JAERI-M
94-051

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４年度作業報告 －

１９９４年３月

秋元　正幸・樋口　健二・藤井　　実・久米　悦雄
神林　　奨・海老原健一・大谷　孝之

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　㈱原子力資料サービス

JAERI-M 94-051

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4年度作業報告 —

日本原子力研究所東海研究所情報システムセンター
秋元　正幸・樋口　健二・藤井　　実・久米　悦雄
神林　　奨・海老原健一・大谷　孝之

(1994年2月14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，命令理解システムの試作，視覚認識，2足歩行ロボットの動作シミュレーション，施設形状データベースの作成，被曝線量計算，さらに被曝線量計算等のモンテカルロ計算の高速化を目的とした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4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

JAERI-M 94-051

A Study on Intelligent Nuclear Systems (HASP : Human Acts Simulation Program) - Progress Report 1992 -

Masayuki AKIMOTO, Kenji HIGUCHI, Minoru FUJII
Etsuo KUME, Shaw KAMBAYASHI, Ken-ichi EBIHARA
and Takayuki OTANI

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

(Received February 14, 1994)

The sixth 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 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 knowledge-base, robot vision, robot kinematics/kinetics, plant geometry data-base, dose evaluation and high speed Monte Carlo machine.

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

目　　次

Contents

1. はじめに

昭和62年度に原子力委員会が改訂した「原子力開発長期計画」においては，原子力技術開発の共通基盤ともなり，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，原子力用レーザー，原子力用人工知能及び放射線リスク評価・低減化技術が挙げられ，これらの4分野について，昭和63年度から関連機関によって研究を開始した。原子力用人工知能については，日本原子力研究所（以下，原研）が既に開始していた「人間動作シミュレーション技術の研究」に，「極限環境センシング・システムの開発」を加え，新しい長期計画の下に研究開発を進めてきた。

本報告は，「人間動作シミュレーション技術の研究」に関する平成4年度作業報告である。また，この研究はHASP（Human Acts Simulation Program）と名付けている。

さて，平成4年度の成果報告を述べる前に，HASPの研究目的など研究概要，平成3年度までの研究概要，ついで平成4年度の成果概要，今後の計画を述べる。

1.1 HASPの研究概要

HASPの概念は，Fig. 1.1に示すように，自然文で書かれた命令を，コンピュータのなかで作り出された模擬人間，即ち，ソフト知能ロボットに与え，知能ロボットは命令を読み，意味を理解し，自己の行動を計画し，動作列を生成する（命令理解，静的経路探索，ロボット作業計画）。次いで，環境の状況を認識・判断しつつ動作列を精密化して目的の位置に，ロボット動力学／運動学に基づいて移動し（動的経路探索，視覚認識，ロボット動力学），点検作業の対象となる装置や機器が有する情報を収集しながら（センサー融合），それが正常かどうか判断するなど点検作業を遂行する（ロボットによる異常診断，ロボットによる修復作業）。この間ロボットが受ける放射線被曝線量計算を含め，これらすべての過程は論理・数値モデルによってコンピュータ・シミュレーションされる。

以上述べたように，HASPの研究目的は，人間理解や人間そのものの動作シミュレーションではない。人的作業の定性的・定量的評価，人的作業の機械化，複雑人工構造物と人間の役割分担の明確化である。具体的には，高放射線下で動作する知能ロボット設計のための要素技術を開発すること，ロボット知能化の程度との関連において，プラントの知能化・自動化技術を開発すること，人工知能関連のシステム化された基盤技術を原子力分野の研究者，技術者に提供することがHASPの目的とするところである。

1.2 平成3年度までの経緯

HASPは，Table 1.1に示すように，10年計画で研究を開始し，平成4年度で6年目になる。当初の2年間（昭和62～63年度）では，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ェアの整備，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ソフトウェアを整備した。

平成元年度から3年度には，研究概要で述べた各要素技術の研究開発を進めた。即ち，命令理解システムについては，日本語で書かれた作業指示文の構文意味解析を行い，結果をフレーム構造で表現し，これを基に，行動目標（ゴール）を生成するシステムの試作に着手し，仮定された人間の知識構造に対応して，プログラムの構造化を進め，プロトタイプ・システムの試作を完了させた。一方，知能ロボットが行う装置・機器等の異常診断及びその修復作業に定性推論応用の検討，R. Shank 等の概念依存関係を拡張し，知能ロボット動作，作業計画及びその評価への応用の検討，さらにロボット移動の経路探索へ真理管理システム（Truth Maintenance System）の応用の検討を行った。

また，視覚認識の研究については，ハードウェアとして既に開発済みの移動ロボットの環境認識技術の現状調査を行うとともに，マーク検出によるロボットの自己位置認識算定プログラムの開発，ニューラル・ネットワークの一種である Hopfield モデルによる両眼立体視対応問題の解法の検討等を行った。特に，両眼立体視の対応問題について，左右走査線上で見える順序が逆転する物体を含むシーンの対応付けが従来手法では解くことが出来なかったが，2段階動的計画法と命名した新しい手法を開発し，問題を解決した。

2足歩行ロボットの動力学的シミュレーションについては，ユーゴスラビアのヴコブラトビッチ（M. Vukobratovic）による人間形2足歩行ロボット・モデルを拡張・整備し，安定歩行を保持しつつ，ロボットの発進・定常歩行・停止の完全歩行をシミュレーションするソフトウェアを完成させた。特に，2足歩行ロボット設計の観点から，与えられた歩容に対して，安定化のために要求される補償動作の計算，逆に与えられた可能な補償動作に対して，安定性のために実現すべき歩容の計算が可能であり，種々の歩行安定性の解析を行った。また，これらのシミュレーション結果の動画化も可能にした。

施設形状データベースについては，知能ロボットの動作空間となる原子力施設として，原研の研究用原子炉 JRR－3 の一部を，施設の設計図面から CSG（Constructed Solid Geometry）手法によってデータベース化した。データベース化にあたっては，比較的放射線レベルが高く，しかも巡回・点検時に作業員が頻繁に立入る場所を主な対象とし，2足歩行知能ロボットの動作環境として，カラー動画化を行った。データベースは，記号処理に適した LISP 及び C 言語によるバージョンと高速数値処理用に適した FORTRAN 言語によってバージョンが準備されている。

被曝線量評価については，2足歩行ロボットに要求される耐放射性を評価することがその研究の目的であるが，現在そのモデルが確定していないため，米国オークリッジ研究所において開発された人体モデル：Cristy phantom を基に，モンテカルロ・コード（MCNP：Monte Carlo for Neutron and Photon）を用いて，ガンマ線の被曝線量計算を行った。この種の計算で問題になるのは，複雑な人体モデルの形状データの作成やその一貫性を確認するために，相当な作業量が必要になることである。これを軽減する観点から，人体モデル・データの可視化プログラムを作成した。このモデルにより，モンテカルロ計算コード MCNP を用いて，実効線量当量の計算を行い，国際放射線防護委員会（ICPR）の基準データと比較し良好な結果を得た。

超高速モンテカルロ装置については，昭和63年度に概念設計，平成元年度に詳細設計を完了させ，平成2年度から3年計画で単一プロセッサの製作を開始した。また，平成3年度にはプロセッサ多重化の製作に着手し，装置完成は平成5年度の予定である。本装置の設計基準として，代表的なモンテカルロ・コードKENO－Ⅳの演算速度が，単一プロセッサで，そのプロセッサのスカラ計算の2.5倍以上，プロセッサの多重化により，スカラ計算の10倍以上と設定している。上記以外の関連コード（MORSE，VIM，GRAPE）に対して，本装置で独自に開発した専用の演算パイプラインである幾何形状，事象分類及び領域判定パイプラインを含め本装置での達成予想演算速度を評価した。

以上が平成3年度までの研究開発の経過であるが，平成元年度から，HASP研究の一部は，科学技術庁原子力基盤技術開発クロスオーバー研究「原子力用人工知能を具備した原子力施設のシステム評価研究」としても位置付けられている。このクロスオーバ研究では，自律型プラントに関する要素技術開発が目標として設定され，参加機関は，原研のほか，動燃，理研，電総研，船研である。原研の担当研究テーマは，「シミュレーション・システムによる知能ロボットの自律型プラントへの適用性評価」であり，上記の開発済みソフトの性能評価をとおして，プラントへの適用性を明らかにしていくこととしている。

1.3 平成4年度の研究概要

詳細は，第2章以降に述べるとおりであるが，ここではその概要を述べる。尚，被曝線量評価については，超高速モンテカルロ装置の完成を持って本格的に研究を進めることになり，平成4年度には，特段の進捗が無かった。

(1) 命令理解システム

平成3年度までに開発完了したプロトタイプ・システムを実ロボット・ハードウェアに搭載した場合を想定し，必要な改良及び性能評価を実施した。即ち，ロボットとの対話上問題点であった自然言語処理の高速化，ロボット・ハードウェアを意識し，ロボットの動作を模擬するソフトウェア（ロボット模擬プロセス）の作成，既システムからグラフィカル・ユーザインターフェイス部分を切離し，ロボット模擬プロセスと併せて，推論エンジンとのプロセス間通信による分散処理システムへの改良，推論過程のデータ操作と関連アルゴリズムの改良による，命令の中止・中断・再開等の割込み命令機能の拡充を行った。自然言語処理の高速化については，処理時間の分析を行い，処理計算既機のグレードアップと処理手順の改良により，ひとつの命令文の平均処理時1.4秒から0.2秒と約7倍の高速化するなど実用化の見通しを得た。

(2) 視覚認識の研究

視覚センサーを用いて，ロボットの位置を認識する手法として，環境内の主要な場所にマークを添付し，それらの検出データとモデルとしてロボットに付与されたマークの正確な位置情報からロボットの位置を検出するシステムを既に開発したが，実用性の観点から処理速度が遅いのが欠点であった。高速かつ高精度に位置を検出するシステムの開発を進めた。また，環境異常等の実測データを収集するため，視覚，聴覚，嗅覚，放射線センサーから構成されるセンシング・システムの概念設計を行った。

(3) 人間形ロボットの2足歩行シミュレーション

現在の2足歩行ロボットの動作制御は，決められた発進位置から停止位置までの安定歩行パターン等を数値シミュレーションによって予め計算し，そのパターンに追従するように関節トルク等が制御される。従って，歩行中の予期しない外乱等に動的に対応することができない点が大きな問題として残っている。この問題を解決する第1ステップとして，計画歩行中の微小外乱に対応可能な制御法の検討を行った。即ち，歩行中の動的重心であるZMP（Zero Moment Point）の位置を計測し，その位置から，続く歩行ステップにおけるZMPを予測し，この予測値に対応する上体の補償動作を算出し，これに追従するように制御する方法の検討である。

(4) 模擬環境の製作

命令理解システムや環境認識用の知的ソフトウェアの性能評価には，具体的な環境において，2足歩行ロボット・ハードウェアに開発済みの知的ソフトウェアを搭載して検証することが不可欠である。しかし，2足歩行ロボットのハードウェアによる総合的な性能評価が容易でないことから，その第1ステップとして，原研研究用原子炉JRR－3一次区画の一部を模擬したロボットの動作環境のモックアップ施設を作成した。このモックアップ環境で実測環境データを上記のセンシング・システムで収集し，完全にデータベース化された施設の詳細情報と組合せて，環境異常の認識ソフト等の性能評価を今後行う。

(5) 施設形状データベース

施設形状データベースの一部には，上記の模擬環境の詳細情報も含まれる。このデータベースは，知能ロボットの動作環境を設定し，この設定環境情報とロボットがセンサーを介して収集する環境情報との対比により環境認識を行うとともに，全体のシミュレーション結果を可視化するために用いられる。平成4年度には，ロボット動作環境である世界モデルを簡略化して高速生成する機能，シミュレーション結果の実時間表示機能等の開発・整備を行った。

(6) 超高速モンテカルロ装置

昭和63年度に概念設計を開始し，次年度の詳細設計を経て製作していた本装置開発が，平成5年4月の装置検収に先立って，平成4年度に完了し，性能評価試験を実施した。強化したロード・ストア・パイプライン及びモンテカルロ・パイプラインの実効性能，さらに中性子輸送モンテカルロ・コードKENO－Ⅳの高速向上率を実測し，設計基準として設定したオリジナル・コードのスカラ計算速度の10倍以上を達成した。

1.4 今後の計画

平成5年度以降の研究計画はTable 1.1に示すとおりである。命令理解，行動計画の研究については，クロスオーバ研究の第Ⅰ期のまとめとして，命令理解システムに環境，ロボット動作等のソフトウェアを統合して，総合的評価研究を行う。環境設計・映像化技術の研究については，ロボットの学習機能の研究として，環境情報学習に最適な，環境・機器等の物体記述法の検討を進める。環境認識技術の研究では，センシング・システムのハードウェアの製作を完了させる。ロボット動作の研究については，2足歩行ロボットの制御法の研究のため，現在のシミュレーショ

ン・システムに多様な動特性解析機能を付与するなど，システムを高度化する。ロボットハードウェアの研究については，模擬環境を用いて，開発済み知的ソフトウェアの性能を検証するため，クローラ走行移動台車を導入・整備する。被曝線量評価の計算については，超高速モンテカルロ装置上，人間型ロボットのファントム・モデルを作成する。

Table 1.1 研 究 ス ケ ジ ュ ー ル

	平成5年度	平成6年度	平成7年度	平成8年度	平成9年度
クロスオーバー研究	原子力用人工知能を具備した原子力施設のｼｽﾃﾑ評価研究（第Ⅰ期）		自律型ﾌﾟﾗﾝﾄの分散協調知能化ｼｽﾃﾑの開発（クロスオーバ研究第Ⅱ期）		
(1) 原子力知識ベースの研究					
① 命令理解・行動計画の研究	開発済ｼｽﾃﾑの実用性評価	分散型知識ﾍﾞｰｽ のｱｲﾃﾞｨｱ 成立性の確認	分散型知識ﾍﾞｰｽ の設計・試作及び知識習得基本ｿﾌﾄ の開発		ｼｽﾃﾑ統合化・作業知識習得化技術の検証
② 環境設定・映像化技術の研究	環境情報学習に最適な物体記述法の検討		環境情報の学習に適したﾃﾞｰﾀﾍﾞｰｽ構造の検討及び改良		ｼｽﾃﾑ統合化・環境情報習得化技術の検証
③ 環境認識技術の研究	ﾊｰﾄﾞｳｪｱ部の開発	色・距離情報からの面情報抽出ｿﾌﾄ 開発		視・聴・嗅覚情報の融合処理による環境認識ｼｽﾃﾑの開発	ｼｽﾃﾑ統合化及び技術の適応性評価
(2) 高速シミュレーション技術の研究					
① ロボット動作の研究	ｼﾐｭﾚｰｼｮﾝ・ｼｽﾃﾑ の高度化		感覚運動学習に基づく制御法の開発、高度二足歩行ﾛﾎﾞｯﾄの機構・制御法の設計のまとめ、開発すべき技術の提言		
② ロボット・ハードウェアの研究	クローラ走行移動台車の整備、視覚情報に基づく走行制御法の検討		二足歩行ﾛﾎﾞｯﾄのﾌﾟﾛﾄﾀｲﾌﾟの基本設計	二足歩行ﾛﾎﾞｯﾄのﾌﾟﾛﾄﾀｲﾌﾟの詳細設計	二足歩行ﾛﾎﾞｯﾄのﾌﾟﾛﾄﾀｲﾌﾟの製作
③ 被曝線量計算の研究	動的ﾌｧﾝﾄﾑ による時間依存ﾓﾝﾃｶﾙﾛ計算手法の開発		実施設に即した放射線場設定手法の開発		ﾛﾎﾞｯﾄ軀体への放射線損傷度評価手法開発
④ 超高速モンテカルロ装置	各種原子力コードによる装置の性能評価				
(3) 原子力コードの知識モジュール化手法の研究	原子炉設計知的支援ｼｽﾃﾑの概念設計	原子炉設計における知識表現法の検討	計算ｺｰﾄﾞ 知識情報ﾓｼﾞｭｰﾙ の試作	知識ﾓｼﾞｭｰﾙ 作成支援ﾂｰﾙ の開発	

JAERI—M 94—051

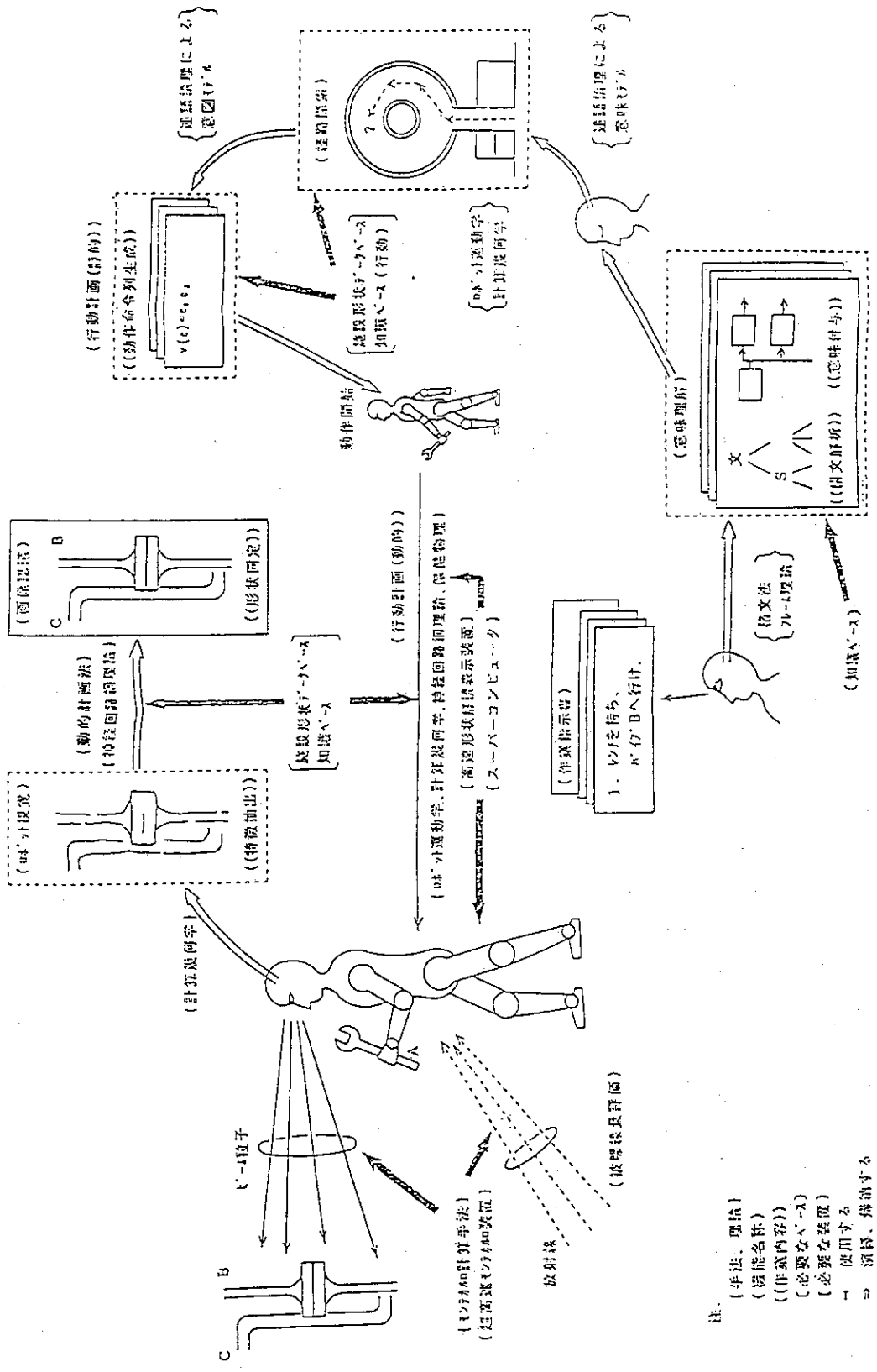


Fig. 1.1 人間動作シミュレーション HASP の概念図

2. 命令理解システム

2.1 はじめに

命令理解の研究では，(1) 日本語で記述された文（文字列）から，計算機で取り扱うことのできるデータ構造で表現されたロボットの行動目標に変換し，そして，(2) 得られた行動目標を，ロボットが実行する具体的な動作内容に展開し，さらに，(3) 具体化された動作内容や，実行時に得られる外界情報，割り込み命令等を取り入れてロボットの行動を制御する，という3つの機能を持つ命令理解システムの研究・開発に取り組んでいる。この研究は昭和62年度から着手され，平成3年度までに，自然言語理解および問題解決の各種手法の調査［1，2］，関連ソフトウェアの導入・整備を行い［3，4］，そして，Table 2.1に示す命令文から，原研の研究用原子炉JRR－3の1部を簡略化した仮想的な動作環境における知能ロボットの行動のシミュレーションを行う命令理解シミュレーション・システムのプロトタイプの試作を完了した［5，6］。

命令理解プロトタイプ・システムは，Kyoto Common Lisp（KCL）言語［7］を用い，SUN3／260Cワークステーション（Sun Microsystems社製）上で構築されたソフトウェアであり，以下の手法を用いて自然言語理解と問題解決を行っている：

- 日本語の意味表現方法として，高木・伊東の手法［8］による構文・意味的な係り受け構造を用い，文字列として与えられた日本語文の意味構造を導出するソフトウェアとして，㈱CSKで開発されたCS-PARSERを利用した［2，3］。
- 問題解決手法として，R.C. Schankらの提唱する概念依存性理論に基づくゴール・プランの理論［9，10］を用いた。
- 非同期的な命令文の入力，すなわち，割り込み命令に対応可能な推論制御構造として，認識－判断－実行サイクルを持つ推論エンジンを導入した［5，6］。

本プロトタイプ・システムは，日本語命令から移動等ロボットの行動を生成するという，命令理解システムの基本機能を実現したものの，ソフトウェア・シミュレーションの評価結果およびロボット運動学シミュレーション・ソフトウェア等との統合化の観点から，以下の問題点を残している：

- 命令文の解析時間が約18秒と非常に長い。
- 単一プロセス内のシミュレーション・ソフトウェアであるため，推論エンジンのサイクルと完全に同期した形式でロボットの動作がシミュレートされる。
- ロボット運動学シミュレーションや映像表示等のソフトウェアとの統合に不可欠な分散処理機能が未整備である。
- シミュレーション・システムをマルチプロセスで実現する場合，命令の中止・中断・再開に際して何らかの形態でデータ通信が必要となるが，このために必要な記憶管理機構や制御手法が未整備である。

そこで，平成4年度の命令理解の研究では，これらの問題点を解決するため，

(1) プロトタイプ・システムの動作環境（ハードウェアおよびソフトウェア）の高速化による日本語解析時間の短縮，
(2) 知能ロボットの動作を模擬するソフトウェア（ロボット模擬プロセス）の作成，
(3) 分散処理機能の整備の一環として，プロトタイプ・システムからグラフィカル・ユーザインタフェース部分を分離し，ロボット模擬プロセスとあわせて，推論エンジントのプロセス間通信による分散処理形式のソフトウェア体系の整備，
(4) 推論過程に関わる記憶の管理に用いる推論情報パケットと呼ばれるデータ構造と，それを操作する制御アルゴリズムの改良による，命令の中止・中断・開始といった割り込み命令への対応，

を行った。以下では，まず，2.2節で日本語解析時間の短縮についてまとめ，次いで，2.3節でロボット模擬ソフトウェアおよびグラフィカルユーザインタフェースの構成，そして推論エンジンとの通信機能について述べる。2.4節では，中止や中断といった実行中の動作に影響を与える割り込み命令に対する処理手法まとめ，2.5節では分散処理環境における推論エンジンの制御方法についてまとめる。

2.2 日本語解析の高速化

平成3年度版プロトタイプ・システムにおいて最も処理時間を費やす部分は，日本語解析システムCS-PARSERを起動する日本語解析サブプロセスである。例えば，命令文「PCS熱交換器付近の線量を測定して来い」の解析時間は，約18秒である。この処理時間の内訳は，以下の通りである:

- keitai，kaiseki等の日本語解析関数に要する処理時間。約1.7秒。
- ガーベジ・コレクション（Garbage Collection: GC）による処理停止時間。2回のGCにより約4秒。
- CS-PARSERをメモリ上にロードするためのページングの時間。約12秒。

このうち，CS-PARSERをメモリ上にロードするためのページングの時間は，次に述べるようなSUN3／260C上における命令理解システムの構築形態によって発生している: SUN3／260C上のKCLを用いてCS-PARSERと推論エンジンの双方を同一のLispイメージにロードした場合，GCが頻繁に起動されシステムのパフォーマンスが著しく低下する。このため，特に負荷の大きな処理を実行した場合，システムの処理時間の約半分がGCによって費やされることが実験によって確認されている。そこで，GCによるパフォーマンスの低下を防ぐために，平成3年度版プロトタイプ・システムでは，CS-PARSERと推論エンジンを別々のLispイメージとしておき，日本語解析の必要に応じて，推論エンジンからCS-PARSERを子プロセスとして呼び出す方式を採用した。CS-PARSERと推論エンジンは共にメモリ・イメージが大きいため，

(1) CS-PARSERを呼び出すたびにページ・インが発生し，
(2) 主メモリの不足が発生すると，推論エンジンのページ・アウトが発生し，
(3) さらに，日本語解析終了時に推論エンジンのページ・インが発生する。

この結果，ページングに伴うディスクI／O処理時間が非常に長くなり，日本語解析の処理時間

の増加が引き起こされているのである。

以上述べてきたように，プロトタイプ・システムにおいて，日本語解析に非常に長い処理時間が必要となる原因は，主としてGCとページングに起因しており，Lisp処理系のメモリ管理方式の効率化によって大幅に日本語解析の処理時間が短縮されるものと考えられる。そこで，Ephemeral GC（EGC）と呼ばれるGC方式を持つLucid Common Lisp（LCL）処理系[11]にプロトタイプ・システムを移植するとともに，SUN3／260Cと比較して約4倍の処理速度を持つS-4／IPワークステーション（富士通製）に処理系を移すことにより，日本語解析時間の短縮を図ることとした。EGCの最大の特徴は，あらゆるタイプのデータを世代毎に管理している点である。このため，関数内部の局所変数のようにガーベジになりやすい若いデータに対して集中的にGCを行うため，1回当たりのGCの所要時間が非常に短くなり，GCの後にはガーベジとならなかったデータ集中して存在するため参照の局所性が向上する。この結果，ページ・フォルトの発生の頻度を低く抑えることができる。処理速度短縮の目標としては，処理ハードウェアの性能比（SUN3／260C: 4 MIPS, S-4／IP: 17.4 MIPS）を参考に，日本語解析時間の平均を0.4秒と設定した。

プロトタイプ・システムのLCL，S-4／IPへの移植作業は，以下の手順で行った:

(1) 推論エンジンの中で関数systemを使って，CS-PARSER用シェルスクリプトを起動している部分を，単純な関数呼び出し形式に変更した。

(2) CS-PARSERのコーディングの効率をあげるために，構文解析規則のNeeLipトランスレータ（構文解析規則を記述するためのNeeLip言語からCommon Lisp言語への変換ツール)の出力結果から，無意味なlet文を削除するとともに，cond文の条件部にある(not (equal 〈FORM〉 nil)を単なる〈FORM〉に変更した。これは，c3004:／home／new-cs-parser／Kaiseki／cobj／neeoptディレクトリに格納されているLispプログラムによって実行することができる。

(3) si:string-concatenate等のKCL固有の関数を除去し，一般的なCommon Lisp言語の関数呼び出しに変更した。

(4) Common Lisp上のハッシュテーブルを用いる形式の単語辞書を採用した。

Table 2.2に，平成3年度版プロトタイプ・システムとLCL，S-4／IPへ移植したプロトタイプ・システムによる日本語解析時間の計測結果と速度向上率を示す。Table 2.2中において，SUN3／260C（KCL）による計測結果には，GCとページングの処理時間は含まれていない。したがって，Table 2.2に示した平均6.96倍という速度向上率は，日本語解析関数の処理時間短縮の度合いを示しており，これは使用した計算機の性能比（17.4 MIPS／4 MIPS＝4.35）を上回る値である。また，移植後のプロトタイプ・システムでは，システム起動直後にページングが発生するものの，シミュレーションの実行中にはページングが発生せず，ページングやGCによる処理速度の低下はほとんど見られない。実際，1回のEGCの所要時間が約0.1秒時間であり，これはEGCの処理頻度から換算すると1文あたり高々0.02秒と極めて短いものである。GCやページングの影響を考慮した実効速度向上率は，

$$\frac{1.358\,(\mathrm{sec}) + 4\,(\mathrm{sec}) + 12\,(\mathrm{sec})}{0.195\ (\mathrm{sec})} = 89\text{倍}$$

となり，実に100倍近い日本語処理時間の短縮が達成できたことになる。

2.3 分散処理に基づくロボット模擬プロセスとグラフィカル・ユーザ・インタフェース

平成3年度版プロトタイプ・システムは，日本語命令から移動等ロボットの行動を生成するという，命令理解システムの基本機能の検証を目指したソフトウェアであり，ロボットの行動に関しては単一プロセス内において簡単なLisp関数によって模擬されている。今後，HASPにおける視覚認識，ロボット運動学シミュレーション，映像生成等のソフトウェアの整備にともなって，ロボットの行動や映像化等をそれぞれ別々のソフトウェアによって系統的に実行するためには，命令理解システムと他のソフトウェアとのシステム化を行う処理形態の整備が不可欠となる。それぞれのソフトウェアは単体でも相当な規模のアプリケーションであるので，統合システムの構築には，ローカル・エリア・ネットワーク上の分散コンピューティング環境の整備が基本と考えられる。そこで，HASPにおける分散処理技術の整備のひな型として，Table 2.3に示すように，(1) ロボットの運動を模擬する部分（ロボット模擬プロセス）と(2) ウィンドウ・インタフェースの部分を独立なプロセスとし，プロセス間通信を用いて推論エンジンとデータ通信を行うクライアント・サーバ型のソフトウェア形態に修正することとした。これにともない，Sun Viewウィンドウ・システムのttyウィンドウによって構成されていたウィンドウ・インタフェースを，クライアント・サーバ・アーキテクチャのXウィンドウ・システムに変更することとした。

従来のプロトタイプ・システムでは，Fig. 2.1に示した命令理解システムの処理概念において，破線で囲まれた部分の視覚処理およびロボット運動学シミュレーションに関する処理をCommon Lisp言語で記述された簡単なプログラムによって模擬していた。このプログラムは，推論エンジンを処理しているLispエバリュエータによって実行され，さらに，行動計画の結果としてロボットの基本動作が［50 cmの移動］や［x°の回転動作］のように細かく分割された形式で与えられるため，推論エンジンとロボットの運動とが完全に同期をとった形でシミュレーションが実行される。しかしながら，単一のLispエバリュエータによる処理では，ロボットの基本動作が終了するまで推論エンジンがブロックされるため，例えば，［数メートルの直進］といった基本動作の形式を想定すると，この移動の間いっさいの割り込み命令が受け付けられなくなる。したがって，Fig. 2.1に示したようにロボットの視覚処理や運動学シミュレーションに関する処理を推論エンジンとは別のプロセスとし，推論エンジンの実行部分が単にそれぞれのプロセスに対してメッセージを送信して，直ちに認識部分に処理を移すという処理形式を実現することが，計算資源の効率的な活用のみならず，ロボットの基本動作の形式にとらわれずに推論エンジンを機能させるために重要である。

複数のプロセスによる分散処理システムでは，プロセス間のデータ通信におけるデータフォーマットおよびデータ通信形態，そして，各プロセスにおけるデータ送受信の制御方式を的確に設定しなければならない。推論エンジンとロボット模擬プロセスの分散処理では，(1) ロボットの

基本動作指令，(2) 基本動作の処理状況および処理結果，そして，(3) データ通信の健全性の確認のために必要なデータ受信状況の 3 種類のデータをソケットによるプロセス間通信によって交換する方法をとった（Table 2.4 参照）。ロボットの基本動作指令に関しては，ロボットの目標位置座標および目標位置におけるロボットの方向と移動形態を指定する形式とし，基本動作の処理状況・処理結果およびデータの受信状況に関しては，Table 2.5 に示すように通信データ中のコマンド識別子および処理フラグによってお互いのプロセスに通知する形式とした。なお，Table 2.5 では，ロボット運動学シミュレーションの処理が，(1) 基本動作指令を展開する処理と (2) 展開結果の数値シミュレーションという 2 つの処理から構成されているものと仮定した。また，推論エンジンでは，以下に示す 6 種類のフラグ（［真偽］）によって問題解決等の処理を制御する形式を採用した:

(1) 送信: ロボット模擬プロセスに基本動作指令を［送信済，未送信］
(2) 受信: ロボット模擬プロセスから基本動作指令の受信済信号を［受信済，未受信］
(3) 開始: ロボット模擬プロセスに動作開始信号を［送信済み，未送信］
(4) 終了: ロボット模擬プロセスから動作終了信号を［受信済み，未受信］
(5) 失敗: ロボット模擬プロセスから受信した動作終了結果が［検証失敗，未検証］
(6) 成功: ロボット模擬プロセスから受信した動作終了結果が［成功，未成功］

これにともない，推論エンジンを駆動する推論情報パケットにおける基本動作指令の表現形式について，上記フラグに対応するフィールドを追加し，これを推論エンジンの判断部分において処理選択の基準に用いる形式とした:

（基本動作指令: key__1 arg__1 ..... : key__n arg__n
送信フラグ　受信フラグ
開始フラグ　終了フラグ
失敗フラグ　成功フラグ
開始時刻）

推論エンジンの処理選択は，Table 2.6 に示す推論情報パケットを使った判断規則および Table 2.7 に示す推論情報パケット中の基本動作指令のフラグ・フィールドによる判断規則を用いて行われる。Fig. 2.2 に推論エンジンとロボット模擬プロセスの分散処理におけるデータフローを示す。

GUI は，XView ToolKit［12］を主体としたCプログラムであり，推論エンジンを実行するホストと同一のホスト上のプロセスとして実行される。また，GUI と推論エンジンの間のデータ通信は，ソケットの一種であるパイプを用いて行われる（Table 2.8 参照）。このパイプは，(1) まず，GUI を起動する際に準備され，(2) システムコール fork を使って子プロセスを起動する際に必要なパイプのみを残し，(3) さらに，Cライブラリ関数 execvp を使って推論エンジンを起動することによって Table 2.8 の形式で形成される。GUI に対して入力された日本語命令文と Lisp コマンドは，XView ToolKit に用意されているイベント処理機能を用いて検出し，推論エンジン側に輸送される。また，推論エンジンから転送される信号に関しては，XView ToolKit のノーティファイア機能を用いて検出し，ウィンドウ上の表示内容の変更等を行う。

2.4 作業の中止・中断・再開に関する割り込み処理

点検作業では，作業環境の動的な変化や点検によって発見された異常事象に対して柔軟な処置を施すことが非常に重要となる。環境の変化や異常の発見は，点検者に対する一種の割り込み情報と考えられる。また，環境の変化への対応や異常に対する処置は，割り込み情報によって生じた新たな行動目標に関する問題解決と見なすことができる。したがって，環境変化への対処や異常事象に対する処置を実行するには，割り込み情報が発生する前に行っていた作業を中止・中断したり，中断した作業を再開することが基本的な処理と考えられる。そこで本節では，処理中の作業の中止・中断および中断された作業の再開に関する割り込み命令を推論情報パケットと推論エンジンの判断規則によって処理する方法について述べる。

中止・中断・再開に分類される割り込み命令を処理する際に問題となるのが，(1) 実行中の作業あるいは動作の特定方法と (2) 中止・中断・再開といった命令を日本語によって記述する方法である。例えば，「Aポンプを検査して，Bポンプを検査しろ」という命令をロボットに与えた後に，「作業を中断しろ」という割り込み命令を与えたとする。この時，「作業」によって示されるものが曖昧であるため，「作業」が「Aポンプを検査して，Bポンプを検査すること」なのか，「Aポンプを検査すること」あるいは「Bポンプを検査すること」なのかを命令者の意図や文脈情報なしに決定することは不可能である。さらに，命令理解プロトタイプ・システムでは，入力された命令文から自然言語解析と問題解決によって，ゴール，Δ-ゴール，基本動作列という階層的な行動の構造が得られるため，割り込み命令によって指示される動作として，以下の可能性が考えられる:

1) 処理中の１つの命令文が支配するすべてのゴール
2) 処理中の１つのゴールが支配するすべてのΔ-ゴール
3) 処理中の１つのΔ-ゴールが支配するすべての基本動作
4) 処理中の１つの基本動作
5) 処理予定の命令文（指定可能な任意個）が支配するすべてのゴール
6) 処理予定の命令文（指定可能な任意個）が支配するすべてのΔ-ゴール
7) 処理予定のΔ-ゴール（指定可能な任意個）が支配するすべての基本動作
8) 処理予定の基本動作（指定可能な任意個）

ここでは，文脈や命令者の意図による曖昧さの少ない方法として，(1) 割り込み命令によって指示される動作として指定可能な任意個のゴールを考えることとし，(2) 割り込み命令文の記述方法と解釈方法に制限を与えることによって，割り込み命令が指示するゴールを特定する方法を採用する。

中止・中断・再開を表す日本語命令文は，以下の意味構造を持つものとする:

(a) 対象格のある中止命令:

（（動詞:［中止］に属する意味素を持つ動詞）
（対象格:（名詞:［行為］に属する意味素を持つ名詞
任意の修飾要素））））

(b) 対象格のない中止命令:

（（動詞:［中止］に属する意味素を持つ動詞））

(c) 対象格のある中断命令:

（（動詞:［中断］に属する意味素を持つ動詞）
（対象格:（名詞:［行為］に属する意味素を持つ名詞
任意の修飾要素））））

(d) 対象格のない中断命令:

（（動詞:［中断］に属する意味素を持つ動詞））

(e) 対象格のある再開命令:

（（動詞:［再開］に属する意味素を持つ動詞）
（対象格:（名詞:［行為］に属する意味素を持つ名詞
任意の修飾要素））））

(f) 対象格のない再開命令:

（（動詞:［再開］に属する意味素を持つ動詞））

ここで，「中断」は，中断された動作が再開されることを前提としていることに注意する。Table 2.9 に，上記割り込み命令が入力された場合の処理をまとめる。推論情報パケットには，そのパケットに対する処理を行うか否かに関する制御情報として推論実行条件が用意されている。この推論実行条件は，

（AND［制御フラグ］［実行条件］）

なる論理表現で記述される。ここで，［制御フラグ］は中断・再開にともなう推論情報パケットの処理のために使用するものであり，［制御フラグ］＝［偽］が中断に対応し，［制御フラグ］＝［真］が再開（中断の割り込みを受けていない状態）に対応する。また，［実行条件］は，条件付きで与えられた命令文に含まれる実行条件を表すものである。

2.5 分散処理環境における推論エンジンの制御方法

平成３年度版プロトタイプ・システムでは，推論情報パケットと名付けた日本語解析や問題解決に関する情報を保存するデータ構造と Table 2.6 に示した推論エンジンの判断規則に基づいて，複数の命令文や複数のゴールの処理を行う。この際に，Table 2.6 に示したように，推論エンジンの実行部分において処理の対象となる推論情報パケットは１つであるので，複数のゴールを処理するといっても，あくまで逐次的な処理であった。しかし，2.3 節で示したロボット模擬プロセスと推論エンジンの分離のように，分散処理環境においては複数のプロセスを同時に実行することが可能なため，命令理解システムにおいても，例えば自然言語処理と問題解決を同時に処理するといった並列的な推論制御方式を整備する必要がある。

複数の処理の並列処理を行う場合，処理対象相互に干渉のないことをあらかじめ確認しておくことが重要である。しかしながら，たとえ独立に制御可能な処理単位でも，あるゴールのもとでそれらの処理単位が依存関係にある場合が想定されるため，干渉のない処理単位の選定は極めて困難である。例えば，「ロボットが移動中にテーブル上の物体をつかむ」という事象を考えると，

移動装置と物体をつかむためのマニピュレータは，互いに独立な運動装置であるが，「テーブル上の物体をつかむ」ためには，２つの運動装置を協調的に制御しなければならない。平成４年度版のプロトタイプ・システムでは，Table 2.10 に示したように処理単位の分類として思考に関するもの（思考系処理）と動作に関するもの（動作系処理）の２種類を考え，２種類の処理単位の組み合せ（３つ）のそれぞれについて，Table 2.11 に示した並列処理に関する判断基準をあらかじめ設定しておく方法を採用した。

具体的な処理方法は次のようにまとめられる：(1) 推論エンジンの判断部分の処理結果である実行命令として複数のサブプロセスおよび基本動作列を含め，(2) 推論エンジンの実行部分においてTable 2.11 の基準にしたがって並列処理を行う。ここで，並列処理が不可能な場合は，従来通り，最も新しく入力された命令を優先して処理を行う。

このような処理を実装することによって，例えば移動中のロボットに他の命令を与えた場合，純粋に逐次的な処理であれば移動動作を一旦停止してから命令の解釈を開始するところであるが，そこで一旦停止せずに移動しながら命令の解釈や今後の行動の計画等を行うことが可能になった。また，システムには未実装であるが，現在実行中の動作の部分的な修正（例えば移動速度の加減）を行うことも原理的に可能になった。こういった機能は，定型的な作業だけしか求められないロボットには不必要なものであるとも考えられるが，人間や他の知能化システムとの協調作業を想像すれば，これらの機能は必要不可欠なものであると言える。

2.6 おわりに

この章では，今年度の命令理解の研究について述べた。前年度までの命令理解の研究では，純然たるソフトウェア上のシミュレーションをシステム開発上の目標としてきたが，今年度の研究成果によって，いよいよハードウェアの実駆動をも可能なシステムに目標を移す基盤が整いつつあるのではないかと考えている。その半面，模擬世界と現実世界とのギャップが残されているのも事実である。今後はこのギャップを埋めることに注力し，またハードウェアとの連係運用を考慮してシステムの操作性の向上にも努め，高度な人工知能技術の具現であると同時に有用性・可用性を兼ね備えたシステムを目指す所存である。

参 考 文 献

[1] 上中淳二，神林 奨：**核データ評価コードに関する知識構造の調査**，JAERI－M 88－143，日本原子力研究所（1988.8）。

[2] 神林 奨，上中淳二：**日本語を受け付ける知識ベースシステムの試作**，JAERI－M 89－218，日本原子力研究所（1989.12）。

[3] 浅井 清他：**原子力知能システム技術の研究－昭和63年度作業報告－**，JAERI－M 89－023，日本原子力研究所（1989.3）。

[4] 浅井 清他：**原子力知能システム技術の研究－平成元年度作業報告－**，JAERI－M 90－060，日本原子力研究所（1990.3）。

[5] 浅井 清他: **原子力知能システム技術の研究－平成2年度作業報告－**, JAERI－M 91－101, 日本原子力研究所 (1991.6)。

[6] S. Kambayashi and Y. Abe, *Instruction Understanding for Intelligent Robots in Nuclear Facilities*, in Proc. of Join International Conference on Mathematical Methods and Supercomputing in Nuclear Applications (Karlsruhe, Germany, 1993).

