

ネットワーク分散処理による海中ロボット用 海中環境シミュレータ

正員 黒田 洋 司* 正員 荒 牧 浩 二**
正員 浦 環* 菅 野 崇**
正員 大 和 裕 幸**

Underwater Environment Simulator for Underwater Robots Based on
Distributed Processings in Computer Networks

by Yoji Kuroda, *Member* Koji Aramaki, *Member*
Tamaki Ura, *Member* Takashi Sugano
Hiroyuki Yamato, *Member*

Summary

This paper introduces a simulator system essential to the research of underwater robots which are required to have highly reliable and efficient ability of information processing. The universe of the simulation consists of independent worlds which include specific agents such as robots and researchers. The features of the proposed system are as follows: 1) The whole system is operated in real time on the computers distributed in the network including the actual underwater robots. 2) Each agent program in the simulation world is connected to the programs called "managers" which handle the interactions among the agents. 3) Underwater environment is modeled as well as the robot itself. 4) Multi-user simulation is available, so that researchers in remote places can share the same simulation world.

With this simulator, for example, the software loaded into a robot can be examined by giving sensor readings in a virtual environment to the actual robot swimming in a testing pool. Since the system continuously provides researchers a virtual underwater environment, they can operate their robots as if they were in the ocean.

1. 緒 言

海中ロボットの研究には、運動の制御や信号処理のようにある程度単独で研究可能なものから、それらを組み合わせて全体として機能するロボットを作り上げ、さらにそれを複数台用いたミッションを構築するなど、様々な局面がある。最終的に実用に耐え得るロボットを完成させる目的のためには、そのどの研究局面においても使用可能な汎用性の高いシミュレータシステムが必要とされる。

一般の操船シミュレータ¹⁾は、船と人と環境の3要素からなる制御系の挙動を模擬するシステムで、主に船員の訓練のために用いられている。この制御系の制御主体は人間であるので、それ自体が制御アルゴリズムの研究に用いられることはなく、制御対象の船のダイナミクスの模擬とマンマシンインターフェースの機能に重点がおかれている。

また、自律的な操船システム²⁾などの研究に利用されるシミュレータは、研究内容に即した専用のプログラム構造であることが多く、様々な研究テーマを持った不特定多数の研究者が手軽に利用できるものとは立場を異にしている。

これらのシミュレータと比較して、実ロボット開発全体のためのシミュレータは、実時間性を確保し、さらには実ロボットと接続するための機能などが要求され、シミュレーション時に陥りやすい「現実離れ」を防ぐための十分な配慮が必要である。シミュレーションの舞台となる仮想的

* 東京大学生産技術研究所

** 東京大学大学院工学系研究科

原稿受理 平成7年7月10日

秋季講演会において講演 平成7年11月16, 17日

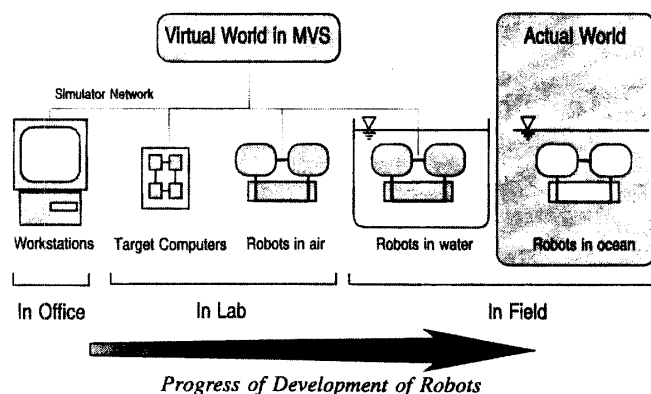


Fig. 1 The integrated research environment

な海は、実際の海がそうであるように、常に存在し、様々な物理的な相互作用が自動的に起こるような、複雑な海であるべきである。

著者らは1993年より、海中ロボット研究のための有効なシミュレーション手法の確立と、そのための道具の構築を目指し、Multi-Vehicle Simulator (MVS)³⁾の開発を進めてきた。本論で提案するこのMVSシステムは「実際の海の代わりに、計算機が作り出す仮想的な海でロボットの様々な実験をするための汎用的な研究ツール」であることが特徴である (Fig. 1)。

2. 仮想環境シミュレータ MVS の概要

2.1 MVS の表す世界

MVS ではシミュレーションの舞台としての抽象的な世界を World と呼び、複数の独立した World が存在する。全 World の集合を Universe と呼び、シミュレーションに関与する主体を Agent と総称する (Fig. 2)。Agent は User と Entity の2種類に分類され、Entity はさらに Vehicle, Station, Obstacle に分類される。User はシミュレーションを行う研究者、Vehicle は海中ロボット、Station は通信

