ごとの選択回数とその比率，

ベクトル化されたDOループの表示，

ベクトル化されたDOループの平均ループ長，

ベクトル化されなかったDOループの理由，

プログラム単位毎のベクトル演算率。

〔静的解析情報〕

プログラム構造図（木構造図）
プログラム単位間の相互参照関係
仮引数と実引数との対応関係
共通ブロック要素の相互参照関係

② PARALLEL ANALYZER

〔動的解析情報〕
プログラム中のタスクの流れ，
プログラムの実行時間，ベクトル化率，パラレル化率，
各タスク毎の実行時間，
各タスク毎の同期待ち時間，同期箇所，
CALL文ごとのプログラム実行時間，
ＤＯループ毎の実行時間，
プログラム中の指定範囲の実行時間。
〔静的解析情報〕
タスクの起動，同期関係の表示
同期／排他制御関係，
共有変数の相互参照関係，
自動パラレル化されたサブルーチン及びループの共有変数の参照関係，
プログラム単位毎のベクトル化，パラレル化の可否と理由。

＊拡張解析情報＊（原研仕様）

本モンテカルロ計算装置を効率良く使用するために，別紙に記載するように利用者が制御可能なPARCASEによるマイクロ・タスキングの機能を付加する。そのために，PARCASE，END PARCASEで囲まれた範囲の文を擬似的なタスクと見做して，上記の動的及び静的情報を出力する機能を付加すること。

7.4.8 Ｃ言語

ワークステーションのシステム及び関連ユーティリティ開発，ワークステーション・モンテカルロ計算装置間のネットワーク・ソフトウェア開発の言語としてＣ言語を使用する。また，モンテカルロ計算装置のシステム・ユーティリティ開発用言語としてもＣ言語を使用する。以下の機能を有するＣ言語コンパイラを用意すること。

① 自動ベクトル化機能
最適化機能

参考文献

1) O. Miwa et al., "FACOM230-75 Array Processor System", FUJITSU, Vol.29, 1978 (in Japanese)
2) M. Ishiguro et al., "Adaptability of Vector Processing to Large-scale Nuclear Codes", JAERI-M 82-018, Feb. 1982 (in Japanese)
3) K. Asai, "A Design of a Computer Complex Including Vector Processors", JAERI-M 82-200, Dec. 1982 (in Japanese)
4) M. Azumi et al., "A Fluid Model Numerical Code System for Tokamak Fusion Research", Proc. 4th Int'l Symp. on Comp. Methods in Appl. Sci. & Eng., Versailles, France, 1979, North-Holland Publ.
5) M. Ishiguro et al., "Performance Analysis of Vectorized Nuclear Codes on a FACOM VP-100 at the Japan Atomic Energy Research Institute", The Int'l Jour. Supercomputer Appl., Vol.1, No.3, 1987
6) K. Asai et al., "A Study on Intelligent Nuclear Systems (HASP: Human Acts Simulation Program) - Progress Report 1988 -, JAERI-M 89-023, Feb. 1989 (in Japanese)

付録　モンテカルロ計算装置の要求仕様

1. 基本的要件

(1) ベクトル・プロセッサ

ベクトル化モンテカルロ・コードを基本に高速化を図る。モンテカルロ計算装置のプロセッサ群の基本単位はベクトル・プロセッサでなければならない。

(2) メモリ・アドレスの指定方法

ロード／ストア命令，アドレス指定の分岐命令の実行を直接アドレス指定で行なえること。

（データのロード／ストア及びベクトル命令実行のために，ベース・レジスタの入れ替えが頻繁に発生することを避ける。）

(3) 対スカラ性能比

ベンチマーク用のモンテカルロ・コードについて，ベクトル化されたコードに関して単一ベクトル・プロセッサとしての処理性能が，スカラ・コードでのスカラ処理性能の 2.5 倍以上であること。