[7] T. Yuasa and M. Hagiya, *Kyoto Common Lisp Report* (TEIKOKU INSATSU, 1984).

[8] 高木 朗, 伊東幸宏: **自然言語の処理** (丸善, 1987)。

[9] R. C. Schank and C. K. Riesbeck, 石崎 俊 監訳: **自然言語処理入門** (総研出版, 1987)。

[10] R. C. Schank and R. P. Abelson, *Scripts, Plans, Goals, and Understanding* (Lawrence Erlbaum Associates, N.J., 1977).

[11] *Lucid Common Lisp/Sun Version 4.0: User's Guide* (Luicid Inc., 1991).

[12] D. Heller, 石川和也 監督: *XView ver.3* **プログラミング・マニュアル** (ソフトバンク, 1992)。

Table 2.1 Sample instructions for the prototype of the instruction understanding system.

No.	例文
1	一次区画に行け.
2	一次区画に着いたら連絡しろ.
3	ＰＣＳ主ポンプの圧力をチェックして来い.
4	ＰＣＳ熱交換器付近の線量を測定して来い.
5	一次区画の線量測定をしてくだされ.
6	ＰＣＳ主ポンプの圧力をチェックしろ.
7	一次系の配管のリークを調べろ.
8	ＲＯＳタンクのレベルを調べて来い.
9	ＰＣＳ熱交換器のメータを調べて来い.
10	重水区画の異常を調べて来い.
11	一次区画を点検して来い.

Table 2.2 Computational time required for syntactic and semantic analyses for the example sentences in Table 2.1. Note that the processing time for paging is not included and each data represents the average of ten different measurements.

No.	real time (sec)		speedup
	(A) Sun3/260C (KCL)	(B) S-4/IP (LCL)	(A/B)
1	0.717	0.107	6.70
2	1.217	0.188	6.47
3	1.500	0.208	7.21
4	1.633	0.237	6.89
5	1.900	0.293	6.48
6	1.200	0.173	6.94
7	1.850	0.258	7.17
8	1.367	0.187	7.31
9	1.383	0.183	7.56
10	1.383	0.188	7.36
11	0.783	0.122	6.42
average	1.358	0.195	6.96

Table 2.3 Functions, programming language, and libraries of three processes in the prototype of the instruction understanding system.

プロセス	機能	記述言語，使用ライブラリ
GUI	Lisp コマンド・リスナー，命令文の入力，各種表示	C, Xlib, XView ToolKit
推論エンジン	問題解決，日本語解析	LCL, C
ロボット模擬プロセス	ロボット動作のシミュレーション	C, Xlib, XView ToolKit

Table 2.4 Specification of the inter-process communication between the inference engine (IE) and action simulator (AS) in the prototype of the instruction understanding system.

入出力方向	用途	データフォーマット
IE→AS	AS に対する基本動作指令，動作開始信号，中止信号を送る．	$((r, \theta), \Delta(r, \theta), com)$. r は目標位置座標，$\theta$ は目標における方向，$\Delta()$ は引数に関する許容誤差，そして，com はコマンド識別子である．
AS→IE	現在位置等の情報，コマンドの処理状況を IE に送る．	$(r, \theta, flag, id)$. r はロボットの現在位置あるいは目標位置，θ は現在の方向，flag は処理フラグ，そして，id はコマンド識別子である．

Table 2.5 Specification of the processing flag for the inference engine (IE) and action simulator (AS) in the prototype of the instruction understanding system. Command means the action-pattern represented in interger less than 10000.

入出力方向	処理フラグ	意味
IE→AS	com=command	基本動作指令
IE→AS	com=command+10000	動作開始信号
IE→AS	com=command+20000	中止信号
AS→IE	flag=1	基本動作指令受信信号
AS→IE	flag=2	動作展開成功信号
AS→IE	flag=3	基本動作終了信号
AS→IE	flag=-1	動作展開失敗信号

Table 2.6 Decision rule for the inference engine in the prototype of the instruction understanding system. (II: input instruction, DS: dependency structure, G: goal, DG: Δ-goals, AIP: active inference packet, IP: inference packet, NLP: natural language processing, GC: goal generation, TP: task planning, PP: path planning, SA: send action command, SIP: stack inference packet, PIP: pop inference packet.)

process	II	DS	G	DG	AIP	IP
NLP	o	—	—	—	o	*
GG	•	o	—	—	o	*
TP,PP	•	•	o	*	o	*
SAC	•	•	o	o	o	*
SIP	Int.	*	*	*	*	*
PIP	—	—	—	—	—	*

o: unprocessed •: processed —: empty

*: unconcern Int.: interruption

Table 2.7 Decision rule for the distributed processing of the inference engine and action simulator. (TF: transmission flag, RF: reception flag, BF: begin flag, EF: end flag, FF: fail flag, SF: success flag, FAS: flag from the action simulator)

TF	RF	BF	EF	FF	SF	FAS	operation
o	o	o	o	*	o	*	generate action command
•	o	o	o	*	o	-1	error
•	o	o	o	*	o	1	RF→ •
•	o	o	o	*	o	2	RF→ •, BF→ •
•	•	o	o	*	o	-1	error
•	•	o	o	*	o	2	BF→ •
•	•	•	o	*	o	-1	error
•	•	•	o	*	o	3	EF→ •
•	•	•	•	o	o	-1	error
•	•	•	•	o	o	≠ 1	if (!verify-result) FF→ •
•	•	•	•	o	o	≠ 1	if (verify-result) SF→ •
•	•	•	•	•	o	-1	error
•	•	•	•	•	o	≠ 1	if (verify-result) SF→ •
•	•	•	•	*	•	*	save result, shift pointer

•: true o: false *: true or false

Table 2.8 Specification of the inter-process communication between the inference engine (IE) and graphical user interface (GUI) in the prototype of the instruction understanding system.

パイプ変数名	入出力方向	用途	データフォーマット
pipe_io[0]	GUI→IE	Lisp リスナーにキーボード入力された内容を IE 側の Lisp リスナー入力ストリームに送る.	任意の文字列
pipe_io[1]	IE→GUI	Lisp リスナー出力ストリームにおける表示内容を GUI 側 Lisp 実効ウィンドウに送る.	任意の文字列
pipe_io[2]	GUI→IE	命令文入力ウィンドウにキーボード入力された内容を IE 側の命令文入力ストリームに送る.	任意の文字列
pipe_io[3]	IE→GUI	応答表示出力ストリームにおける表示内容を GUI 側のロボット応答ウィンドウに送る.	任意の文字列
pipe_io[4]	IE→GUI	全体地図表示出力ストリームにおけるエリア名とビットマップファイル名を GUI 側の全体地図表示ウィンドウに送る.	sscanf(buf, "%s␣%s", arg1, arg2) で読み込める形式. arg1 はエリア名, arg2 はビットマップファイル名を表す.
pipe_io[5]	IE→GUI	部分地図表示出力ストリームにおける表示位置と表示内容を GUI 側の部分地図表示ウィンドウに送る.	fscanf(stream, "%d␣ %d␣ %n", &x, &y, &n) と fgets(buf, BUFSIZ, steram) を連続した呼び出しで読み込める形式. x と y は離散化地図のます目単位の座標成分, n は fscanf で読みとったデータのバイト数, そして, buf が表示内容を表す.
pipe_io[6]	IE→GUI	推論過程表示出力ストリームにおける表示内容を GUI 側の推論過程モニタに送る.	任意の文字列

Table 2.9 Processing rule for interruptive instructions for stopping, suspending, and resuming tasks.

割り込み命令	対象格の有無	処理
中止命令	有	対象格に含まれる [行為] に属する意味素を持つすべてのゴールについて，そのゴールを含む推論情報パケットを削除する
中止命令	無	すべての推論情報パケットを削除する
中断命令	有／無	すべての推論情報パケットの制御フラグを [偽] とする
再開命令	有／無	すべての推論情報パケットの制御フラグを [真] とする

Table 2.10 Categorization of sub-processes and elementary actions defined in the instruction understanding system into mental and physical processes.

分類	内容
思考系処理	環境状態メモリから入力命令文文字列を削除 推論情報パケット生成 日本語解析サブプロセス呼出 行動目標決定サブプロセス呼出 ゴールの Δ-ゴール列への展開 Δ-ゴール処理履歴の初期化 基本動作列生成サブプロセス呼出 DO-METHOD マクロ展開
動作系処理	基本動作 check 基本動作 close 基本動作 grasp 基本動作 open 基本動作 operate 基本動作 ptrans 基本動作 search 基本動作 speak 基本動作評価結果の保存

Table 2.11 Rule for parallel processing in the inference engine of the instruction understanding system.

組み合わせ	並列処理の可否
思考系処理 ＋ 思考系処理	同一のものでなければ可能
思考系処理 ＋ 動作系処理	可能
動作系処理 ＋ 動作系処理	不可能

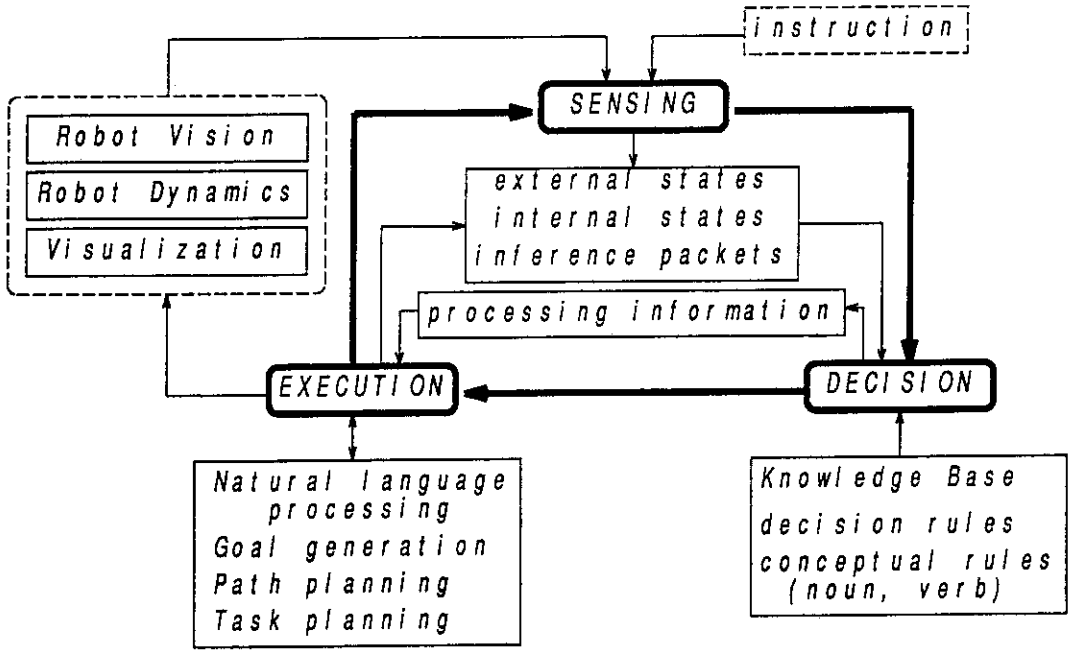


Fig. 2.1 System configuration of the instruction understanding system. Thick arrows denote flow of processing and thin arrow denote data flow.

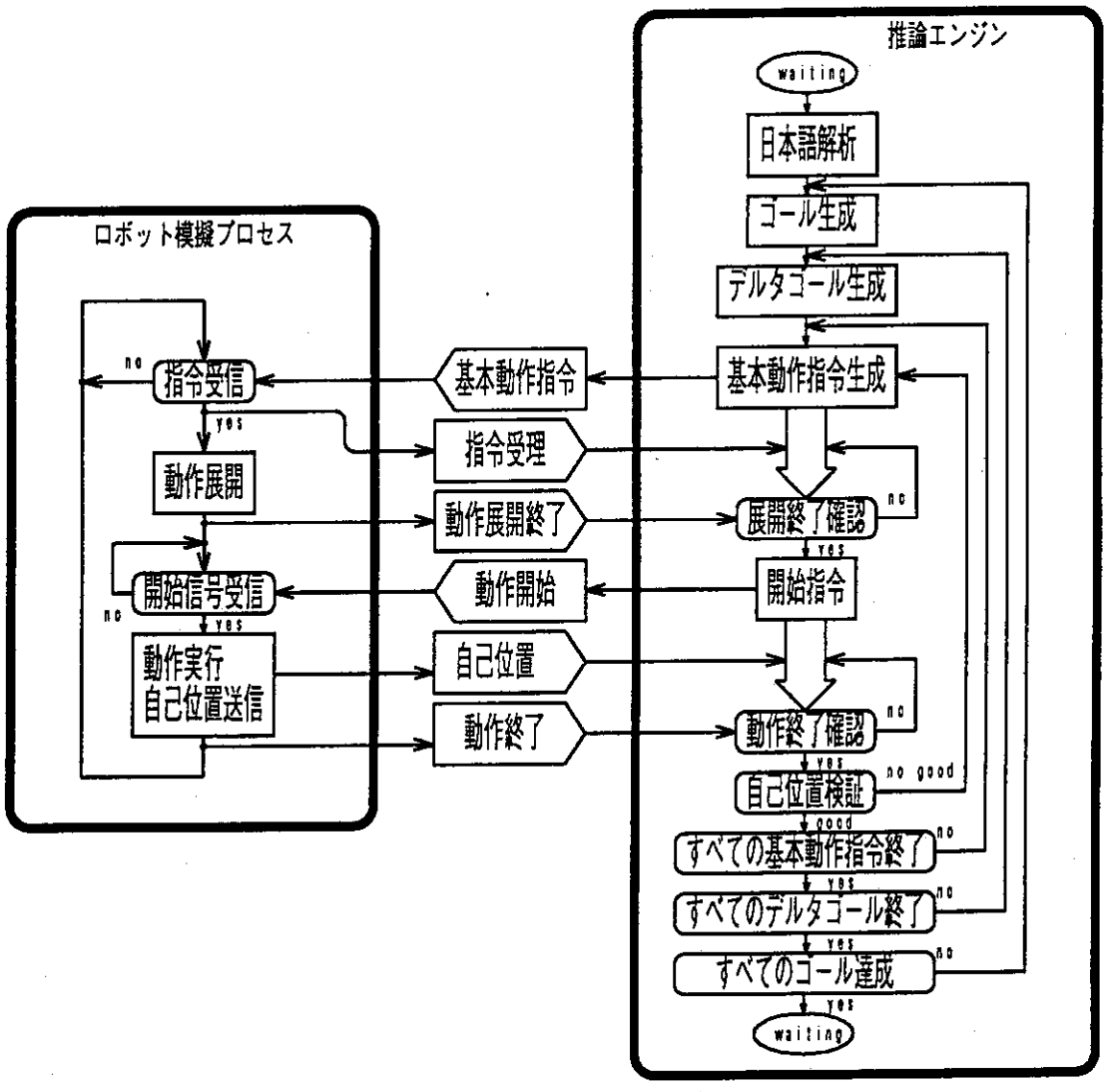


Fig. 2.2 Data flow of the instruction understanding system under distributed processing of the inference engine and action simulator.

3. 視覚認識の研究

3.1 はじめに

人間動作シミュレーション技術の研究（HASP）では，原子力施設で働く自律型知能ロボットを研究対象とした，知能ロボット設計に必要な基盤的要素技術の開発を行ってきている[1]。視覚認識の研究では，これまで，(a)既知環境下における視覚認識方法（Fig. 3.1）としてマークを利用した自己位置認識方法について技術開発[2]を，(b)半既知環境下における視覚認識方法（Fig. 3.2）として両眼立体視による空間認識方法について技術開発を行ってきた。前者は実用を狙った技術開発，後者は人間と共同作業する将来ロボットに必要な技術の研究開発として位置付けている。これまでの研究経過は，前年度報告書[3]に詳しく記述されているので，ここでは，今年度の研究概要を示す。

HASPでは，今年度から，ロボット・ハードウェアに実際に搭載する知的ソフトウェアの整備を始めた。具体的には，視覚認識研究においては，(1)ロボットの自己位置認識システムの設計，製作及び(2)環境異常等を検出するセンシングシステムの設計，製作である。(1)の自己位置認識システムに関しては，既存の白黒カメラを使用し，環境に設置したマークを手掛かりに自己位置を高速かつ高精度に認識するシステムの設計と自己位置算出ソフトウェアの作成を行った。これを3.2節に記述する。(2)のセンシングシステムに関しては，機能，構成機器等について概念設計を行った。これを3.3節に記述する。

3.2 ロボットの自己位置認識システム

3.2.1 はじめに

ロボットに搭載する自己位置認識システムの設計，製作を今年度から次の計画で開始した。

［第1段階］：既存のロボットに搭載でき，JRR－3模擬環境における巡回移動実験に使用できるシステムの開発。具体的には，既存の白黒カメラを使用し，環境に設置したマークを手掛かりに自己位置を高速かつ高精度に認識するシステムを設計，製作する。

［第2段階］：環境に存在する機器等の特徴点を手掛かりに自己位置を認識するシステムの開発（環境に手を加えない）。

ここで，記述する自己位置認識システムは，第1段階のもので，使用する白黒カメラ等から以下の制約条件を設ける。

［ハードウェアの制約条件］

① 小型白黒カメラ（縦480＊640画素）を使用する。

② カメラ制御は，上下方向が正面水平方向から真下まで90°，左右方向が正面から左右それぞれ約24°の範囲で可能とする。

③ ロボットのデッドレコニングによる推定自己位置の精度は，自己位置認識要求が出される場所で前後左右それぞれ20 cm 以内，進行方向は10° 以内とする。

また，自己位置認識システムに要求される要件として以下のものがある。

［ソフトウェアの要求要件］

① 原子力施設内で実際に使用できること。

② カメラ制御等のハードウェア関連部分を除く，自己位置算出ソフトウェアの処理をできるだけ高速（１秒以内を目標）にすること。

③ 自己位置認識の精度は，段階付近等ではできるだけ高精度（１cm，1° 以内を目標）にすること。

④ 任意の位置で自己位置認識できること。

HASP では，これまで，壁張りマークを使用した自己位置算出方法を既に開発し，計算機シミュレーションでその性能評価を行っている[1),2)]。しかし，この方法は，カラー望遠カメラを前提とした方法で，計算時間もかかるため，上記のハードウェア制約条件及びソフトウェア要求要件を満足しない。そこで，ここではその性能評価結果を実用システムの設計，製作に利用する。また，搭載する小型白黒カメラを使用した自己位置認識ソフトウェアがメーカで既に試作れているが，環境に存在する機器等の特徴点を手掛かりに自己位置を認識する方法を採用しているため，計算時間がかかる，照明条件によって特徴点が検出できないケースが出るなどの問題点があり，まだ前述のソフトウェアの要求要件①，②を満足できない状況である。この機器等の特徴点を手掛かりに自己位置を認識する方法については，実用化がすぐには困難であるので，第２段階で研究開発を進める計画である。

以下に今年度の設計，製作内容について記述する。

3.2.2 ロボットの自己位置認識システムの概要

ロボットに搭載する小型白黒（CCD）カメラをコントロールして，床面に貼った白黒マークを検出し，その見え方からロボットの自己位置を認識するシステムについて全体設計を行った。設計したロボットの自己位置認識システムの概要を Fig. 3.3 に示す。自己位置認識システムは，ロボット制御システム，命令理解システム，自己位置算出システムから構成される。このシステムでは，ロボットは，立ち止まった場所で毎回マークを利用した自己位置算出を行う。その手順を以下に示す。

① ロボット制御システムは，静止するとデッドレコニングによる推定自己位置情報（部屋座標系でのX，Y，Z座標と進行方向 TH）を命令理解システムに送信する。

② 命令理解システムでは，受信した推定自己位置情報を基に，その位置からカメラの制御範囲に入るマークで最も近いものを，カメラ設置情報を検索して選択する。

③ 命令理解システムは，選択したマーク情報とカメラ制御角情報をロボット制御システムに送信する。

④ ロボット制御システムは，受信したカメラ制御角情報を基にカメラをマーク中心が存在すると推定される方向に向け，その方向での画像情報を得る。

⑤ ロボット制御システムは，マーク情報と画像情報に加え，ロボットの姿勢を反映させた座

標変換情報（カメラ座標からロボット座標への変換情報）を自己位置算出システムに送信する。

⑥ 自己位置算出システムでは，それらの受信情報を基にして，画像中に写っているマークを検出し，ロボットの部屋座標系での位置と方向を算出する。

⑦ 自己位置算出システムは，算出した自己位置情報をロボット制御システムに送信する。

視覚認識の研究では，このうち⑤の受信，⑥，⑦の送信，即ち自己位置算出システムを担当する。

3.2.3 設置するマークの検討

ここでは，設置するマークに関して行った検討内容について記述する。

(1) マークの設置場所の検討

壁張りマークを使用したケースについては，HASPにおけるこれまでの研究[1),2)]から，ロボットがマークから5ｍ離れると自己位置認識誤差が最大35㎜（ズーム無しの場合）に達することがわかっている。1㎝以内の精度を目標にすると2ｍ以内の壁又は機器上にマークを貼る必要があり，原子力施設中で配管等の多い場所ではこの条件を確保することは難しい。また，壁張りマークを利用する場合，見えるマークの選択に計算時間がかかる，ロボットの方向の算出精度が良くない等の問題点もあることがこれまでの研究等からわかっている。

天井にマークを設置可能であれば，天井全体をマークとみなすこともでき，天井に白黒（又はカラー）模様を描くことで，自己位置を高速かつ高精度に算出可能である。例えば，カメラを真上に向け，そのレンズ中心部に写った天井の模様からその地点の絶対位置が直接算出できるように天井の模様をコード化する方法が作成可能である。天井の電球を格子状に配置し，ある時間間隔でそれらの電球を順番に点灯させてロボットの自己位置認識に利用しているシステムもある。ロボットは，カメラを真上に向け，自分の上の電球の点灯した時刻から自己位置を大まかに検出する。天井に配置するものを電球でなく模様にすれば，精度はカメラの解像度まで良くすることが可能である。ただし，この方法は，原子力施設の中で配管等の多い場所では，真上の天井部分が見えない箇所が多いこと等からここでは採用できない。

このため，マークは床面に設置する方法を採用する。床面に設置する場合は，ロボットの近くに置くことも可能である。ただし，マークがロボットや人間によって踏まれた場合，マークの表面が汚れるという欠点がある。そこで，マークはできる限り通路脇に設置する，汚れのつきにくいものを採用することにした。

(2) マークの形状とマーク検出方法の検討

マークの形状については，Fig. 3.4の4種類を順に検討した。Fig. 3.4の(1)は，従来のシミュレーション研究で使用したマークの形状で，その検出方法はFig. 3.5に示される。この方法は，エッジ検出，細線化，テンプレート・マッチングを含むため，マークが存在すると推定された一定領域については全画素の処理を必要とするので計算時間がかかる，任意の方向から見た場合にテンプレート・マッチングに難しい等の問題点がある。一般に線分抽出（エッジ検出，細線化）は，環境条件の変化に弱く，計算時間がかかる等の問題点があるため，探索線を利用する方法を次に検討した。

Fig. 3.4 の (2) は，横探索線から得られる白黒強エッジ（●印）をつなぐ直線と縦探索線から得られる白黒強エッジ（▲印）をつなぐ直線との交点からマーク中心が検出できるもので，白黒強エッジがマークのものであるかどうかは次に黒白強エッジ（×印）がくることで確認される。しかし，この1探索線について2か所のチェックでは，実写画像における実験でマーク近辺の黒線を誤って検出することが意外に多い頻度で生じた。

このため，1探索線について複数箇所のチャックが可能な Fig. 3.4 の (3) のマーク形状を検討した。このチェック方法は，最終的に採用した Fig. 3.4 の (4) のマークのチェック方法と同じなので省略するが，黒枠部分が幅1cmと狭いので，照明の強い場所ではマークを3m以上離して見たときに中央の黒帯部分が白くボケ，明確に検出できないことが生じた。

そこで，最終的には，Fig. 3.4 の (4) のマークを採用した。このマークの検出方法は，3.2.5 の (5) に示す。

3.2.4 床置き白黒マークを利用したロボットの自己位置算出システム

ロボットの自己位置算出システムは，次の2つの部分に大きく分けられる。

(1) Cプログラム：ロボット制御システムとの通信インタフェース及びマーク検出過程等を表示するマンマシン・インタフェースを担当する主なプログラムである。

(2) FORTRAN プログラム：画像情報からマークを検出し，自己位置を算出する。Cプログラムから呼び出される。

以下，3.2.5 に今年度作成した FORTRAN プログラムで記述される自己位置算出ソフトウェアの概要を記述する。来年度，マーク検出過程を詳しく表示するマンマシン・インタフェース部分等を追加して，完成させる計画である。

3.2.5 自己位置算出ソフトウェアの概要

(1) 自己位置算出アルゴリズムの概要

ロボットのデッドレコニングが正確であれば，ロボットに搭載したカメラを床面に設置したマークの中心に向けて画像を撮ると，マークの中心は画像の中心に写るはずである。実際には，デッドレコニングが正確でないため，マーへの中心が画像の中心からズレて写る。この画像上のズレは，デッドレコニングによる自己位置（推定値）を基にカメラを制御したために起きる。このため，この画像上のズレの大きさを検出し，それを実世界（部屋座標系）の大きさに変換し，ロボットのデッドレコニングによる推定自己位置から差し引いてロボットの現在位置を算出する。

このため，マーク設置情報を記述する部屋座標系とマークが写る画像座標系との対応付けが必要になる。以下に，マーク設置情報，座標系の定義，入力パラメータ，マーク探索アルゴリズムの概要，座標系の変換について記述し，最後に作成した FORTRAN プログラムの概要を記述する。

(2) マーク設置情報

・ファイル名：mark__param

・メモリ上：markd（100，7）
・１枚のマークに以下の情報が付けられる。
① マーク番号（markno），
② 設置（部屋）X座標（markx）
③ 設置（部屋）Y座標（marky）
④ 設置（部屋）Z座標（markz）
⑤ 設置（部屋）方向（markth）
⑥ マークタイプ（１）
⑦ マークサイズ（marksize）

(3) 座標系の定義（①，②，③は，右手座標系）
① 部屋座標系　：A（x，y，z，th）：a（３），ath
② ロボット座標系：R（x，y，z）　：r（３）
・ロボットの進行方向がx軸
・ロボットの部屋座標系での位置，方向　：walkpos（３），th
③ カメラ座標系　：C（x，y，z）　：c（３）
・カメラの光軸がx軸
・カメラレンズ中心のロボット座標系での位置：vct（３）
・カメラ座標系からロボット座標系への変換（回転）行列：vc（３，３）
④ 理想画像座標系：ＩＩ（x，y）　：riix，riiy
・カメラレンズに歪み等のない場合の画像座標系である。
・画像の中心を原点，横軸をx軸（右が正），縦軸y軸（上が正）。
⑤ 補正画像座標系：ＩＨ（x，y）　：rihx，rihy
・カメラレンズの補正後の画像座標系である。
・レンズの中心補正（＋x０，＋y０）と周辺部の縮み補正（ｋ１）がなされている。
⑥ 左上方原点座標系：（x，y）　：x，y
・画像の左上方を（0.5，0.5），横軸をx軸（右が正），縦軸をy軸（下が正）。
⑦ 画素座標系　：Ｉ（x，y）　：rx，ry
・画素番号が座標値。画像の左上方が（０，０）。

(4) 入力パラメータ
・iparam (1)　：NLINE；横探索線の本数＝２＊NLINE＋１
iparam (2)　：NINT；横探索線の間隔＝NINT（mm）
iparam (3)　：ERRY；横探索線の探索幅の拡張分（mm）
iparam (4)　：エッジの閾値（絶対値）
iparam (5)　：強エッジの閾値（絶対値）
・param (1)　：fc；カメラの焦点距離（mm）
param (2)　：ｘ０；CCD基板上のレンズ中心点のx軸補正（mm）

param (3) ：ｙ０；CCD基板上のレンズ中心点のｙ軸補正（㎜）
param (4) ：dsx；CCD基板上の１画素のｘ軸方向の長さ（㎜）
param (5) ：dsy；CCD基板上の１画素のｙ軸方向の長さ（㎜）
param (6) ：ｋ１；画像の周辺部の縮みパラメータ
param (7) ：エッジの閾値（相対比率値）
param (8) ：強エッジの閾値（相対比率値）

(5) マーク探索アルゴリズムの概要

現在使用しているマークは，Fig. 3.4の(4)に示す白黒マーク１種類である。マークサイズは，１辺16㎝を標準とし，階段等の狭い場所では８㎝のものを使用する。

部屋座標系でマークを探索するが，処理は部屋座標系のマーク位置等を画素座標系に変換して行う。

ロボットのデッドレコニングの最大許容誤差は以下を仮定する。

進行方向 ＝ ERRX（㎜）＝ 200㎜
横方向 ＝ ERRY（㎜）＝ 200㎜
進行角度 ＝ ERRTH（degree）＝ 10°

マークの探索と自己位置算出は，以下の手順で行う。

① 探索するマーク情報をマーク設置情報から読む。
② マーク中心から左右（ERRY ＋ marksize／２），
上下（ERRX ＋ marksize／２）
の範囲の探索を行う。
③ 探索は，まず，Fig. 3.6に示す横方向の探索線（本数：２＊NLINE＋１，間隔：NINT（㎜））を部屋座標上に引き，その地点が写る画素座標系の画素列を対象として，マーク中央右の白黒強エッジを見つけることで行う。即ち，横探索線上の黒白強エッジでかつ16㎝マークにおいて左８㎝，左４㎝，右４㎝にそれぞれ黒白強エッジ，白黒強エッジ，白黒強エッジがある画素（●印，a_i）を探す。
④ マーク中央右の黒白強エッジを検出した複数の横探索線の画素群位置から，画像上に写るマークのほぼ正確な方向がわかる。この方向を次の縦探索線に使用する。
⑤ 縦探索は，Fig. 3.6に示すように④でマーク中央右の黒白強エッジを検出した横探索線のうち上から２番目の線において，マーク中央右の黒白強エッジから16㎝マークにおいて左６㎝，右２㎝の位置を写す左右２つの画素（▲印）から④で得た上下方向に計４本の探索線を行う。即ち，各探索線での最初の白黒強エッジがある画素（×印，b_i）を探す。
⑥ 縦探索で得られた４つの画素（b_1，b_2，b_3，b_4）から，ロボットの推定位置を基準とした場合のマーク中心の位置とマークの方向を算出する。即ち，４つの画素の重心となる画素の部屋座標系での位置，及び上下方向の各探索線で得られたそれぞれ２つの画素（b_1とb_3，b_2とb_4）から出したマークの方向の平均値を部屋座標系に変換して算出する。

⑦ マークの部屋座標系での位置，方向について，画像から算出された値と実際に定義された値とのズレをロボットの推定位置から差し引くことにより，ロボットの現在位置を算出する。

(6) 座標系の変換

(3) で記述した座標系の定義から，各座標系に変換する式は，以下のとおりである。

① ロボット座標系 → 部屋座標系

$$A\begin{pmatrix}x\\y\\z\end{pmatrix} = VR*R\begin{pmatrix}x\\y\\z\end{pmatrix} + WLKPOS\begin{pmatrix}x\\y\\z\end{pmatrix}$$

WLKPOS：ロボットの部屋座標系での位置

ここで，VR，VR^{-1}は，以下のとおりである。

$$VR = \begin{pmatrix}\cos(th), & -\sin(th), & 0\\ \sin(th), & \cos(th), & 0\\ 0, & 0, & 1\end{pmatrix}, \quad VR^{-1} = \begin{pmatrix}\cos(th), & \sin(th), & 0\\ -\sin(th), & \cos(th), & 0\\ 0, & 0, & 1\end{pmatrix}$$

② 部屋座標系 → ロボット座標系

$$R\begin{pmatrix}x\\y\\z\end{pmatrix} = VR^{-1}*\left(A\begin{pmatrix}x\\y\\z\end{pmatrix} - WLKPOS\begin{pmatrix}x\\y\\z\end{pmatrix}\right)$$

③ カメラ座標系 → ロボット座標系

$$R\begin{pmatrix}x\\y\\z\end{pmatrix} = VC*C\begin{pmatrix}x\\y\\z\end{pmatrix} + VCT\begin{pmatrix}x\\y\\z\end{pmatrix}$$

ここで，VC，VC^{-1}は，以下のとおりであり，VCTは入力である。

$$VC = \begin{pmatrix}xx, & xy, & xz\\ yx, & yy, & yz\\ zx, & zy, & zz\end{pmatrix}, \quad VC^{-1} = \begin{pmatrix}xx, & yx, & zx\\ xy, & yy, & zy\\ xz, & yz, & zz\end{pmatrix}$$

④ ロボット座標系 → カメラ座標系

$$C\begin{pmatrix}x\\y\\z\end{pmatrix} = VC^{-1}*\left(R\begin{pmatrix}x\\y\\z\end{pmatrix} - VCT\begin{pmatrix}x\\y\\z\end{pmatrix}\right)$$

⑤ カメラ座標系 → 理想画像座標系

$$\begin{pmatrix} IIx \\ IIy \end{pmatrix} = \frac{FC}{Cx} * \begin{pmatrix} -Cy \\ Cz \end{pmatrix}$$

ここで，FC は，焦点距離で入力である。

⑥ 理想画像座標系 → カメラ座標系

$$\begin{pmatrix} Cy \\ Cz \end{pmatrix} = \frac{Cx}{FC} * \begin{pmatrix} -IIx \\ IIy \end{pmatrix}$$

⑦ 理想画像座標系 → 補正画像座標系

$$\begin{pmatrix} IHx \\ IHy \end{pmatrix} = VL * \begin{pmatrix} IIx \\ IIy \end{pmatrix} + \begin{pmatrix} x0 \\ y0 \end{pmatrix}$$

ここで，$VL = \dfrac{1}{1 + RR * k1}$

$RR = IIx^2 + IIy^2$

k1，x0，y0 は，入力する。

⑧ 補正画像座標系 → 理想画像座標系

$$\begin{pmatrix} IIx \\ IIy \end{pmatrix} = \frac{1}{VL} * \left(\begin{pmatrix} IHx \\ IHz \end{pmatrix} - \begin{pmatrix} x0 \\ y0 \end{pmatrix} \right)$$

⑨ 補正画像座標系 → 左上方原点座標系 → 画素座標系

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 320.5 \\ 240.5 \end{pmatrix} + \begin{pmatrix} IHx/Dsx \\ -IHy/Dsy \end{pmatrix}$$

ここで，Dsx，Dsy は，入力

$$\begin{pmatrix} Ix \\ Iy \end{pmatrix} = \begin{pmatrix} x + 0.5 \\ y + 0.5 \end{pmatrix}$$

⑩ 画素座標系 → 左上方原点座標系 → 補正画像座標系

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} Ix \\ Iy \end{pmatrix} + 0.5$$

$$\begin{pmatrix} IHx \\ IHy \end{pmatrix} = \begin{pmatrix} Dsx * (x - 320.5) \\ -Dsy * (y - 240.5) \end{pmatrix}$$

(7) FORTRAN プログラムの概要

FORTRAN プログラムは，以下の４つのサブルーチンから構成され，C言語で記述された主プログラムから①の mark が呼び出される。②，③，④は，①の mark から呼び出される。