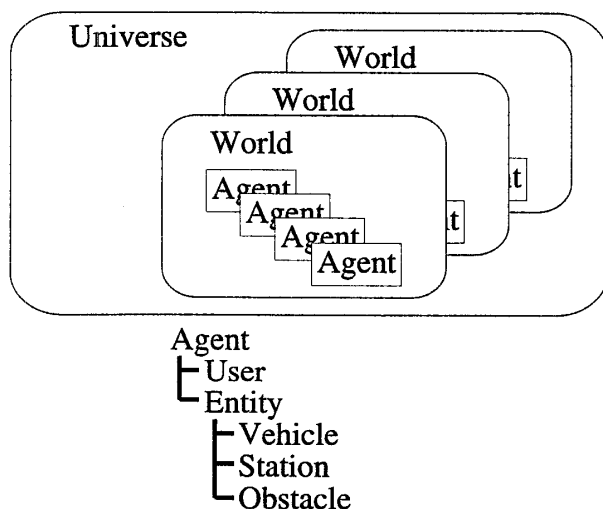


Fig. 2 The universe of the MVS system

機能などを備えた海中プラットフォーム、そして Obstacle は海底地形や試験水槽などを含まれる障害物である。

2.2 MVS の構成

MVS を構成する主たるプログラム群は、Manager と Performer の2種類に大別される。Manager は Agent 同士の相互作用関係を仲介するためのもので、仮想的な海中環境を計算機ネットワーク上に実現するための中心的なプログラムである。Manager は数種類あり、それぞれの詳細は3.1節で述べる。Performer は、Agent の役割をするプログラムで、一つの Agent につき一つの独立した Performer が実行される。

MVS では、Fig. 3のように、各 Performer が Manager を通じて他の Performer と情報のやり取りを繰り返すことにより、全体のシミュレーションが進む。これらのやり取りは、Table 1に示されるような、イベント駆動型の一連の手続きとして定義されており、ロボットに内蔵されたインターバルタイマや、研究者がユーザインターフェース (UI) を通じて与える操作などをきっかけに実行される。

2.3 MVS の特徴

MVS は計算機ネットワーク上に恒常的に展開された仮想的な海を舞台に、利用者が自由にアクセスしてシミュレーションを行うためのシステムである。実際の海で行われる実験と同様のスタイルでシミュレーションを行えるように、以下の機能を備えることを設計方針の柱とした。

- (1) マルチワールド
- (2) マルチエージェント
- (3) 遠隔マルチユーザ
- (4) マルチ CPU
- (5) 実時間動作
- (6) 実ロボットなど外部システムとの接続機能

同時に複数の独立したシミュレーションを行えるように、(1)の機能としてネットワーク上に複数の World を展開させる (Fig. 2)。そして任意の形状の海底地形や障害物が存在する環境を想定したり、複数ロボットによる協調行

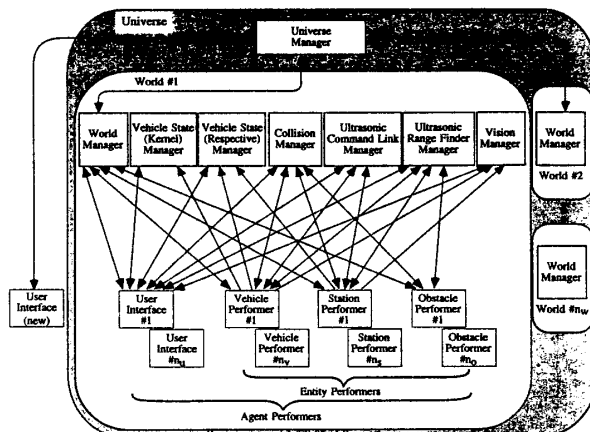


Fig. 3 The software structure of MVS

動の研究に対応するために、MVS には任意の個数の任意のタイプの障害物やロボットを動的に World に加えたり削除する機能(2)を盛り込んだ (Fig. 4)。

実際の海で実験を行う際に複数の研究者が立ち会い、実験の手順を指示したり実験の様子を観察するように、シミュレーションの場合も複数の利用者が同じ世界の出来事に対し働きかけ、あるいは観察することができるべきである。MVS では(3)の機能として、複数の利用者が計算機ネッ

トワーク上の任意の場所から任意の時刻にシミュレーションに参加することができる。ネットワークに接続している端末であれば利用できるため、原理的には地球上のどの研究者も同一のシミュレーション世界を共有することが可能である。

MVS は(4)の機能、つまり複数の計算機に情報処理の負荷を分散させることが特徴の一つである。Agent の役割をする Performer 一つ一つを計算機ネットワーク上の適切な場所で実行させることにより、高い処理速度を確保することができる。また、個々の計算機の特性を活かせるので、グラフィクスに適した計算機で Graphical User Interface (GUI) を実行したり、実際のロボットなどの外部システムと接続する機能を実現できる。