（現用の中性子・光子輸送計算コードに対して，少数台の密結合プロセッサで１０倍上の処理性能を得るためには，単一プロセッサで 2.5 倍以上の処理能力が必要。）

(4) 密結合プロセッサ・システム

密結合プロセッサ・システムとして２台以上の並列処理可能なベクトル・プロセッサ及びこれを制御するためのソフトを有すること。

（現用のモンテカルロ・コードに対しては，単一プロセッサでは１０倍の目標値は達成できない。並列処理の導入が必要である。）

(5) 高速演算性能

基本となる単一ベクトル・プロセッサのスカラ演算性能は 100 MIPS以上，ベクトル演算の最高性能は 1 GFLOP 以上であること。

（製作後５年間は世界最高速のモンテカルロ計算装置であることが望ましい。）

2. ハードウェアの構成

概略，Fig. 7.1 に示す構成の装置とする。

(1) 制御プロセッサCPU１

ジョブの入出力管理，ファイル管理，コンパイル結合・編集，タイムシェアリング処理，通信制御等を行う。

(2) 入出力プロセッサCPU2

周辺装置と主記憶装置の間にあって，制御プロセッサからの指令にしたがい，入出力処理を行うとともに，ＯＳ の初期ロード，モンテカルロ計算装置の起動・停止制御等を行う。

(3) 演算プロセッサＡＰ

ＡＰ の概要は Table 7.1 のとおり。この他に，マイクロ・タスキング用の AP 間通信レジスタ及び関連命令を有すること。

(4) ローカル・メモリ

ＡＰとメイン・メモリMMUとの間に在って，ＡＰで使用するプログラム命令及びデータ，即ち単一のユーザ・プログラムをロードするためのメモリである。

このメモリの存在によって，

(i) ユーザ・プログラムを大幅に変更することなく並列処理可能となる，

(ii) メイン・メモリMMUでのメモリ・コンフリクトを減少させ得る

などの効果がある。

各ローカル・メモリには，単一プログラムの実行開始前には，そのプログラムの同一コピーがロードされている。

ソフトウェアによって，これとほぼ等価な機能を実現する方法もある。両方の方法を比較検討し，より安価かつ機能，性能を低下させないものを採用すること。

3. ソフトウェア

3.1 運用に係るソフトウェア

下記のソフトウェアを有すること。

(1) ジョブ管理

ただし，モンテカルロ計算装置の初期の運用においては，First Come First Served（FCFS）のユニプログラミング方式で運用する。

(2) プログラム実行管理

(3) データ管理

(4) ＲＡＳ機能

(5) 運用管理

(6) ネットワーク管理

(7) 対話情報処理機能

3.2 言語処理に係るソフトウェア

下記の言語処理系及び関連ソフトウェアを有すること。

(1) Fortran 77

Fortran 77 の言語仕様に準拠したFortran 言語のためのコンパイラ。言語及びコンパイラは，自動ベクトル化機能，自動並列化機能，最適化機能を有すること。モンテカルロ計算のための拡張機能は 5.のとおり。

(2) 関連支援ツール

ベクトル化，並列化のためのデバッグ及びチューニング・ツールを有すること。

(3) ベクトル，並列処理用科学計算ライブラリ

Fig. 7.1 の幾何形状分類パイプライン，事象分類パイプラインのベクトル化ライブラリを含むこと。領域判定パイプラインについてはライブラリ化の有効性について調査後にライブラリ化を検討する。