① mark：Robotの位置を算出するサブルーチン

（初期設定データ）

markd（100,7）：マーク設置情報

iparam（5）：整数型パラメータ

param（8）：実数型パラメータ

（入力パラメータ）

wlkpos（3）：ロボットの部屋座標系での推定位置

th：ロボットの部屋座標系での推定方向

markno：探索するマーク番号

vc（3，3）：ロボット座標から部屋座標への変換行列

vct（3）：ロボット座標から部屋座標への平行移動ベクトル

（出力パラメータ）

calpos（3）：ロボットの部屋座標系での計算位置

calth：ロボットの部屋座標系での計算方向

istatus：処理状態コード

② convai：部屋座標系を画素座標系に変換するサブルーチン

③ convia：画素座標系を部屋座標系に変換するサブルーチン

④ err：エラーが発生した場合に処理状態コードを設定するサブルーチン

3.3 センシングシステムの概念設計

原子力施設内の巡回・点検作業には，定期的な点検作業で行われている定められた経路を通り，定められた点検を行う作業以外に，通常の点検時と異なる状況を発見した場合の対処等の不定期な作業が含まれる。そのような不定期な作業が発生した場合，作業員は，その状況に応じて自ら判断し，次にとるべき，おそらく定期的な作業内容とは異なる行動を決定し，行わなくてはならない。このような原子力施設内の巡回・点検作業を，人間に代わって，知能ロボットにさせることを想定すると，知能ロボットは，状況に応じて，とるべき行動を決定するための自律的な判断能力及び，定期点検作業において定められている行動以外の行動を行うことができるための能力を備えている必要がある。特に，後者の能力のうち，定められた経路以外の経路でも移動することができる任意位置移動能力は，欠くことができない能力である。そして，これらの能力を備えるために，知能ロボットは，まず，周囲の状態を検出し，状況を認識する能力を持つ必要がある。

よって，原子力施設内で自律的に巡回・点検作業を行う知能ロボットの実現化を目指した場合，ロボットによる周囲の状態の検出に関する技術，つまりロボットセンシングの技術は必要不可欠な要素技術の1つである。そこで原研情報システムセンターでは，知能ロボットによる原子力施設の巡回・点検技術の研究の一環として，ロボットセンシングの研究を行っている。現在，我々のロボットセンシングの研究では，ロボットが，常に静止した状態で，周囲の状態を検出するとする“静的センシング”及び，ロボットは，巡回点検の対象となる環境の地図やその環境内の機器等に関する情報をモデルとして持っているとする“モデルベースド”を研究の規範としている。

しかし，実際の巡回・点検作業では，モデルとして持つことができない機器の破損や動的に回避しなくてはならない障害物等に出会す可能性があることから，ロボットセンシングの研究を進めてゆく過程で，これらの規範を変更していく必要があると考えている。

そこで，このようなロボットセンシングの研究の一環として，周囲の状態を検出する装置であるセンシングシステムを開発することを計画した。このセンシングシステムは，原研研究炉JRR－3の一部を模して製作した模擬環境内［3］を自由に移動可能な車輪型移動台車に搭載可能であり，視覚的，聴覚的，嗅覚的及び放射線センサを備えた装置として設計・開発される計画である。本報告書では，センシングシステムの製作に先だち，平成4年度に行った概念設計に関して記述する。なお，より詳細については，文献［4］を参照されたい。

3.3.1 ハードウェアの概念設計

原研研究炉JRR－3及び東海第二発電所において実施されている巡回・点検作業を調査した結果，実際の巡回・点検作業では，作業員の五感，特に非接触的な感覚である視覚，聴覚，嗅覚に依存する点検項目が，著しく多いことが分かった。そこで，センシングシステムでは，人間の視覚，聴覚，嗅覚に類似した情報を得ることが可能な，受動的かつ非接触的なセンサを使用することとした。受動的かつ非接触的なセンサを使用することによって，巡回・点検を行う原子力施設に対する影響やその施設への依存性が少なくなる。さらに，センシングシステムは，模擬環境内を巡回する車輪型移動台車に搭載されることから，使用するセンサは，実現かつ実用可能で，さらに小型軽量でなくてはならない。これらのことを考慮し，センシングシステムのハードウェアに関する概念設計では，以下に述べる，視覚的，聴覚的，嗅覚的，及び放射線センサを使用することとした。

- 視覚的センサ　視覚的センサは，パン，チルトの回転自由度を持つ旋回台に設置されたイメージセンサと距離測定定機の二種類のセンサによって構成される。

  イメージセンサとしては，カラーCCDカメラを使用し，距離測定機としては，レーザー光走査による対応点投影法［5］を用いたものを使用する。さらに，センシングシステムの軽量化のため，距離測定機に必要なカメラとイメージセンサのカメラを共有することとした。
- 聴覚的センサ　聴覚的センサのセンサデバイスとして，可聴音を検出する小型軽量な有指向性のコンデンサマイクロホンを，視覚的センサと同一の旋回台に搭載し使用する。
- 嗅覚的センサ　嗅覚的センサは，感度がよく，応答速度が速く，小型軽量である半導体ガスセンサ［6］を複数個使用する。さらに，気体成分の空間分布に大きな影響を与える，温度及び湿度を測定するセンサも嗅覚的センサに含む。
- 放射線センサ　電離箱式サーベイメータを使用する。

以上が，センシングシステムに使用するセンサである。さらに，これらのセンサを含めたセンシングシステムのハードウェア全体の構成について述べる。

センシングシステムのハードウェアは，上記のセンサを含むセンサ部の他に，通信部，前処理部の3つの部分によって構成されているとし，それら各々の機能は，以下に示すとおりである。

センサ部　旋回台に設置されている視覚的センサ，聴覚的センサからなる可動センサと，嗅覚的センサ，放射線センサからなる固定センサによって構成される。センサ部は，センシングの命令に従って，各々のセンサを用い，周囲の状態を検出し，その結果を出力する。また，旋回台もその回転結果及び鉛直方向に対する旋回台自身の傾きを出力する。

通信部　センサ部に接続され，移動機構に搭載される移動ロボット側通信装置と，センシングシステムや移動機構システムを含めた知能ロボット全体の制御やセンシングデータの処理等を行う情報処理用計算機及び以下に述べる前処理部に接続される固定局通信装置によって構成される。通信部は，センシング命令，センシングデータ，さらに，移動機構の制御や応答の信号の送受信を行う。また，使用する状況によって，無線通信，有線通信のいずれかを選択できる。

前処理部　前処理部の一部は，ソフトウェアによる処理を含むが，その明確に定まった機能のため，便宜上，ハードウェアの概念設計に含めることとする。

前処理部は，画像データ前処理装置，距離データ前処理装置，音響データ前処理装置，嗅覚データ前処理装置，放射線データ前処理装置によって構成される。それらの機能は，センサ部から送られてきたセンシングデータを情報処理用計算機によって処理できる型に変換することであり，各々の前処理装置の出力は，以下のとおりである。

画像データ前処理装置	検出画像の3原色（赤，緑，青）に対する輝度値
距離データ前処理装置	検出画像の対象物までの画素ごとの距離値
音響データ前処理装置	検出音に対する周波数分布
嗅覚データ前処理装置	検出気体に対する成分濃度値
放射線データ前処理装置	周囲の放射線の線量当量率

3.3.2　今後の問題

知能ロボットによる原子力施設内の巡回・点検作業の研究におけるロボットセンシングの研究の目標は，実際の原子力施設において作業員が行っている点検作業を計算機に行わせることである。さらに言えば，計算機がセンサによって検出した情報に基づき，周囲の状態を認識し，その状態が正常であるかを判断することである。このことを考慮し，前節で述べたセンシングシステムのハードウェアを用いたロボットセンシングの研究の今度の課題についてまとめる。

まず，センシングシステム・ハードウェアが完成後，その特性・性能及び検出されるデータの精度に関する調査を行う必要がある。前者については，カメラやマイクロホンの特性，通信装置の性能等であり，後者については，画像や距離データ等の精度である。そして，これらの結果は，センシングデータ処理を考える上で考慮すべき重要な要因となる。

次に，各センサごとのセンシングデータ処理の内容について考察することとする。

視覚的センサについて考えた場合，まず輝度値や距離値の列びのデータから，画像内の機器やその他の物体を抽出し，各々を識別し，認識しなくてはならない。さらに具体的には，画像や距離のデータを，計算機内で扱え，各々の機器それ自身やそれらの配置等の情報を含んだ内部表現

に変換しなくてはならない。そして，データベースとして計算機が保持しているモデルと得られた内部表現と比較することにより，認識した機器等の状態が正常であるかを判断できるようになる。また，認識した機器の位置とデータとして持っている環境内の地図から，センシングをした位置，つまり知能ロボットの位置が同定可能となるはずであり，この機能は，知能ロボットが任意位置移動能力を持つために，欠くことのできない機能である。

聴覚センサについては，センシングを行った周囲の可聴音の周波数分布をデータとして得ることができる。よって，それらの得られたデータとあらかじめ測定し，計算機内に保持している音響データとを比較することにより，周囲の音の状態における異常の有無を判断することができるだろう。また，マイクロホンの指向性を使うことによって，異常音の音源の方向を同定することも可能となると考えられる。

嗅覚センサ，放射線センサによって検出されたデータは，異常の種類や内容を同定するためよりは，むしろ，異常の有無を察知するために用いることになる。つまり，嗅覚センサ，放射線センサにおけるセンシングデータ処理は，計算機内に保持している正常時における気体成分の濃度や線量当量率の上限の値をしきい値とし，センシングデータがその値を超えた場合に，知能ロボット全体を制御しているシステムに異常の発生を知らせるものとなる。そのため，これら２つのセンサについては，静的センシングの規範を超え，動的にセンシングを行う必要がある。

以上において，各センサが，個々に行うセンシングデータ処理について考察した。しかし，実際の原子力施設の点検作業において，作業員は，個々の感覚器を個別的に用い，点検作業を行っているのではなく，それぞれの感覚器を協調的に用い，個々の感覚器からの情報を総合し，環境の状態を認識し，異常の有無やその内容を判断している。よって，作業員に代わり，知能ロボットに点検作業をさせる場合にも，やはり，各センサからの情報を総合的に処理し，環境状態の認識や異常の内容等の判断を行うセンシングデータ処理を考えるべきである。そして，このような複数のセンシングデータを総合的に処理し，単一のセンシングデータから得られる情報とは違った情報や，より信頼性高い情報を得るという考え方は，センサフュージョンと呼ばれ，最近のロボット工学の分野において，その重要性・必要性が盛んに論じられている［７］。そこで，今後センシングシステムのセンサによって得られるセンシングデータの処理ソフトウェアを開発していく際，各センシングデータの個別的な処理の他に，センサフュージョンの考え方を考慮した，各々のセンシングデータの総合的な処理を含んだソフトウェアを開発していくことが必要であろう。例えば，カラー画像と距離画像を融合し，カラー距離画像を生成することや，視覚センサの情報と聴覚センサの情報を総合的に使い，配管からの蒸気漏れ異常を発見，認識すること等，さまざまなレベルにおいてセンサフュージョンの考え方に基づいたセンシングデータ処理が考えられる。

センサフュージョンの考え方に基づくセンシングデータ処理を行う際，複数のセンシングデータを処理するため，それらのセンシングデータを１つ１つ直列的に処理するのではなく，まず並列分散的に処理し，次いでその結果を総合的に処理することが必要である。そこで，そのような並列分散的な情報処理や異なるデータの融合を実現するために適していると思われる方法の１つとして，人間の脳内における情報処理を行っている神経細胞網をモデル化したニューラルネットワークが考えられる。しかし，単にニューラルネットワークと言っても多種多様であり，センシ

ングシステムによる異常の知覚や認識，またその内容の判断を可能とするモデルを構成するためには，今後，ニューラルネットワークの理論的研究及び計算機による多くのシミュレーションが必要になると考えられる。

参考文献

［1］ 浅井 清他，“原子力知能化システム技術の研究 － 平成元年度作業報告 － ”，JAERI－M 90－060（日本原子力研究所，1990.3）。

［2］ 大谷孝之，“ディジタル画像処理とその人間動作シミュレーション技術の研究（HASP）への応用”私信（1990.3）。

［3］ 秋元正幸他，“原子力知能化システム技術の研究 － 平成3年度作業報告 － ”，JAERI－M 92－198（日本原子力研究所，1993.1）。

［4］ 海老原健一，“知能ロボットによる原子力施設内巡回点検用センシングシステムの概念設計”，JAERI－M 93－225（日本原子力研究所，1993.11）。

［5］ 小関 修，中野倫明，山本 新，“光切断法を用いた実時間距離検出装置”，信学論，J－68D 5，pp.1141－1148（1985）。

［6］ 新田正義他，“ガスセンサとその応用［原理・製法から応用システムまで］”（パワー社，1987）。

［7］ 日本ロボット学会誌，Vol.8，No.6（1990）。

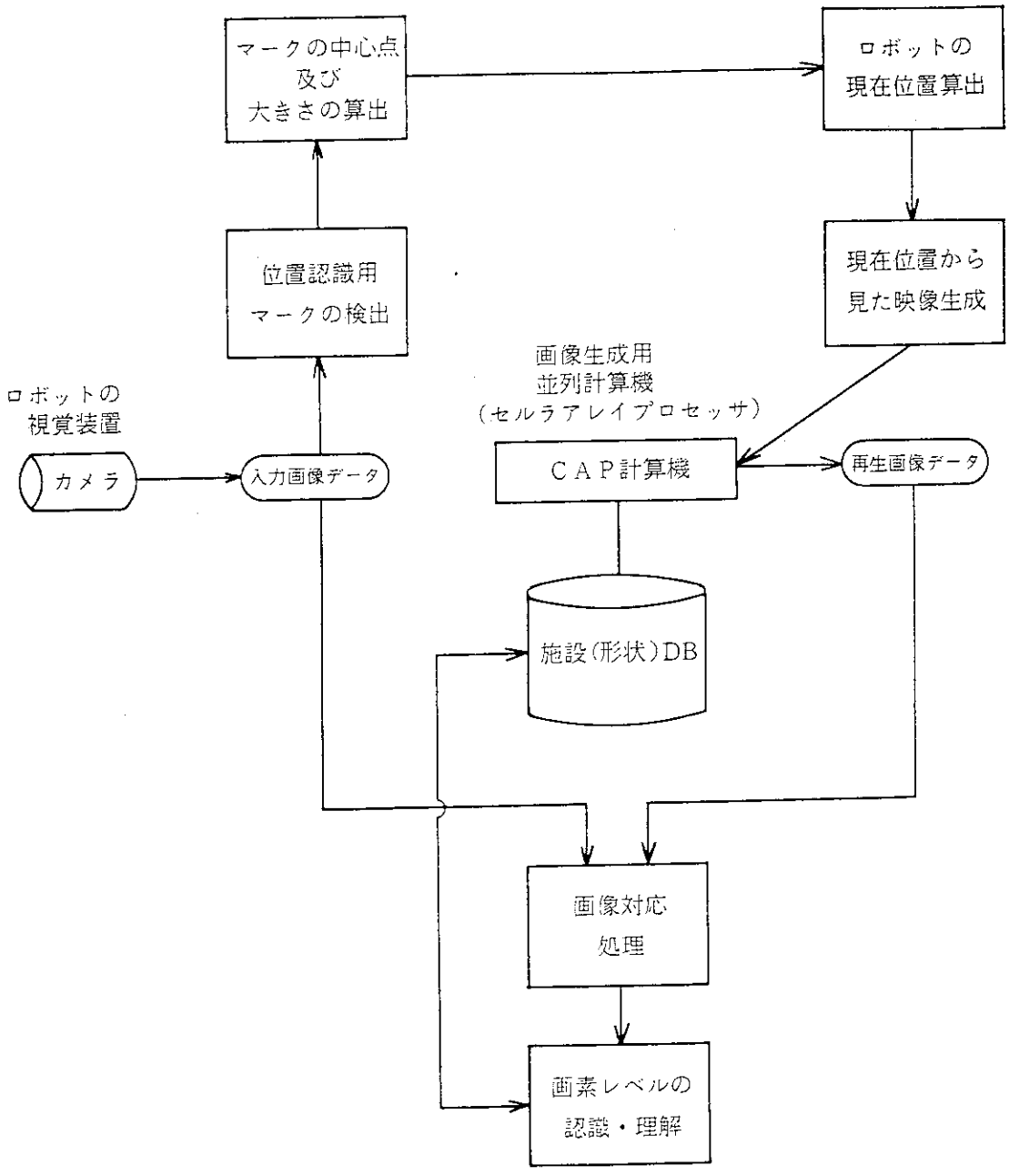


Fig. 3.1 A recognition process of the vision in a structured environment

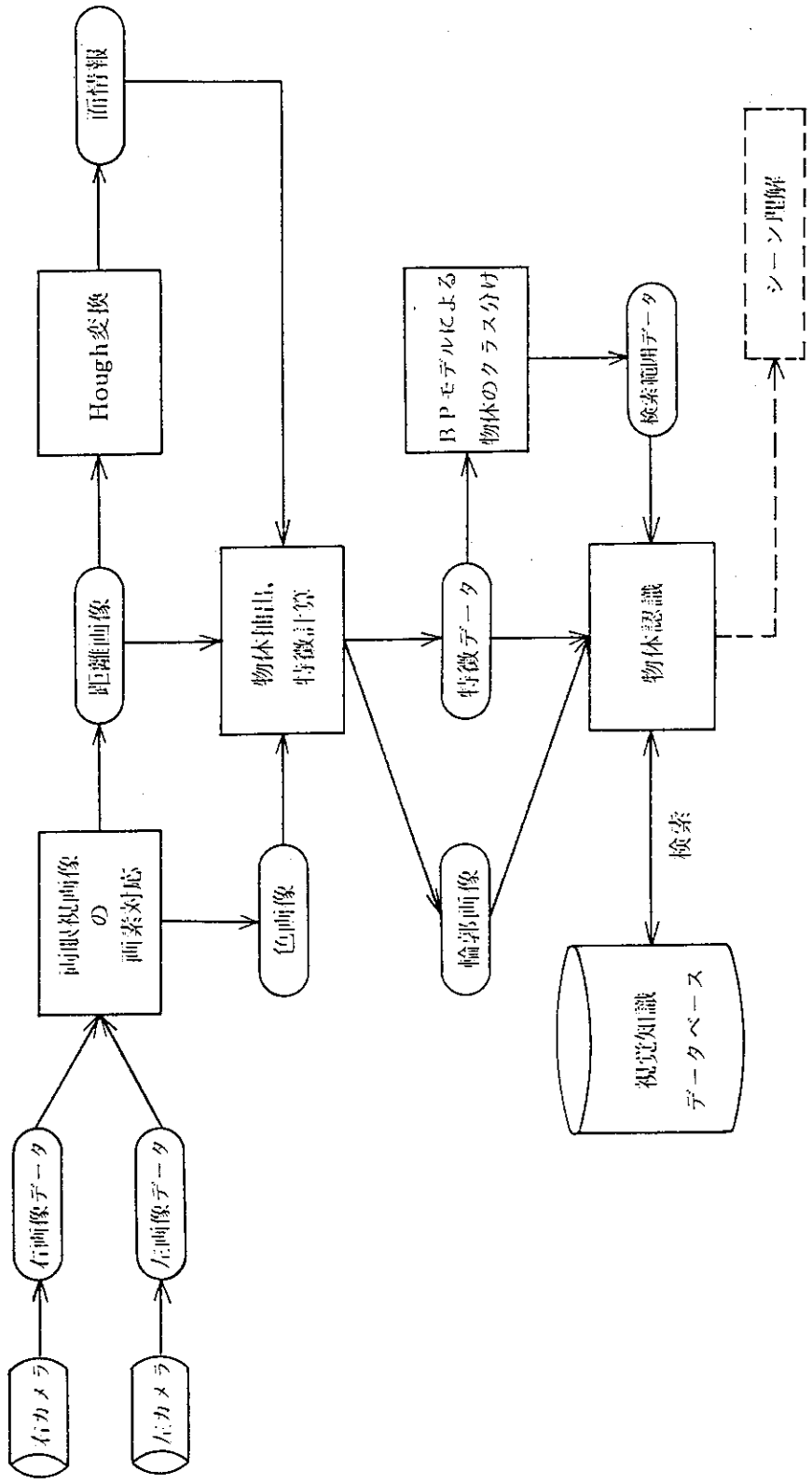


Fig. 3.2 A recognition process of the vision in a semi-structured environment.

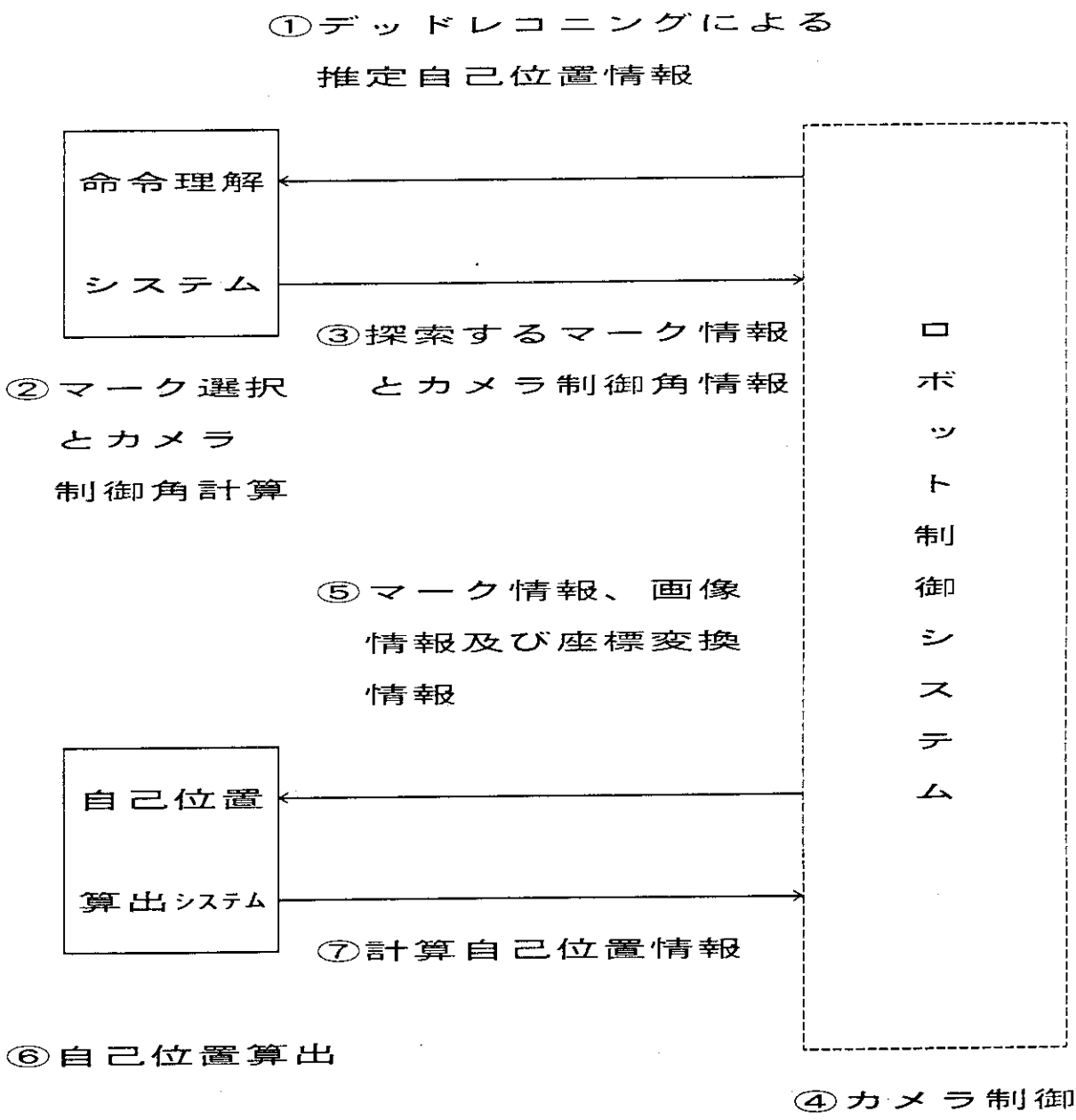
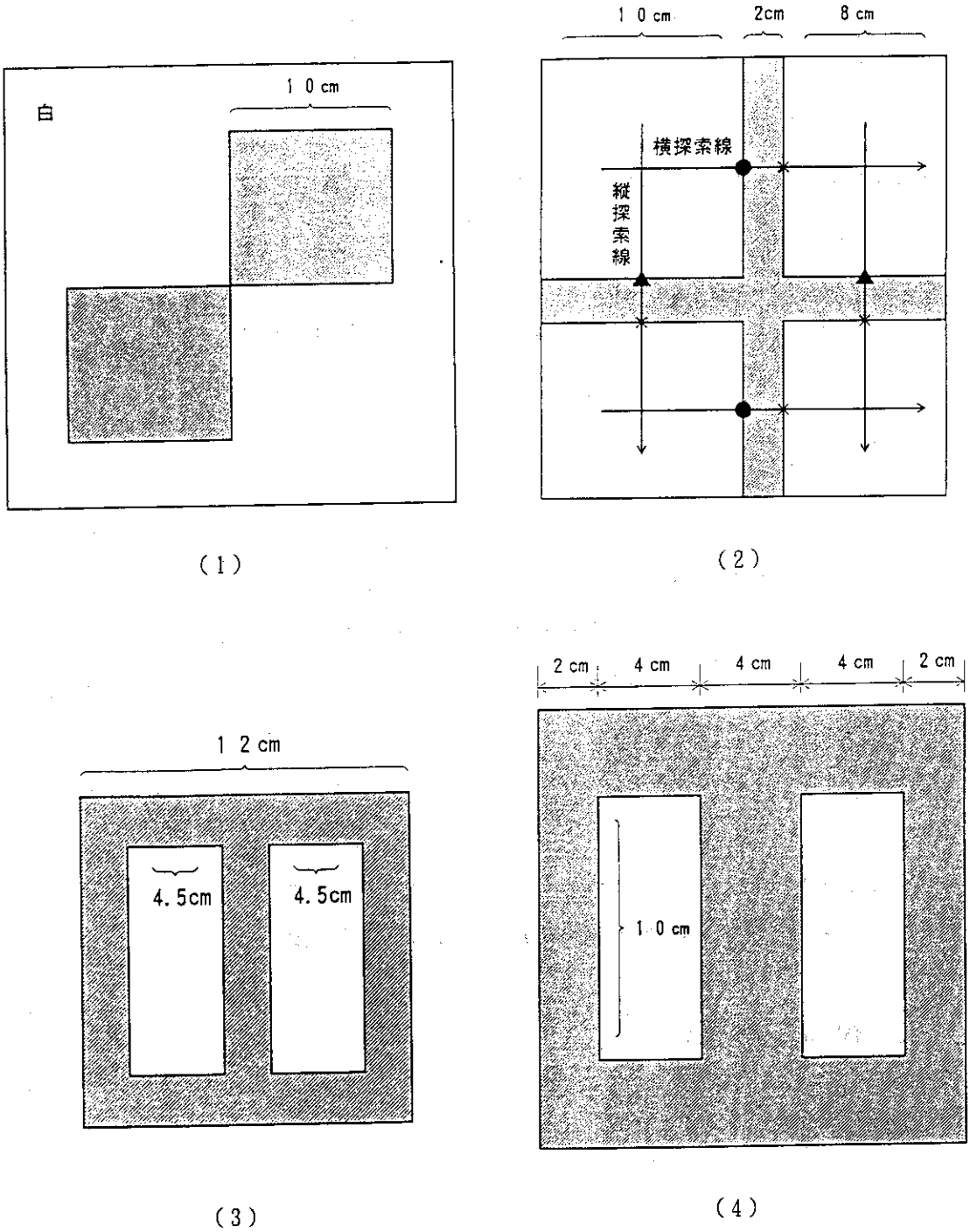


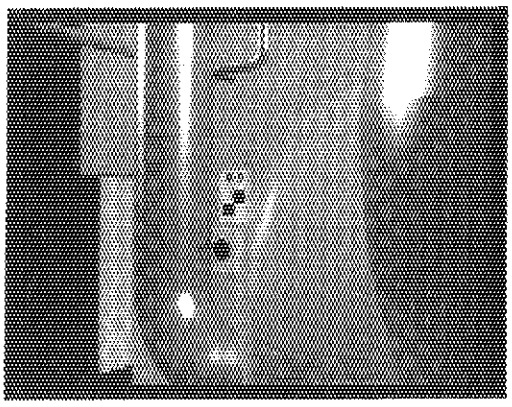
Fig. 3.3 Outline of self-location recognision system.



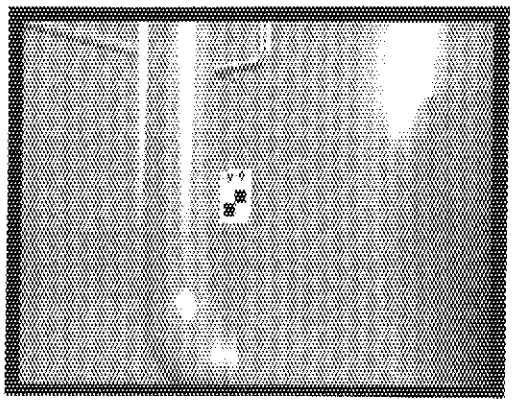
(1)　(2)

(3)　(4)

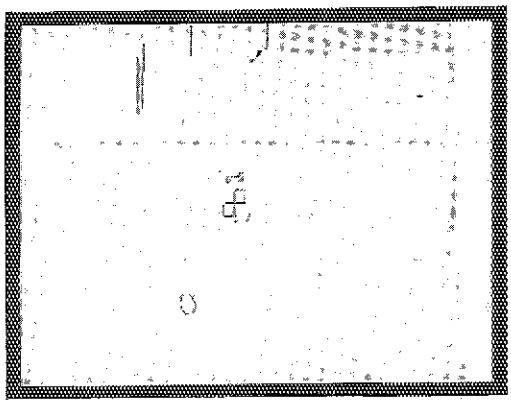
Fig. 3.4 Examined marks.



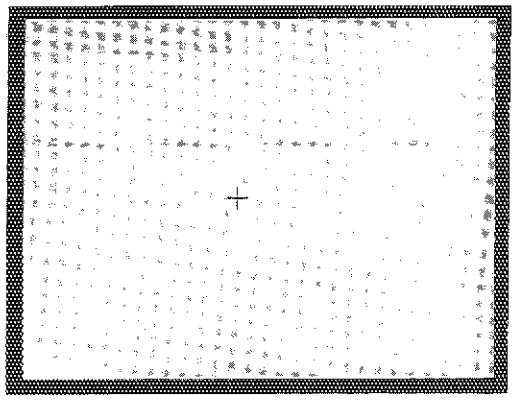
a) visual image



b) color matching



c) edge detection



d) template matching

Fig. 3.5 Mark detection processes.

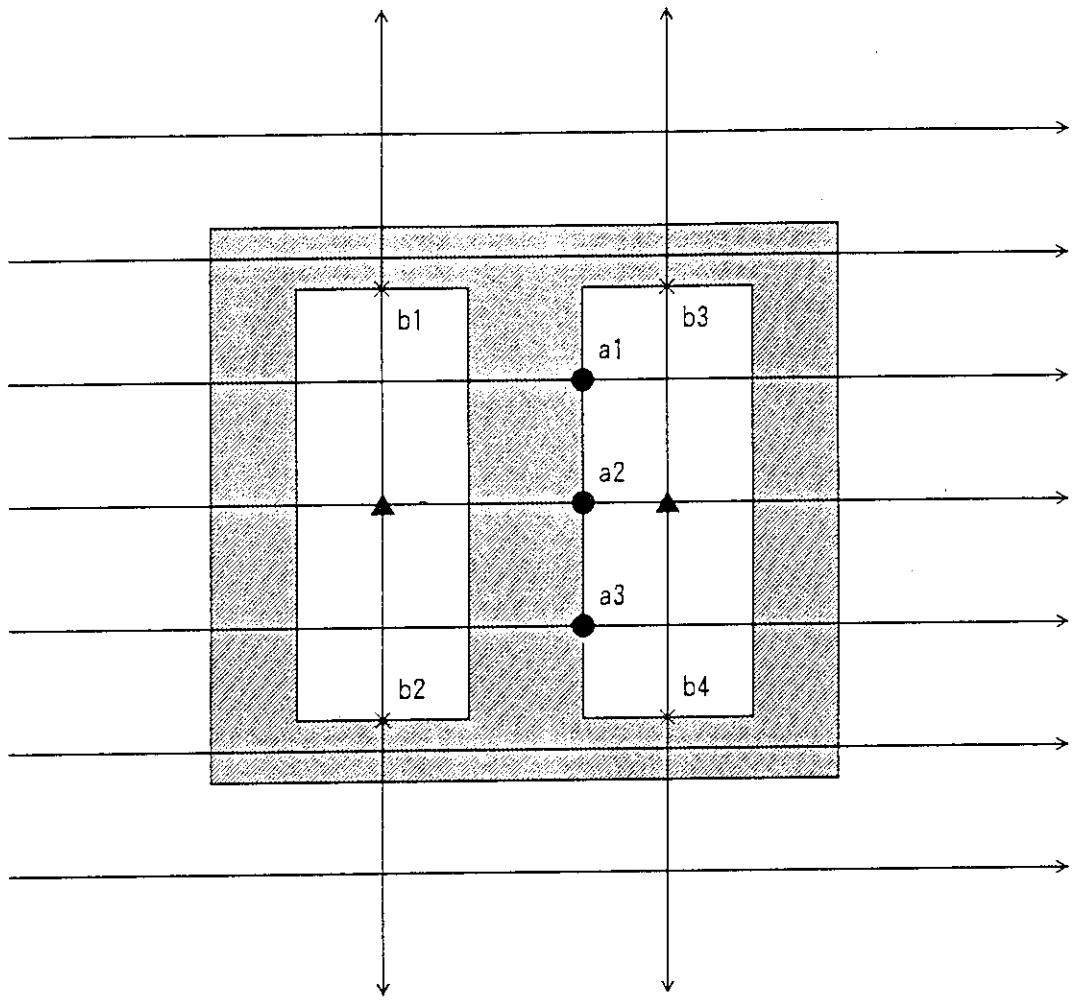


Fig. 3.6 Mark search lines.