(5)および(6)は MVS の最も特徴的な機能である。ロボットの制御アルゴリズムを一旦シミュレータ内で構築、検証したのち、実ロボットに搭載された CPU のためのプログラムという形にそれを移植する方法では手間がかかり、シミュレーションと実験を交互に繰り返すという研究スタイルをとりにくい。MVS の(6)の機能により、最初から実ロボット用の制御プログラムを組み、実ロボットを計算機ネットワークに接続して MVS の Agent の一つとしてシミュレーションに参加させられるので、非常に能率よく研究を進めることができる。また、いかにすぐれたシミュレーションシステムでも、実際の環境下でロボットが受ける外乱や、センサあるいはアクチュエータなどの特性を忠実に再現することは不可能である。この問題に関して MVS では、外乱やそれに伴うダイナミクスはシミュレーションでなく試験水槽などで航行している実際のロボットに受け持たせ、ロボットが獲得すべきセンサ計測値、つまり実際には存在しない障害物を検知したかのような情報を MVS 側が算出し、ロボットに提供するという実験スタイルが可能である。これは、実験の手間を軽減する効果と、シミュレーションの精度を確保する効果を期待することができ、MVS の大きな特徴である。

2.3 利用形態

MVS を利用して研究を進める際、研究者は Fig. 1 に示すように研究開発の各段階で、

(A) MVS 内の仮想的なロボットを用いて仮想的な海でシミュレーションを行う。

(B) MVS に実ロボットを接続して仮想的な海でシミュレーションを行う。

(C) MVS に接続された実ロボットを実際に水中で泳がせ、仮想的な計測値などを与えるスタイルでシミュレーションを行う。

(D) MVS から完全に独立に実ロボットで実験を行う。という形態のいずれかを適宜選択して、シミュレーションや実験を行う。そして、一回のシミュレーションでは、

(1) シミュレーションの準備として、検証したい研究

Table 1 List of principal events

No.	Eventの内容
01	System configurationの登録
02	System configurationの登録
10	UIがSystem configurationを入手
11	UIが全Worldの簡単な情報を入手
12	UIがWorldの詳細な情報を入手
15	UIがWorldに参加
16	UIのパスワードの確認
20	UIが新規UIに対するメッセージを登録
21	UIが他のUIにメッセージを送信
25	Managerが全UIにメッセージを送信
26	Entityが全UIにメッセージを送信
30	新たにEntityが起動
40	UIがEntityの位置・姿勢を変更
41	UIがObstacleの大きさを変更
42	UIがEntityにトリガを送信
50	UIがMission Modeを変更
51	UIが外乱パラメータを変更
60	定期的にVehicleが位置・姿勢などを報告
61	UIがVehicleの位置・姿勢などを入手
63	定期的にVehicle/Stationが内部状態量を報告
64	UIがVehicle/Stationの内部状態量を入手
67	定期的にVehicleが位置・姿勢を報告
68	定期的な衝突判定
70	定期的な超音波測距
75	通信用超音波の発信
90	UIがEntityの削除を指示
91	UI/Entityの終了

UI: User Interface

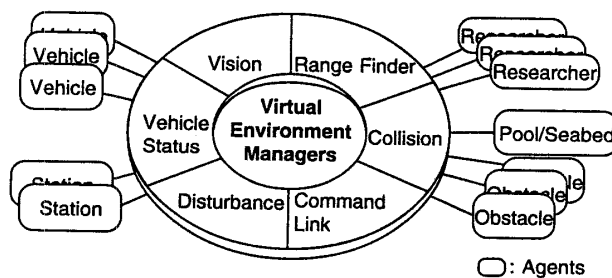


Fig. 4 The distributed system architecture

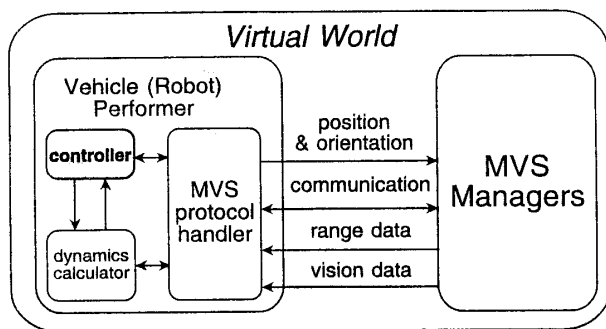


Fig. 5 MVS-robot interface

内容をロボットの制御プログラムの形にする。

(2) 準備した制御プログラムをMVS上で実行し、ユーザインターフェースを通じて観察、検討する。という二段階の手順を踏む。

研究者が(1)で制御プログラムを作成する際に、シミュレータシステムとのインターフェース機能の作成までも研究者に要求するスタイルは望ましくない。シミュレータ利用者側は研究の本質と直接関係のある要素のみに専念できるような体制が、シミュレータ側に整っていなければならない。Fig. 5は、MVS主要部を構成するManagerとVehicle Performerの関係を表している。研究者が作成する制御プログラムはVehicle Performerの中にあり、シミュレーションに関与する全プログラムのごく一部で済む。MVSではソフトウェアライブラリを充実させ、研究者のプログラミング作業が必要最小限に抑えられている。