4. モンテカルロ法における代表的計算要素についての要求性能

演算プロセッサの計算の速さをMP，そのスカラ計算の速さをＳＰとするとき，比MP／SPを

性能向上比と呼ぶ。代表的な計算要素について，単一プロセッサのベクトル計算において以下の目標値を達成すること。

(1) 粒子飛距離計算

間接番地指定の粒子飛程距離計算（Fig. 7.2）において，100粒子についてMP／SP＞4であること。

(2) 散乱角計算

間接番地指定の粒子散乱角計算（Fig. 7.3）において，100粒子についてMP／SP＞4 であること。

(3) 粒子通過領域判定計算

(a) 円筒領域

条件分岐を含む円筒領域の粒子通過判定計算（Fig. 7.4）において，間接番地指定の100粒子についてMP／SP＞4 であること。

(b) 立方体領域

条件分岐を含む立方体領域の粒子通過判定計算（Fig. 7.5）において，間接番地指定の100粒子についてMP／SP＞4 であること。

(c) 球領域

球領域の粒子通過判定計算（Fig. 7.6）において，間接番地指定の100粒子についてMP／SP＞4 であること。

5. 並列処理に関するFortran拡張機能

並列処理のためのユーザ・プログラムの変更を最小限に留めるために，モンテカルロ計算装置の利用においてはコンパイラ指示行PARCASEを主として用いるものとする。この考えのもとで以下に記述する機能を検討し，同等あるいは代替となる拡張機能を付加すること。

(1) プロセッサ番号の陽指定

利用者が最適なスケジューリングを行えるようCP，AP 1，・・・，APn のプロセッサ番号を陽に指定可能とする（後述のPLACTIVATEで使用）。

(2) CASE番号の陽指定

PARCASE nn，……，END CASE mmとCASE番号 nn（nnは整数変数）を陽に指定可能とする。また，整数変数 mmには当該CASE bodyの実行終了後に整数値1が入るものとする。

(3) SERIAL番号の陽指定

SERIAL，END SERIAL指示行についてもCASEと同様とする。

(4) PARCASEの優先順位

PARCASEは1次元の優先順位をプログラム単位中で与えられること。

(5) PLACTIVATE var 指示行の付加

利用者がプログラム実行中に動的に並列処理のPARCASE body列を再編成するためのコンパイラ指示行PLACTIVATE varを付加すること。var は1次元配列で，（FFF0，FFF1，実プロセッサ番号，PARCASE番号，……，FFF1，実プロセッサ番号，PARCASE番号，…，

FFFF）の形式とする。

(6) ユーザ指定の並列処理スケジューリング

PLACTIVATE var 指示行を含むPARCASE body はユーザ指定のスケジューリング・ブロックである。このPARCASE body には当該プログラム単位中の最高の優先順位を与えるものとする。スケジューリング・ブロックは，プログラム単位で定義可能とする。サブルーチンに含まれるスケジューリング・ブロックから実プロセッサを要求した場合には，現在実行中のPARCASE bodyを割り込み中断し，サブルーチンのスケジューリング・ブロックの指示に従い実行を開始する。スケジューリング・ブロックは，通常はメイン・メモリに置かれているユーザ・プログラムのものが実行される。

Table 7.1 Charactoristics of Monte Carlo machine

最大ベクトル演算性能			1GFLOPS以上
AP台数			1～4台
演算プロセッサ（AP）	レジスタ	ベクトルレジスタ	36Kバイト
		ベクトルマスクレジスタ	64ビット×8個
		スカラレジスタ	64ビット×128個
	データ形式	固定小数点	32／64ビット
		浮動小数点	32／64／128＊ビット（＊スカラ命令のみ）
		論理	64ビット
	ベクトル演算パイプライン		1～2本
	スカラ演算パイプライン		1セット
	マスク付ベクトル処理		可
	リストベクトル処理		可
	スカラ演算用キャッシュメモリ		64Kバイト
	モンテカルロパイプライン		1セット（検討）
	アドレスプリセット演算器		1セット（検討）

ベクトル計算機改造型モンテカルロ装置

従来のスーパーコンピュータ（ベクトル計算機、下図の細線部分）に粒子モデル高速計算機用の５種の機能（太線部分）を付加する。ベクトル・レジスタ、従来型演算パイプラインの簡素化及びロード／ストア・パイプラインの高速化を行う。さらに、回路を多重化する。