4. 人間形ロボットの２足歩行シミュレーション

4.1 はじめに

HASP[1]では，原子力プラントにおける巡回点検用人間形知能ロボットの研究開発を行っている。本章では，この人間形知能ロボットの移動機構を構成する２足歩行について報告する。

我々の研究の目的は，プラント内巡回点検用人間形知能ロボットの設計であり，はじめに設計に有用なソフトウェアの開発を行っている[2),3)]。即ち，２足歩行をハードウェアで実現する前に，数値シミュレーションによってロボット設計パラメータの考察を行い，歩行安全性を十分に検討する。次にこの結果を基にプロトタイプ・モデルの試作を行いシミュレーション・モデルの妥当性及び制御性の検証，ハードウェアの問題点及び知能化の検討を行う。そして，命令理解や視覚認識等，他の要素技術を統合化した人間形知能ロボットの設計を行うものである。

これまでに設計に有用なシミュレーション・ソフトウェア開発の観点から，直進定常歩行に関して，種々のタイプの補償動作の計算法を示してきた。しかし，この手法に代表されるような，従来の２足歩行ロボットの制御法は，事前のシミュレーションによって安定歩行パターンを解析し，それをプログラム制御することでロボットを歩行させるものであり，ロボットと環境との相互作用が予め分かっている必要がある。特に，ロボットが歩行中に自身の状態（状況）を的確に把握し，それに基づいて歩行安定化制御をしているのではないため，予期できない外乱等には対処できない。また，安定歩行のための補償動作の計算は，非線形の微分方程式の初期値問題を定常歩行という仮定の下に周期解として扱っているため，解析対象とすることができる歩行パターンは限定される。さらに初期値問題に対する収束計算が必要であり，計算コストの増大など，多くの問題を持っている。

そこで，我々は，歩行パターンの多様化・高度化には，これまでのようなプログラム制御ではなく，ロボットの状態を検出しこれを基に安定化を行うリアルタイム制御法が必要であると考え，今年度新たにリアルタイム２足歩行制御法を提案した。本章ではこの新たに開発した制御法の概要と，この制御法のシミュレーションによる妥当性の検証，そして問題点及び今後の方針について述べる。なお，本制御法に関しては現在JAERI－Mレポートを執筆中であるので，詳細についてはそちらに譲るとし，本稿では概略を記述することにとどめる。また，本制御法による２足歩行シミュレーション・ソフトウェアを含め，これまでに開発してきたソフトウェアのI／O関係についてTable 4.1にまとめておく。

4.2 ロボット・モデル及び運動方程式

研究の対象とする２足歩行ロボット・モデルをFig. 4.1に示す。このモデルは上肢なし人間形モデルで，上体おもり，アクチュエータ等を質点とし，リンク構造部材を鋼体とする準質点系モデルである。本２足歩行ロボット・モデルは８ボディ，７関節からなり，下肢の前頭面内の動

きは両脚（legs）が平行で，足（foot）は常に床面と水平であると仮定し，下肢５自由度（ピッチ軸４自由度，ロール軸１自由度），腰２自由度（ピッチ軸・ロール軸それぞれ１自由度）の合計７自由度を持つ（Fig. 4.2）。なお，現段階では直進歩行を前提とし方向転換の自由度は与えていない。

一般に剛体系の運動方程式は式（4.1）で表せる。

$$M=\sum_{i=1}^{n}(r_i-\Delta r)\times m_iG+\sum_{i=1}^{n}(r_i-\Delta r)\times m_i\frac{d^2r_i}{dt^2}+\sum_{i=1}^{n}\frac{d(J_i\cdot\omega_i)}{dt} \qquad (4.1)$$

$\Delta r=(X,\ Y,\ Z)$：任意の点の位置ベクトル，

$M=(M_X,\ M_Y,\ M_Z)$：Δr点におけるモーメント，

n　：ボディの数，

m_i　：ボディｉの質量，

$r_i=(x_i,\ y_i,\ z_i)$：ボディｉの重心位置ベクトル，

$G=(0,\ 0,\ g)$：重力加速度ベクトル，

$J_i=\mathrm{diag}(J_{xi}\ J_{yi}\ J_z)$：慣性テンソル，

$\omega_i=(\omega_{xi},\ \omega_{Yi},\ \omega_Z)$：ボディｉの角速度ベクトル，

式（4.1）に関し床面上において，左辺のＸ成分とＹ成分が共に零になる点Δrを，Zero Moment Point（ZMP）[4]と呼び歩行中の安定な状態の規範として用いた。このZMPがロボットの足底接地面の形成する多角形の安定領域内に存在していれば，ロボットは安定に歩行できる[5]。

4.3　ZMPの動的監視に基づく制御法（直進完全歩行）

これまで採用されている２足歩行制御方式のほとんどはZMP規範に基づくプログラム制御である。即ち，ZMP規範に基づく安定歩容，補償動作等をロボット運動のコンピュータ・モデルによって計算し，与えられた動作と設定すべき各関節などの駆動力との関係を事前に求める。プログラム化されたこの関係をロボットに搭載したマイクロ・コンピュータによって処理し，ロボット動作を制御する方式である。この方法では，予め設定されたパターンで歩行を行うため，歩行中のロボットに対する予期しない外乱等動的な変化に頑強に対処できない。このような動的変化に対処するためには，時々刻々と変化するロボットの状態を検出し，それを基に安定化を行うリアルタイム制御法が必要であると考える。

今回開発した制御法では，ロボットの歩行状態を動的に検出し，それを基にZMPを算出し，このZMPが安定領域に存在するよう補償動作を制御する。即ち，既に算出されたZMP軌跡から将来のZMPを推定し，この推定ZMPが安定領域を外れないような安定化補償動作を，ロボット動作の数学モデル等を用いて動的に生成するものである。本手法は，予期しない外乱に対しても適応可能であるが，今回は外乱の無い直進完全歩行に本制御法を適用し，その妥当性と有効性をシミュレーションによって検証する。

4.3.1 制御法の概略

ここで取り扱うモデルは，直進完全歩行における任意の歩容に対する補償動作を上体で行うものであり，歩行中制御周期毎に補償動作を算出し制御するリアルタイム制御を前提としている。本制御法のフローチャートをFig. 4.3に示す。はじめに定常歩行部の1歩を計算し，これに接続する境界条件を満足する過渡歩行部を作成する。遊脚の足の軌跡の作成にはスプライン関数による補間を用い，腰の高さは一定とし定常歩行時の腰の速度は等速として，逆運動学計算により下肢の歩容を作成する。ZMPは安定領域内のほぼ中央を滑らかに移動する軌跡（設定ZMPと呼ぶ）を作成する。また，過渡歩行部は発進・停止それぞれ1歩とし，発進の下肢動作を行う前及び停止の下肢動作が終了した後に，下肢は運動しないが上体だけ運動を行う歩容静止期を設ける[2]。

この基本歩行パターンに基づいて歩行させた場合の上体補償動作のシミュレーションを行う。歩行動作中常にZMPを監視し，設定ZMPと実ZMPの偏差が，あるしきい値を越えると予想されるとき及び越えたときに，上体動作（前頭面θ，矢状面ϕ，$\alpha=0$；一定）によって補償を行い，偏差がしきい値内にあるときに補償動作を収束に向かわせる。

なお，歩行中のZMPの算出については，ロボットに作用する外力やモーメント等全ての情報を含んでいる床反力に着目し，これを検出することでZMPを算出（実モデルでは床反力センサ情報を用いて算出）するのが適当であるが，シミュレーションではロボットの運動状態を示す関節角度情報から式（4.1）を基に算出することとする。

4.3.2 ZMPの予測

ZMPの予測には，ZMPの変化速度を用いる方法が考えられるが，直接ZMPの変化速度を検出することは困難であり，数値計算に頼らざるを得ない。また，ZMPの時間微分値を求めることは，シミュレーションにおいては角加速度の微分値が必要になるなど計算時間がかなり増大する。

しかし，本制御法ではそれほど正確な予測値を必要としないため，簡易で計算時間のあまりかからない手法が望ましい。そこで，今回は1）多項式近似（ラグランジェ補間）による推定，2）微分値近似＋オイラー法による推定の2つの場合[6]について検討を行う。1）の場合は3ステップ前までのZMPの値を用い，これを2次関数で近似し次のステップでのZMPを推定する。2）の場合は，微分値を1ステップ前の値で近似し，オイラー法によって推定する方法である。

4.3.3 補償動作の生成

ロボットの運動方程式は非線形連立常微分方程式である。この式から分かるように，修正すべきZMP変更量とそれに対する補償動作の角度との関係を一般的に求めることは困難である。しかし，角度と角速度が与えられたときに，角加速度について解を求めることは可能である。つまり，ZMPの変更量と補償動作の角加速度との関係を定量化することができる。実際に補償動作の角加速度曲線とZMP軌跡及び床反力曲線（X・Y成分）は，その形が非常によく類似している。ここでは，運動方程式により直接制御量（角加速度）を算出する方法を用いるが，ZMP変更量と角加速度との関係を定量的に評価しテーブル化する事ができれば，制御の高速化やファジィ

制御による安定歩行も可能になると考えられる。

ZMP 偏差に対する実際の ZMP 修正量はこの偏差に修正用ファクターを乗じて決定し，制御量を決める。この修正用ファクターは，シミュレーションを緩用して決定する。制御入力の型としては，パルス，ランプ，ステップなど種々のタイプが考えられるが，パルスはその時点のみの修正に終わってしまうこと，ランプは応答が遅くその間に安定領域を外れてしまう恐れがあること，ステップは補償動作がすぐに発散してしまい収束させるのが難しいことなどから，ここでは以下に述べるような制御入力の型とする。即ち，角加速度はステップ状に立ち上がるその後数秒の後に 0 に収束させる。このとき，同時に角速度，角度も 0 に収束させるのである。この条件を満足する制御入力を多項式を用いて作成する。条件は初期値と最終値の角度・角速度・角加速度となるため，5 次関数での記述が可能であるが，角速度・角加速度の滑らかな収束を考慮すれば 7 次以上とするのが妥当である。そこで，制御入力の型を次式で記述し，各パラメータをシミュレーションにより決定する。

$0 \leqq t \leqq t_c$

$$\theta(t) = a_1 (t-t_c)^n + a_2 (t-t_c)^{n-1} + a_3 (t-t_c)^{n-2}$$

$$\dot{\theta}(t) = a_1 n (t-t_c)^{n+1} + a_2 (n-1) (t-t_c)^{n-2} + a_3 (n-2) (t-t_c)^{n-3}$$

$$\ddot{\theta}(t) = a_1 n (n-1) (t-t_c)^{n-2} + a_2 (n-1) (n-2) (t-t_c)^{n-3} + a_3 (n-2) (n-3) (t-t_c)^{n-4}$$

$$a_1 = \frac{t_c^2 \ddot{\theta}_0 + 2 t_c \dot{\theta}_0 (n-2) + \theta_0 (n^2 - 3_n + 2)}{2 (-t_c)^n}$$

$$a_2 = \frac{t_c^3 \ddot{\theta}_0 + t_c^2 \dot{\theta}_0 (2n-3) + \theta_0 t_c n (n-2)}{(-t_c)^n}$$

$$a_3 = \frac{t_c^4 \ddot{\theta}_0 + 2 t_c^3 \dot{\theta}_0 (n-1) + \theta_0 t_c^2 n (n-1)}{2 (-t_c)^n} \qquad (4.2)$$

$t \leqq t_c$

$$\theta(t) = 0.0$$

$$\dot{\theta}(t) = 0.0$$

$$\ddot{\theta}(t) = 0.0$$

t：制御入力時からの経過時間，

t_c：収束までの時間，

n：制御入力の次数；$n \geqq 7$

$\ddot{\theta}_0$：初期角加速度（制御入力），

$\dot{\theta}_0$：初期角速度，

θ_0：初期角度

ϕについても同様にして作成する。

4.4 制御法の妥当性の検証

4.4.1 ZMPの予測

予測方法1），2）について検討する。歩容を歩幅300［mm］，1.0［s／step］とし上体を不動（$\theta=\phi=0$ ；一定）とした定常歩行を対象に，1歩間のZMPの予測値について，両者の実ZMPとの偏差の比較を行った結果をTable 4.2に示す。これらの結果から分かるように両者ともかなりよい近似であり，実用に値するものである。両者の違いは僅かであるが2）の微分値近似＋オイラー法の方がやや精度がよい，本制御法ではこの2）の方法を用いるものとする。

4.4.2 制御入力の決定

制御入力に要求されるのは，第一に補償動作が発散しないこと，第二に角度の極値の絶対値が小さいこと，第三に角加速度の変動が小さいことである。今回は，定常歩行部歩幅300［mm］，1.0［s／step］の完全歩行を中心にシミュレーションを行い，n，t_cの値をそれぞれ10，2.0［s］で歩行中一定とした。しかし，制御のタイプや歩容の違いによってこれらの最適な値を決定することで，より適切な制御が可能になると思われる。

4.4.3 歩行シミュレーション

本制御法を適用した歩行のZMP軌跡のシミュレーション結果をFig. 4.4に示す。X方向のZMPについては安定領域内に入るように制御が適正に行われているが，Y方向のZMPについては安定領域を外れる部分がある。特に発進の1歩から安定領域を外れており，制御法の改良が必要である。X方向については移動による慣性力によって補償効果が期待できるのに対し，Y方向には移動による慣性力が作用しないためである。

ここで，簡単なモデルを用いてZMPの変動について考察してみる。Fig. 4.5は補償動作のθの1自由度について簡略化した1質点モデルである。このようにモデル化するとZMPは式（4.3）となる。

$$YZMP = -\frac{L\left[\sin\theta\left(g-L\dot{\theta}^{2}\cos\theta-L\ddot{\theta}\sin\theta\right)\right]}{g-L\dot{\theta}^{2}\cos\theta-L\ddot{\theta}\sin\theta} - \frac{L^{2}\left[\cos\theta\left(\dot{\theta}^{2}\sin\theta-\ddot{\theta}\cos\theta\right)\right]}{g-L\dot{\theta}^{2}\cos\theta-L\ddot{\theta}\sin\theta} \tag{4.3}$$

式（4.3）において歩行開始期を考えると，つまり$\theta=\dot{\theta}=0$とすると，

$$YZMP = -\frac{L^{2}\ddot{\theta}}{g} \tag{4.4}$$

となる。即ち，YZMPを正方向に移動させる場合$\ddot{\theta}$は正となる。これはYZMPの移動方向と

逆方向に上体を振ることになる。この場合，角度は時間の2乗に比例して増大するため，重力項が徐々に支配的となり，YZMP は正方向への移動量が減少して，やがて負の方向に向かい出す。これを正の方向に戻すために，角加速度は更に正方向に大きくなりさらなる角度増大を招き，ついには YZMP は安定領域を外れてしまう。これは，角速度・角加速度を無視できるとした場合に，式（4.3）が式（4.5）に示すように重心を表す静歩行の式となり，YZMP の移動方向と上体の移動方向が一致するためである。

$$YZMP = -L \sin\theta \tag{4.5}$$

つまり，ZMP の方程式は式（4.4）と（4.5）から分かるように角度と角加速度に関して，相反する性質を持つためこのようなことが起こるのである。これらのバランスによって発散しない補償動作が成立している。実際に YZMP においては角速度と角加速度が小さく静歩行と一致する（ZMP 軌跡と重心の軌跡が一致）時期が存在している。

今回の直進歩行においては，この問題を発進の歩容静止期に前頭面のみ重心を支持脚側に移動させることで回避する。この移動においては，ZMP への影響が少ない角加速度の範囲をシミュレーションにより決定する。この角加速度の範囲を－0.4～0.4［rad／s²］とし，重心移動のための動作をこの範囲内で指定した角加速度 $\ddot{\theta}_{0a}$ と発進の歩容静止時間 t_{sa} から以下のように与える。

$$0 \leqq t \leqq \frac{t_{sa}}{2}$$

$$\ddot{\theta}(t) = \frac{2\,\ddot{\theta}_{0a}}{t_{sa}}\,t$$

$$\dot{\theta}(t) = \frac{\ddot{\theta}_{0a}}{t_{sa}}\,t^2$$

$$\theta(t) = \frac{\ddot{\theta}_{0a}}{3\,t_{sa}}\,t^3$$

$$\frac{t_{sa}}{2} < t \leqq t_{sa}$$

$$\ddot{\theta}(t) = -\frac{2\,\ddot{\theta}_{0a}}{t_{sa}}\,t + 2\,\ddot{\theta}_{0a}$$

$$\dot{\theta}(t) = -\frac{\ddot{\theta}_{0a}}{t_{sa}}\,t^2 + 2\,\ddot{\theta}_{0a}\,t - \frac{\ddot{\theta}_{0a}}{2}\,t_{sa}$$

$$\theta(t) = -\frac{\ddot{\theta}_{0a}}{3\,t_{sa}}\,t^3 + \ddot{\theta}_{0a}\,t^2 - \frac{\ddot{\theta}_{0a}}{2}\,t_{sa}\,t + \frac{\ddot{\theta}_{0a}}{12}\,t_{sa}^2$$

$$t > t_{sa}$$

$$\ddot{\theta}(t) = 0.0$$

$$\dot{\theta}(t) = 0.0$$

$$\theta(t) = 0.0 \tag{4.6}$$

Fig. 4.6～4.10に完全歩行の7歩のシミュレーション結果を示す（スティック線図は0.2［s］間隔）。また，これらの21歩の場合についても，補償動作に発散はなく，歩行停止とともに収束する安定歩行が可能となっている。なお，スティック線図は今年度開発した「ロボット運動可視化システム」[7]を用いて，シミュレーション結果を動画化したもののハード・コピーである。このシステムは，ワークステーション上でロボットの歩行速度に合わせたリアルタイム表示や表示速度の変更などが可能であり，シミュレーション結果の確認及び解析に有用なシステムである。

修正用ファクター及びしきい値については，シミュレーションにより安定歩行となり補償動作の発散しない，かつ収束までの時間が短い場合についてサーベイし，最適値を決定した。修正用ファクターは0.2～0.5程度が適当で，これ以上になると角加速度が過渡的な動きとなるため，補償動作がぎこちなくなるうえ，発散する傾向がある。しきい値については，ここではXZMP：0.05［m］，YZMP：0.04［m］としたが，この値も修正用ファクターと同様に，あまり小さくし過ぎると補償動作は発散する傾向にある。これは，補償動作の収束期間が短くなるためである。

補償動作の計算は上記7歩の場合でCPU時間が約2.3［s］（FACOM M－780／20；スカラ計算）であり，実用に十分値する。なお，補償動作を歩容にまで拡張する場合には，逆運動学計算に比較的多くのCPUを費やしているため，これらのアルゴリズムについて検討が必要である。

前述のように本制御法の有効性と妥当性が確認された。しかし，事前に修正用ファクターやしきい値などのいくつかのパラメータ・サーベイを必要とするためこれらについて定量化が必要である。これらの値は，歩行速度などのいくつかの歩容パラメーターに依存する値であるから，さらにシミュレーションを進めることでこれらの定量化は可能であると考えられる。

4.5 今後の課題

本章では，従来の2足歩行ロボットの制御法である，プログラム制御での歩行による問題点から，ロボット自身の現在の動作状況を検出しながら制御する方法の一つとして，ZMPの動的監視に基づく制御法を提案した。そして，外乱の無い直進完全歩行に本制御法を適用し，シミュレーションによってその妥当性を確認した。

しかし，まだいくつかの残されている問題がある。第1に腰及び足の軌道についてである。今回は，腰の軌道は一定高を等速（定常歩行時）で動くと仮定し，足の軌跡は接地時の相対速度が0となるようにスプライン関数による補間を行って作成した。これらの軌道をエネルギーやアクチュエータのトルク・パワーなどに関する評価関数の下に最適化し，より安定で自然な歩行とすべきでる。第2に外乱などによって，上体のみでは補償しきれない場合に対処するための，制御法の開発である。これは，歩幅や周期を変更することによって補償するか，腰の位置の変更でも補償することが可能であると考えている。また，上体の胴の長さを長くすることも補償範囲の拡大につながる。第3に歩行の高速化や方向転換の自由度付加にともなう，ヨー軸まわりのモーメント補償の問題である。ヨー軸まわりのモーメントの補償は，早稲田大学のWL－12RV[8]のような方法や，腕を付加することによって可能になると思われる。

今後の方針としては，本制御法の外乱等への適応性についての検討，上記問題点の解決及び方向転換動作の付加を行い，プロトタイプ・モデルの設計，2足歩行ロボット・ハードウェアの試

作へと進めていく計画である。

参考文献

1) 例えば，秋元，久米　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）　－平成3年度作業報告－，JAERI－M 92－198，1993.
2) 久米：人間型ロボットの2足歩行シミュレーション，JAERI－M 91－047，1991.
3) 久米：人間型ロボットの2足歩行シミュレーション　－シミュレーションモデルの開発－，第9回日本ロボット学会学術講演会，PP. 315－318，1991.
4) M. Vukobratovic（加藤，山下　訳）：歩行ロボットと人工の足，日刊工業新聞社，1975.
5) 高西，加藤　他：2足歩行ロボット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.
6) 長島：数値計算法，槇書店，1986.
7) ロボット運動可視化システムの製作　－作業報告書－，1993.
8) 高西，寸土　他：上体運動により3軸モーメントを補償する2足歩行ロボットの開発　～高速動歩行の実現～，第6回知能移動ロボットシンポジウム，PP. 1－6，1993.

Table 4.1 Input／Output of our computer simulation software systems.

安定歩行動作解析			上体補償型	歩容補償型	上体・歩容補償型	ZMP監視 上体補償型
解析可能な歩行種別	直進歩行（平地・階段）	定常	○	○	○	
		発進	○			
		停止	○			
		完全	○			○
	その他					
実行コマンド			ｃｔ	ｃｇ	ｃｔｇ	ｎｃｔ
入力			1) 歩容に関するﾃﾞｰﾀ・ﾌｧｲﾙ名 ・歩幅，歩行時間 ・腰動作 ・遊脚動作 ・歩数 など	1) 上体動作に関するﾃﾞｰﾀ・ﾌｧｲﾙ名 ・上体動作 ・歩幅，歩行時間 ・遊脚動作 など	1) 上体動作に関するﾃﾞｰﾀ・ﾌｧｲﾙ名 ・上体動作（矢状面） ・歩幅，歩行時間 ・遊脚動作 など	1) 歩容に関するﾃﾞｰﾀ・ﾌｧｲﾙ名 ・歩幅，歩行時間 ・腰動作 ・遊脚動作 ・歩数 など
			2) 上体補償動作に関するﾃﾞｰﾀ・ﾌｧｲﾙ名 ・歩行種別の選択 ・補償動作推定初期値 など	2) 歩容補償動作に関するﾃﾞｰﾀ・ﾌｧｲﾙ名 ・ZMP偏差許容値 ・腰高さ など	2) 上体・歩容補償動作に関するﾃﾞｰﾀ・ﾌｧｲﾙ名 ・ZMP偏差許容値 ・腰高さ など	2) 上体補償動作に関するﾃﾞｰﾀ・ﾌｧｲﾙ名 ・ZMP偏差しきい値 ・補償動作の初期値 など
			3) ﾛﾎﾞｯﾄ・ﾎﾞﾃﾞｨに関するﾃﾞｰﾀ・ﾌｧｲﾙ名 ・ﾎﾞﾃﾞｨ寸法 ・材質，強度 ・質量，慣性ﾓｰﾒﾝﾄ など			
			4) 出力ﾃﾞｰﾀ・ﾌｧｲﾙ名 （ﾛﾎﾞｯﾄ・ﾎﾞﾃﾞｨ位置時系列ﾃﾞｰﾀ）			
出力（ファイル）			1) ﾛﾎﾞｯﾄ・ﾎﾞﾃﾞｨ位置時系列ﾃﾞｰﾀ 2) 関節角時系列ﾃﾞｰﾀ 3) 歩容計算結果 4) 上体補償動作計算結果 5) ﾓｰﾒﾝﾄ計算結果 6) ｱｸﾁｭｴｰﾀ・ﾄﾙｸ，ﾊﾟﾜｰ計算結果	1) ﾛﾎﾞｯﾄ・ﾎﾞﾃﾞｨ位置時系列ﾃﾞｰﾀ 2) 関節角時系列ﾃﾞｰﾀ 3) 上体動作計算結果 4) 歩容補償動作計算結果 5) ﾓｰﾒﾝﾄ計算結果 6) ｱｸﾁｭｴｰﾀ・ﾄﾙｸ，ﾊﾟﾜｰ計算結果	1) ﾛﾎﾞｯﾄ・ﾎﾞﾃﾞｨ位置時系列ﾃﾞｰﾀ 2) 関節角時系列ﾃﾞｰﾀ 3) 上体動作計算結果 4) 上体・歩容補償動作計算結果 5) ﾓｰﾒﾝﾄ計算結果 6) ｱｸﾁｭｴｰﾀ・ﾄﾙｸ，ﾊﾟﾜｰ計算結果	1) ﾛﾎﾞｯﾄ・ﾎﾞﾃﾞｨ位置時系列ﾃﾞｰﾀ 2) 関節角時系列ﾃﾞｰﾀ 3) 歩容計算結果 4) 上体補償動作計算結果 5) ﾓｰﾒﾝﾄ計算結果 6) ｱｸﾁｭｴｰﾀ・ﾄﾙｸ，ﾊﾟﾜｰ計算結果

Table 4.2 Errors of ZMP trajectory.

	X_{ZMP} 最大誤差[mm]	Y_{ZMP} 最大誤差[mm]	X_{ZMP} 累積誤差	Y_{ZMP} 累積誤差
多項式近似	4．1	0．4	1 1 4．0	5 9．9
微分値近似	3．4	0．3	1 1 3．3	5 9．7

300[mm/step]，1.0[S/STEP]，両脚支持時間1歩の20[%]

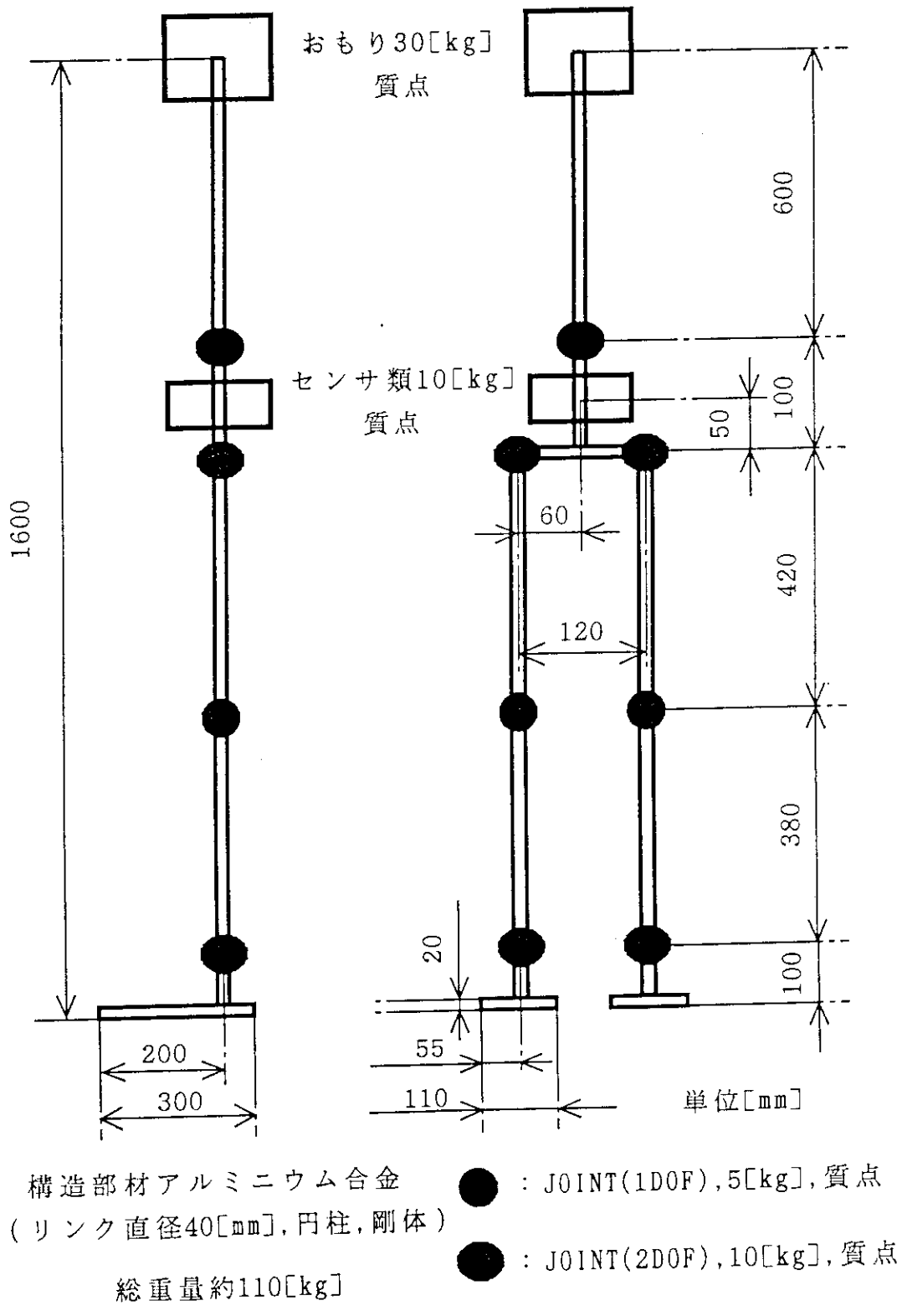


Fig. 4.1 A model of biped locomotion.

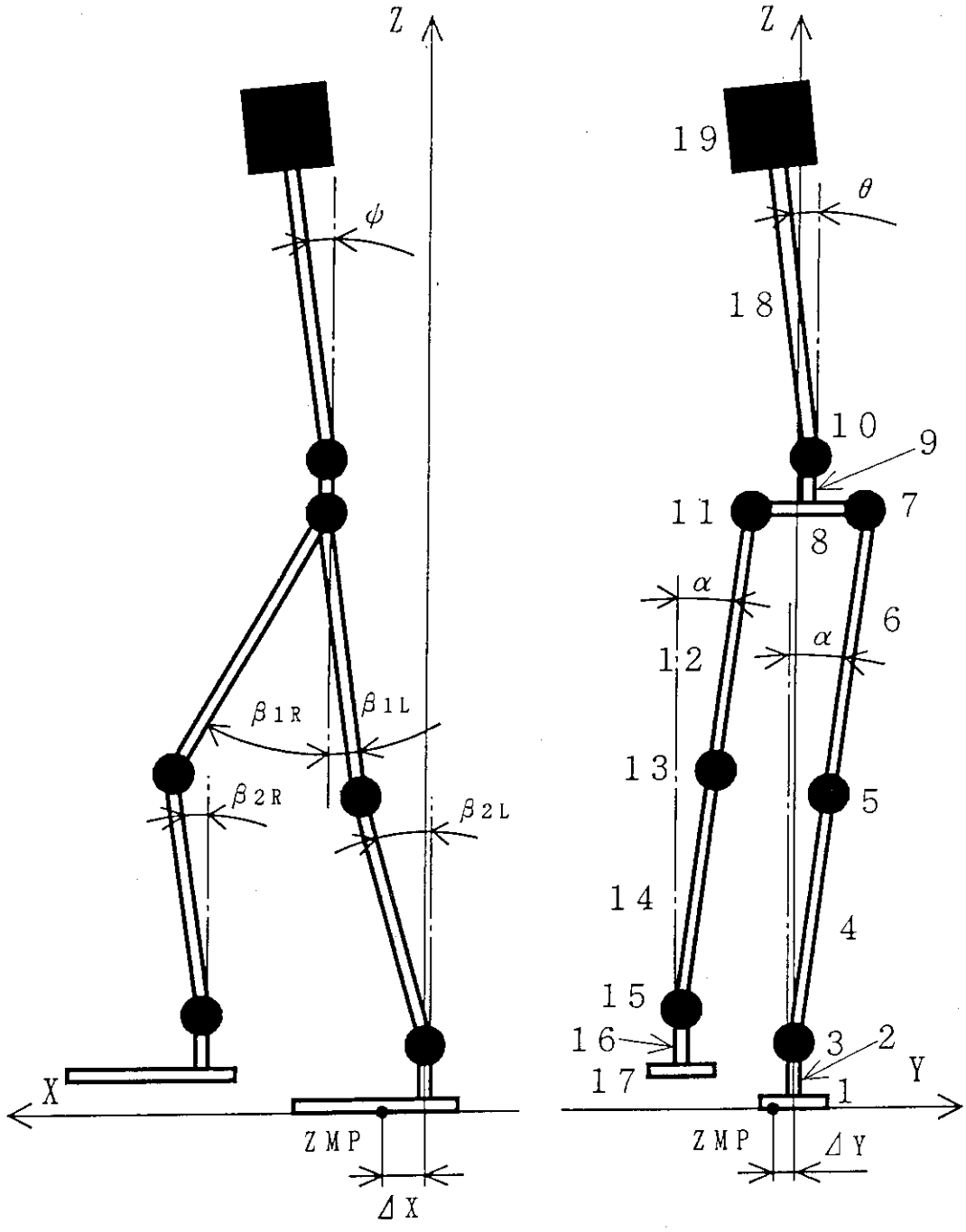


Fig. 4.2 Configuration of robot joints and bodies.

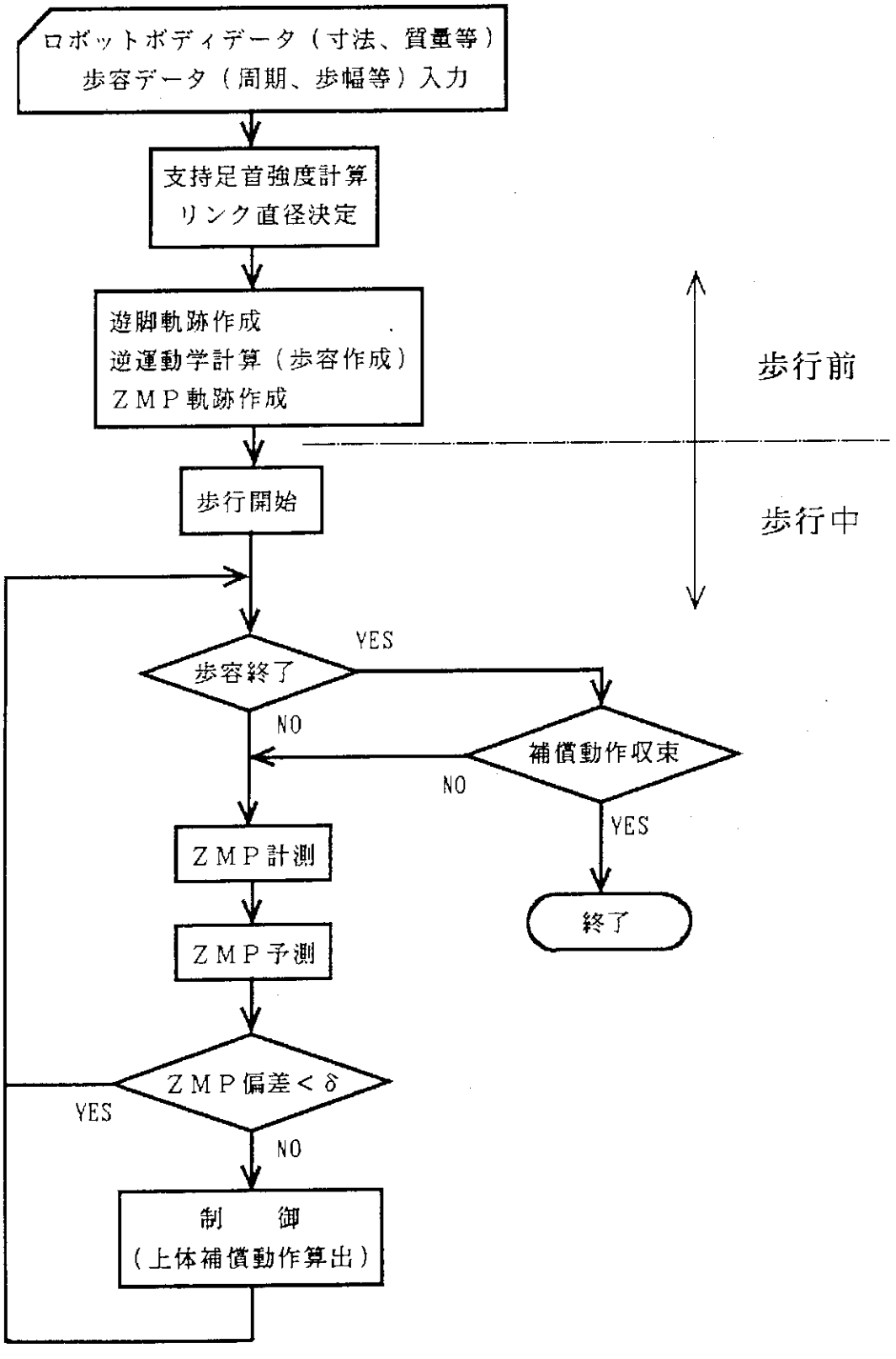


Fig. 4.3 Flow chart of new control method.

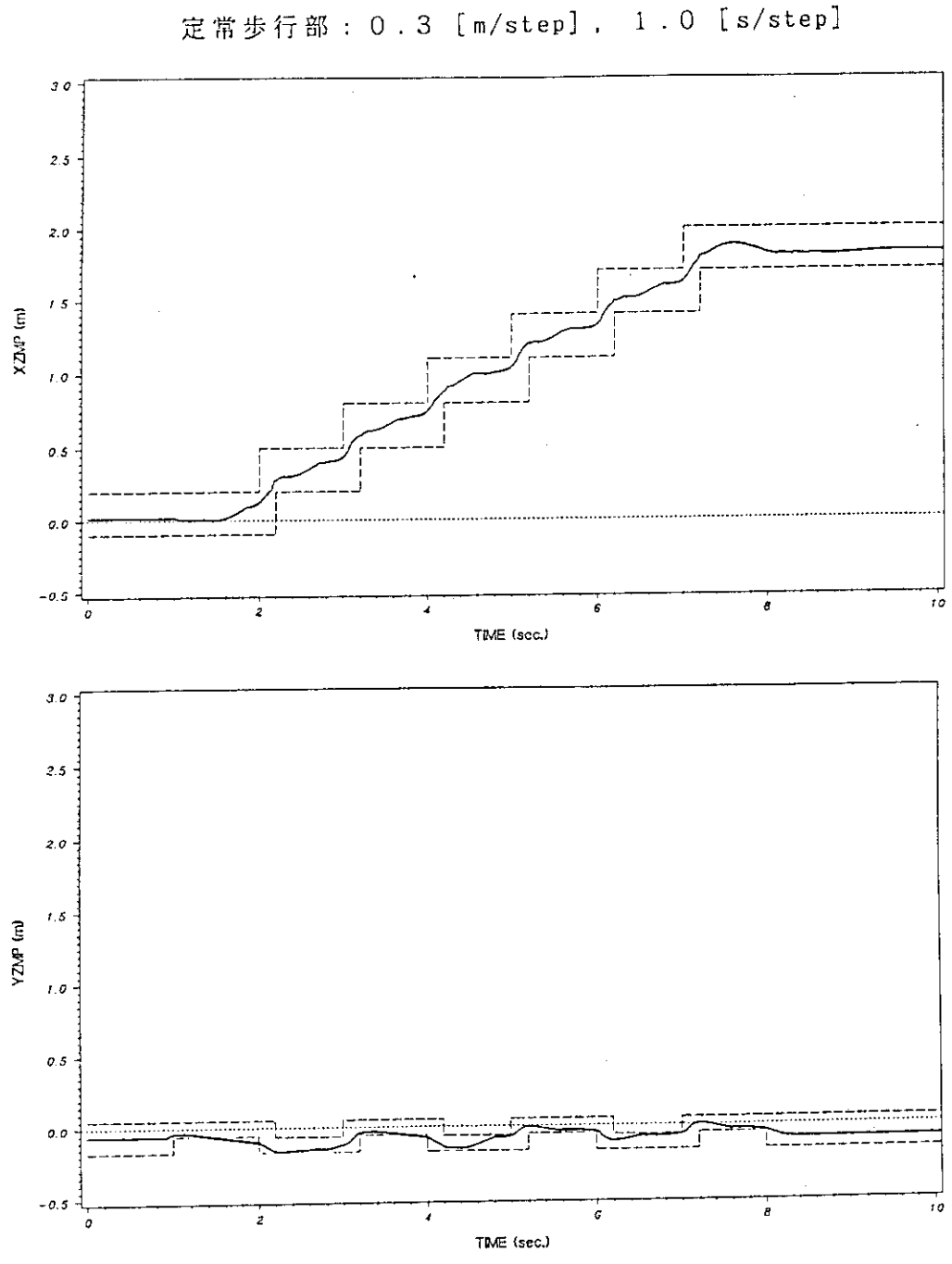


Fig. 4.4 Simulated result based on the new control method.
(ZMP trajectory)

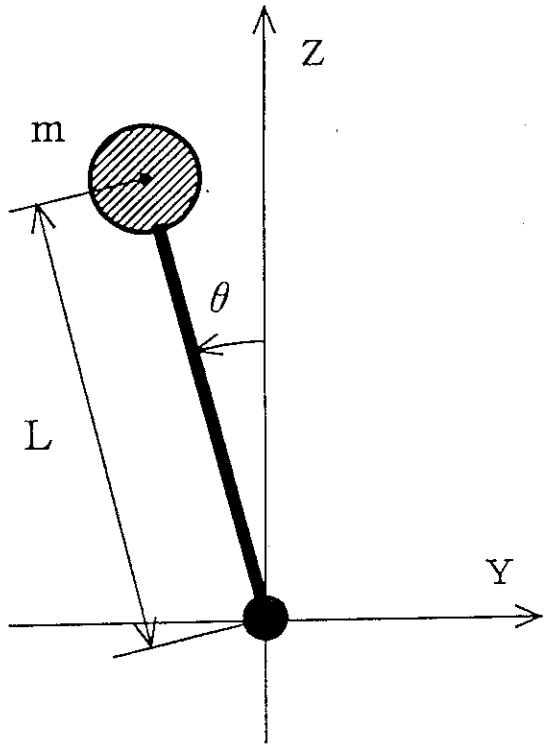


Fig. 4.5 One particle model of robot's trunk.