シミュレーションを実行する(2)の段階は、実験の場合と同様に、さらに以下のように細分化される。

(2a) Agentの生成：シミュレーションに必要な障害物やロボットの役割をするPerformerプログラムを実行し、AgentをWorld内に登場させる。

(2b) Agentの配置：障害物やロボットに初期位置を与える。

(2c) シミュレーション開始：仮想的な海としてのシミュレータ機能は恒常的に作動状態であるので、ロボットに対して開始の指示を与えることを意味する。

(2d) シミュレーション終了

(2e) Agentの撤去：Performerプログラムを終了する。

基本的にはこれらは全てユーザインターフェースを通じて行われるが、ユーザインターフェースは必要な時のみWorldにアクセスすればよい。

3. MVSのソフトウェア

MVSシステムでは、要求された機能・性能（マルチロボット対応、接続性、実時間性）を実現するため、クライアント/サーバモデル⁹⁾に基づいたプロセス間通信による分散システムアーキテクチャを採用している。

MVSではPerformerの一つ一つがクライアントであり、通信関係や超音波測距センサなどの仮想情報をロボットに提供したり、ロボットの内部状態変数をユーザインターフェースに提供するための、複数の独立したサーバがManagerである。機能毎にManagerが独立しているので、将来サポートするセンサの種類を増やすときにも、相応するManagerのモジュールをMVSシステムに追加するだけで良い。このように、MVSシステムは小型のプログラムの集合であるため、システム自体の設計、製作、改良が容易である。

3.1 シミュレーション主要部(Manager)

Managerには以下の8種類がある。これらはMVSを用いたシミュレーションを支える重要な構成要素であり、常に計算機上で実行状態にあることにより、恒常的な仮想環境を実現している。

(1) Universe Manager (UNVM)：どのWorldにも属さず、主にUserに対して各Worldでのシミュレーション状況の通知サービスを行う。

(2) World Manager (WLDM)：World内の総合的なサービスを行う。User相互間のメッセージ授受や、Entityの初期位置の設定など、シミュレーションに伴う非定常的なオペレーションを仲介する。

(3) Vehicle State Kernel Manager (VSKM)：Vehicleのタイプに依らない位置、姿勢などの内部状態データをUserに中継する (Fig. 6, Table 2。Fig. および Table 中の番号は、Table 1 に示すイベントの番号である。

(4) Vehicle State Respective Manager (VSRM)：

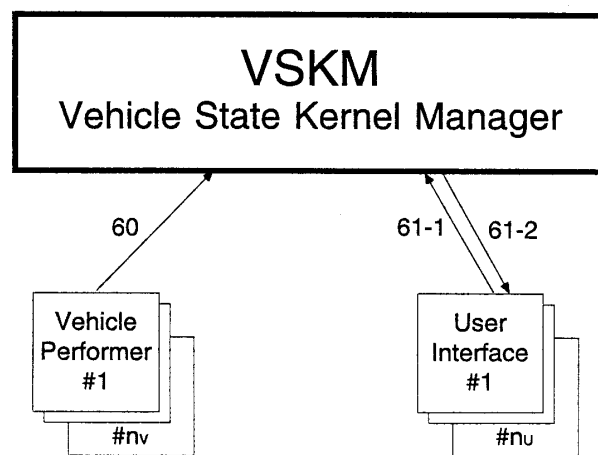


Fig. 6 Vehicle State Kernel Manager

Table 2 Events of the Vehicle State Kernel Manager

No.	Direction	Data
60	各Vehicle → VSKM	Vehicleの位置・姿勢など
61-1	UI → VSKM	希望のVehicleのID
61-2	VSKM → UI	VehicleのID, 位置・姿勢などの集合

Vehicle タイプに固有の内部状態データを User に中継する (Fig. 7, Table 3)。

(5) Collision Manager (COLM): Vehicle と他の Entity との衝突判定を援助するための Manager。COLM は Vehicle の現在位置を中継し、実際に衝突判定計算をするのは各 Entity である (Fig. 8, Table 4)。

(6) Ultrasonic Command Link Manager (UCLM): Vehicle や Station の間の超音波による通信機能を司る。

(7) Ultrasonic Range Finder Manager (URFM): ナロービームの超音波測距センサの機能を司る。URFM は Vehicle の現在位置や距離データなどの取りまとめと中継を行うだけで、実際に距離を計算するのは Entity である (Fig. 9, Table 5)。

(8) Vision Manager (VISM): Vehicle や Station に搭載されたビデオカメラが捉える画像情報を司る。

このうち、(2)～(8)は、各 World に一つずつ存在する。また、(1)～(4)はシミュレーション時の利便性のための Manager であるのに対し、(5)～(8)は物理的な現象を司るものである。

3.2 シミュレーションを行う動作主体 (Agent)

Agent はロボットや障害物など、シミュレーションに直接関与する主体であるが、これらの役割をするクライアントプログラム (Performer) には、以下の 4 種類がある。