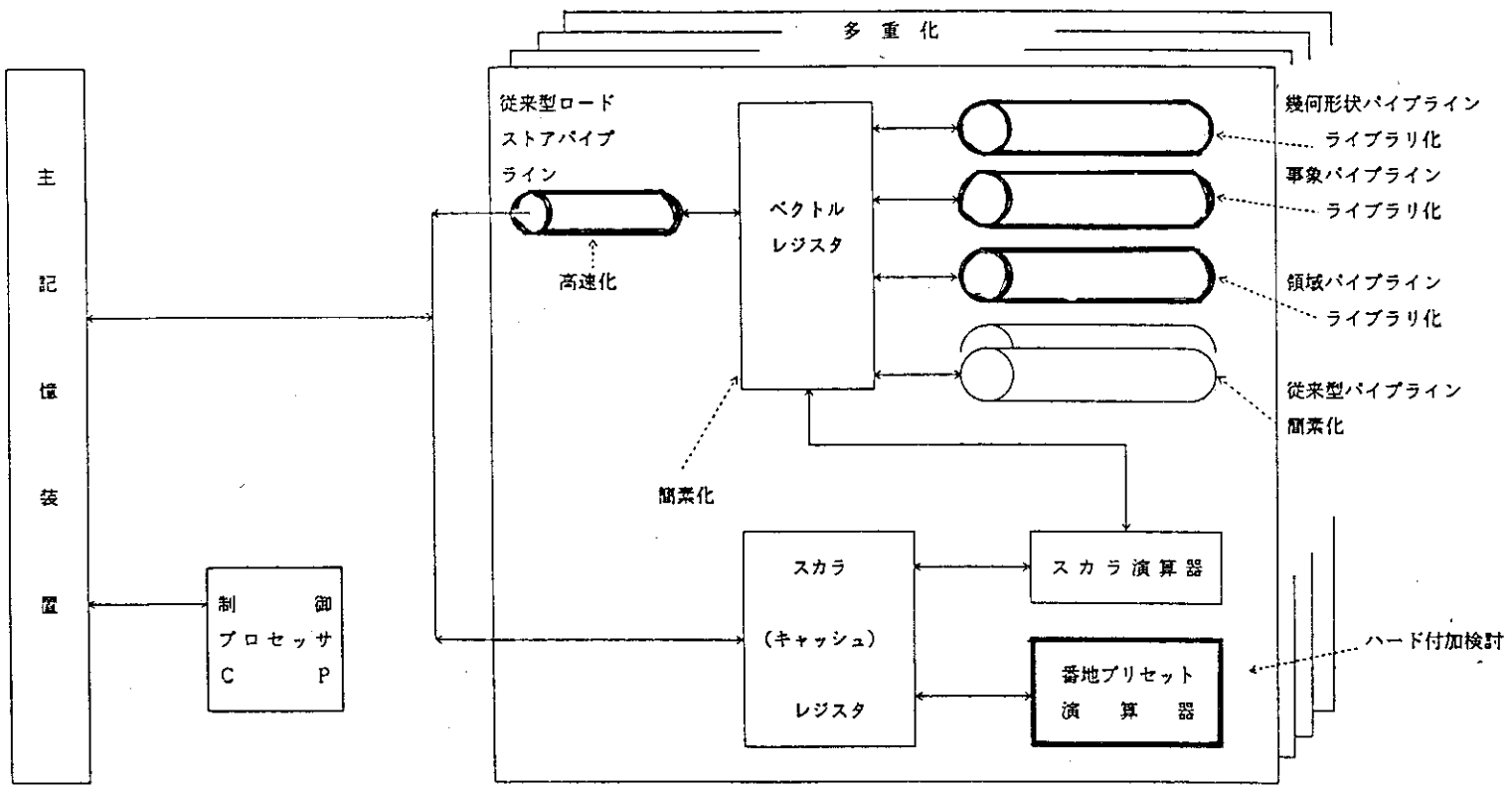


Fig. 7.1 Concept of Monte Carlo machine

```
      DO 9320 IV=1,NPATH
      JV=PATHBA(IV)
      KR(JV)=MAT(K(JV))
      IF(KR(JV).NE.0) THEN
      PTH(JV)=RPTH(JV)*RSIGT(KR(JV),IG(JV))
      ELSE
      PTH(JV)=BIG
      ENDIF
      X1(JV)=X(JV)+PTH(JV)*U(JV)
      Y1(JV)=Y(JV)+PTH(JV)*V(JV)
      Z1(JV)=Z(JV)+PTH(JV)*W(JV)
      IF(X(JV).EQ.X1(JV)) X1(JV)=X(JV)+SIGN(X(JV),U(JV))*(1.0E-6)
      IF(Y(JV).EQ.Y1(JV)) Y1(JV)=Y(JV)+SIGN(Y(JV),V(JV))*(1.0E-6)
      IF(Z(JV).EQ.Z1(JV)) Z1(JV)=Z(JV)+SIGN(Z(JV),W(JV))*(1.0E-6)
 9320 CONTINUE
```