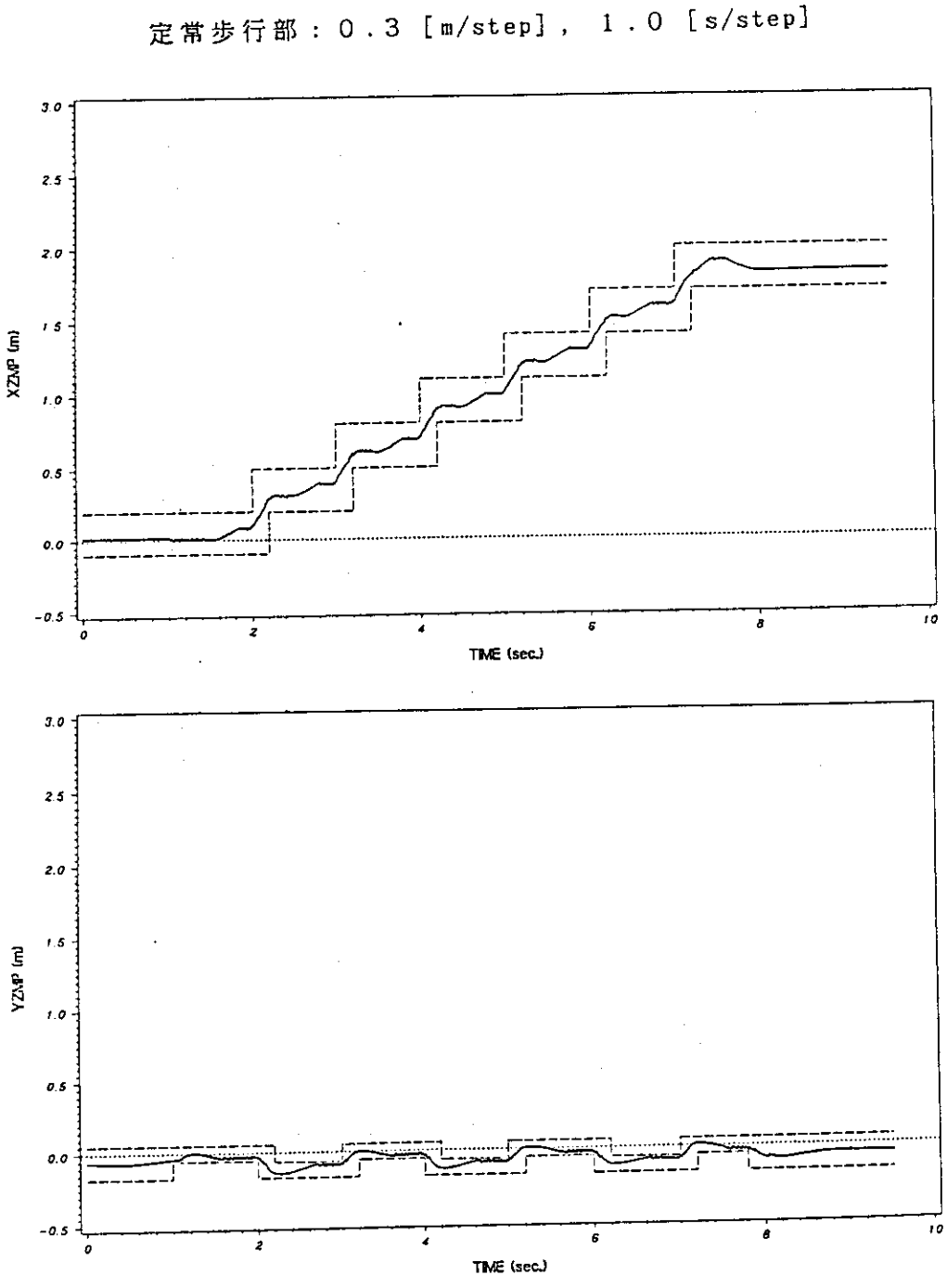
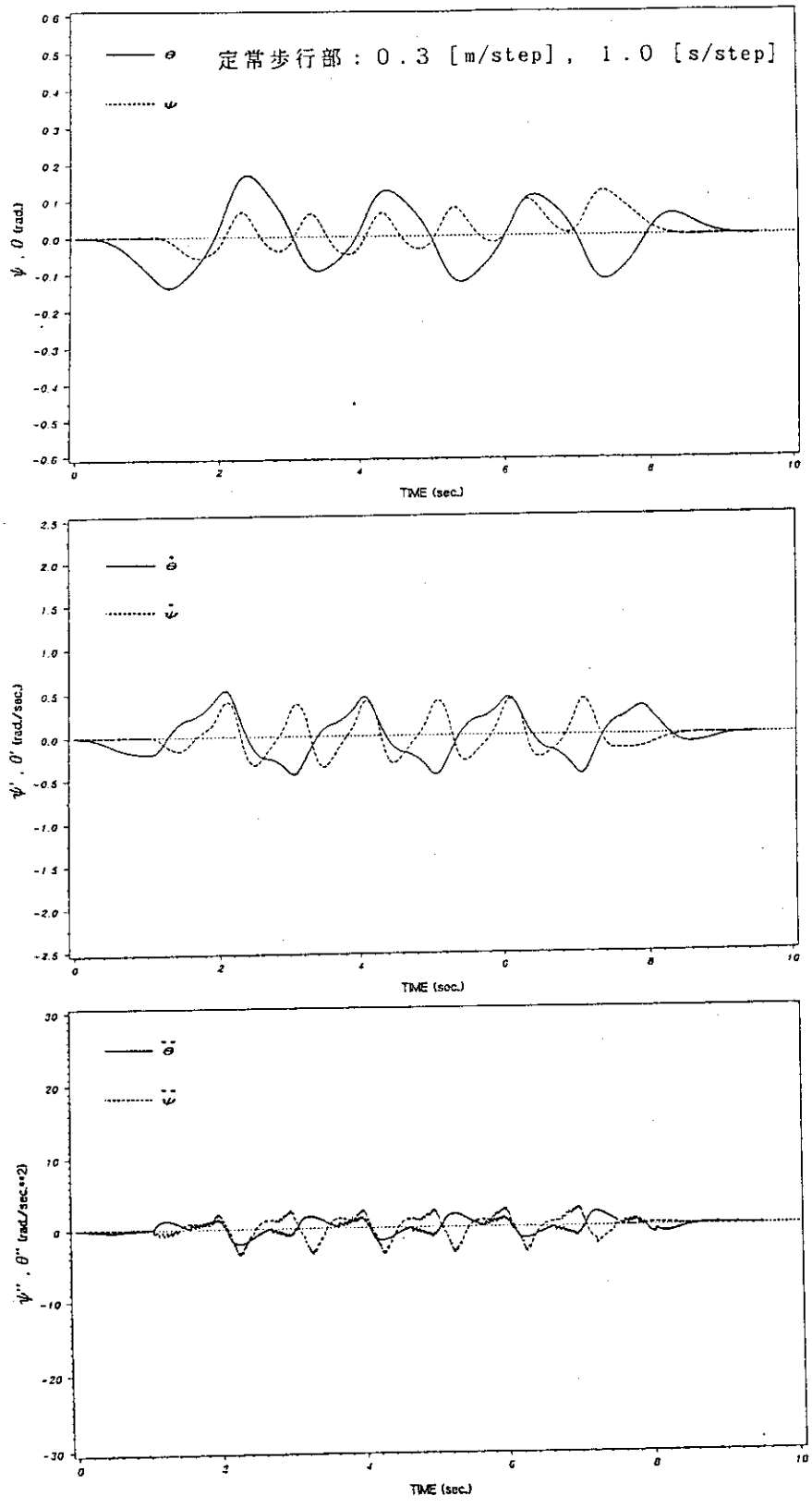
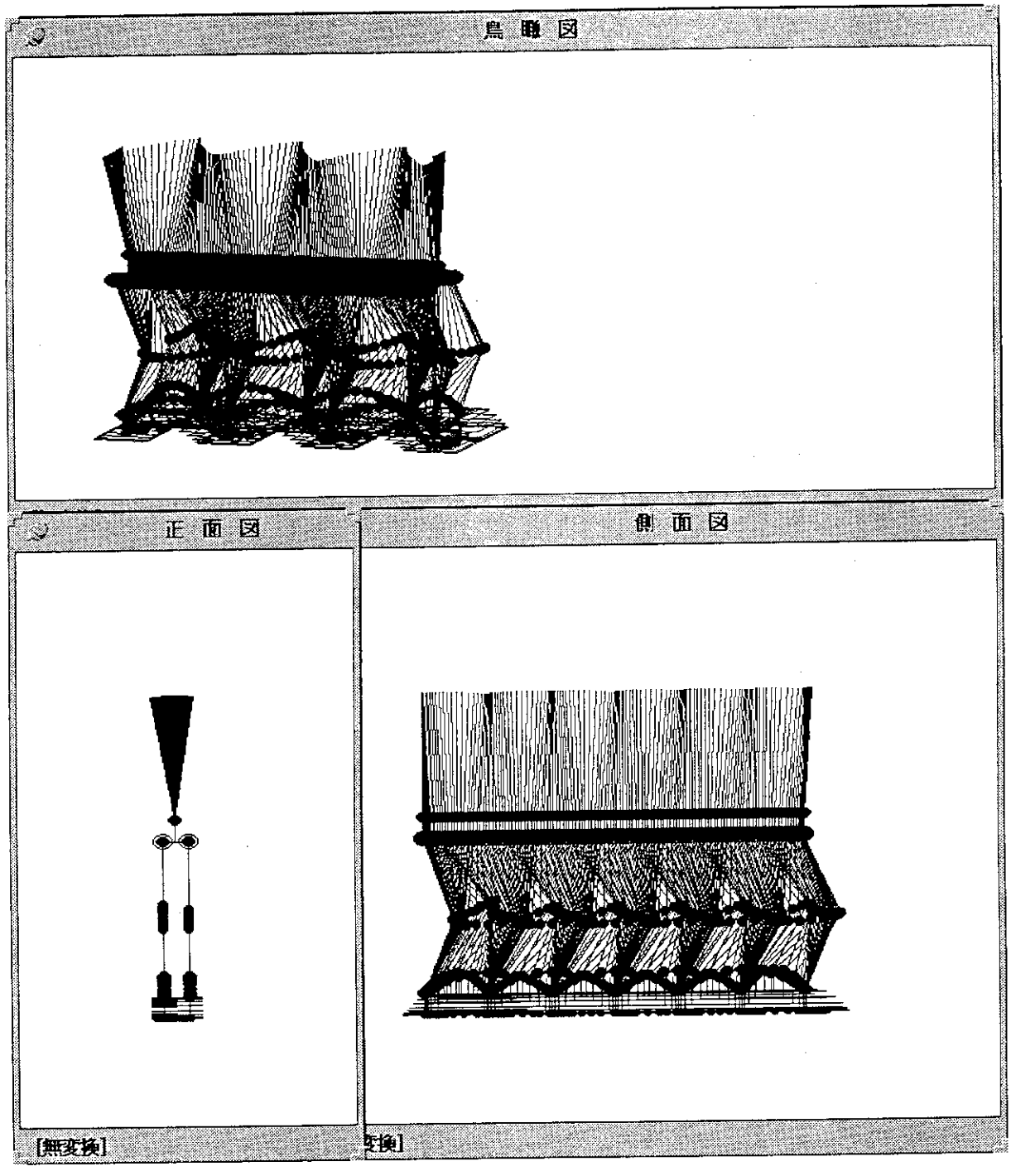


Fig. 4.6 Simulated results based on the new control method.
(0.3 m／step, 1.0 s／step)
(a) ZMP trajectory



(b)

Fig. 4.6 Simulated results based on the new control method.
(0.3 m／step, 1.0 s／step)
(b) Compensative trunk motion



(c)

Fig. 4.6 Simulated results based on the new control method.
(0.3 m／step, 1.0 s／step)
(c) Walking

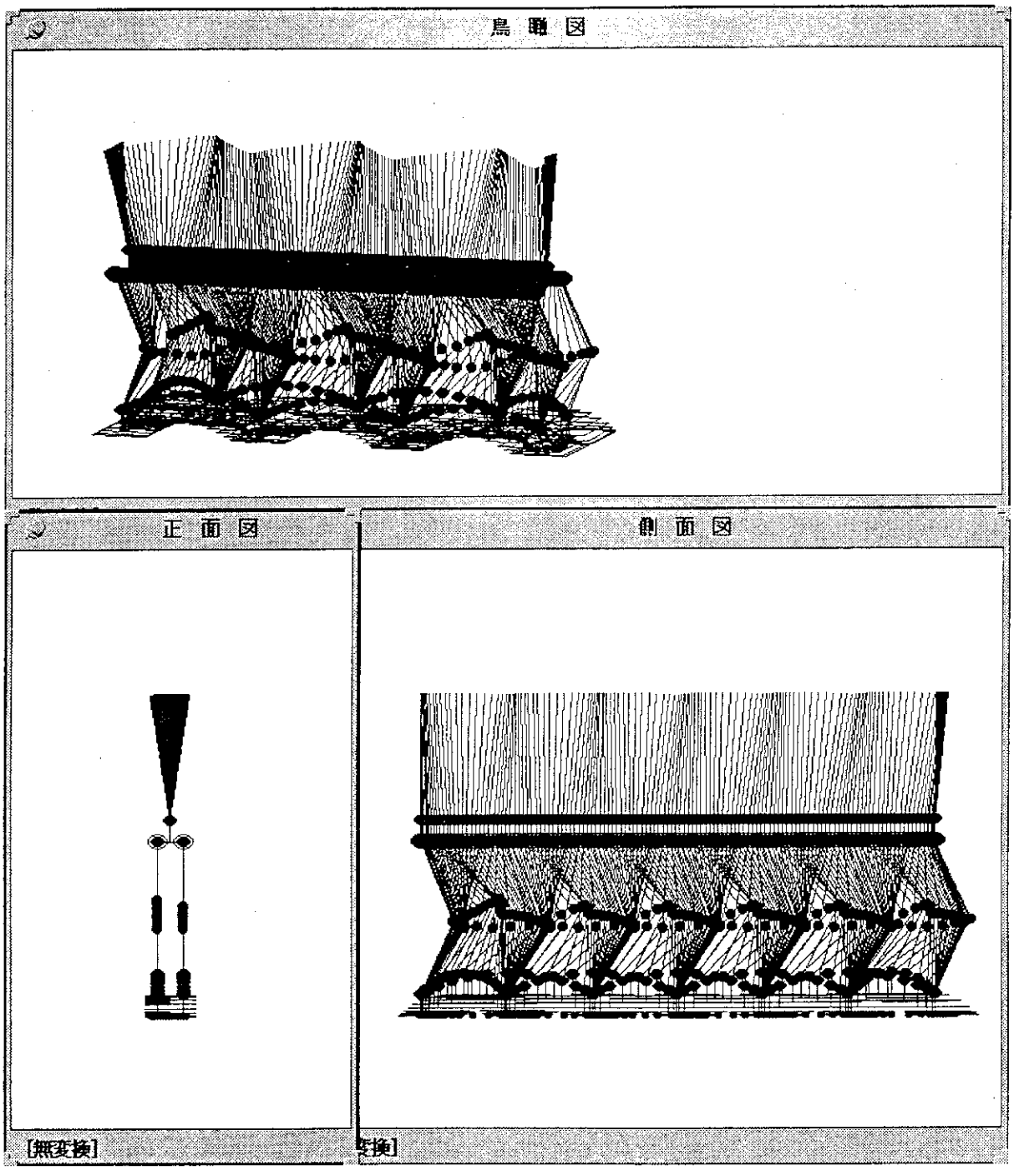


Fig. 4.7 Simulated results based on the new control method.
(0.4 m／step, 1.0 s／step)

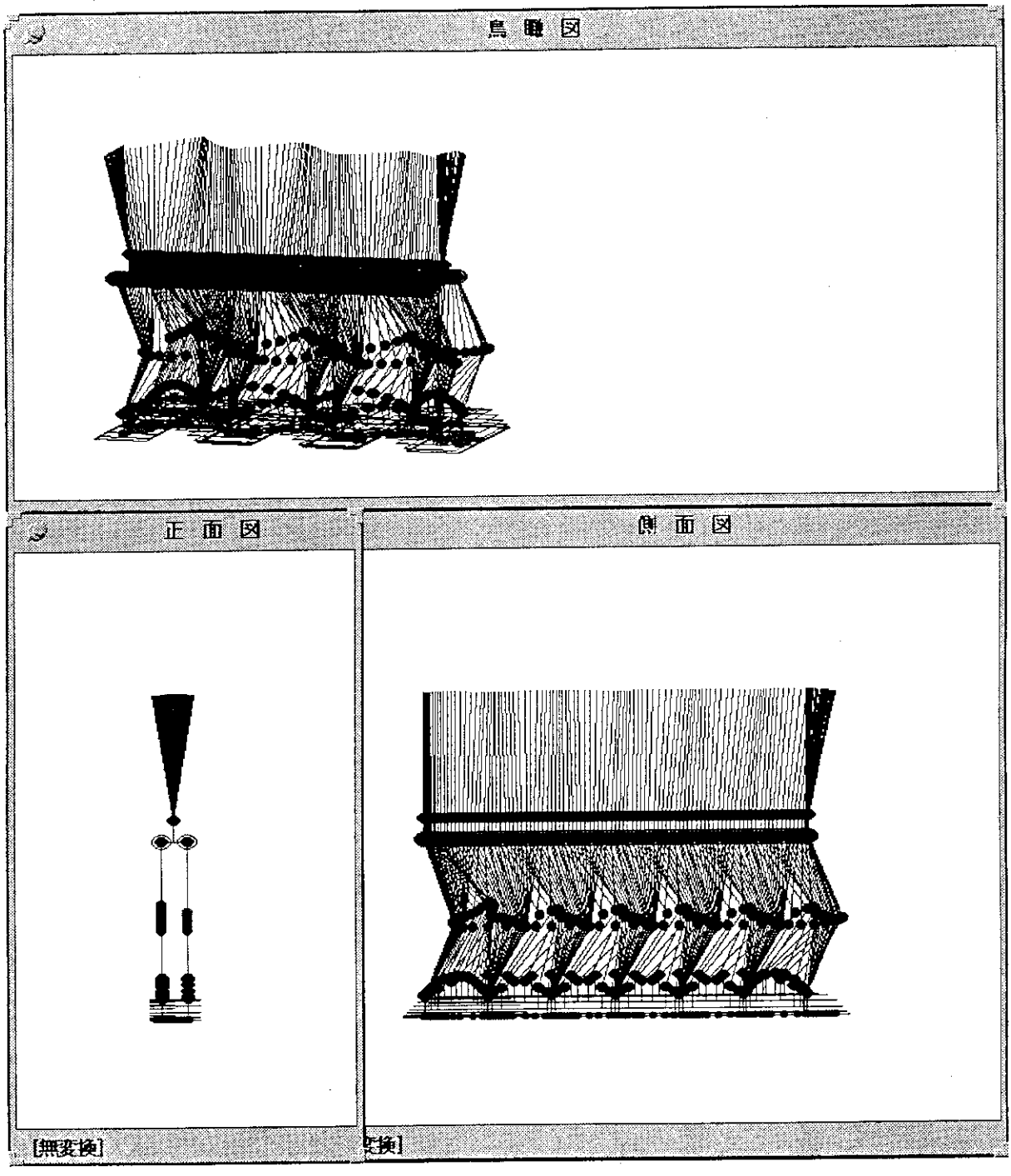


Fig. 4.8 Simualted results based on the new control method.
(0.3 m/step, 0.75 s/step)

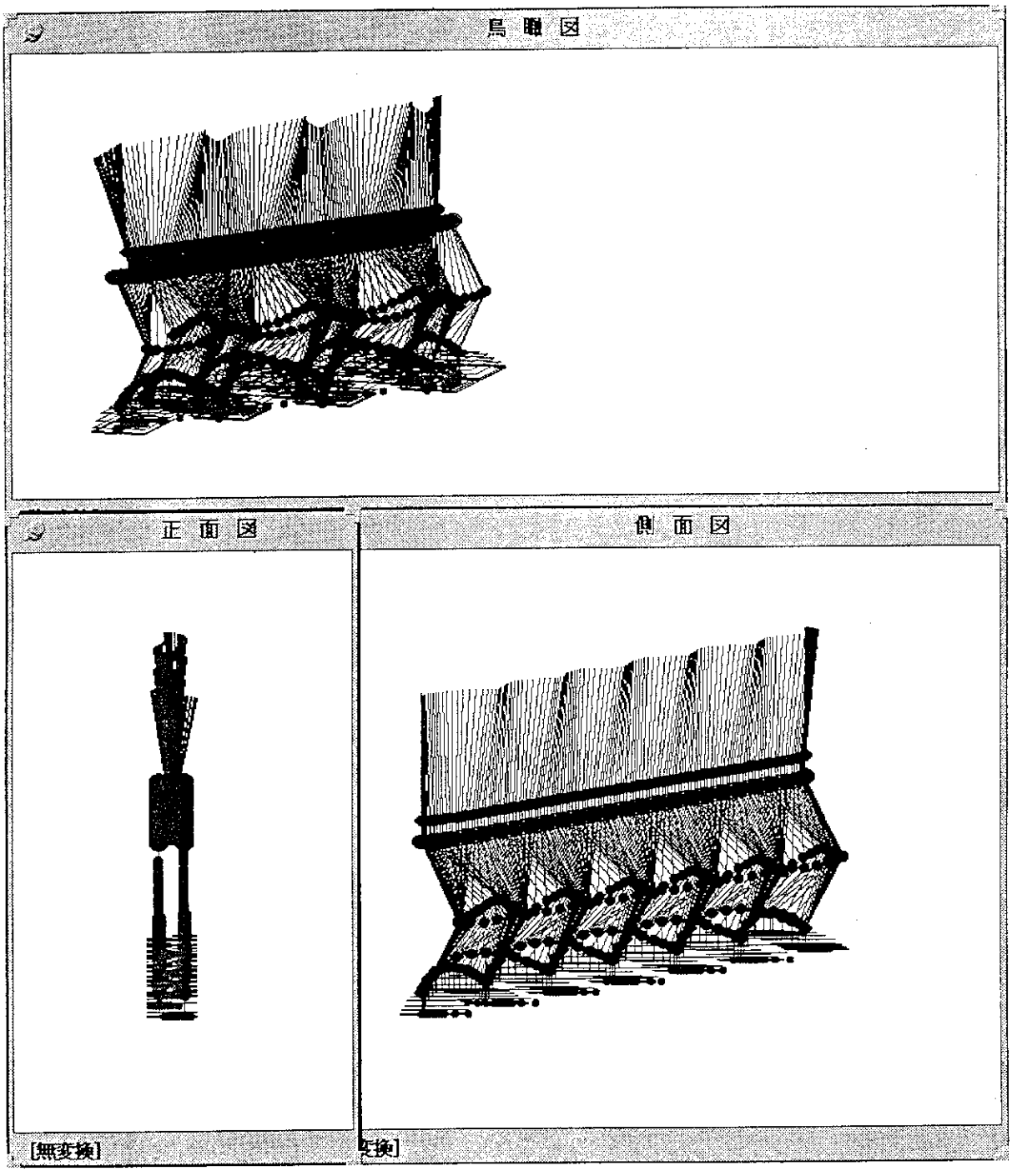


Fig. 4.9 Simulated results based on the new control method.
(Up stairs: 0.05 m, 0.3 m／step, 1.0 s／step)

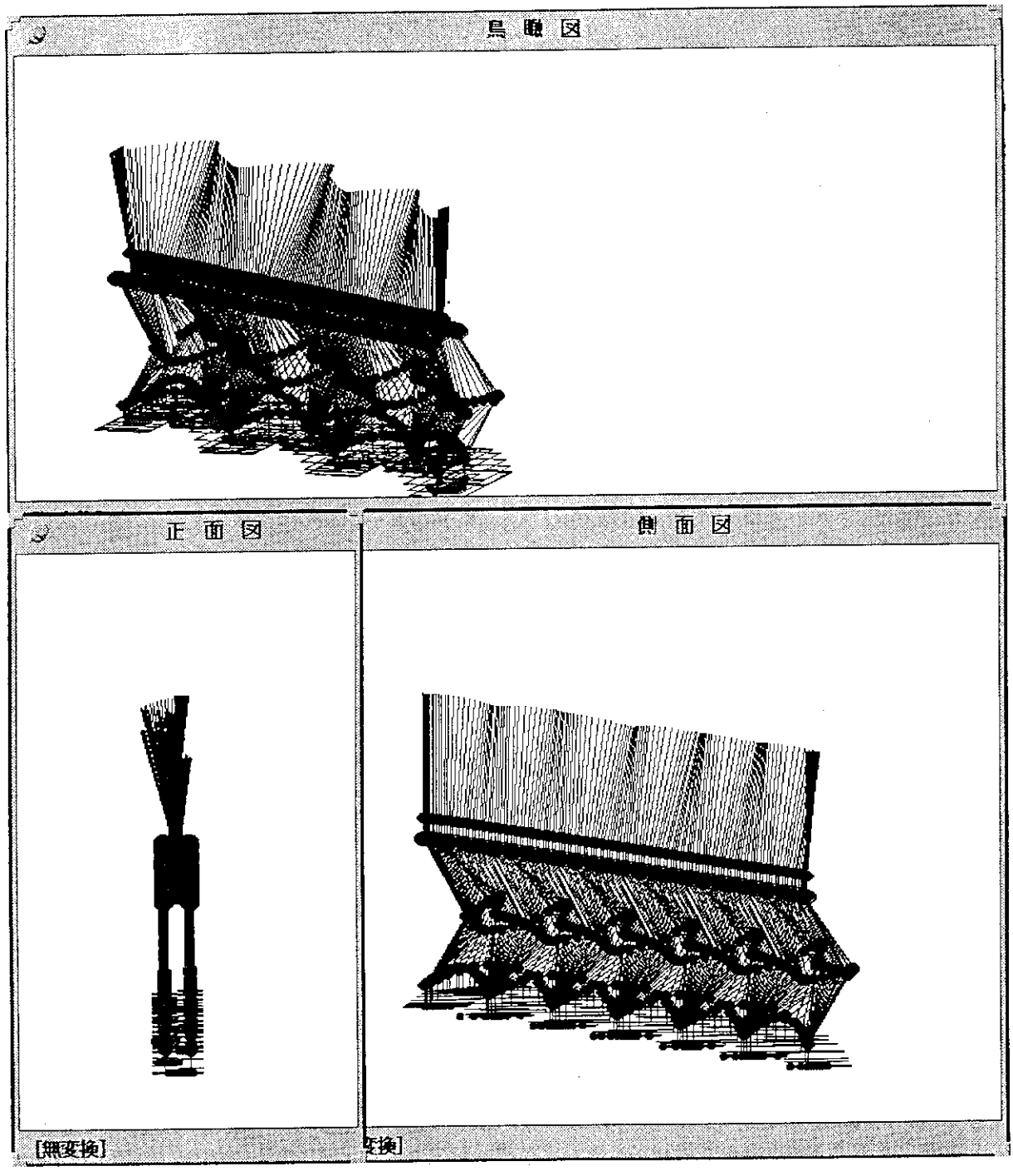


Fig. 4.10 Simulated results based on the new control method.
(Down stairs: 0.05 m, 0.3 m／step, 1.0 s／step)

5. 模擬環境の製作

5.1 模擬環境の必要性

これまで，人工知能の分野においては，(1)視覚認識等，正確な形状あるいは色データを必要とする問題においては，円柱や直方体といった基本立体のみで構成される簡単な環境，所謂，「積木の世界」において，(2)超音波センサに基づく移動実験等，遮蔽物の有無だけが問題となる場合は，机や椅子の点在する研究室や居室において研究開発が行われてきた。これは，現実世界を正確に記述し，計算機データとして登録することが困難なためである。しかし，実用レベルの人工知能技術を開発するために，あるいは，開発した技術の有効性を検証するために，現実世界に即した環境モデルを用いて研究開発を行う必要がある。また，原子力施設における巡回点検ロボットの要素技術開発を目的とする本研究においては，原子力施設内機器の特徴を捕らえた環境において，開発したソフトウェアの性能評価を行うことが望ましい。このような観点から，本研究においては，開発したソフトの性能評価を行うハードウェア実験のため，JRR－3施設地下一階を模擬した環境を整備することとなった。

本模擬環境の製作は，これまで開発したソフトの性能評価，あるいは実用的なソフト開発に必要なデータ収集を行うためのロボット動作環境の整備として位置づけられる。本模擬環境において，平成5年度に導入予定のクローラ走行型ロボットを用いて，ハードウェア実験による各システムの性能評価を行う。また，このロボットに，同じく平成5年度にハードウェア製作予定のセンシング・システムを搭載し，センシング・システムの性能評価及びデータ収集を行い，開発した技術の問題点を把握する予定である。

5.2 模擬環境製作における必要条件

本模擬環境は，以下の留意点に基づき設計・製作された。

(1) 形状及び色データの把握

言うまでもなく，知的ソフト開発のためのロボット動作環境は，その環境内のすべての物体に関する形状及び色データが計算機に登録されていることが必要不可欠である。しかし，現実世界に存在する物体を計算機データとして正確に記述することは，多くの場合，非常に困難である。そこで，本模擬環境の設計においては，図面作成に先立ち施設形状データベースを用いて概念設計を行った。次に，施設形状データベースに登録された形状データを用いて，製作する機器を記述した。即ち，製作した機器は，既に施設形状データベースに環境モデルとして完全に記述されている。したがって，ケーブルのように計算機データとして記述不可能な三次元物体は製作していない。

(2) 製作する機器類の選択

本模擬環境は，原子力施設内機器の特徴を含む機器で構成される。機器の形状，色及び配置

については，JRR－3 施設地下一階一次区画を参考にした。製作した模擬機器は，以下のとおり。

① CRS イオン交換樹脂塔ユニット，② 溢流タンク，③ SFS 熱交換器，④ SFS ポンプ，⑤ PCS 主ポンプ，⑥ PCS 補助ポンプ，⑦ RCS 前置フィルタ，⑧ SFS 前置フィルタ

また，原子力施設巡回点検ロボットに必要な移動機構の検討や走行制御における問題点把握のため，段階や原子力施設において水漏れ事故に備えて設置された堰等が必要である。このため，上記機器類に加え，JRR－3 施設地下 1 階エレベータ前の段階と一次区画遮蔽扉（跨ぎ越え部分）を製作した。さらに，キャタピラあるいは車輪等の路面接地部分の滑りによるデッド・レコニングの精度及び走行制御に与える影響を検討するため，不整地として，グレーチングを配置した。

(3) 機器の表面仕上げ

製作した模擬機器表面の色及び反射の性質（散乱，鏡面，金属面等の質感）は JRR－3 施設に設置された実物と同等とするため，マンセル票による比較を行い，機器の表面仕上げに用いる塗料の色を決定した。

(4) 照　　明

照度不安定な室外光は，視覚認識処理に多大な影響を及ぼす。また，通常，原子力施設には外部環境との遮蔽を考慮し，窓は配置されていない。以上の理由から，模擬環境においては，室外光を完全に遮蔽し，照明は，蛍光灯のみとした。この照明も，実環境において照明として用いられている蛍光灯と位置（設置高さ）及び照度等の照明条件は同等である。すなわち，光源の形状及び照度分布は実環境と同等である。

(5) 床　　面

床面の表面仕上げは，機器同様，色及び反射率を実施設と同等のものにするため，同じ塗料を用いて表面を仕上げた。

床面の強度は，約 1,000 kgの荷重に耐えられるフリーアクセスである。

5.3 製作方法

(1) 機 器 類

タンクやポンプ類（台座も含む）は，合板を加工して製作した。イオン交換樹脂塔は，鉄骨と塩化ビニール管を用いて製作した。

(2) グレーチング

グレーチングは実施設において使用されているものと同一である。

(3) 段　　階

段階は実物（JRR－3 施設地下一階エレベータ前）と同じ構造部材を用いて製作した。ただし，第一段目は，実物がコンクリートのため，厚さ 3 mmの鉄板を用いて製作した。

(4) 照　　明

模擬環境に設置された蛍光灯は，実環境において用いられているそれと同一のものであり，その位置も同等である。製作後，照度計を用いて床面における照度が実環境と同等であること

を確認した。

5.4 製作した模擬環境

Fig. 5.1 に，製作した模擬環境の設計図を示す。また，Fig. 5.2 に，CG によって可視化された模擬環境の三次元画像を示す。

前述したように，本模擬環境内機器の形状及び色（反射の性質も含む）データは，すべて施設形状データベースに登録済である。Fig. 5.3 に，製作した模擬環境の CCD カメラによる実写画像と CG による画像の比較を示す。この CG は，通常ワークステーションにおいて三次元画像生成に用いられるZ－バッファ・アルゴリズムによるものではなく，光の反射をより忠実にシミュレーションする光線追跡法によるものである。2つの画像を比較することにより，本模擬環境の形状及び色データが施設形状データベースにおいて正確に記述されていることがわかる。

5.5 今後の予定

今後，本模擬環境において，CCD カメラやレンジ・ファインダを用いた視覚処理を想定し，センシング・システムによる収集を行い，このデータを用いて自己位置あるいは環境認識のためのソフトウェア開発を行う。実データを用いることによって，カメラのノイズ，色及び画面解像度の影響，シミュレーション不可能なハイライト及び影の視覚処理における影響を検討することが可能である。

また，本模擬環境において，平成5年度に導入予定のクローラ走行型ロボットにセンシング装置を搭載し，人間動作シミュレーションにおける命令理解等の各システムを統合した総合実験を行い，開発したソフトウェアの性能評価を行う予定である。

本模擬環境に関する形状及び色データ等は，データベース化されているため，上記ハードウェアを用いた実験だけでなく，2足歩行ロボット（まだ実際に製作はされていない）の動作シミュレーションがワークステーション上で可能である。このシミュレーションにおいては，施設形状データベースの開発として平成4年度に開発した実時間表示システムにより，ロボット運動の動力学的計算結果，行動計画，干渉チェック等の移動ロボットに必要な処理結果が確認可能である。

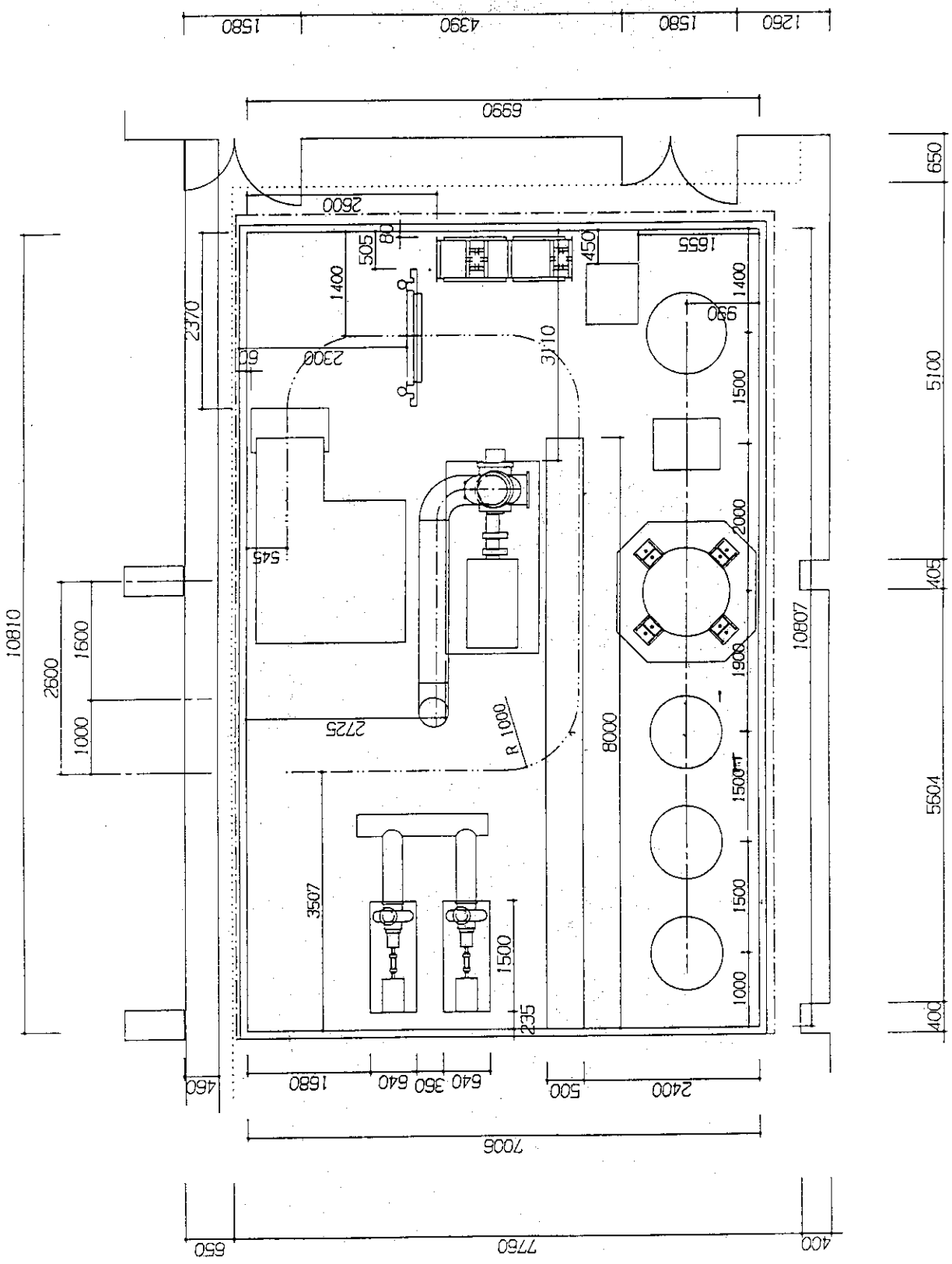


Fig. 5.1 Design drawing of the mockup enbironment.

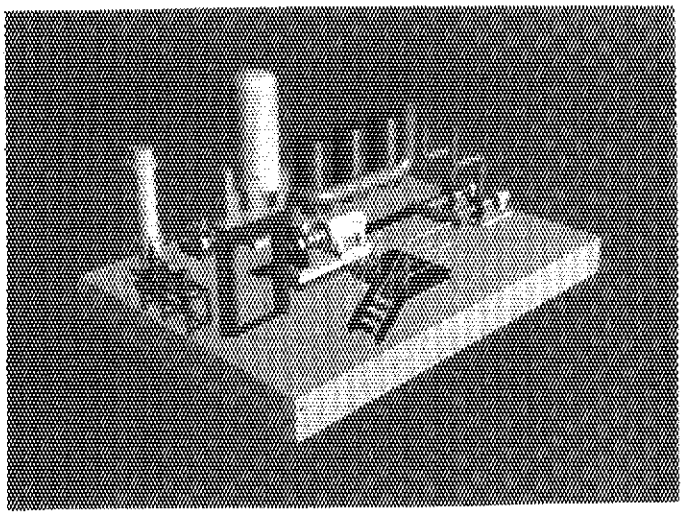


Fig. 5.2 3-D image of the mockup enbironment.