(1) Obstacle Performer: 海底地形や試験水槽を含

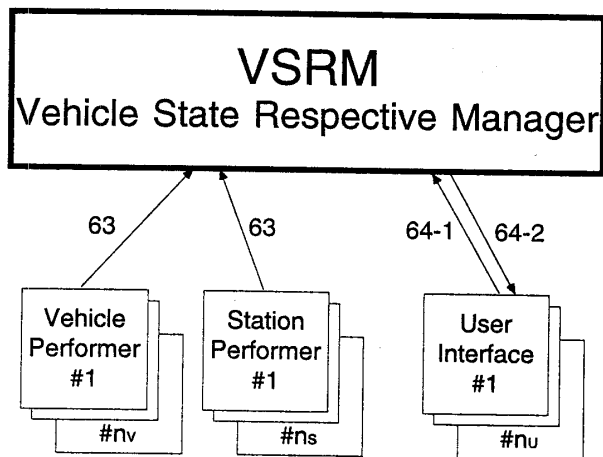


Fig. 7 Vehicle State Respective Manager

Table 3 Events of the Vehicle State Respective Manager

No.	Direction	Data
63	各 Vehicle/Station → VSRM	Vehicle/Station の内部状態量
64-1	UI → VSRM	希望の Vehicle/Station の ID と状態変数番号リスト
64-2	VSRM → UI	Vehicle/Station の ID と内部状態量の集合

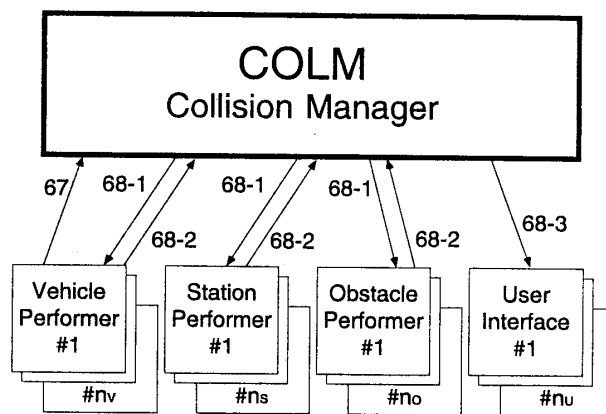


Fig. 8 Collision Manager

Table 4 Events of the Collision Manager

No.	Direction	Data
67	全 Vehicle → COLM	Vehicle の位置・姿勢
68-1	COLM → 全 Entity	全 Vehicle の ID, Type, 位置・姿勢
68-2	各 Entity → COLM	当 Entity と衝突している全 Vehicle の ID
68-3	COLM → 全 UI	衝突している Entity の ID

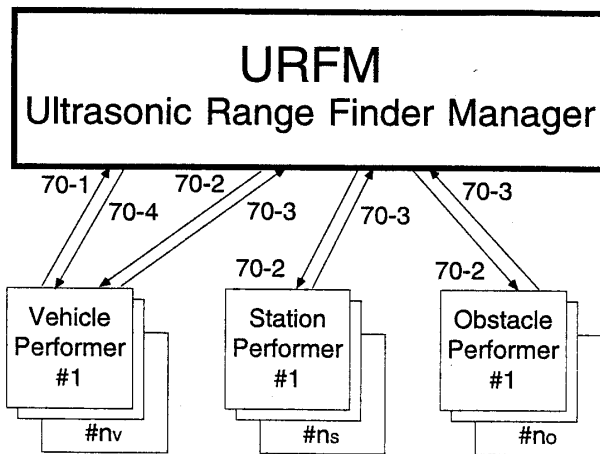


Fig. 9 Ultrasonic Range Finder Manager

Table 5 Events of the Ultrasonic Range Finder Manager

No.	Direction	Data
70-1	各 Vehicle → URFM	Vehicle の Type, 位置・姿勢
70-2	URFM → 全 Entity	Vehicle の Type, 位置・姿勢
70-3	各 Entity → URFM	当 Entity までの全測距センサからの距離
70-4	URFM → Vehicle	全測距センサの最小化された計測値

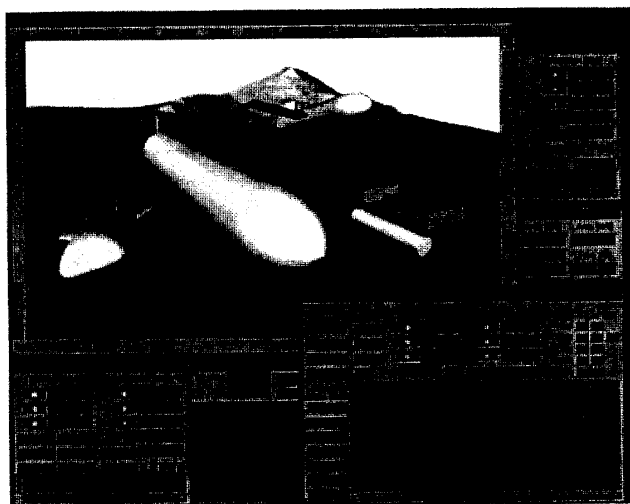


Fig. 10 Graphical user interface

む障害物の役割をするプログラムで、幾何学的な任意の形状を持つ。衝突判定や超音波測距センサの計測値の具体的な計算は、このプログラムが行う。

(2) Vehicle Performer: 海中ロボットの役割をするプログラムで、ロボットの制御アルゴリズムとダイナミクス計算機能を合わせ持っている。このプログラムがロボット研究者の研究対象を表すものである。MVS利用者としての研究者がシミュレーションに先立って行うことは、このプログラムに自身の研究内容を組み込むことのみであるので、手軽にシミュレーションを実行できる。(1)と同様に衝突判定などの計算も行う。