Fig. 7.2 Particle transport calculation

```
        DO 9912 IV=1,NSURV
        JV=SURVBA(IV)
     *VOCL STMT,IF(100)
        IF(MUBARX(IV).NE.0.0) THEN
          FMU=MUBARX(IV)
          SINPSI=SQRT(1.0-FMU*FMU)
     C..  CALL AZIRN(SINETA(JV),COSETA(JV))
          SINETA=SIN(PI2*WK1(IV))
          COSETA=COS(PI2*WK1(IV))
          STHETA=SQRT(1.0-U  (JV)*U  (JV))
     *VOCL STMT,IF(100)
          IF(STHETA.GT.0.0) THEN
            COSPHI=V(JV)/STHETA
            SINPHI=W(JV)/STHETA
          ELSE
            COSPHI=1.0
            SINPHI=0.0
          ENDIF
          V(JV)=V(JV)*FMU+U(JV)*COSPHI*COSETA*SINPSI
         +        -SINPHI*SINPSI*SINETA
          W(JV)=W(JV)*FMU+U(JV)*COSETA*SINPHI*SINPSI
         +        +COSPHI*SINPSI*SINETA
          U(JV)=U(JV)*FMU-COSETA*SINPSI*STHETA
        ENDIF
 9912 CONTINUE
```

Fig. 7.3 Particle scattering angle calculation

```
      DO 9290 IV=1,J280
      JV=CRO280(IV)
      Q=TX(JV)*RR(JV) + TY(JV)*S(JV)
      F=XXK4(JV)-TX(JV)*TX(JV)-TY(JV)*TY(JV)
      P = RR(JV)*RR(JV) + S(JV)*S(JV)
      DIS = Q*Q + P*F
      L1=    TZ(JV).GE.XXK2(JV)
     +   .OR.TZ(JV).LE.XXK3(JV)
     +   .OR.Q.GE.0.0
     +   .OR.DIS.LE.0.0
      IF(L1) GO TO 451
      ETAUSD(JV)=(-Q-SQRT(DIS))/P
*VOCL STMT,IF( 0)
      IF ( F.GE.0.0 ) ETAUSD(JV) = 0.0
*VOCL STMT,IF(29) GO TO 451
      IF ( ETAUSD(JV).GT.1.0 ) GO TO 451
      TZ1(JV)=ETAUSD(JV)*T(JV)+TZ(JV)
*VOCL STMT,IF( 0) GO TO 451
      IF(TZ1(JV).GT.XXK2(JV).OR.TZ1(JV).LT.XXK3(JV)) GO TO 451
      TX(JV)=ETAUSD(JV)*RR(JV)+TX(JV)
      TY(JV)=ETAUSD(JV)* S(JV)+TY(JV)
      TZ(JV) = TZ1(JV)
      M(JV) = 1
  451 X(JV)=TX(JV)
      Y(JV)=TY(JV)
      Z(JV)=TZ(JV)
 9290 CONTINUE
```

Fig. 7.4 Particle crossing check calc. for cylindrical region

```
      DO 8080 IV=1,J20
      JV=CRO20(IV)
60    IF(XXK3(JV).LT.Y(JV).AND.XXK3(JV).LT.Y1(JV)) GO TO 8080
      IF(XXK3(JV).GT.Y(JV).AND.XXK3(JV).GT.Y1(JV)) GO TO 8080
      RHOY=(XXK3(JV)-Y(JV))/S(JV)
      XTEMP =RHOY*RR(JV)+X(JV)
      ZTEMP = RHOY*T(JV)+Z(JV)
      IF(XTEMP.GT.XXK1(JV).OR.XTEMP.LT.XXK2(JV).OR.ZTEMP.GT.XXK5(JV).OR.
     1ZTEMP.LT.XXK6(JV)) GO TO 8080
      IF(M(JV).EQ.1) GO TO 70
      M(JV)=1
      ETAUSD(JV)=RHOY
      KB(JV)=3
      GO TO 8080
70    IF(ETAUSD(JV).LE.RHOY) GO TO 8080
      ETAUSD(JV)=RHOY
      KB(JV)=3
8080  CONTINUE
```