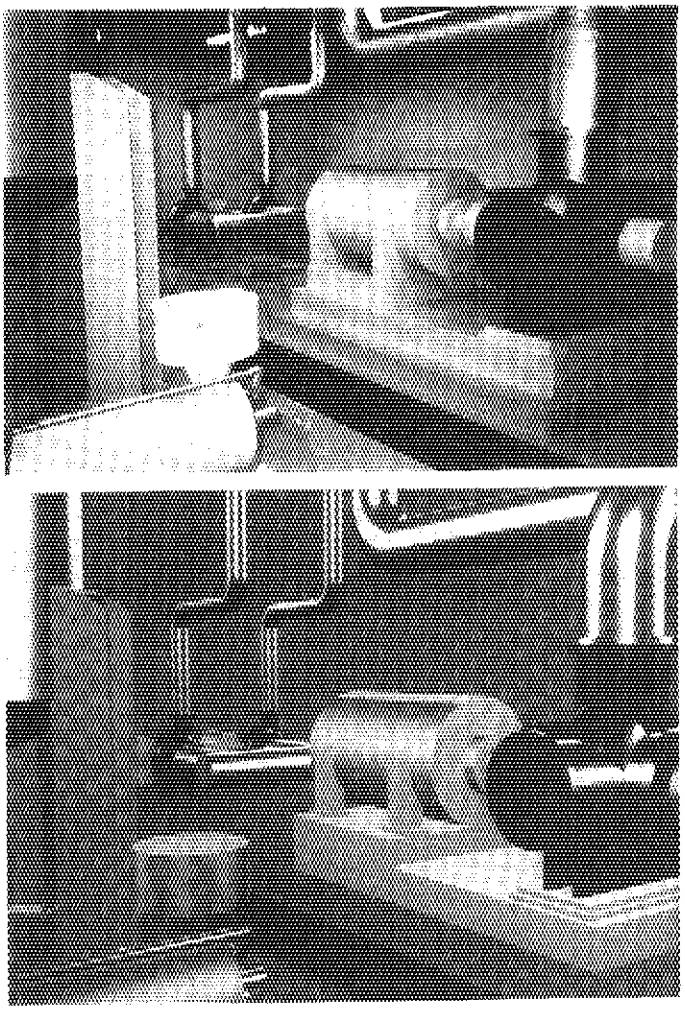


Fig. 5.3 Comparison between computer graphics and photo of the mockup enbironment

6. 施設形状データベース

6.1 はじめに

人間動作シミュレーション技術の研究において，施設形状データベースは，巡回点検を行う知能ロボットの動作環境（世界モデル）を設定する他，シミュレーション結果を可視化する役割を持つ。これらの機能開発として，平成4年度は，以下の作業を行った。

(1) 世界モデルの構築

本データベースにおいては，施設の建屋，機器等の三次元物体は，CSG（Constructive Solid Geometry）モデルによって記述されている。CSGとは，三次元の基本立体の集合演算によって物体を記述する手法であり，そのモデルは，点と線からなるCADデータ等の形状データとは異なり，三次元情報が縮退していない。このため，本研究において，6.2で述べる種々の形状データを生成するためのSourceとして，あるいは，狭隘な空間においてロボット躯体と施設内機器との詳細な干渉チェックを行うための形状データとして，CSGモデルにより三次元物体を記述している。しかし，CSG表現による三次元モデルは，二次曲面が複雑に組み合わされた構造となるため，高速処理を必要とする行動計画やシミュレーション結果の可視化には，不適当である。平成4年度は，CSGモデルから，ロボット行動計画の高速処理のためのボクセル・マップ及び三次元画像の高速生成を実現するためのポリゴン・データ生成機能を開発した。

① ボクセル・マップ生成機能の開発

CSGモデルとして記述されているJRR－3施設等に関する形状データから，ロボット行動計画のためのボクセル・マップを生成する機能を開発した。ボクセル・マップの説明を6.2(2)に，生成アルゴリズムの概要及び変換プログラムの処理内容を6.3に示す。

② ポリゴン・データ生成機能の開発

CSGモデルとして記述されているJRR－3施設等に関する形状データから，可視化のためのポリゴン・データを生成する機能を開発した。6.4に，ポリゴン・データを生成する変換アルゴリズムの概要を示す。

(2) 実時間表示機能の開発

これまで，本データベースにおいては，光線追跡法と画像生成用並列計算機CAP（Cellular Array Processor）により，シミュレーション結果の可視化のための三次元画像生成を行ってきた。しかし，このシステムにおいては，1枚の画像を生成するために数十分の計算時間を要する。即ち，数分の動画を生成するために，数百～数千時間が必要であった。平成4年度は，可視化における，この膨大な計算時間を短縮するため，ワークステーションを用いた三次元画像生成機能を開発した。ここでは，上記②において生成したポリゴン・データを用いて，ワークステーション上で知能ロボットの動きを動画として実時間表示する機能を開発した。6.5に，ポリゴンの集合体として表現された三次元物体の可視化アルゴリズムの概要を示す。

6.2 施設形状データベースにおける種々の形状データ

人間動作シミュレーション技術の研究において，命令理解や視覚認識等の各システムは，使用目的に応じて次の４つの形式データが利用可能である。

- MORSE形式のソリッド・モデル
- ボクセル・マップ
- 簡易モデル
- ポリゴン・データ

① MORSE形式のソリッド・モデル

前述したように，CSGとは，三次元の基本立体の集合演算によって物体を記述する方法である。人間動作シミュレーション技術の研究においては，粒子輸送モンテカルロ・コードを用いて，ロボット躯体に対する放射線損傷度を評価する。通常，粒子輸送モンテカルロ・コードにおいては，体系中の領域は，三次元の基本立体の集合演算によって記述されている。このため，人間動作シミュレーション技術の研究においても，ロボットの動作環境を同等の表現法によって記述する必要がある。このCSGモデルは，上記目的だけでなく，ロボット躯体と環境中の物体との詳細な干渉チェックにも有用である。また，以下に述べる種々の形状データを本モデルによって一元的に管理している。

② ボクセル・マップ

ボクセル・マップとは，二次元／三次元の格子状マップであり，各ボクセル（三次元の直方体セル）中に存在する機器IDが登録されている。命令理解システムでは，大まかな行動計画を行う際，このボクセル・マップを用いてロボットの可動空間を検索している。

③ 簡易モデル

簡易モデルとは，各機器を包含する球や円筒といった基本立体によって，その機器の形状を代表させるモデルである。本モデルは，大まかな干渉チェック，あるいは，解析が困難なロボット躯体の揺れ等を考慮し，十分に安全な可動範囲を高速に決定するための干渉チェックに有用である。

④ ポリゴン・データ

これまで，本研究において可視化のために用いてきた光線追跡法は，新鮮な画像を得ることができる，という長所を持つ反面，処理に多大の計算時間を要するとうい点が欠点である。三次元物体を高速に可視化する手法として，画像生成用の専用ハードウェアが開発されているZバッファ・アルゴリズムを用いて，ポリゴン・データをワークステーション上で可視化する手法がある。本データベースにおいては，この可視化処理のため，ポリゴン・データをCSGモデルから生成している。近似の度合い，即ち，物体表現に用いられるポリゴンの枚数は可変である。

6.3 ボクセル・マップ生成機能の開発

以下にボクセル・マップ生成アルゴリズムの説明を示す。

(1) ボクセル・マップ生成アルゴリズムの概要

まず，各ボクセルを構成する頂点と環境を構成するすべての三次元物体（領域）との包含関係を調べる。ボクセルを構成する8点のうち少なくとも1点がある三次元物体iに含まれれば，物体iは，そのボクセルを占有する。各ボクセルに対して，この処理を行い，ボクセル・マップを作成する。ボクセルを構成する各点が，ある物体に含まれるか否かの判定は，次の手法によって行う。

(2) 点と三次元領域の包含関係の検出方法

基本立体の集合演算で表現される三次元領域は，各基本立体を構成する曲面に対する論理式で表現することができる。いま，判定の対象となる領域表現を

a ∪ b ∩ c ∩ d ∪ e（a～eは，符号付き曲面番号）

とする。(3)において後述する関数を用いて，対象となる点を各曲面が作る半空間との関係を調べ，次のような論理式を作成する。この論理式における各要素は，各曲面に関する判定結果を示す。即ち，点が問題点の半空間に含まれる時，1，それ以外の時，0である。

0 ∪ 1 ∩ 1 ∩ 1 ∪ 0

上記論理式に対する評価を行い，対象となる点がその領域に含まれるか否かを判定する。論理式の評価においては，交わり（∩）部分の演算を合併（∪）部分の演算に優先して処理する。この処理において使用する演算は，交わり部分に対しては，論理積，合併部分に対しては，排他的論理和である。上記論理式の評価をFig. 6.1に示す。また，円筒に対する包含関係評価の例を次に示す。

［円筒に対する包含関係評価の例］底面の中心（0，0，0），底面の半径10，中心軸がZ軸に平行，高さ20の円筒は，以下3つの曲面に分解される。

$g_1(x, y, z) = z - 20$

$g_2(x, y, z) = X^2 + Y^2 - 100$

$g_2(x, y, z) = z$

ここで，a，b，cをそれぞれg_1，g_2，g_3で表される曲面の番号とすると，三次元領域iの内部は，以下のように表現される。

－a ∩ －b ∩ c

ここで，上記表現に対する論理式を後述の関数を用いて作成し，その論理式を評価する。

(3) 点と曲面との位置関係を調べる関数

① 関数仕様

機能：点r（r＝（x，y，z））が，曲面iによって作られる半空間に含まれるか否かを判定する。含まれる場合，f＝1，含まれない場合，f＝0

引数：rの三次元座標，曲面番号i（符号を含む）

② 球面に対する関数の例

対象となる点をr_1，aを球面内部を表現する符号付き曲面番号，h（r）を半径R_1，中心位置C_1の球面の式とする。つまり，

h（r）：｜r－c_1｜－R_1（球）

である。曲面aによって作られる半空間と点との包含関係を調べるために，次のブール関

数を考える。

$g(r_1) = \mathrm{sign}(a) * \mathrm{sign}(h(r_1))$

ここで，$\mathrm{sign}(X) = 1 (X \geqq 0)$

$= -1 (X < 0)$

である。ここで，

$f(r_1) = (g(r_1) + 1) / 2$

とすると，関数fは，上記①の機能を満足する。

6.4 ポリゴン・データ生成機能の開発

以下に，施設形状データベースにCSGモデルとして登録された形状データからポリゴン・データを生成する変換アルゴリズムの概要を示す。また，変換プログラムにおける詳細な処理内容を付録1に示す。

(1) ある物体を構成する各基本立体を複数のパッチ（三次元空間における有限な二次曲面）に分解する。（例：円筒は2枚の平面パッチと1枚の二次曲面パッチに分解される。）

(2) 各パッチを近似する複数のポリゴン（三角形）を生成する。

(3) 基本立体間の論理演算を行った結果，物体表面に位置するポリゴンを決定する。各ポリゴンを構成する3つの頂点のうち少なくとも1点が，物体を構成する正の負号を持つすべての基本立体に包含されず，かつ論理式を満足する時，そのポリゴンは物体表面に位置するポリゴンであり，その他の場合，そのポリゴンは物体表面に位置するポリゴンではない。点と基本立体との包含関係の判断には，上記6.3(2)の手法を用いる。

(4) あるポリゴンを構成する3つの頂点のうち，1点または2点が物体表面に位置する場合，1枚または，2枚の新たなポリゴンによる再構成を行う。

6.5 ポリゴン・データ可視化アルゴリズム

上記アルゴリズムによって生成されたポリゴン・データ，即ち，ポリゴンの集合体として表現された三次元物体の可視化アルゴリズムの概要について述べる。このアルゴリズムは，Zバッファと呼ばれるポリゴン・データ可視化アルゴリズムで，殆どのワークステーション上で専用ハードウェアにより高速化されている。処理プログラムは，C言語及びPHIGS等のグラフィック・ライブラリで記述されている。ここでは，各ポリゴンに対して，以下の処理が行われる。

(1) ポリゴンをスクリーン上に投影する。三次元空間内のポリゴンをスクリーンに対応する二次元平面にアフィン変換した時，そのポリゴンが，スクリーンの外にある場合，処理を終了する。

(2) 視点から，そのポリゴンまでの距離を計算し，その距離が，既にデプス・バッファ（注1参照）に書き込まれた距離より短い時，デプス・バッファにその値を書き込み，次の処理に移る。長い時，そのポリゴンに対する処理を終了する。

(3) ポリゴンがスクリーン上で占有する画素に対する輝度を計算し，その値をフレーム・バッ

ファ（注2参照）に書き込む。

注1）デプス・バッファ：各画素に対する距離情報が，書き込まれたバッファ。初期値として，無限遠点に相当する値が書き込まれている。

注2）フレーム・バッファ：各画素に対する輝度値が，書き込まれたバッファ。

6.6 可視化システムの性能評価

5章において述べた模擬環境の形状データを用いて，開発した可視化システムの性能評価を行った。ここでは，5章で述べた模擬環境における2足歩行ロボットの動きを可視化した。ロボットの動きは，4章で述べた2足歩行運動学システムにおいてシミュレーションされたものである。使用計算機は，SUN－4／10　モデル30 GT（主記憶128MB，ディスク4.4 GB，スワップ260 MB，グラフィック・アクセラレータ付き）である。このシステムを用いて，約5,000枚のポリゴン・データによって表現された模擬環境とロボットを動画表示した。出力された画像は以下のとおりである。

- 画面解像度：横1024 × 縦1024
- RGB解像度：各8ビット
- 動画：1秒あたり10枚の画像をダブル・バッファを使用し，連続表示
- 描画アルゴリズム：Zバッファ
- 形状データ：ポリゴン（三角形）
- 陰影づけモデル：Gouraud または Phong

Table 6.1に模擬環境及びJRR－3施設地下一階の形状データを用いて行ったシステムの性能評価結果を示す。この表において，CSGモデルの詳細度における簡易形状とは，描く画時間を短縮するため簡単なCSGモデルを用いた場合を，ポリゴン・データへの変換において簡易近似とは，同じく描画時間を短縮するため少ない枚数のポリゴンによって物体を近似した場合を示す。表における描画時間が，画像1枚を生成・表示するために必要な時間である。

6.7 おわりに

(1) 平成4年度の結果

平成4年度に行った本データベースの機能開発の概要とその有用性は以下のとおり。

① ボクセル・マップ生成機能の開発

ロボット行動計画に用いられているボクセル・マップ生成機能を開発した。このボクセル・マップは，命令理解システムにおいて，ロボットの静的かつ大まかな行動計画に実際に用いられ，処理の高速化に貢献している。

② ポリゴン・データによる可視化機能の開発

CSGモデルからポリゴン・データを生成し，これを用いてワークステーション上で動画表示する機能を開発した。この機能開発により，ロボットの動きの実時間表示が可能になった。動画表示でなく，静止画像を表示する場合，ポリゴンの数を大きくし，詳細に近似され

た画像も表示可能である。

(2) 今後の計画

平成４年度までの機能開発により，使用目的に応じた種々の形状データが整備され，実時間表示機能も利用可能になった。即ち，人間動作シミュレーション技術の研究における施設形状データベースの基本機能が整備された。今後は，各機能の性能向上，あるいは，画像生成技術における新手法の開発を目的とした研究を行う。平成５～６年度は，その一つとして，光線追跡法による三次元画像生成システムをモンテカルロ装置，Monte－４上に構築する予定である。Monte－４は，７章に述べるように，粒子輸送モンテカルロ・コードを高速化するために開発されたベクトル・並列計算機であるが，本装置に搭載された特殊なハードウェアは，直線と三次元領域との交差点を求める幾何学的計算を高速処理する。このため，光線あるいは視線と体系中の物体との交差テストを行い，スクリーン中の各画素に対する輝度値を計算している光線追跡法も同様に高速化可能である。

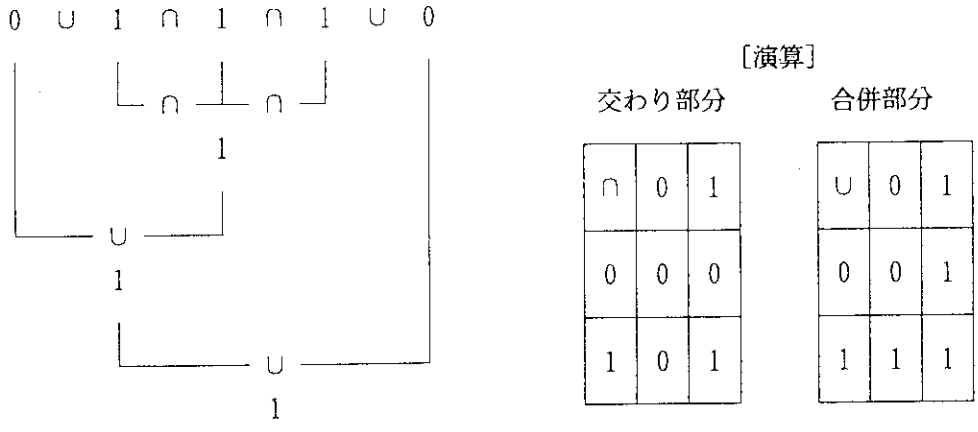


Fig. 6.1 Evaluation of logical expression to investigate a relation between point and region.

Table 6.1 Performance in visualization of plant geometry database using S-4/10

光源の種類	可視化対象モデル	CSGモデルの詳細度	ポリゴン・データへの変換	プリミティブ数	ポリゴン数	前処理時間 (sec.)*	描画時間 (sec.)**
環境光のみ	模擬環境	簡易形状	簡易近似	80	1,807	9.57	0.067
		複雑形状	簡易近似	1,075	15,551	9.53	0.560
			精密近似	1,075	29,471	9.84	0.695
	JRR-3施設 地下1階 1次区画	複雑形状	簡易近似	1,303	33,524	10.21	1.109
			精密近似	1,303	40,689	9.64	1.204
環境光及び4つの点光源	模擬環境	簡易形状	簡易近似	80	1,807	8.43	0.136
		複雑形状	簡易近似	1,075	15,551	8.44	1.404
			精密近似	1,075	29,471	8.41	1.907
	JRR-3施設 地下1階 1次区画	複雑形状	簡易近似	1,303	33,524	8.45	3.010
			精密近似	1,303	40,689	8.40	3.420

* 画像生成ライブラリーへ供給される構造体の生成、各ポリゴンの頂点の法線ベクトル計算等の前処理時間

** スクリーンへのアフィン変換、シェーディングのための輝度計算等グラフィッ・アクセラレータを使用した3次元画像生成時間
（描画させたロボットデータは、10コマ／秒の歩行データを用いた）

7. 超高速モンテカルロ装置

7.1 はじめに

複雑な三次元領域の取り扱いが可能である，という点は，粒子輸送モンテカルロ・コードの大きな特長であるが，このために生ずる複雑なプログラム構造は，しばしば効率の良いベクトル処理を妨げる。粒子輸送問題を解くモンテカルロ・シミュレーションの計算内容は，

1) 粒子追跡
2) 衝突のシミュレーション
3) 統計計算

の3つに分かれる。このうち，最も計算時間の比率が高い計算部分は，粒子追跡であり，通常の問題においては，全体の50～70％の計算時間が費やされる。これは，複雑な空間（二次曲面の論理演算によって表現された三次元領域）における粒子の追跡過程において，三次元空間における二次元曲面と直線の交差問題を解く必要があるためである。この問題においては，

1) 曲面の種類によって直線との交差点を求める計算式が異なるため，曲面ごとに多分岐の条件分岐文によって粒子を分類する。
2) 領域を構成する各曲面に対して求めた交差点がその領域の境界となっているか否かを再帰的処理を要する条件分岐文によって判定する。

といった処理が必要となるが，従来のベクトル計算機においては，これらの処理（多分岐や再帰的処理を要する条件分岐文）を高速化することが不可能であった。超高速モンテカルロ装置においては，モンテカルロ・パイプラインと呼ばれる特殊なハードウェアによって，上記処理部分，即ち，粒子輸送問題を解くモンテカルロ・シミュレーションにおいて計算コストが高くしかも従来のベクトル計算機において高速処理が不可能な部分の高速化を図っている。また，本装置においては，粒子輸送モンテカルロ・コードのベクトル処理上のもう一つの問題点である間接番地参照データの輸送能力についても改善を図っている。粒子輸送モンテカルロ・コードのベクトル処理においては，複数の粒子に対する計算を並列に行うことによって高速処理を実現しているが，この際，各粒子の属性データは，間接番地によって参照される。間接番地参照データは，順番に規則正しく参照される直接番地参照データと異なり，主記憶からベクトル・プロセッサへの輸送速度が遅い。このため，ベクトル・プロセッサに遊びが生じ，プロセッサの性能が落ちる，という点が従来のベクトル計算機上の粒子輸送モンテカルロ・コードのベクトル処理における問題点の一つであった。本装置においては，主記憶から演算データをベクトル・プロセッサへ輸送するロード／ストア・パイプラインを強化し，間接番地参照データ転送能力を高めている。

このように，本装置は，粒子輸送モンテカルロ・コードを従来のベクトル計算機において処理した際の問題点を解消する，という方針のもとに開発されたベクトル・並列計算機である。昭和63年度から概念検討を開始し，詳細設計を経て，平成4年度に製作を完了した。本章では，粒子輸送モンテカルロ・コードのベクトル処理上の問題点，これを解消するために開発されたモン

テカルロ・パイプラインと呼ばれる特殊パイプライン等のハードウェア上の特長及び汎用モンテカルロ・コードKENO-IV及びMCNPを用いた本装置の性能評価結果について述べる。

7.2 粒子輸送モンテカルロ・コードのベクトル処理上の問題点

KENO-IVコードやMCNPコード等の既存の粒子輸送モンテカルロ・コードは，ベクトル処理可能なDOループを含まないために，ベクトル化においては，ひとつの粒子の誕生から消滅までを追跡しているオリジナル・コードのアルゴリズムを，複数の粒子を同時に追跡するよう変更する必要がある。ベクトル処理は，散乱，吸収等の事象ごとに作られた粒子バンクを使ったDOループ計算に対して行われる。しかし，このようにしてベクトル化したコードを従来のベクトル計算上において処理しても，ベクトル化率及び加速率が低いために，高い高速向上率を得ることは極めて難しい。

(1) 低いベクトル化率

粒子輸送モンテカルロ・コードにおいては，多様な物理現象をシミュレーションするため，あるいは複雑な幾何形状において粒子を追跡するために条件分岐文を多用している。これらの条件分岐文は，①多分岐の条件分岐文であるため，あるいは再帰的処理を要するためベクトル化できない，②ベクトル化は可能であるが，高い速度向上率を期待できない，という場合が多い。このため，既存の粒子輸送モンテカルロ・コードをベクトル化した際，そのベクトル化率は，低い値に留まる。これらの条件分岐文による処理は，以下のように大別される。

・幾何形状分類
・事象分類
・領域判定

ここで，幾何形状分類とは，粒子の現在位置から次の境界までの距離を計算する際，その領域を構成する曲面の種類によって粒子を分類する処理であり，ベクトル処理不可能な多分岐の条件分岐文によって行われている。事象分類とは，乱数と断面積データを用いて衝突のシミュレーションを行う際，粒子バンクを作成するために粒子を散乱，吸収等の事象によって分類する処理であり，二分岐の条件分岐文によって行われる。この処理は，ベクトル化可能であるが，条件分岐命令に対して，制御変数（マスク）を用いてベクトル・データの圧縮（Compress）または伸長（Expand）等の処理を行うため，高い速度向上率は望めない。領域判定とは，ある点が複数の曲面によって構成された三次元領域に含まれるか否かを判定する処理であり，二値論理式の評価を行う処理である。その処理を行うプログラムは，条件分岐文が複雑に組み合わされ，再帰的処理を要する構造となっているためベクトル化は極めて困難である。

(2) 低い加速率

粒子輸送モンテカルロ・コードのベクトル処理において，加速率（ベクトル・プロセッサの性能）が低い理由は，①ベクトル計算においてベクトル長が短い，②間接番地データを用いてベクトル計算を行っているためでる。

① 短いベクトル長

ベクトル処理は，前述した粒子バンクに対して行われるが，個々の粒子の振る舞いは，そ

れぞれ異なるため，追跡している粒子が幾つかの粒子バンクに分散してしまう。分散した粒子がさらに分散する場合も多い。また，計算が進むにつれ，粒子の削減に伴い，処理粒子が減少する。このため，粒子輸送モンテカルロ・コードのベクトル処理においては，ベクトル長さが短い。特に，臨界計算においては，数百という少ない数の粒子を100～200程度の世代にわたって追跡することが多いため，この傾向が強い。遮蔽計算においても，追跡する粒子を増やすことは，メモリ容量の増加につながるため，処理粒子数には，やはり限度がある。筆者らの経験では，粒子輸送モンテカルロ・コードのベクトル処理における平均ベクトル長は，50～100程度である。

② 間接番地参照

複数の粒子を同時に取り扱うため，粒子の位置，飛方向，エネルギー等の物理的属性を粒子番号によって参照する。この結果，ベクトル・プロセッサへのデータ転送速度の遅い間接番地によってデータを参照することになり，ベクトル・プロセッサの「待ち」が生じ，その結果，ベクトル処理性能が低下する。

KENO-IVコードやMCNPコードをベクトル化した筆者らの経験[1)]に拠れば，ベクトル化率は，70～80％であり，加速率は3倍程度である。これらの因子によるベクトル計算においては，理論上，2倍以上の速度向上率は望めない。実際，KENO-IV，MCNPコード等の既存の粒子輸送モンテカルロ・コードの従来のベクトル計算機上における速度向上率は，1.5倍程度であった。

7.3 本装置の設計思想及びアーキテクチャ

超高速モンテカルロ装置は，上記に述べた問題点を克服するために開発された特殊なハードウェアを装備したベクトル・並列計算機である。本装置は，以下の設計思想に基いて開発された。

(1) モンテカルロ装置の設計思想

① 現在大型計算機上において実行されている粒子輸送モンテカルロ計算を高速化する本装置においては，これと同等以上の処理能力が要求される。従って，現在のスーパーコンピュータと同等のベクトル演算機能，大容量メモリ，速いマシーン・サイクルを有すること。

② 主記憶から演算プロセッサへのオペランド（演算対象データ）転送の遅れは，演算プロセッサの性能低下の原因となる。従って，間接番地データを多用する粒子輸送モンテカルロ計算において，演算プロセッサの能力に応じたデータ輸送能力を有すること。

(2) モンテカルロ装置のアーキテクチャ

以上の観点から，本装置の開発は，ベクトル・並列計算機SX－3をベース・プロセッサとして粒子輸送モンテカルロ・コード高速化のための改造を行う，という指針のもとに行われた。本装置の開発において，変更あるいは新規開発したLSI（大規模集積回路）は，演算装置部分について15％，主記憶制御部分について32％であり，MCP（マルチ・チップ・パッケージ：70～80個のLSI集合体）は，演算装置部分について67％，主記憶制御部分について100％である。本装置の構造をFig. 7.1に示す。本装置の開発においては，従来の演算パイプラインに，モンテカルロ・パイプラインを付加，ロード／ストア・パイプラインを強化した。本装

置は，さらに，4台のプロセッサを用いてベクトル処理速度を加速する共有メモリ方式ベクトル・並列計算機である。新規開発あるいは強化されたハードウェアの特長及び性能評価結果を以下に示す。

① モンテカルロ・パイプライン

7.2において述べたように，粒子輸送モンテカルロ・コードにおいては，条件分岐文のためにベクトル化が不可能であるが，ベクトル処理による速度向上率が望めないためにスカラ処理されるDOループが多い。これらの処理を高速化するため，本装置は，モンテカルロ・パイプラインと呼ばれる以下の3種類の特殊パイプラインを装備している。

・幾何形状パイプライン

粒子の属する領域形状によって粒子を分類するための多分岐の条件分岐文を含むDOループは，ベクトル化が不可能である。本パイプラインは，従来スカラ処理によって行われていたこの種のDOループを高速処理する。Fig. 7.2に，本パイプラインの性能評価に用いたFORTRANプログラムを示す。スカラ処理時間に対する速度向上率は，ループ長100の場合，7.5倍であった

・事象分類パイプライン

本パイプラインは，粒子を事象によって高速に分類し，粒子バンクを作成する。二分岐の条件分岐文を含むDOループは，ベクトル処理可能であるが，この処理は頻繁に行われるため，さらに高速化が必要である。Fig. 7.3に，本パイプラインの性能評価に用いたFORTRANプログラムを示す。速度向上率は，ループ長100の場合，従来のベクトル処理時間と比較して，2.2倍であった。

・領域判定パイプライン

本パイプラインは，Fig. 7.4に示される二値論理式の評価を高速に行う。幾何形状パイプラインと同様，複雑な三次元領域を取り扱うための処理を高速化する。本パイプラインの性能評価において，スカラ処理時間に対する速度向上率は，長さ88の二値論理式を処理した場合，12.9倍であった。

② ロード／ストア・パイプラインによるデータ転送機能の強化

前述したように，粒子輸送モンテカルロ・コードのベクトル処理においては，ベクトル・プロセッサの処理性能を高めるため，間接番地参照データの主記憶－ベクトル・プロセッサ間の転送能力を高める必要がある。本装置の開発において，主記憶－ベクトル・プロセッサ間の間接番地参照データ転送能力は，ベース・プロセッサと比較し，8倍に強化された。

7.4 汎用モンテカルロ・コードによる本装置の実効性能評価結果

本装置の実効性能評価においては，臨界計算多群モンテカルロ・コードKENO-IV及び連続エネルギー粒子輸送モンテカルロ・コードMCNPを用いた。この結果について述べる。

(1) 実効性能評価において用いた入力データ

① KENO-IVコードにおいて用いた入力データ

KENO-IVコードによる実効性能評価において用いた入力データは，TCA（Tank-type

Critical Assembly）における臨界計算であり，KENO-IV コードによるモンテカルロ計算に用いられている典型的な入力体系である。低濃縮ウラン（2.3％の UO_2）からなる19×19本の燃料ピンのある水位に対する実効増倍係数を求めるために，1世代あたり1,200個の中性子を53世代にわたり追跡している。

② MCNP コードにおいて用いた入力データ

MCNP コードを用いた性能評価においては，人体模型を用いた実効線量当量の計算を行った。Fig. 7.5 に示す人体模型に対し，γ線の平行ビームを前方から照射した場合の各臓器に対する線量当量を求める問題である。この人体模型は，約200枚の二次曲面によって表現された約110の三次元領域から構成される。γ線の入射エネルギーは，1 MeV，ヒストリー数は百万である。

(2) 実効性能評価結果

① 強化されたロード／ストア・パイプラインの実効性能

Table. 7.1 に，KENO-IV 及び MCNP コードの本装置（1プロセッサのみ）上における計算時間と改造を行ったベース・プロセッサである SX－3／11 上における計算時間を示す。両者を比較することで，強化されたロード／ストア・パイプラインによって，計算時間が約13％短縮されたことがわかる。

② モンテカルロ・パイプラインの実効性能

Table 7.2 に，本装置（1プロセッサのみ）上において，強化されたロード／ストア・パイプラインのみを用いた場合と強化されたロード／ストア・パイプライン及びモンテカルロ・パイプラインを用いた場合の KENO-IV 及び MCNP コードの計算時間を示す。両者を比較することで，モンテカルロ・パイプラインの実効性能を知ることができる。KENO-IV コードにおける約18％の計算時間短縮は，幾何形状パイプラインに依るものである。MCNP コードを用いた性能評価においては，約48％の計算時間の短縮があるが，これは，主に領域判定パイプラインの効果である。

③ ベクトル処理及びベクトル・並列処理による KENO-IV コードの速度向上率

Table 7.3 に，本装置上で1プロセッサを使用した場合（スカラ処理及びベクトル処理）の KENO-IV コードの実行時間と4プロセッサを使用した場合の実行時間を示す。Table 7.3 からわかるように，本装置上の速度向上率は，1プロセッサ使用時，オリジナル・コードのスカラ計算に対し，3.7倍，4プロセッサ使用時，10倍であった。4プロセッサ使用時に，並列処理による速度向上率が，2.7（＜4＝プロセッサ台数）倍である理由は，実効増倍係数を求めるために各世代に関する計算終了時に生じる同期のオーバーヘッドと粒度の低下，及びメモリ競合である。

④ ベクトル処理及びベクトル・並列処理による MCNP コードの速度向上率

Table 7.4 に，本装置上で1プロセッサを使用した場合（スカラ処理及びベクトル処理）の MCNP コードの実行時間と4プロセッサを使用した場合の実行時間を示す。4プロセッサ使用時に，並列処理による速度向上率が，KENO-IV コードと比較して高い値である理由は，遮蔽計算のため，同期待ちによる並列処理性能の低下が小さいことに起因する。

7.5 おわりに

従来のベクトル計算機上で極めて困難であった粒子輸送モンテカルロ・コードの高速処理を実現するため，超高速モンテカルロ装置を開発し，設計基準であり，目標性能である「既存コードのスカラ処理に対して10倍」を達成した。KENO-IV，MCNPコードの他，汎用コードVIM，MORSE等の粒子輸送モンテカルロ・コードによる実効性能評価も現在行っている。また，本装置は，従来のスーパーコンピュータに粒子輸送モンテカルロ・コード高速化のための特殊パイプラインを付加した構造となっているため，粒子モデル計算のみならず，連続体モデル計算に対する高速処理も十分可能である。さらに，本装置は，粒子輸送モンテカルロ計算と同様，複雑な幾何形状において直線を追跡し，三次元画像を生成する光線追跡法の高速化も可能である。今後，本装置は，人間動作シミュレーション技術の研究における粒子モデル計算や画像生成処理に大きく貢献することが期待される。

参考文献

1) 例えば，Asai, K. et al., Vectorization of the KENO IV Code, Nucl. Sci. & Eng., 92, 298 (1986).

Table 7.1 Comparison of execution time using only load／store enhancement feature on Monte Carlo machine with SX-3／11.

	ＳＸ－３／１１上における計算時間	超高速モンテカルロ装置（１プロセッサ）上における計算時間
KENO-IV コード	６２．８ 秒	５４．８ 秒
MCNPコード	１０２０．３ 秒	８８９．９ 秒

注）超高速モンテカルロ装置とＳＸ－３／１１のマシーン・サイクルは異なるため、表におけるＳＸ－３／１１上の計算時間は、超高速モンテカルロ装置のマシーン・サイクルである2.5 nsに矯正したものである。

Table 7.2 Execution time using enhanced load／store and three Monte Carlo pipelines.

	モンテカルロ・パイプラインを使用しない場合の計算時間	モンテカルロ・パイプラインを使用した場合の計算時間
KENO-IV コード	５４．８ 秒	４４．９ 秒
MCNPコード	８８９．９ 秒	４６３．４ 秒

Table 7.3 Speedup ratios of the KENO-Ⅳ code on Monte Carlo machine.

	1プロセッサ使用時		4プロセッサ使用時
処理モード	スカラ処理	ベクトル処理	ベクトル処理
実行時間	167.1 秒	44.9 秒	16.7 秒
速度向上率（A）	1.0	3.7	10.0
速度向上率（B）		1.0	2.7

注1）表における167.1 秒は、オリジナル・コードのスカラ処理時間を示す。
注2）速度向上率Aは、オリジナル・コードのスカラ処理時間を1とした時の速度向上率を、同じくBは、1プロセッサを用いたベクトル処理時間を1とした時の速度向上率を示す。

Table 7.4 Speedup ratios of the MCNP code on Monte Carlo machine.

	1プロセッサ使用時		4プロセッサ使用時
処理モード	スカラ処理	ベクトル処理	ベクトル処理
実行時間	1284.2 秒	463.4 秒	137.8 秒
速度向上率（A）	1.0	2.8	9.3
速度向上率（B）		1.0	3.4

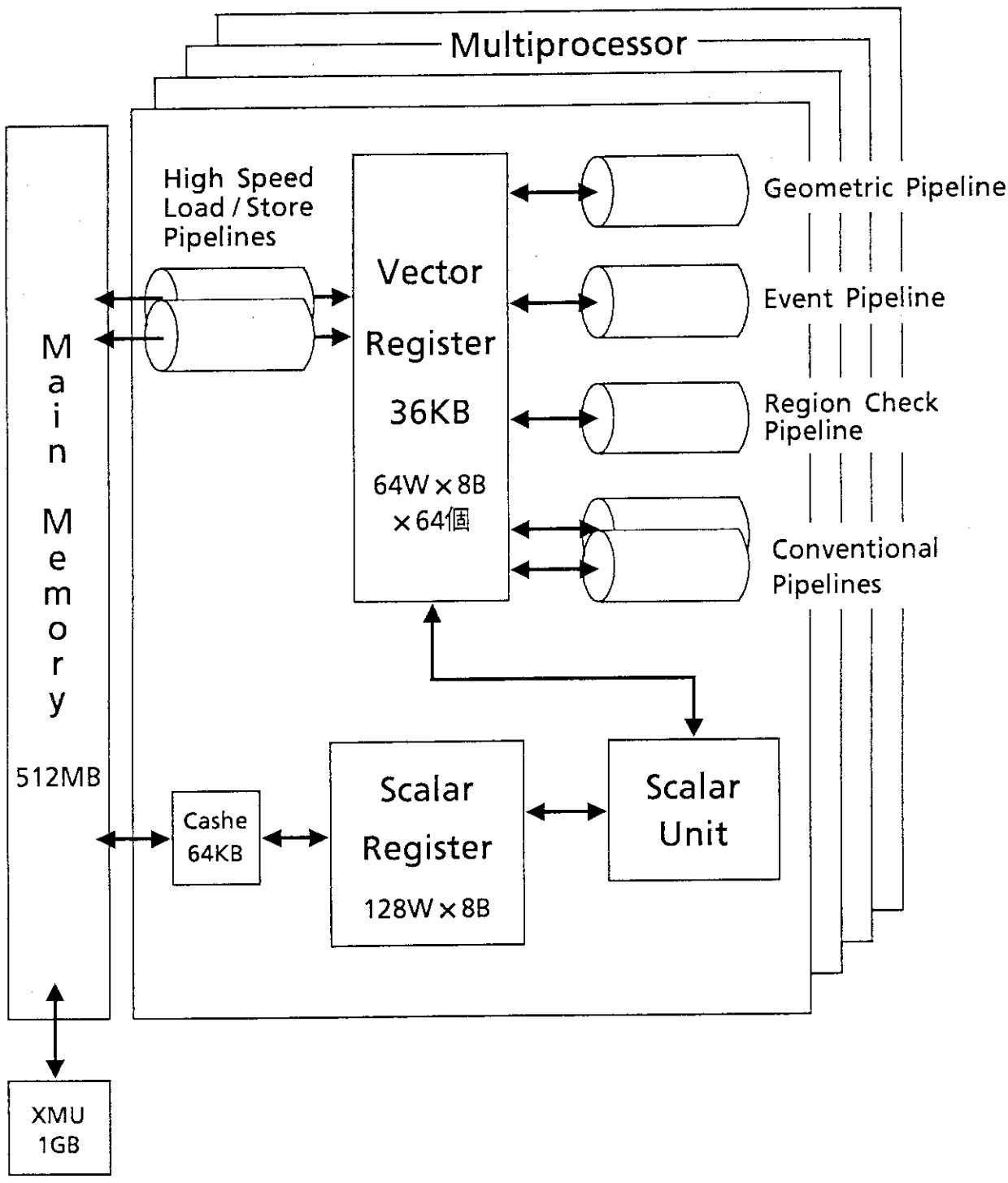


Fig. 7.1 Configuration of Monte Carlo machine

```
      DO 100 I=1,N
      IV=BANK(I)
      GO TO (10, 20, 30, 40, 50, 60),IGEO(IV)
       N7=N7+1
      BANK7(N7)=IV
      GO TO 100
 10   N1=N1+1
      BANK1(N1)=IV
      GO TO 100
 20   N2=N2+1
      BANK2(N2)=IV
      GO TO 100
 30   N3=N3+1
      BANK3(N3)=IV
      GO TO 100
 40   N4=N4+1
      BANK4(N4)=IV
      GO TO 100
 50   N5=N5+1
      BANK5(N5)=IV
      GO TO 100
 60   N6=N6+1
      BANK6(N6)=IV
100   CONTINUE
```

Fig. 7.2 Fortran statements for particle selection by geometry.

```
      DO 100 I=1,N
      IV=BANK(I)
      IF(NFLAG(IV).NE.0) THEN
          N1=N1+1
          BANK1(N1)=IV
      ELSE
          N2=N2+1
          BANK2(N2)=IV
      ENDIF
100   CONTINUE
```

Fig. 7.3 Fortran statements for particle selection by event type.

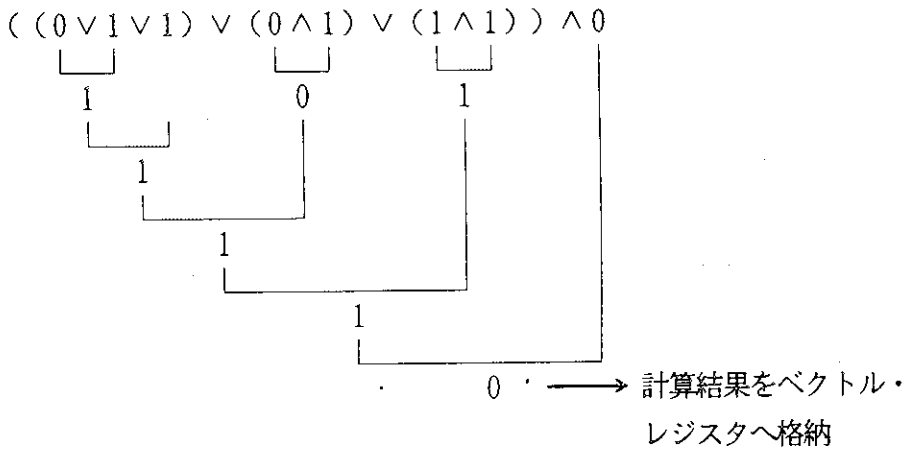


Fig. 7.4 Evaluation of a logical expression

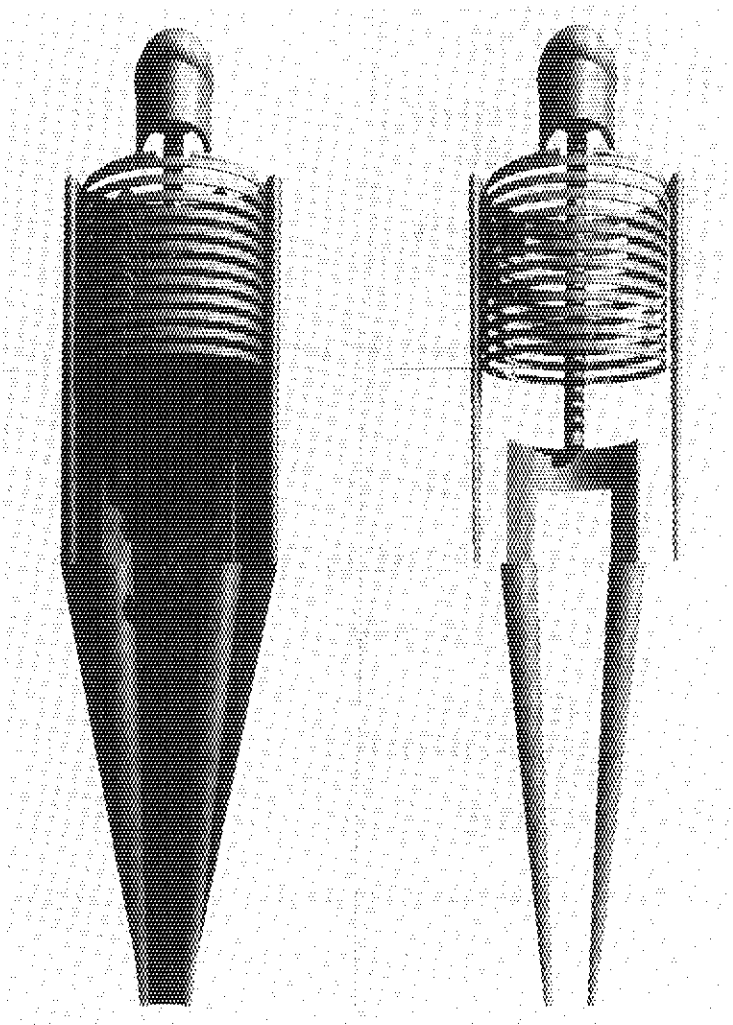


Fig. 7.5 Human phantom used for the estimation of effective performance of Monte Carlo machine.