(3) Station Performer: 海中プラットフォームの役割をするプログラムで、移動機能がない点を除けば Vehicle Performer と全く同じである。プラットフォームを含んだミッションコントロールの研究をする場合に利用する。

(4) User Interface: シミュレーションに立ち会う研究者の窓口となるプログラムで、実行する計算機の特성에合わせて GUI (Fig. 10) や、文字ベースの端末装置に対応した汎用的なもの (Fig. 11) など、多様な形態をとる。

4. MVS のハードウェア

MVS は、様々な機能を持った独立したプログラム群が通信を行いながら全体のシミュレーションを実行する分散システムであることはすでに述べた。つまり MVS では、シミュレーションを行うために必要な数のプログラムが同時に実行し、かつ、そのプログラム間で MVS が規定するプロトコルに則った情報の伝達が可能であればよく、特定のハードウェアに依存するものではない。従ってプログラムを実行させるためのハードウェアを規定することは基本的には行わない。これは MVS の汎用性および様々なシステムとの接続性を高める重要なポイントである。

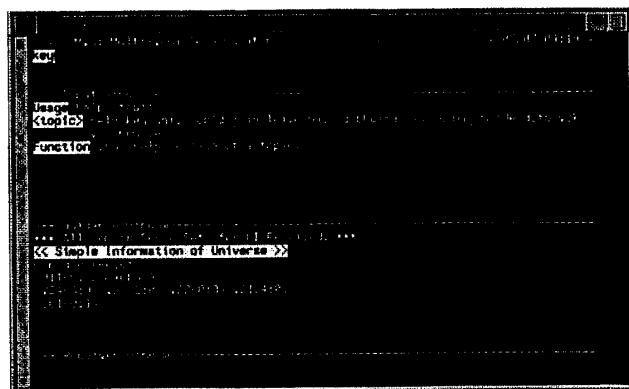


Fig. 11 Text based user interface

5. MVS の応用

MVS は単純な数値解を算出するためのシミュレータではなく、実ロボットを用いた実験というシナリオを仮想的な環境で遂行するための、極めて汎用性に富んだシステムである。従って、研究者が行う実験の目的や内容が研究テーマによりそれぞれ異なるように、MVS も様々な形で利用される。

5.1 システム構成

著者らの研究室ではグラフィックスおよび計算処理用に SGI 社の IRIS Indigo を 1 台、同じく Indy を 3 台用いている。また、計算処理専用として SUN SPARC システムを 2 台活用している。これらの UNIX ワークステーションはイーサネットにより接続され、MVS システムのプロセス間通信には TCP/IP プロトコルによるソケット通信法を用いている。また、著者らの研究室で開発された汎用テストベッドロボット“Twin-Burger”⁵⁾とのインターフェースが用意されており、実ロボットを Agent としたシミュレーションも行っている。

プログラム開発は基本的に C++ 言語を用い、GUI 用に 3 次元リアルタイムグラフィックスパッケージ Inventor などにも利用している。

5.2 MVS による研究開発

5.2.1 海中ロボットの開発

MVS は不特定のロボットを対象にした汎用的なシミュレーションシステムであり、Twin-Burger をはじめとして、航行型プロトタイプ“PTEROA 150”⁶⁾、シャトル型ロボット“ALBAC”⁷⁾、そして現在開発中の長時間航行型ロボット“R-one”⁸⁾がモデル化されている。研究者はこれらのロボットに組み込まれる制御アルゴリズムを研究する際、MVS によるシミュレーションでプログラムの検証を繰り返すことができる。

実海域における実験は、海中ロボット研究には欠かせない要素であるが、これには莫大な金銭的、労力的なコストがかかるため、頻繁に行うことが困難である。そのため実

験に望む前に万全の準備を整え、限られた時間と手間で実験内容を消化しなければならない。このような制約に対し、実験手順に近いスタイルで手軽にシミュレーションを行える MVS は、非常に有効な研究ツールといえる。

5.2.2 マルチロボット研究

海中は広大で電磁波の減衰が著しいため、一台のロボットで調査できる海域は限られたものになる。このため、海洋調査においてはマルチロボットシステムの利点は大きい。マルチロボットの研究^{9),10)}は今後の海洋開発に重要な意味を持つと予想されるが、実験のためには複数のロボットとそれを活動させるための設備、人員が必要であり、マルチエージェント対応型のシミュレータは必要不可欠といえる。

Fig. 12 は、5 台の Twin-Burger による探査活動をシミュレートしている様子である。ロボット個々の制御のみならず、ミッション全体の制御の研究にも MVS を利用した一例である。MVS では動的に Vehicle や Obstacle の追加、位置変更、削除ができるため、手軽にシミュレーション条件を変えられ、利便性が高い。