Fig. 7.5 Particle crossing check calc. for cubic region

```
      DO 9400 IV=1,J380
      JV=CR0380(IV)
      Q =RR(JV)*X(JV) + S(JV)*Y(JV) + T(JV)*Z(JV)
      F=X(JV)*X(JV)+Y(JV)*Y(JV)+Z(JV)*Z(JV)-XXK2(JV)
      P=RR(JV)*RR(JV) +S(JV)*S(JV) + T(JV)*T(JV)
      DIS=Q*Q-P*F
      IF ( DIS.LE.0.0 ) GO TO 9400
      IF ( N.GT.0 ) GO TO 390
      IF ( Q.GE.0.0 ) GO TO 9400
      ETAUSD(JV)=(-Q-SQRT(DIS))/P
      IF(ETAUSD(JV).GE.1.0) GO TO 9400
      GO TO 400
390   ETAUSD(JV)=(-Q+SQRT(DIS))/P
      IF ( ETAUSD(JV).LE.1.0 ) GO TO 400
      ETAUSD(JV) = 1.0
      X(JV) = X1(JV)
      Y(JV) = Y1(JV)
      Z(JV) = Z1(JV)
      GO TO 9400
400   X(JV)=ETAUSD(JV)*RR(JV)+X(JV)
      Y(JV)=ETAUSD(JV)*S(JV)+Y(JV)
      Z(JV)=ETAUSD(JV)*T(JV)+Z(JV)
      M(JV) = 1
9400  CONTINUE
```

Fig. 7.6 Particle crossing check calc. for sphere region

8. おわりに

HASPの研究開発も3年を経過し，初年度，2年度に比べると本報告に見られる通り，幾らか形をなしてきた。この間に，研究所内外の多くの方々から貴重なコメント，助言を頂いたのが研究推進に役立っている。ハードウェアとしてのロボット設計に関する知見を得るためには，ソフトウェア・シミュレーションのみではデータが不足である。現在のハードウェアのレベルでのデータを収集するために，マイクロ・ロボットの製作を検討することとしている。本報告の執筆担当は，

第2章　神林　奨，上中淳二
第3章　藤井　実，大谷孝之
第4章　久米悦雄
第5章　樋口健二
第6章　樋口健二
第7章　浅井　清，樋口健二，秋元正幸
第1，8章　浅井　清

である。

謝　　辞

HASPの研究開発に関し，中部大学西原　宏教授，東京大学吉川弘之教授，九州大学松尾文顕教授から御批判，助言を頂いた。厚くお礼を申し上げます。

8. おわりに

HASPの研究開発も3年を経過し，初年度，2年度に比べると本報告に見られる通り，幾らか形をなしてきた。この間に，研究所内外の多くの方々から貴重なコメント，助言を頂いたのが研究推進に役立っている。ハードウェアとしてのロボット設計に関する知見を得るためには，ソフトウェア・シミュレーションのみではデータが不足である。現在のハードウェアのレベルでのデータを収集するために，マイクロ・ロボットの製作を検討することとしている。本報告の執筆担当は，

第2章　神林　奨，上中淳二
第3章　藤井　実，大谷孝之
第4章　久米悦雄
第5章　樋口健二
第6章　樋口健二
第7章　浅井　清，樋口健二，秋元正幸
第1，8章　浅井　清

である。

謝　　辞

HASPの研究開発に関し，中部大学西原　宏教授，東京大学吉川弘之教授，九州大学松尾文顕教授から御批判，助言を頂いた。厚くお礼を申し上げます。