8. おわりに

HASPの研究開発も6年を経過し，研究も幾分進展をみた。特に，これまでの研究成果を7編の論文にまとめ二つの国際会議に投稿し受理された。ひとつは，Joint International Conference on Mathematical Methods and Supercomputing in Nuclear Applicationsで6編の論文を投稿し，他はANS Fifth Topical Meeting on Robotics and Remote Systemsで，1編の論文を投稿し受理された。この間に研究所内外の多くの方々から貴重なコメント，助言を頂いたのは研究推進に役立っている。今後，蓄積していくソフトウェアの検証・評価のためには，ロボット・センサ，具体的な移動ロボット等のハードウェアが不可欠であり，外部機関との共同研究等新たな突破口も模索しなければならない。尚，本報告書の執筆担当者は，

第2章	神林　奨
第3章	藤井　実，海老原健一
第4章	久米　悦雄
第5，6，7章	樋口　健二
第1，8章	秋元　正幸

である。また，全章に渡って内容のチェック等補筆訂正を大谷孝之が行った。

謝　　辞

HASPの研究開発に関し，中部大学西原宏教授，九州大学松尾文顕教授からご批判，ご助言を頂いた。厚くお礼を申し上げます。

8. おわりに

HASPの研究開発も6年を経過し，研究も幾分進展をみた。特に，これまでの研究成果を7編の論文にまとめ二つの国際会議に投稿し受理された。ひとつは，Joint International Conference on Mathematical Methods and Supercomputing in Nuclear Applicationsで6編の論文を投稿し，他はANS Fifth Topical Meeting on Robotics and Remote Systemsで，1編の論文を投稿し受理された。この間に研究所内外の多くの方々から貴重なコメント，助言を頂いたのは研究推進に役立っている。今後，蓄積していくソフトウェアの検証・評価のためには，ロボット・センサ，具体的な移動ロボット等のハードウェアが不可欠であり，外部機関との共同研究等新たな突破口も模索しなければならない。尚，本報告書の執筆担当者は，

第2章	神林　奨
第3章	藤井　実，海老原健一
第4章	久米　悦雄
第5，6，7章	樋口　健二
第1，8章	秋元　正幸

である。また，全章に渡って内容のチェック等補筆訂正を大谷孝之が行った。

謝　　辞

HASPの研究開発に関し，中部大学西原宏教授，九州大学松尾文顕教授からご批判，ご助言を頂いた。厚くお礼を申し上げます。

付録１　ＣＳＧモデルからポリゴン・データを生成する処理の詳細

１　ポリゴンデータ作成プログラム　（ｍｏｒｓｅｄｂ）

ポリゴンデータ作成プログラムは、施設を構成する基本プリミティブ面をポリゴンデータに表現した後、そのポリゴンデータを用いてＣＳＧ演算に基づいた合成を行い、新しいポリゴンを作成している。

基本プリミティブ面を構成するポリゴンデータを作成する際には、描画速度の向上を図るため三角片(N個の頂点からなるN-2個の三角形の集合：ｋを1≦ｋ≦N-2とすれば、第k, k+1およびk+2番目の頂点が１個の三角形の頂点になるものである。図1参照)を用いてプリミティブ面を構成するようにしたが、合成する場合は三角辺を分解した１つの三角形について考えた。以下に基本的考え方およびプログラムの説明を示す。

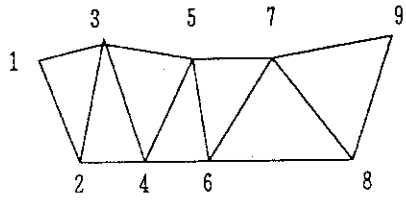


図1 三角片

１．１　基本プリミティブ

１．１．１　基本プリミティブの種類および形状

施設を構成する基本プリミティブは全部で１１種類である。その一覧および形状を以下の表および図に示す。これらプリミティブの原点は、ＨＥＩＧＨＴの始点および球の中心になっている。

表　１　　基本プリミティブ一覧

ﾌﾟﾘﾐﾃｨﾌﾞ No.及び名		特性長No.	特性長の内容	対応する軸**
0	BOX	1	ＷＩＤＴＨのｘ座標	
		2	〃　　ｙ座標	Y
		3	〃　　ｚ座標	
		4	ＤＥＰＴＨのｘ座標	
		5	〃　　ｙ座標	－Ｘ
		6	〃　　ｚ座標	
		7	ＨＥＩＧＨＴのｘ座標	
		8	〃　　ｙ座標	Z
		9	〃　　ｚ座標	
1	CONE	1	ＨＥＩＧＨＴのｘ座標	
		2	〃　　ｙ座標	Z

		3	〃 z座標	
		4	RADIUSの値	
2	CYLINDER	1	HEIGHTのx座標	Z
		2	〃 y座標	
		3	〃 z座標	
		4	RADIUSの値	
3	SPHERE	1	RADIUSの値	
5	ELLIPS	1	WIDTHのx座標	Y
		2	〃 y座標	
		3	〃 z座標	
		4	HEIGHTのx座標	Z
		5	〃 y座標	
		6	〃 z座標	
		7	DEPTHのx座標	－X
		8	〃 y座標	
		9	〃 z座標	
6	E_CLINDER	1	HEIGHTのx座標	Z
		2	〃 y座標	
		3	〃 z座標	
		4	WIDTHのx座標	Y
		5	〃 y座標	
		6	〃 z座標	
		7	DEPTHのx座標	－X
		8	〃 y座標	
		9	〃 z座標	
8	HYPER	1	HEIGHTのx座標	Z
		2	〃 y座標	
		3	〃 z座標	
		4	WIDTHのx座標	Y
		5	〃 y座標	
		6	〃 z座標	
		7	DEPTHのx座標	－X
		8	〃 y座標	
		9	〃 z座標	
		10	RATIO*	
9	PARABOLA	1	HEIGHTのx座標の1/10	Z
		2	〃 y座標の1/10	

		3	〃 z座標の1/10	
		4	ＷＩＤＴＨのｘ座標の1/10	Y
		5	〃 ｙ座標の1/10	
		6	〃 ｚ座標の1/10	
		7	ＤＥＰＴＨのｘ座標	－X
		8	〃 ｙ座標	
		9	〃 ｚ座標	
11	TORUS	1	ＨＥＩＧＨＴのｘ座標	Z
		2	〃 ｙ座標	
		3	〃 ｚ座標	
		4	ＲＡＤＩＵＳの値	
12	TORUS_E	1	ＨＥＩＧＨＴのｘ座標	Z
		2	〃 ｙ座標	
		3	〃 ｚ座標	
		4	ＲＡＤＩＵＳ１	
		5	ＲＡＤＩＵＳ２	
13	TORUS_Q	1	ＨＥＩＧＨＴのｘ座標	Z
		2	〃 ｙ座標	
		3	〃 ｚ座標	
		4	ＳＴＡＲＴのｘ座標	Y
		5	〃 ｙ座標	
		6	〃 ｚ座標	
		7	ＥＮＤのｘ座標	－X
		8	〃 ｙ座標	
		9	〃 ｚ座標	

* HYPERのＲＡＴＩＯは、一葉双曲面の式（$x^2/a^2+y^2/b^2-z^2/c^2=0$）の内、ＤＥＰＴＨをａ、ＷＩＤＴＨをｂとおいたときのｃの値である。

** 対応する軸とは、基本プリミティブを下図に示すようにｘｙ平面に平行に置いたときに、そのDEPTH,WIDTH,HEIGHTの示すベクトルがどの軸に対応するかを示したものである。

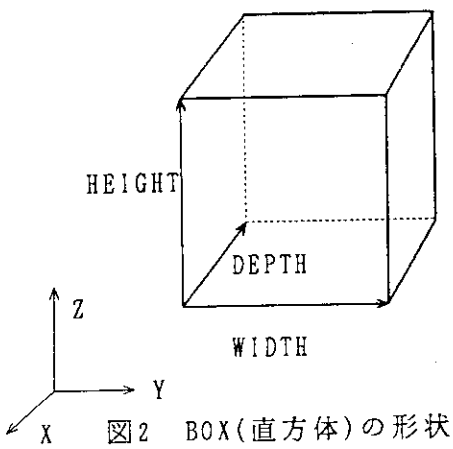


図2 BOX(直方体)の形状

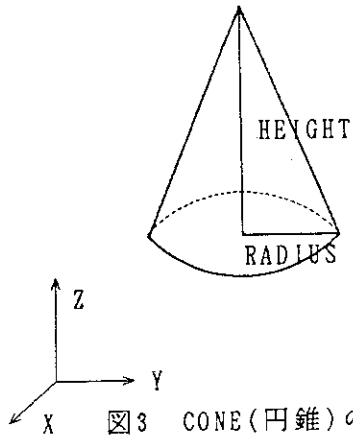


図3 CONE(円錐)の形状

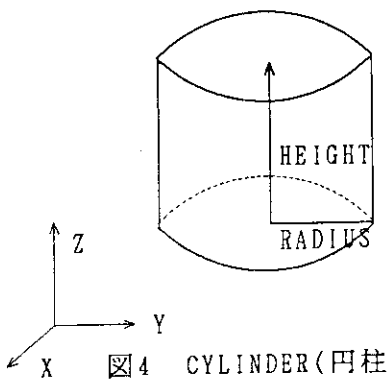


図4 CYLINDER(円柱)の形状

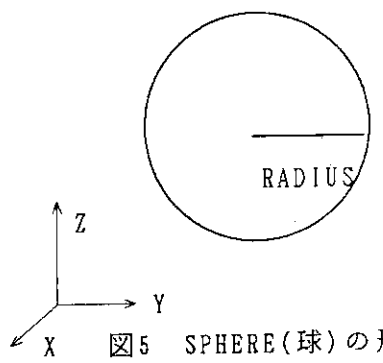


図5 SPHERE(球)の形状

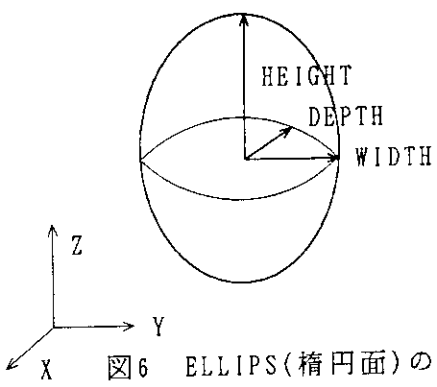


図6 ELLIPS(楕円面)の形状

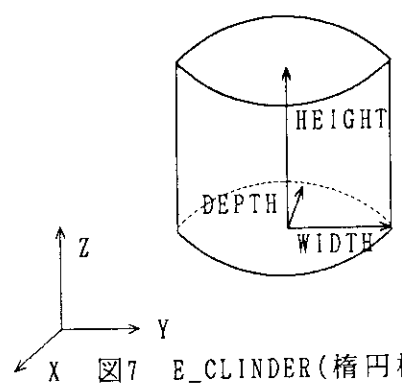


図7 E_CLINDER(楕円柱)の形状

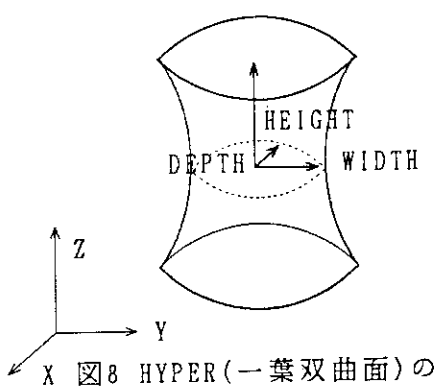


図8 HYPER(一葉双曲面)の形状

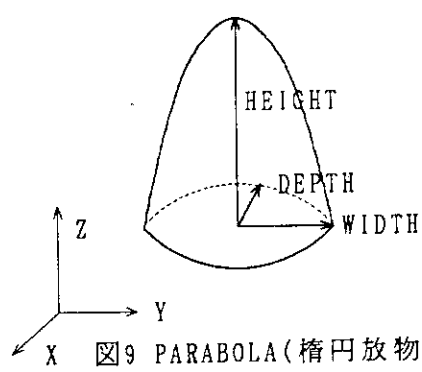


図9 PARABOLA(楕円放物面)の形状

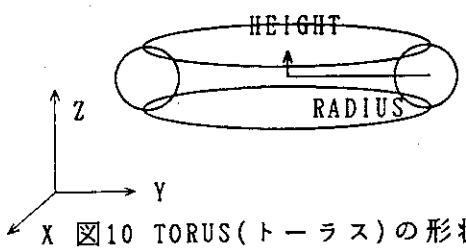


図10 TORUS(トーラス)の形状

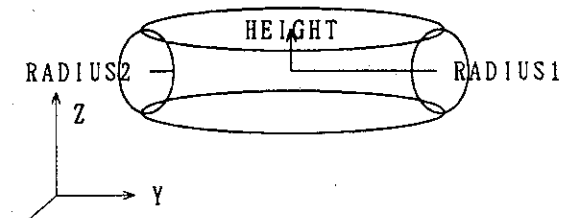


図11 TORUS_E(楕円トーラス)の形状

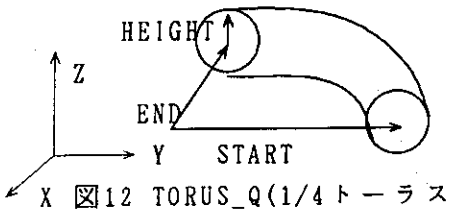


図12 TORUS_Q(1/4トーラス)の形状

1．1．2 基本プリミティブを構成する方程式

各基本プリミティブを構成する方程式を以下に示す。ここで用いているDEPTH,WIDTH,HEIGHTは、各々そのベクトルの大きさ(絶対値)を示している。

（1） ＢＯＸの場合

６面を６つの方程式で構成する。

$$x = -DEPTH \quad (0 \leqq y \leqq WIDTH,\ 0 \leqq z \leqq HEIGHT) \tag{1}$$

$$x = 0 \quad (0 \leqq y \leqq WIDTH,\ 0 \leqq z \leqq HEIGHT) \tag{2}$$

$$y = 0 \quad (-DEPTH \leqq x \leqq 0,\ 0 \leqq z \leqq HEIGHT) \tag{3}$$

$$y = WIDTH \quad (-DEPTH \leqq x \leqq 0,\ 0 \leqq z \leqq HEIGHT) \tag{4}$$

$$z = 0 \quad (-DEPTH \leqq x \leqq 0,\ 0 \leqq y \leqq WIDTH) \tag{5}$$

$$z = HEIGHT \quad (-DEPTH \leqq x \leqq 0,\ 0 \leqq y \leqq WIDTH) \tag{6}$$

（2） ＣＯＮＥの場合

底面 ： $x^2 + y^2 = (RADIUS)^2$ (7)

側面 ： $x^2 + y^2 - (RADIUS/HEIGHT)^2 (z - HEIGHT)^2 = 0$

$(0 \leqq z \leqq HEIGHT)$ (8)

（3） ＣＹＬＩＮＤＥＲの場合

底面 ： $x^2 + y^2 = (RADIUS)^2 \quad (z = 0, HEIGHT)$ (9)

側面 ： $x^2 + y^2 = (RADIUS)^2 \quad (0 \leqq z \leqq HEIGHT)$ (10)

（4） ＳＰＨＥＲＥの場合

原点の座標を (x_0, y_0, z_0) とおくと、

$(x - x_0)^2 + (y - y_0)^2 + (z - z_0)^2 = (RADIUS)^2$ (11)

（5） ＥＬＬＩＰＳの場合

$x^2/DEPTH^2 + y^2/WIDTH^2 + z^2/HEIGHT^2 = 1$ (12)

（６）　Ｅ＿ＣＬＩＮＤＥＲの場合

底面 ： $x^2/DEPTH^2 + y^2/WIDTH^2 = 1 \quad (z=0, HEIGHT)$ (13)

側面 ： $x^2/DEPTH^2 + y^2/WIDTH^2 = 1 \quad (0 \leqq z \leqq HEIGHT)$ (14)

（７）　ＨＹＰＥＲの場合

底面 ： $x^2/DEPTH^2 + y^2/WIDTH^2 = 1 \quad (z = \pm HEIGHT)$ (15)

側面 ： $x^2/DEPTH^2 + y^2/WIDTH^2 - z^2/RATIO^2 = 1$

$(-HEIGHT \leqq z \leqq HEIGHT)$ (16)

（８）　ＰＡＲＡＢＯＬＡの場合

底面 ： $x^2/(10*DEPTH)^2 + y^2/(10*WIDTH)^2 = 1 \quad (z=0)$ (17)

側面 ： $x^2/(10*DEPTH)^2 + y^2/(10*WIDTH)^2 + (z-HEIGHT)/HEIGHT = 0$

$(0 \leqq z \leqq HEIGHT)$ (18)

（９）　ＴＯＲＵＳの場合

$$x^4 + y^4 + z^4 + 2x^2y^2 + 2x^2z^2 + 2y^2z^2 - 2(RADIUS^2+HEIGHT^2)(x^2+y^2) + 2(RADIUS^2-HEIGHT^2)z^2 + 2(RADIUS^2-HEIGHT^2)^2 = 0 \quad (19)$$

（１０）　ＴＯＲＵＳ＿Ｅの場合

$$x^4 + y^4 + (RADIUS2^4/HEIGHT^4)z^4 + 2x^2y^2 + 2(RADIUS2^2/HEIGHT^2)x^2z^2 + 2(RADIUS2^2/HEIGHT^2)y^2z^2 - 2(RADIUS1^2+RADIUS2^2)(x^2+y^2) + 2(RADIUS1^2-RADIUS2^2)(RADIUS2^2/HEIGHT^2)z^2 + 2(RADIUS1^2-RADIUS2^2)^2 = 0 \quad (20)$$

（１１）　ＴＯＲＵＳ＿Ｑの場合

$$x^4 + y^4 + z^4 + 2x^2y^2 + 2x^2z^2 + 2y^2z^2 - 2(END^2+HEIGHT^2)(x^2+y^2) + 2(END^2-HEIGHT^2)z^2 + 2(END^2-HEIGHT^2)^2 = 0$$

$(-END-HEIGHT \leqq x \leqq 0,\ 0 \leqq y \leqq START+HEIGHT)$ (21)

１．１．３　基本的な曲線の分割方法および基本ポリゴンの作成方法

３次元モデルの変換法を述べる前に、円や楕円などの２次元の基本図形からポリゴンを作成する手法を説明する。

（１）　円の分割方法

円の場合には、パラメータとして角度をとると、一定角度の変化に対して内接するポリゴンの面積を最大にしうる。

許容できる最大の長さをδとした場合、$8n = 2\pi / d\theta$となるｎを用いて、$\delta \geqq$円周$(2\pi r)/n$になるｎを求める。このｎを成り立たせる$d\theta$を用い、原点を中心とする円周上の点の座標は次式で与えられる。

$$x_{n+1} = r\cos(\theta + d\theta) \quad (22)$$

$$y_{n+1} = r\sin(\theta + d\theta) \quad (23)$$

倍角公式を用いて、(22),(23)式を書き換えると、

$$x_{n+1} = x_n\cos d\theta - y_n\sin d\theta \quad (24)$$

$$y_{n+1} = x_n\sin d\theta + y_n\cos d\theta \quad (25)$$

従って、$(x_1, y_1)=(r, 0)$と与えることにより、次の座標値が求められる。

（２）　楕円周の分割方法

楕円の場合には表示点を等角度間隔で計算すると、離心率の高い楕円では楕円の端部で曲率が非常に大きくなり、点数が少なすぎるという問題が出てくる。このため以下のアルゴリズムを用いて、原点を中心とする楕円周上の点の座標を求める。

楕円の長半径をａ、短半径をｂとおく。許容できる最大の長さをδとした場合、$\delta \geq a/\sqrt{2}n$になるｎを求める。このｎを用いてｄｘ、ｄｙを求める。

$$dx = a/\sqrt{2}n \quad (26)$$

$$dy = b/\sqrt{2}n \quad (27)$$

初期値を$(x_1, y_1)=(a, 0)$と与える。

・　$y < b/\sqrt{2}$ならば、$y_{n+1}=y_n+dy$とし、楕円の方程式からｘを求める。

・　$y \geq b/\sqrt{2}$ならば、$x_{n+1}=x_n+dx$とし、楕円の方程式からｙを求める。

（３）　双曲線の分割方法

正のｘ軸を対称軸とするような原点に中心のある双曲線を発生することを考える。直角座標系によるノンパラメトリックな式を、以下の式で表すと、

$$x^2/a^2 - y^2/b^2 = 1 \quad (28)$$

最大内接面積を実現する双曲線のパラメトリック表現として、以下の式が考えられる。

$$x = a\cosh\theta \quad (29)$$

$$y = b\sinh\theta \quad (30)$$

双曲線関数の倍角公式の関係より、

$$x_{n+1} = a(\cosh\theta\cosh d\theta + \sinh\theta\sinh d\theta) \quad (31)$$

$$y_{n+1} = b(\sinh\theta\cosh d\theta + \cosh\theta\sinh d\theta) \quad (32)$$

上式をさらに書き換えると、

$$x_{n+1} = x_n\cosh d\theta + a/b\cdot y_n\sinh d\theta \quad (33)$$

$$y_{n+1} = b/a\cdot x_n\sinh d\theta + y_n\cosh d\theta \quad (34)$$

$0 \leq \theta \leq \infty$に対し、全双曲線が描かれる。双曲線の範囲を指定するには、最大値$\theta = \theta_{max}$を定める必要がある。双曲線の第１および第４象限部分のみを考察の対象とすることとし、$a \leq x \leq a + c$の部分を描くものとすれば、

$$\theta_{max} = \cosh^{-1}((a + c)/a) \quad (35)$$

許容できる最大の長さをδとした場合、$\theta=0$と$\theta=d\theta$間の距離をδ以下になるように分割数を定める。

（４）　放物線の分割方法

双曲線と同様に、正のｘ軸を対称軸とし、原点を中心に右に開いた放物線を考察する。直角座標系におけるノンパラメトリックな放物線表現は、次式となる。

$$y^2 = 4ax \quad (36)$$

最大内接面積を生ずるパラメトリック表現は次式である。

$$x = a\theta^2 \quad (37)$$

$$y = 2a\theta \quad (38)$$

$\theta_{n+1} = \theta_n + d\theta$ とすれば、(37),(38)式は、次のようになる。

$$x_{n+1} = a\theta_n^2 + 2a\theta_n d\theta + a(d\theta)^2 \quad (39)$$

$$y_{n+1} = 2a\theta_n + 2a\,d\theta \quad (40)$$

上式を書き直すと、

$$x_{n+1} = x_n + y_n d\theta + a(d\theta)^2 \quad (41)$$

$$y_{n+1} = y_n + 2a\,d\theta \quad (42)$$

放物線も、$0 \leq \theta \leq \infty$ に対し全線が描かれる。放物線の範囲を指定するには、最大値 $\theta = \theta_{max}$ を定める必要がある。これにはｘ座標の範囲を制限したり、ｙ座標の範囲を制限するなどの方法があるが、本プログラム中では、ｘ座標の範囲を制限する。$0 \leq x \leq x_{max}$ の部分を描くものとすれば、

$$\theta_{max} = (x_{max}/a)^{1/2} \quad (43)$$

許容できる最大の長さを δ とした場合、$\theta=0$ と $\theta=d\theta$ 間の距離を δ 以下になるように分割数を定める。

（５）　円および楕円の作成方法

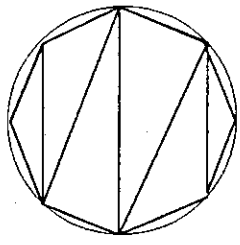


図　13

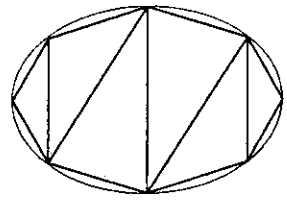


図　14

（１）（２）で求めた円周の座標を用いてポリゴンを作成する。最大分割数を n_{max} とすると、n_{max}-2個のポリゴンを作成する。これで、１組の三角片となる。

１．１．４　基本プリミティブのポリゴンデータ作成方法

すべての基本プリミティブにおいて、まずｘｙ平面に平行なプリミティブ面の基本的なポリゴンデータを作成し、次にDEPTH,WIDTH,HEIGHT軸の方向を用いて、ｘｙｚ軸から、DEPTH,WIDTH,HEIGHT軸に座標変換させる回転マトリックスを求め、基本面のポリゴンデータを回転した後原点移動して、実際の面を作成する過程をとっている。各プリミティブの作成方法を以下に示す。ここで用いているDEPTH,WIDTH,HEIGHTは、各々そのベクトルの大きさ(絶対値)を示している。

（１）　ＢＯＸの場合

原点をＯ(0,0,0)とし、DEPTH,WIDTH,HEIGHTの３方向に、１２個のポリゴンを作成している。１２個のポリゴンの座標は以下の通りであり、各平面を１組とし、６組の三角片で構成する。

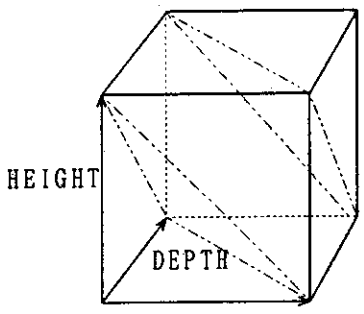


図 15

1. (0,0,0) (-DEPTH,0,0) (0,WIDTH,0)
2. (-DEPTH,0,0) (0,WIDTH,0) (-DEPTH,WIDTH,0)
3. (0,0,HEIGHT) (-DEPTH,0,HEIGHT) (0,WIDTH,HEIGHT)
4. (-DEPTH,0,HEIGHT) (0,WIDTH,HEIGHT) (-DEPTH,WIDTH, HEIGHT)
5. (0,0,0) (-DEPTH,0,0) (0,0,HEIGHT)
6. (-DEPTH,0,0) (0,0,HEIGHT) (-DEPTH,0,HEIGHT)
7. (0,WIDTH,0) (-DEPTH,WIDTH,0) (0,WIDTH,HEIGHT)
8. (-DEPTH,WIDTH,0) (0,WIDTH,HEIGHT) (-DEPTH,WIDTH, HEIGHT)
9. (0,0,0) (0,WIDTH,0) (0,0,HEIGHT)
10. (0,WIDTH,0) (0,0,HEIGHT) (0,WIDTH,HEIGHT)
11. (-DEPTH,0,0) (-DEPTH,WIDTH,0) (-DEPTH,0,HEIGHT)
12. (-DEPTH,WIDTH,0) (-DEPTH,0,HEIGHT) (-DEPTH,WIDTH,HEIGHT)

以上のように基本的なポリゴンデータを作成した後、後述の座標変換その２を行っている。

（２） ＣＯＮＥの場合

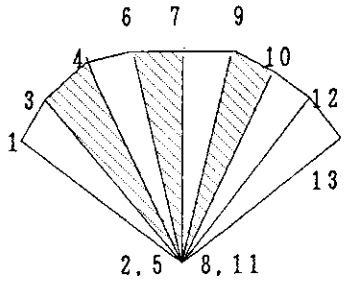


図 16

原点をＯ(0,0,0)とし、(0,0,HEIGHT)の方向に円錐の頂点があるとする。底面の円は、RADIUSをｒとおき3.1.3節の(5)の方法でn_{max}-2個のポリゴンデータで近似する。側面は、円周のｘ，ｙ座標と頂点の座標を用いて、(0,0,HEIGHT)，(x_n, y_n,0)，(x_{n+1}, y_{n+1},0)の３点で構成するが、三角片を作成関係上3/2*n_{max}-1個のポリゴンで近似する。図3.16に側面の展開図を示す。斜線部は、ポリゴンが重なる部分である。

以上のように基本的なポリゴンデータを作成した後、後述の座標変換その1を行っている。三角片は２組作成される。

（３） ＣＹＬＩＮＤＥＲの場合

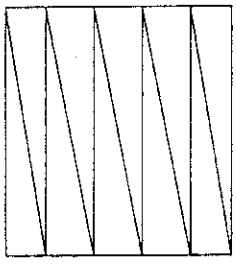


図 17

原点をＯ(0,0,0)とし、(0,0,HEIGHT)の方向に円柱の上面があるとする。底面および上面は、RADIUSをｒとおき1.1.3節の(5)の方法で、ｚ=0およびｚ=HEIGHTの各一組の三角片で近似する。側面は、円周のｘ，ｙ座標を用いて(x_n, y_n,0)，(x_{n+1}, y_{n+1},0)，(x_n, y_n,HEIGHT)の３点と、(x_{n+1}, y_{n+1},0)，(x_n, y_n,HEIGHT)，(x_{n+1}, y_{n+1},HEIGHT)の３点で図17で示すように近似し、１組の三角片を作成する。以上のように基本的なポリゴンデータを作成した後、後述の座標変換その１を行っている。

（４）　ＳＰＨＥＲＥの場合

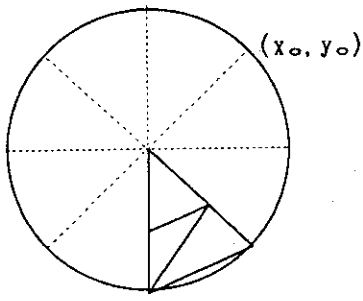


図　18 Z=0平面での断面

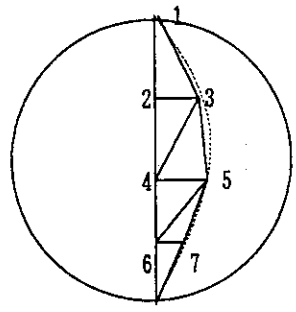


図　19 X=0平面での断面

原点をＯ(0,0,0)としたとき、ｚ=0平面上でRADIUSをｒとおき、1.1.3節の(1)の方法で円周上のｘ。，ｙ。座標を定める。図に示すように、球の側面を経線を作る方向に8n分割し、8n-2個のポリゴンデータ即ち１組の三角片を作成する。合計で、8n個の三角片が構成される。球の場合は回転しても形が変わらないので、最初から原点の座標を組込んでポリゴンデータを作成している。

（５）　ＥＬＬＩＰＳの場合

ＳＰＨＥＲＥの場合と同様に原点をＯ(0,0,0)としたとき、ｚ=0平面上で1.1.3節の(2)の方法で楕円周上のｘ。，ｙ。座標を定める。楕円体の側面を経線を作る方向に分割し、１組の三角片を作成する。合計で、分割数分の三角片が作成される。

以上のように、基本的なポリゴンデータを作成した後、後述の座標変換その２を行っている。

（６）　Ｅ＿ＣＬＩＮＤＥＲの場合

ＣＹＬＩＮＤＥＲの場合と同様、原点をＯ(0,0,0)とし、(0,0,HEIGHT)の方向に円柱の上面があるとする。底面および上面は1.1.3節の(5)の方法で、ｚ=0およびｚ=HEIGHTの各一組の三角片で近似する。側面は円周のｘ，ｙ座標を用いて、図18で示すように１組の三角片を作成する。以上のように基本的なポリゴンデータを作成した後、後述の座標変換その２を行っているその１を行っている。

（７）　ＨＹＰＥＲの場合

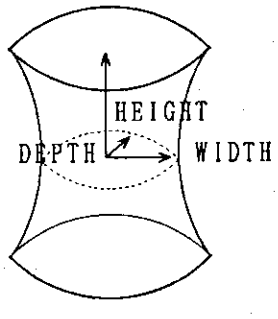


図　20

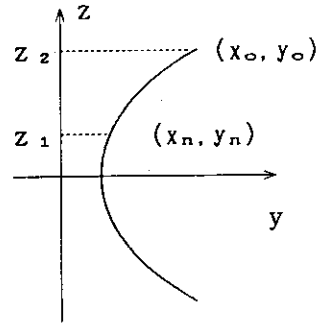


図　21

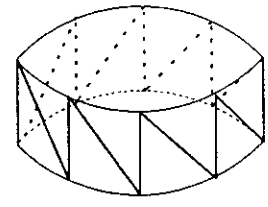


図　22

原点をＯ(0,0,0)とし、(0,0,±HEIGHT)の方向に上面および底面があるとする。底面および上面は1.1.3節の(5)の方法で、ｚ=±HEIGHTにおいて、２組の三角片を作成する。

側面については1.1.3節の(3)の方法で、正のｙ軸を対称とした双曲線上の座標ｘ，z_nを求め、$z = z_2$(=HEIGHT)時の楕円周上のx_o，y_oは、(5)の方法で既に求まっているので、$z = z_1$時の楕円周上のx_n，y_n座標を、図21に示すように比例配分によって求め、側面を構成する（x_o，y_o，z_1），（x_{o+1}，y_{o+1}，z_1），（x_n，y_n，z_2）の３点と、（x_n，y_n，z_2），（x_{n+1}，y_{n+1}，z_2），（x_{o+1}，y_{o+1}，z_1）の３点の２n_{max}個のポリゴンで近似し図22で示すように１組の三角片を作成する。それを側面の分割数だけ繰り返し、分割数分の三角片が作成される。

以上のように基本的なポリゴンデータを作成した後、後述の座標変換その２を行っている。

（８）　ＰＡＲＡＢＯＬＡの場合

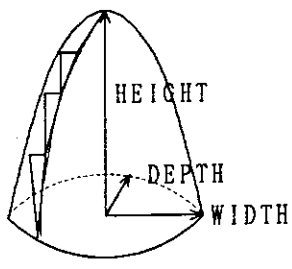


図　23

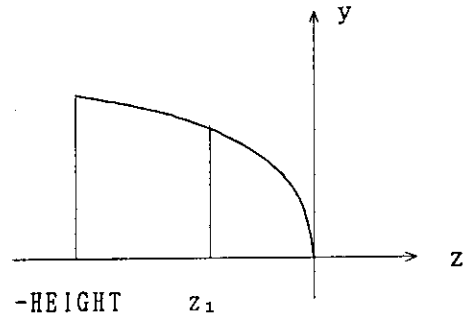


図　24

原点をＯ(0,0,0)とし、底面は1.1.3節の(5)の方法で、ｚ=-HEIGHTの１組の三角片を作成する。側面については1.1.3節の(4)の方法で、負のｚ軸を対称とした放物線上の座標ｘ，z_nを求め、$z = z_2$(=HEIGHT)時の楕円周上のx_o，y_oは、(5)の方法で既に求まっているので、$z = z_1$時の楕円周上のx_n，y_n座標を図24に示すように比例配分によって求め、球の場合と同様に図23に示すように、経線方向に三角片を作成する。

以上のように基本的なポリゴンデータを作成した後、後述の座標変換その２を行っている。

（９）　ＴＯＲＵＳ、ＴＯＲＵＳ＿Ｑの場合

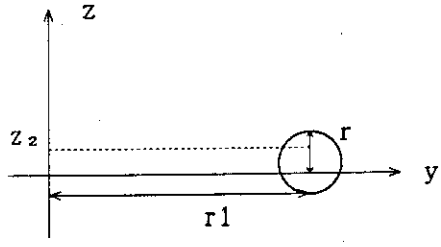


図　25

原点をＯ(0,0,0)とし、1.1.3節の(1)の方法で最外周の半径をr1+rとおいたときの最外周上の座標x_o，y_oを求め、同様に、半径をｒとおいたときの小円の座標ｙ，z_nを求める。側面については、ｚ=0時の外円周上のx_o，y_o座標はすでに求めているので、$z = z_2$時の外円周上のx_n，y_n座標を比例配分によって求め、側面を構成する（x_o，y_o，z_1），（x_{o+1}，y_{o+1}，z_1），（x_n，y_n，z_2）の３点と、（x_n，y_n，z_2），（x_{n+1}，y_{n+1}，z_2），（x_{o+1}，y_{o+1}，z_1）の３点の２n_{max}個のポリゴンを近似し、１組の三角片を作成し、それを側面の分割数だけ繰り返す。$z = r$になったら、次に内円周上のx_n，y_n座標を比例配分によって求め、外周と同様に側面を

構成する（x_o, y_o, z_1），（x_{o+1}, y_{o+1}, z_1），（x_n, y_n, z_2）の３点と、（x_n, y_n, z_2），（x_{n+1}, y_{n+1}, z_2），（x_{o+1}, y_{o+1}, z_1）の３点の２n_{max}個のポリゴンを近似し、同様に側面の分割数だけ繰り返す。三角片は、分割数分作成される。

以上のように基本的なポリゴンデータを作成した後、ＴＯＲＵＳの場合は、後述の座標変換その１を行っている。ＴＯＲＵＳ＿Ｑの場合は、ポリゴンデータの範囲を$-r1-r \leqq x \leqq 0$、$0 \leqq y \leqq r1+r$に限定し、断面の円のポリゴンデータを近似して付け加えている。座標変換もその２を用いている。

（１０）　ＴＯＲＵＳ＿Ｅの場合

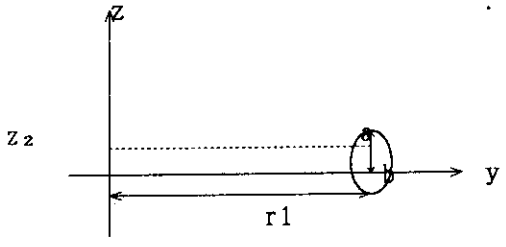


図　26

ＴＯＲＵＳ、ＴＯＲＵＳ＿Ｑと同様に、原点をＯ(0,0,0)とし、1.1.3節の(1)の方法で最外周の半径を$r1+b$とおいたときの最外周上の座標x_o, y_oを求め、同様に、径をa, bとおいたときの小楕円の座標y, z_nを求める。側面については、z=0時の外円周上のx_o, y_o座標はすでに求めているので、$z = z_2$時の外円周上のx_n, y_n座標を比例配分によって求め、側面を構成する（x_o, y_o, z_1），（x_{o+1}, y_{o+1}, z_1），（x_n, y_n, z_2）の３点と、（x_n, y_n, z_2），（x_{n+1}, y_{n+1}, z_2），（x_{o+1}, y_{o+1}, z_1）の３点の２n_{max}個のポリゴンを近似し、１組の三角片を作成し、それを側面の分割数だけ繰り返す。$z = a$になったら、次に内円周上のx_n, y_n座標を比例配分によって求め、外周と同様に側面を構成する（x_o, y_o, z_1），（x_{o+1}, y_{o+1}, z_1），（x_n, y_n, z_2）の３点と、（x_n, y_n, z_2），（x_{n+1}, y_{n+1}, z_2），（x_{o+1}, y_{o+1}, z_1）の３点の２n_{max}個のポリゴンを近似し、同様に側面の分割数だけ繰り返す。三角片は分割数分作成される。