5.2.3 超音波センサによる地形図作成

著者らは、ロボットに搭載された超音波測距センサを用いて周囲の局所的な障害物形状を確率論的手法で認識し、ロボットに一種の好奇心を与えることにより行動を計画する研究テーマを扱っている^{11),12)}。この研究においては、MVS システムは測距センサの計測値の算出はもとより、GUI による認識結果の 3 次元的な可視化にも役立っている。Fig. 13 はその一例で、仮想環境の地形は海洋科学技術センター提供による八丈凹地の実測データをもとにしており、点々と並んでいる泡状のものは、障害物認識のシミュレーションの結果、ある一定深度における海底面の存在確率が閾値を越えている点を表している。このように、ロボットの認識した地形形状と実際との差を研究者は容易に確認できる。

5.2.4 実験データの可視化による検証

ロボットの研究を行う上で、実験やシミュレーションの

結果を研究者に解りやすい形に可視化することにより、ロボットの活動中の挙動を把握し、その問題点を調査しやすくなる。MVS では、リアルタイム表示の他に、時間ステップ毎や早送り、逆再生などの機能も備えている。

Fig. 14 は、1993 年 8 月に船舶技術研究所において行われた Twin-Burger の実験航行結果の一部を GUI により可視化したものである。航跡と共に超音波測距センサの計測情報が可視化されている。このグラフィック表示により実際の測距センサの故障や、実際の方角センサの仕様の問題点が発見されたという実績がある。

5.2.5 実仮想世界の合成によるロボットの行動実験

MVS の主要な特徴として、実時間処理と実ロボットとの接続機能があることはすでに述べた。ここでは、この特徴を活用した研究例の一つとして、実仮想世界の合成によるロボットの行動実験¹³⁾を示す。

これは、実ロボットを試験水槽などで実際に泳がせるのと同時に、MVS で作成した仮想環境中に実ロボットの分身ともいべき、仮想ロボットを泳がせるというものである。このとき、実際には仮想環境から得られた測距センサデータを実ロボットに送り込み、合成することによって、