以上のように基本的なポリゴンデータを作成した後、後述の座標変換その１を行っている。

１．１．５　座標変換

（１）　その１

－　プリミティブの方向がHEIGHT(ｚ軸)のみによって決まる場合　－

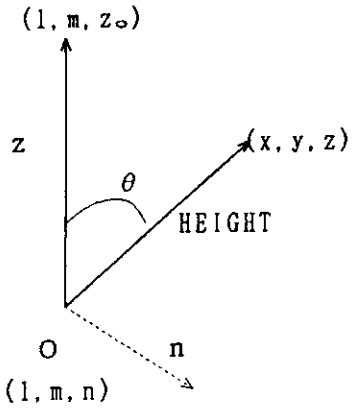


図　27

ベクトルHEIGHTを、ｚ軸ベクトルに重ねるために、双方の法線ｎの回りにθ回転するマトリックスを求める。

ベクトルHEIGHTを、ｚ軸ベクトルに重ね合わせるには回転角度は＋θ、逆にｚ軸ベクトルを、ベクトルHEIGHTに重ね合わせるには回転角度は－θになる。

回転軸が、(l, m, n)を通るとすれば、変換式は次のようになる。

$$[X\ Y\ Z\ H]=[x\ y\ z\ 1]\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\-l&-m&-n&1\end{bmatrix}\begin{bmatrix}R\end{bmatrix}\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\l&m&n&1\end{bmatrix} \tag{44}$$

法線ベクトルｎ(n_1, n_2, n_3)とすると、回転マトリックスＲは次式で表される。

$$\begin{bmatrix}R\end{bmatrix}=\begin{bmatrix}n_1^2+(1-n_1^2)\cos\theta & n_1n_2(1-\cos\theta)+n_3\sin\theta & n_1n_3(1-\cos\theta)-n_2\sin\theta & 0\\ n_1n_2(1-\cos\theta)-n_3\sin\theta & n_2^2+(1-n_2^2)\cos\theta & n_2n_3(1-\cos\theta)+n_1\sin\theta & 0\\ n_1n_3(1-\cos\theta)+n_2\sin\theta & n_2n_3(1-\cos\theta)-n_1\sin\theta & n_3^2+(1-n_3^2)\cos\theta & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \tag{45}$$

また、法線ベクトルｎは、(l, m, n), (x, y, z), (l, m, z_0)を通る平面の方程式から求めることができる。原点をＯ(l, m, n)としたときの、ベクトルHEIGHTおよびｚの成分をそれぞれ(L_1, M_1, N_1), (L_2, M_2, N_2)とすれば、平面の方程式は、以下に示すようになる。

$$\begin{vmatrix}x-l & y-m & z-n\\ L_1 & M_1 & N_1\\ L_2 & M_2 & N_2\end{vmatrix} = 0 \tag{46}$$

従って、法線ｎの成分は、$(n_1, n_2, n_3)=(M_1N_2-M_2N_1, L_2N_1-L_1N_2, L_1M_2-L_2M_1)$となる。

（２）　その２

－　プリミティブの方向がDEPTH(ｘ軸)、WIDTH(ｙ軸)、HEIGHT(ｚ軸)によって決まる場合　－

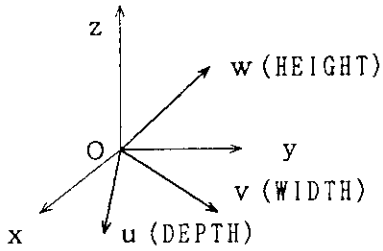


図　28

ｕｖｗ座標系をｘｙｚ座標系に変換する回転マトリクスを求める。

回転軸が、(l,m,n)を通るとすれば、変換式は次のようになる。

$$[X\ Y\ Z\ H]=[x\ y\ z\ 1]\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\-l&-m&-n&1\end{bmatrix}\begin{bmatrix}R\end{bmatrix}\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\l&m&n&1\end{bmatrix} \tag{47}$$

回転マトリックスＲは、次のように求められる。

$$\begin{bmatrix}R\end{bmatrix}=\begin{bmatrix}U_x&V_x&W_x&0\\U_y&V_y&W_y&0\\U_z&V_z&W_z&0\\0&0&0&1\end{bmatrix} \tag{48}$$

ただし、

U_x : $\vec{u}$のｘ軸に対する方向余弦　$u_x/|u|$
U_y : $\vec{u}$のｙ軸に対する方向余弦　$u_y/|u|$
U_z : $\vec{u}$のｚ軸に対する方向余弦　$u_z/|u|$
V_x : $\vec{v}$のｘ軸に対する方向余弦　$v_x/|v|$
V_y : $\vec{v}$のｙ軸に対する方向余弦　$v_y/|v|$
V_z : $\vec{v}$のｚ軸に対する方向余弦　$v_z/|v|$
W_x : $\vec{w}$のｘ軸に対する方向余弦　$w_x/|w|$
W_y : $\vec{w}$のｙ軸に対する方向余弦　$w_y/|w|$
W_z : $\vec{w}$のｚ軸に対する方向余弦　$w_z/|w|$

元々のｘｙｚ座標系は左手系でｘ座標が負になっている。従ってまず、ｘ座標を正に変換して、座標変換後、再度ｘ座標を変換し負の値にするので、実際にＲマトリックスの値は、以下のようになる。

$$\begin{bmatrix}R\end{bmatrix}=\begin{bmatrix}-1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}\begin{bmatrix}U_x&V_x&W_x&0\\U_y&V_y&W_y&0\\U_z&V_z&W_z&0\\0&0&0&1\end{bmatrix}\begin{bmatrix}-1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}=\begin{bmatrix}U_x&-V_x&-W_x&0\\-U_y&V_y&W_y&0\\-U_z&V_z&W_z&0\\0&0&0&1\end{bmatrix} \tag{49}$$

逆に、ｘｙｚ座標系をｕｖｗ座標系に変換する回転マトリクスでは、Ｒが以下のようになる。

$$\begin{bmatrix}R\end{bmatrix}=\begin{bmatrix}U_x&-U_y&-U_z&0\\-V_x&V_y&V_z&0\\-W_x&W_y&W_z&0\\0&0&0&1\end{bmatrix} \tag{50}$$

１．２　プリミティブの合成

施設内の設備の形状は、2.1節で説明した基本プリミティブをＣＳＧ演算で合成することにより表示される。本プログラム中では、基本プリミティブをポリゴンデータに落としてあるので、２次もしくは４次の曲面のＣＳＧ演算を、平面同士の演算に置き換えることができる。従って、プリミティブの合成は、平面と直線の交点を求めることを基本に据えて行う。

１．２．１　基本的な考え方

今、仮にプリミティブＡ(BOX)と、プリミティブＢ(CYLINDER)があるとして、Ａ＋ＢのＣＳＧ演算を行うとすると、その形は図29の斜線部に示すような形になり、Ａ－ＢのＣＳＧ演算を行うとすると、その形は図30の斜線部に示すような形になる。

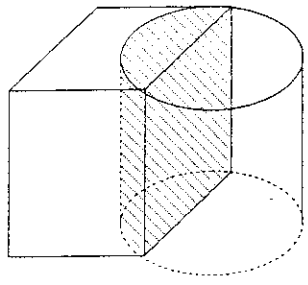


図　29　Ａ＋Ｂ

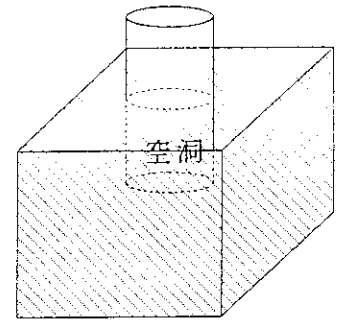


図　30　Ａ－Ｂ

基本プリミティブＡの物体を構成する式群を$f_A(x,y,z)$(実際は、1.1.2で示したように複数の式になる場合もある。)、基本プリミティブＢの物体を構成する式群を、$f_B(x,y,z)$とおくと、Ａ＋Ｂの部分は、$f_A(x,y,z)\leqq 0$ & $f_B(x,y,z)\leqq 0$を成り立たせる(x,y,z)を用いて、ポリゴンデータを作成することであり、Ａ－Ｂの部分は、$f_A(x,y,z)\leqq 0$ & $\overline{f_B(x,y,z)}\leqq 0$を成り立たせる$(x,y,z)$を用いて、ポリゴンデータを作成することである。しかし、、新しい物体Ａ＋ＢもしくはＡ－Ｂを構成する式$f_{A+B}(x,y,z)$もしくは、$f_{A-B}(x,y,z)$を作り直し、ポリゴンデータを再構成することは難しいので、本プログラムでは、基本プリミティブを構成する式の正負の判定に加えて、ポリゴンの面および辺同士の交点の関係を用いてポリゴンデータの再構成を行うことにした。

３つ以上の項からなるＣＳＧ演算、例えばＡ＋Ｂ－Ｃの場合は、演算は左から順に行われる。まずＡ＋Ｂの演算を行いその結果に対して、（Ａ＋Ｂ）－Ｃが行われるので、－Ｃの項の演算を行う際には、ＡおよびＢの原形は保証されていない。

１．２．２　ポリゴンデータの合成方法

ＣＳＧ演算で合成するポリゴンデータは、基本プリミティブＡのポリゴンデータの頂点が基本プリミティブＢの内部にあるか外部にあるか、また基本プリミティブＢのポリ

ゴンデータの頂点が基本プリミティブＡの内部にあるか外部にあるかを全てのポリゴンについて場合分けし、あるポリゴンデータが、他のプリミティブのポリゴンデータと交わっているかどうかを判定して作成し直している。

１つのプリミティブが他のプリミティブを貫いたり、貫かれたりしている場合は、ポリゴンデータの頂点が３点ともプリミティブ外に出ているので、前もってポリゴンデータをプリミティブ内部にあるものとないものに分割しておく必要がある。

これ以降説明のために、プリミティブＡとプリミティブＢの演算、即ちＡ＋Ｂもしくは Ａ－Ｂを行うとし、どの場合もＡは必ず正で、プリミティブＡを構成する式を $f_A(x, y, z)$、プリミティブＢを構成する式を $f_B(x, y, z)$ とし、プリミティブＡ中のポリゴンデータ座標を (x_{A1}, y_{A1}, z_{A1}), (x_{A2}, y_{A2}, z_{A2}), (x_{A3}, y_{A3}, z_{A3})、プリミティブＢ中のポリゴンデータ座標を (x_{B1}, y_{B1}, z_{B1}), (x_{B2}, y_{B2}, z_{B2}), (x_{B3}, y_{B3}, z_{B3}) とする。

（１）　プリミティブＢの１ポリゴンデータが３点共プリミティブＡの外部にある場合

（ $f_A(x_{B1}, y_{B1}, z_{B1}) > 0$ ＆ $f_A(x_{B2}, y_{B2}, z_{B2}) > 0$ ＆ $f_A(x_{B3}, y_{B3}, z_{B3}) > 0$ ）

Ａの１ポリゴン平面とＢの１ポリゴン平面が交わり、Ｂのポリゴンデータ３点共Ａのプリミティブ外部にあり、またＡのポリゴンデータも３点共Ｂのプリミティブ外部にある場合において、Ｂのポリゴンの２辺がポリゴンＡを貫いている場合(図31)、もしくはＡポリゴンの２辺がポリゴンＢを貫いている場合(図32)、もしくはＡＢのポリゴンの１辺がお互いを貫いている場合(図33)は、（３）（４）（５）の場合に当てはまらないが、当然ＣＳＧ演算の対象になるので、（３）（４）（５）の場合に当てはまるようにポリゴンを分解する。Ａ，Ｂのポリゴン共に３個ずつに分解する。

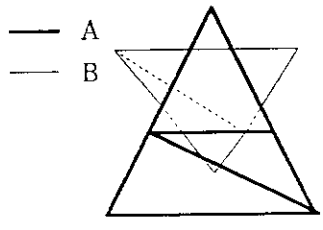


図　31

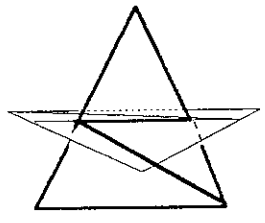


図　32

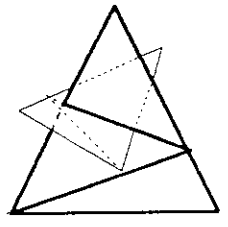


図　33

（２）　プリミティブＢの１ポリゴンデータが３点共プリミティブＡ上にあり、
プリミティブＡのポリゴンデータが３点共プリミティブＢ内にない場合

（ $f_A(x_{B1}, y_{B1}, z_{B1}) = 0$ ＆ $f_A(x_{B2}, y_{B2}, z_{B2}) = 0$ ＆ $f_A(x_{B3}, y_{B3}, z_{B3}) = 0$
＆ $f_B(x_{A1}, y_{A1}, z_{A1}) > 0$ ＆ $f_B(x_{A2}, y_{A2}, z_{A2}) > 0$ ＆ $f_B(x_{A3}, y_{A3}, z_{A3}) > 0$ ）

Ｂの１ポリゴン平面がＡのプリミティブの１平面上にあり、Ａのポリゴンデータが３点共プリミティブ外部にある場合も（３）（４）（５）の場合に当てはまらないがやはり、ＣＳＧ演算の対象になるので、（３）（４）（５）の場合に当てはまるようにＡのポリゴンを分解する。

（３）　プリミティブＢの１ポリゴンデータが３点共、プリミティブＡの面上にある場合

（ $f_A(x_{B1}, y_{B1}, z_{B1}) = 0$ ＆ $f_A(x_{B2}, y_{B2}, z_{B2}) = 0$ ＆ $f_A(x_{B3}, y_{B3}, z_{B3}) = 0$ ）

① プリミティブBの1ポリゴンデータの3点共、プリミティブAの1ポリゴンデータ内部にある場合。

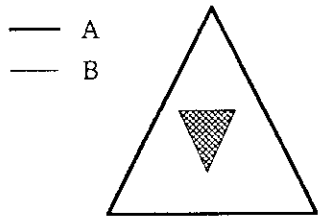


図 34 A＋B

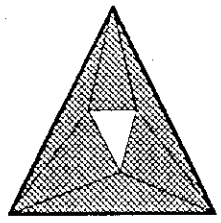


図 35 A－B

Aの1ポリゴン平面とBの1ポリゴン平面が重なり、Bのポリゴンデータ3点共、Aの1ポリゴン平面内にある場合は、図に示すようにA＋Bの場合は1個、A－Bの場合は、6個のポリゴンに分ける。

② プリミティブBの1ポリゴンデータの内2点が、プリミティブAの1ポリゴンデータ内部にある場合。

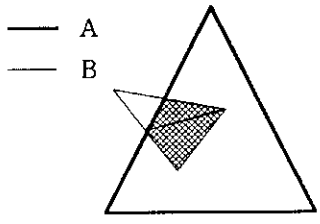


図 36 A＋B

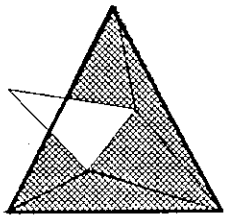


図 37 A－B

Aの1ポリゴン平面とBの1ポリゴン平面が重なり、Bのポリゴンデータの内2点が、Aの1ポリゴン平面内にある場合において、Bのポリゴンの2辺がAの、ポリゴンの1辺と交わっている場合は、図に示すようにA＋Bの場合は2個、A－Bの場合は、5個のポリゴンに分ける。

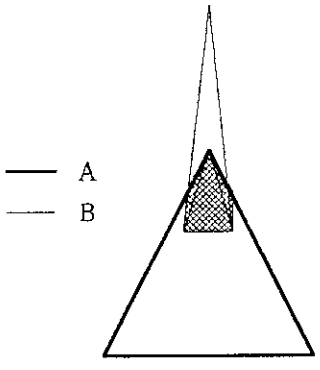


図 38 A＋B

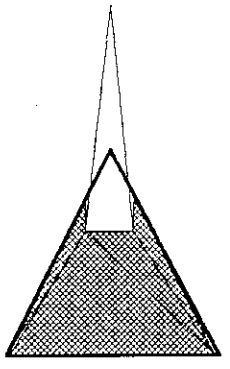


図 39 A－B

Aの1ポリゴン平面とBの1ポリゴン平面が重なり、Bのポリゴンデータの内2点が、Aの1ポリゴン平面内にある場合において、Bのポリゴンの2辺がAの、ポリゴンの2辺と交わっている場合は、図に示すようにA＋Bの場合は3個、A－Bの場合は、4個のポリゴンに分ける。

③ プリミティブBの1ポリゴンデータの内1点が、プリミティブAの1ポリゴンデータ内部にある場合。

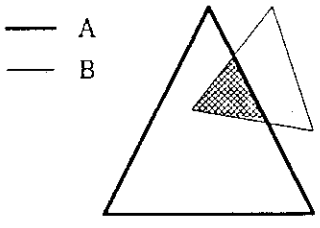


図 40 A＋B

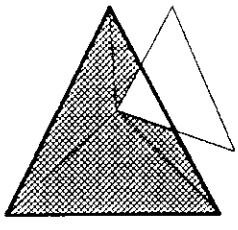


図 41 A－B

Aの1ポリゴン平面とBの1ポリゴン平面が重なり、Bのポリゴンデータの内1点が、Aの1ポリゴン平面内にある場合において、Bのポリゴンの2辺がAの、ポリゴンの1辺と交わっている場合は、図に示すようにA＋Bの場合は1個、A－Bの場合は、4個のポリゴンに分ける。

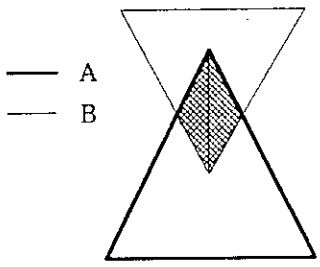


図 42 A＋B

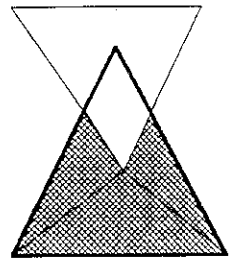


図 43 A－B

Aの1ポリゴン平面とBの1ポリゴン平面が重なり、Bのポリゴンデータの内1点が、Aの1ポリゴン平面内にある場合において、Bのポリゴンの1辺ずつが、それぞれAのポリゴンの1辺ずつと交わっている場合は、図に示すようにA＋Bの場合は2個、A－Bの場合は、3個のポリゴンに分ける。

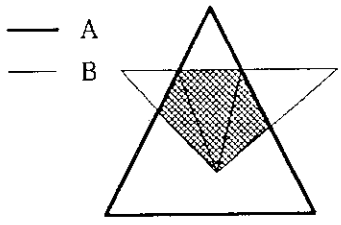


図 44 A＋B

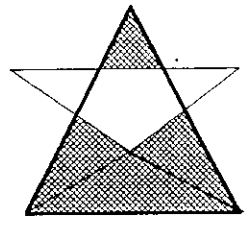


図 45 A－B

Aの1ポリゴン平面とBの1ポリゴン平面が重なり、Bのポリゴンデータの内1点が、Aの1ポリゴン平面内にある場合において、Bのポリゴンの2辺ずつが、それぞれAのポリゴンの1辺ずつと交わっている場合は、図に示すようにA＋Bの場合は3個、A－Bの場合は、4個のポリゴンに分ける。

④ プリミティブBの1ポリゴンデータの内3点が、プリミティブAの1ポリゴンデータ外部にある場合。

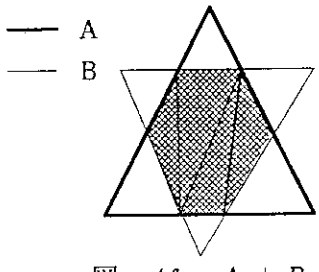


図 46 A＋B

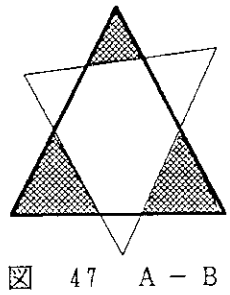


図 47 A－B

Aの1ポリゴン平面とBの1ポリゴン平面が重なり、Bのポリゴンデータの内3点が、Aの1ポリゴン平面外にある場合において、Bのポリゴンの1辺ずつが、それぞれAのポリゴンの2辺ずつと交わっている場合は、図に示すようにA＋Bの場合は4個、A－Bの場合は、3個のポリゴンに分ける。

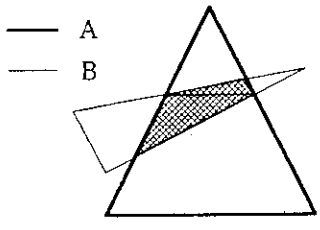


図 48 A＋B

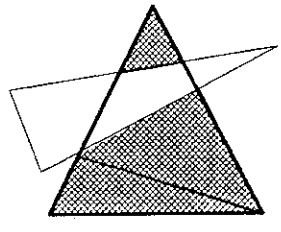


図 49 A－B

Aの1ポリゴン平面とBの1ポリゴン平面が重なり、Bのポリゴンデータの内3点が、Aの1ポリゴン平面外にある場合において、Bのポリゴンの2辺が、それぞれAのポリゴンの2辺と交わっている場合は、図に示すようにA＋Bの場合2個、A－Bの場合は、3個のポリゴンに分ける。

（4） プリミティブBの1ポリゴンデータの内1点が、プリミティブA内にある場合

$(f_A(x_{B1}, y_{B1}, z_{B1}) \leqq 0 \ \& \ f_A(x_{B2}, y_{B2}, z_{B2}) > 0 \ \& \ f_A(x_{B3}, y_{B3}, z_{B3}) > 0)$

① プリミティブAの1ポリゴンデータの内1点が、プリミティブBの1ポリゴンデータ内部にある場合。

$(f_B(x_{A1}, y_{A1}, z_{A1}) \leqq 0 \ \& \ f_B(x_{A2}, y_{A2}, z_{A2}) > 0 \ \& \ f_B(x_{A3}, y_{A3}, z_{A3}) > 0)$

Ａの１ポリゴン平面とＢの１ポリゴン平面が交わり、Ｂのポリゴンデータの内１点がＡのプリミティブ内にあり、Ａのポリゴンデータの内１点がＢのプリミティブ内部にある場合について考える。(以下の図内で・で示してある点が相手のプリミティブ内に入っている点である。)

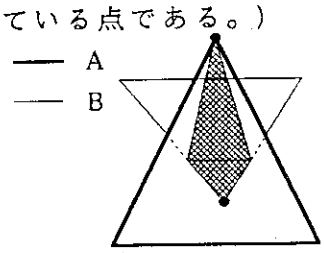


図　50　Ａ＋Ｂ

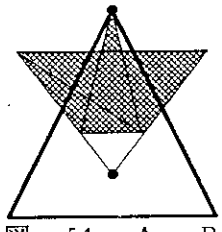


図　51　Ａ－Ｂ

Ｂのポリゴンの２辺がポリゴンＡを貫いている場合は、図に示すようにＡ＋Ｂの場合２個、Ａ－Ｂの場合３個のポリゴンに分ける。

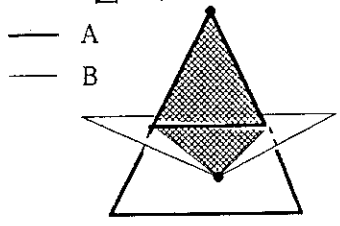


図　52　Ａ＋Ｂ

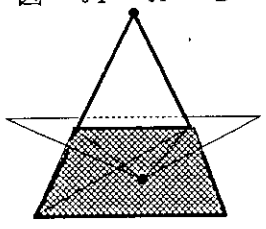


図　53　Ａ－Ｂ

Ａのポリゴンの２辺がポリゴンＢを貫いている場合は、図に示すようにＡ＋Ｂの場合２個、Ａ－Ｂの場合３個のポリゴンに分ける。

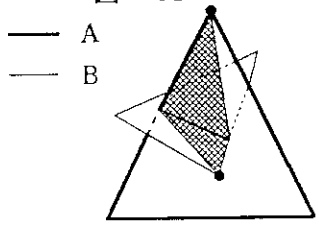


図　54　Ａ＋Ｂ

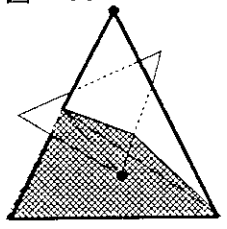


図　55　Ａ－Ｂ

Ｂのポリゴンの１辺がポリゴンＡを貫き、Ａのポリゴンの１辺がポリゴンＢを貫く場合は、図に示すようにＡ＋Ｂの場合２個、Ａ－Ｂの場合３個のポリゴンに分ける。

②　プリミティブＡの１ポリゴンデータの内２点が、プリミティブＢの１ポリゴンデータ内部にある場合。

$$\left(f_B(x_{A1}, y_{A1}, z_{A1}) \leqq 0 \ \& \ f_B(x_{A2}, y_{A2}, z_{A2}) \leqq 0 \ \& \ f_B(x_{A3}, y_{A3}, z_{A3}) > 0 \right)$$

Ａの１ポリゴン平面とＢの１ポリゴン平面が交わり、Ｂのポリゴンデータの内１点がＡのプリミティブ内にあり、Ａのポリゴンデータの内２点がＢのプリミティブ内部にある場合について考える。(以下の図内で・で示してある点が相手のプリミティブ内に入っている点である。)

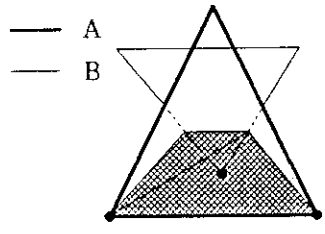


図　56　Ａ＋Ｂ

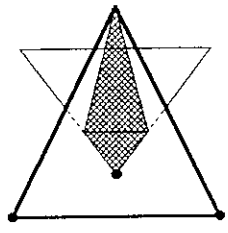


図　57　Ａ－Ｂ

Ｂのポリゴンの２辺がポリゴンＡを貫いている場合は、図に示すようにＡ＋Ｂの場合３個、Ａ－Ｂの場合２個のポリゴンに分ける。

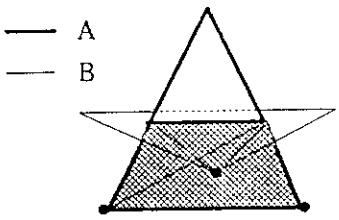


図 58 A＋B

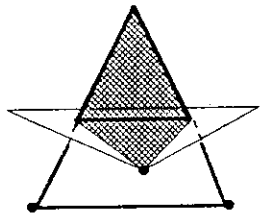


図 59 A－B

Aのポリゴンの2辺がポリゴンBを貫いている場合は、図に示すようにA＋Bの場合3個、A－Bの場合2個のポリゴンに分ける。

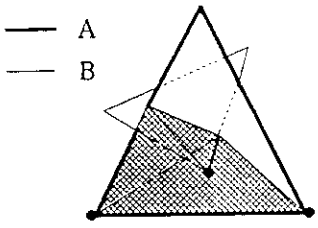


図 60 A＋B

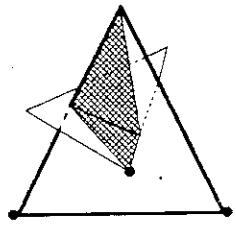


図 61 A－B

Bのポリゴンの1辺がポリゴンAを貫き、Aのポリゴンの1辺がポリゴンBを貫く場合は、図に示すようにA＋Bの場合3個、A－Bの場合2個のポリゴンに分ける。

③ プリミティブAの1ポリゴンデータの内3点共、プリミティブBの内部にある場合。

($f_B(x_{A1}, y_{A1}, z_{A1}) \leqq 0$ & $f_B(x_{A2}, y_{A2}, z_{A2}) \leqq 0$ & $f_B(x_{A3}, y_{A3}, z_{A3}) \leqq 0$)

Aの1ポリゴン平面とBの1ポリゴン平面が交わり、Bのポリゴンデータの内1点がAのプリミティブ内にあり、Aのポリゴンデータの内3点共Bのプリミティブ内部にある場合について考える。(以下の図内で・で示してある点が相手のプリミティブ内に入っている点である。)

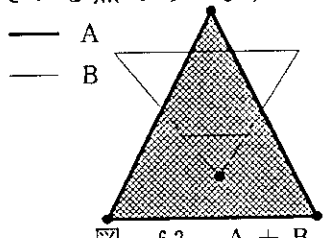


図 62 A＋B

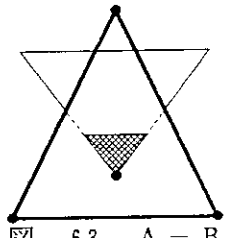


図 63 A－B

Bのポリゴンの2辺がポリゴンAを貫いている場合は、図に示すようにA＋Bの場合2個、A－Bの場合1個のポリゴンに分ける。

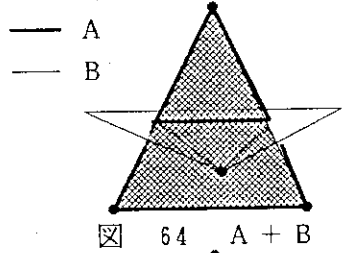


図 64 A＋B

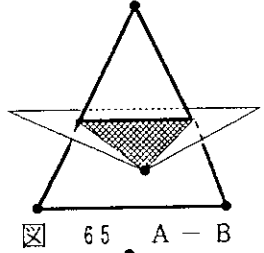


図 65 A－B

Aのポリゴンの2辺がポリゴンBを貫いている場合は、図に示すようにA＋Bの場合2個、A－Bの場合1個のポリゴンに分ける。

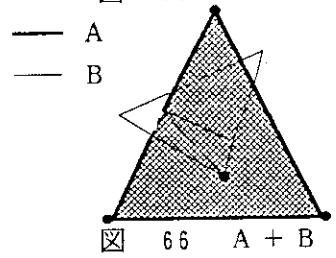


図 66 A＋B

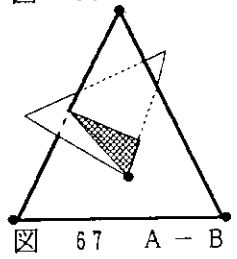


図 67 A－B

Bのポリゴンの1辺がポリゴンAを貫き、Aのポリゴンの1辺がポリゴンBを貫く場合は、図に示すようにA＋Bの場合2個、A－Bの場合1個のポリゴンに分ける。

(5) プリミティブBの1ポリゴンデータの内2点が、プリミティブA内にある場合

($f_A(x_{B1}, y_{B1}, z_{B1}) \leqq 0$ & $f_A(x_{B2}, y_{B2}, z_{B2}) \leqq 0$ & $f_A(x_{B3}, y_{B3}, z_{B3}) > 0$)

① プリミティブAの1ポリゴンデータの内1点が、プリミティブBの1ポリゴンデータ内部にある場合。

($f_B(x_{A1}, y_{A1}, z_{A1}) \leqq 0$ & $f_B(x_{A2}, y_{A2}, z_{A2}) > 0$ & $f_B(x_{A3}, y_{A3}, z_{A3}) > 0$)

Aの1ポリゴン平面とBの1ポリゴン平面が交わり、Bのポリゴンデータの内2点がAのプリミティブ内にあり、Aのポリゴンデータの内1点がBのプリミティブ内部にある場合について考える。(以下の図内で・で示してある点が相手のプリミティブ内に入っている点である。)

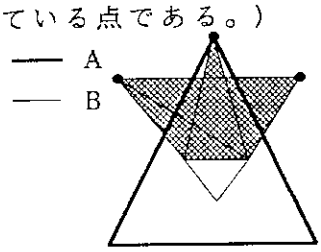


図 68 A＋B

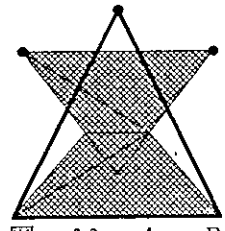


図 69 A－B

Bのポリゴンの2辺がポリゴンAを貫いている場合は、図に示すようにA＋Bの場合3個、A－Bの場合4個のポリゴンに分ける。

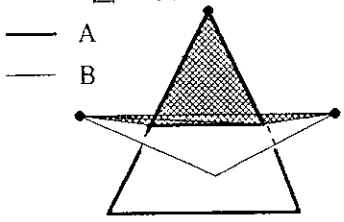


図 70 A＋B

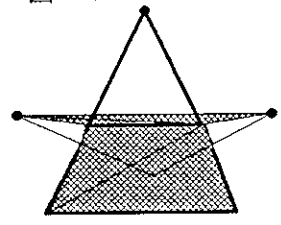


図 71 A－B

Aのポリゴンの2辺がポリゴンBを貫いている場合は、図に示すようにA＋Bの場合3個、A－Bの場合4個のポリゴンに分ける。

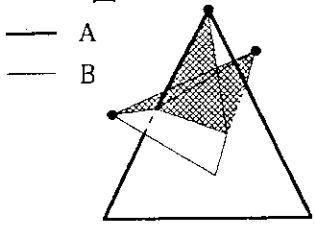


図 72 A＋B

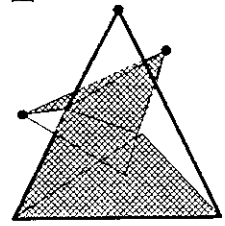


図 73 A－B

Bのポリゴンの1辺がポリゴンAを貫き、Aのポリゴンの1辺がポリゴンBを貫く場合は、図に示すようにA＋Bの場合3個、A－Bの場合4個のポリゴンに分ける。

② プリミティブAの1ポリゴンデータの内2点が、プリミティブBの1ポリゴンデータ内部にある場合。

($f_B(x_{A1}, y_{A1}, z_{A1}) \leqq 0$ & $f_B(x_{A2}, y_{A2}, z_{A2}) > 0$ & $f_B(x_{A3}, y_{A3}, z_{A3}) > 0$)

Aの1ポリゴン平面とBの1ポリゴン平面が交わり、Bのポリゴンデータの内2点がAのプリミティブ内にあり、Aのポリゴンデータの内2点がBのプリミティブ内部にある場合について考える。(以下の図内で・で示してある点が相手のプリミティブ内に入っている点である。)

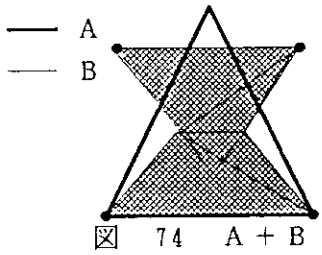


図 74 A＋B

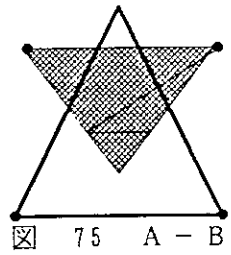


図 75 A－B

Bのポリゴンの2辺がポリゴンAを貫いている場合は、図に示すようにA＋Bの場合4個、A－Bの場合3個のポリゴンに分ける。

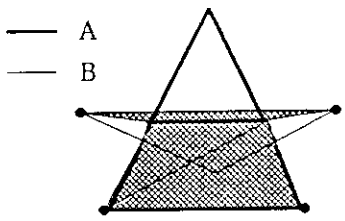


図 76 A＋B

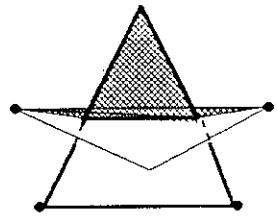


図 77 A－B

Aのポリゴンの2辺がポリゴンBを貫いている場合は、図に示すようにA＋Bの場合4個、A－Bの場合3個のポリゴンに分ける。

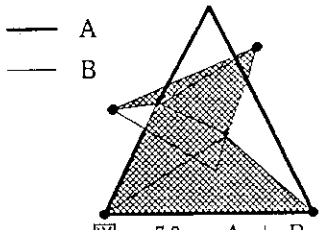


図 78 A＋B

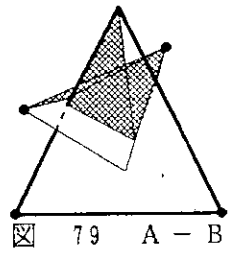


図 79 A－B

Bのポリゴンの1辺がポリゴンAを貫き、Aのポリゴンの1辺がポリゴンBを貫く場合は、図に示すようにA＋Bの場合4個、A－Bの場合3個のポリゴンに分ける。

③ プリミティブAの1ポリゴンデータの内3点共、プリミティブBの内部にある場合。

($f_B(x_{A1}, y_{A1}, z_{A1}) \leqq 0$ & $f_B(x_{A2}, y_{A2}, z_{A2}) \leqq 0$ & $f_B(x_{A3}, y_{A3}, z_{A3}) \leqq 0$)

Aの1ポリゴン平面とBの1ポリゴン平面が交わり、Bのポリゴンデータの内2点がAのプリミティブ内にあり、Aのポリゴンデータの内3点共Bのプリミティブ内部にある場合について考える。(以下の図内で・で示してある点が相手のプリミティブ内に入っている点である。)

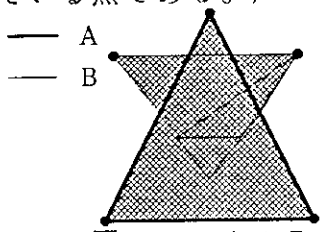


図 80 A＋B

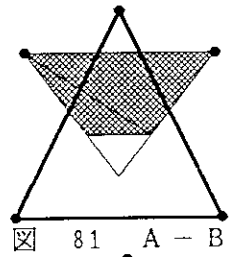


図 81 A－B

Bのポリゴンの2辺がポリゴンAを貫いている場合は、図に示すようにA＋Bの場合3個、A－Bの場合2個のポリゴンに分ける。

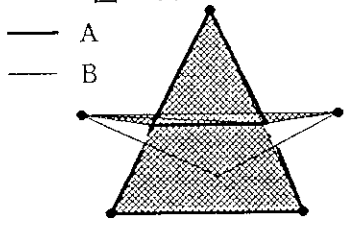


図 82 A＋B

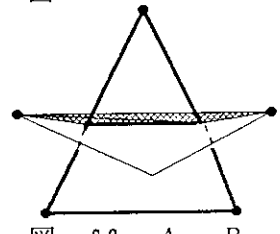


図 83 A－B

Aのポリゴンの2辺がポリゴンBを貫いている場合は、図に示すようにA＋Bの場合3個、A－Bの場合2個のポリゴンに分ける。

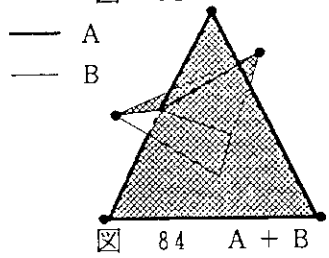


図 84 A＋B

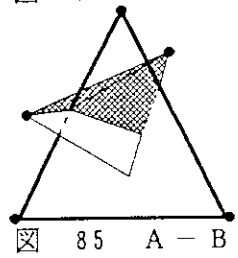


図 85 A－B

Bのポリゴンの1辺がポリゴンAを貫き、Aのポリゴンの1辺がポリゴンBを貫く場合は、図に示すようにA＋Bの場合3個、A－Bの場合2個のポリゴンに分ける。