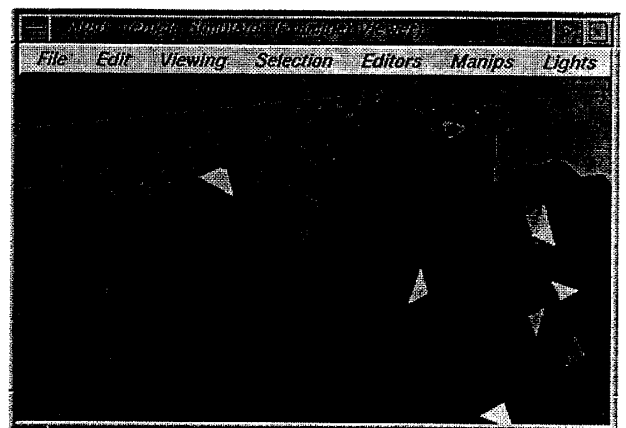


Fig. 13 An MVS application on map-making

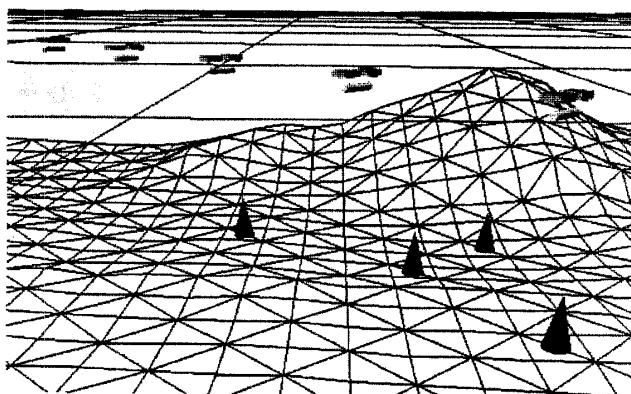


Fig. 12 An MVS application on multi-robot operation

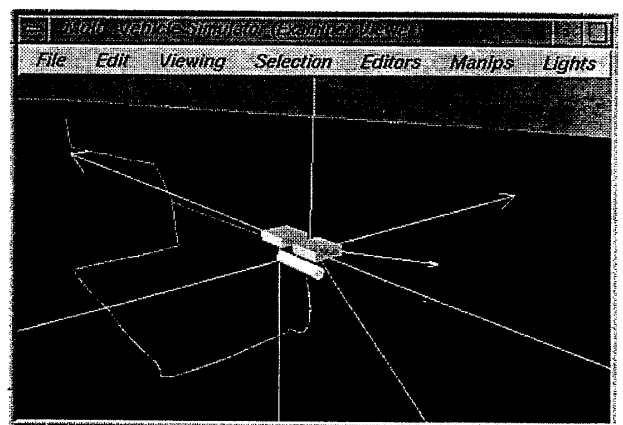


Fig. 14 An MVS application on data visualization

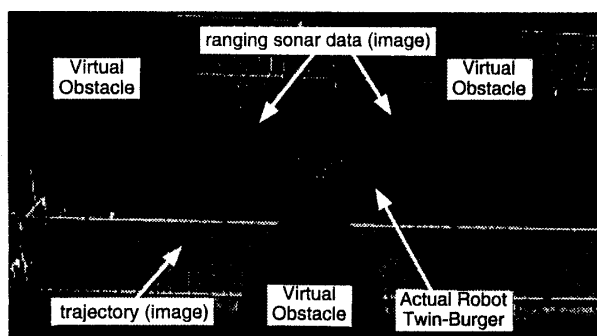


Fig. 15 An image of the merged world

ロボットにとっては実環境中に仮想的な物体を知覚することになる。この合成された状態は、Fig. 15 に示すようなイメージとなる。

この手法により、試験水槽にいながらにして、実際の海にいるのに近い複雑な環境におけるロボットの行動試験を行ったり、また他の仮想ロボットを導入することで、実ロボットと仮想ロボットを用いた複数ロボットによる行動試験を行うこともできる。これは、実用ロボットの研究開発へ向けての強力な研究手段であると考えられる。

6. 結 言

本研究では、海中ロボットの研究開発の様々な局面で利用することを前提にした、汎用性の高い海中環境シミュレータシステムを構築した。その結果、以下の知見が得られた。

(1) マルチワールド機能、マルチエージェント機能、遠隔マルチユーザ機能、マルチCPU機能を実現することにより、計算機ネットワーク上に恒常的に仮想的な海を展開し、研究者が実際の海でロボットの実験を行う場合と同様のスタイルでシミュレーションを行うことが可能になった。すなわち、複数の研究者が任意の場所に居ながらにして同一のシミュレーション世界に立ち会うことができるようになった。

(2) 実時間動作機能、実ロボットなど外部システムとの接続機能を実現することにより、プログラムの開発や移植の手間を軽減し、研究能率を向上させることが可能になった。また、航行中の実ロボットに仮想的なセンサ計測値を提供することにより、実験の手間を軽減するとともに、シミュレーションの精度を確保する新しい実験スタイルを確立した。

本研究で対象にしているロボットは海中用の自律型移動ロボットであるが、上に述べた特徴を備えたシミュレーションシステムは、原子力施設内の作業や惑星探査などの様々な分野で、自律型移動ロボットの研究開発に幅広く応用することができる。

謝 辞

本研究に際し、東京大学大学院工学系研究科船舶海洋工学専攻の小山健夫教授より、ご指導ご鞭撻を賜りました。また、同教授研究室の榎本昌一氏に、計算機の利用に際してご便宜をはかっていただきました。ここに深く感謝致します。

参 考 文 献

- 1) 例えば、小瀬邦治：“船舶運航技術の革新と操船シミュレータ”，日本造船学会誌第781号，1994
- 2) 例えば、T. Koyama, Y. Jin：“On the Design of the Marine Traffic Control System (1st Report)”，日本造船学会論文集，Vol. 162 (1987)，pp. 183-192
- 3) 黒田洋司，浦環，荒牧浩二：“複数海中ロボット対応型仮想環境シミュレータの開発”，第12回海洋工学シンポジウム，1994，pp. 543-548
- 4) 例えば、D. Comer, D. Stevens 著，村井純，楠本博之訳：“TCP/IPによるネットワーク構築 Vol. 3”，共立出版，1995
- 5) 藤井輝夫，浦環，黒田洋司，荒牧浩二，能勢義昭，千葉裕之：“海中ロボットの知能化に関する研究(その1：汎用テストベッドの開発と水槽実験)”，日本造船学会論文集，Vol. 174 (1993)，pp. 903-916
- 6) 浦環，能勢義昭，坂巻隆：“航行型無策無人潜水艇「PTEROA 150」”，第10回海洋工学シンポジウム，1991，pp. 349-353
- 7) K. Kawaguchi, T. Ura, Y. Tomoda and H. Kobayashi：“Development and Sea Trials of a Shuttle Type AUV ‘ALBAC’”，Proc. Of 8th Int. Symp. On Unmanned Untethered Submersible Technology，1993，pp. 7-13
- 8) 小原敬史，浦環，山本喜多男，前田久明，大和裕幸：“長時間稼働が可能な航行型海中ロボットの研究開発”，第12回海洋工学シンポジウム，1994，pp. 549-554
- 9) Y. Kuroda, T. Ura, K. Aramaki：“Vehicle Control Architecture for Operating Multiple Vehicles,” Proc. Of IEEE Autonomous Underwater Vehicle Technology’ 94，1994，pp. 323-329
- 10) 黒田洋司，浦環：“複数海中ロボットに対応する行動決定機構”，日本造船学会論文集，Vol. 177 (1995)，pp. 447-455
- 11) K. Aramaki, T. Ura：“Local Map Drawing by AUV and Path Planning,” Recent Advances in Marine Science and Technology ’94，PACON International and James Cook University of North Queensland，1994，pp. 35-45
- 12) 荒牧浩二，浦環：“潜水機による局所海底地形図作成およびパスプランニング”，日本造船学会論文集，Vol. 177 (1995)，pp. 437-445
- 13) Y. Kuroda, K. Aramaki, T. Fujii, T. Ura：“A Hybrid Environment for the Development of Underwater Mechatronic Systems,” Proc. Of IECON’ 95，1995，(近刊)