# レスキューロボットにおける デバイス管理を容易にするためのシステム開発

前田 弘文*・小林 滋**・高森 年***

# Development of system for rescue robots to facilitate device management

Hirofumi Maeda* , Shigeru Kobayashi** , Toshi Takamori

## Abstract

This paper describes about decentralized processing system "Marionette" for rescue robots to facilitate device management. This system is made using the shared memory. As a result, the rescue robot can be easily developed. Moreover, rescue robot UMRS-2010 that uses this system has been described in this paper. UMRS-2010 is a robot developed by IRS(International Rescue System Institute). This robot is made aiming to survey the disaster scene by remote control.

## 1．緒　　言

阪神淡路大震災をきっかけに，日本国内ではレスキューロボットの開発が盛んに行われている．しかし，人命救助を直接行えるレスキューロボットは少なく，実際に活躍できる場面としては，おもに二次災害を防ぐための現場調査という形がもっとも一般的である．そのため，多くのレスキューロボットは，操縦者が直接ロボットに接近した状態での操縦は行わず，操縦者が遠く離れた場所から無線を用いて遠隔操縦する形を採用している[1][2].

しかしこのことはレスキューロボットを開発する上で，多くの問題を生むこととなる．1つには，ロボット・操縦者間の通信の問題が上げられる．通信距離を伸ばすことはもちろんのこと，通信が途絶えた際のロボットの緊急行動を組み込む必要が出てくる．このことは，プログラムを複雑化する要因となり，開発に莫大な時間を費やすことを意味する．また，通信による遅延などから操縦者側の遠隔操作機器(以後，操作卓)にすべてのロボット制御を組み込むことは暴走を招く原因となり，実質不可能である．そのため，ロボット内部にローカルで複雑な制御システムを構築する必要がある．もちろん，ロボットを遠隔操作する必要があることから操作卓についても，通信を含めた複雑なシステムを構築する必要が出てくる．

さらに，災害現場においてロボット 1 台のみで，すべてを調査することは現実的に不可能であることから，複数ロボットの連携も必要となる．これらのことから，レスキューロボットの開発は，複雑なシステムの構築を必要とし，デバッグ作業に多くの時間を費やすこととなる．この問題を解決する手法として分散処理システムを採用するケースが上げられる．

今回我々もこの分散処理システムに着目し，以前開発したミドルウェア grouse[3]とはまったく別路線として，共有メモリを用いた分散処理システム Marionette の開発を行い，このシステムを用いてレスキューロボットのソフトウェアの再構築を行った．本論文において，実際に開発したレスキューロボット UMRS-2010 について紹介するとともに，システム全体の構成とその中枢となる Marionette について述べる．

## 2．ロボットシステム

今回，非営利活動法人国際レスキューシステム研究機構(以下 IRS)にて開発した UMRS-2010 を図 1 に示す．

UMRS-2010 は，地下街で発生した災害に対応す

*　情報工学科
**　神戸市立工業高等専門学校 機械工学科
***特定非営利活動法人国際レスキューシステム研究機構 理事（神戸大学名誉教授）

平成 23 年 9 月 30 日受理

ることを目的とした探査ロボット UMRS-2009 の後継機(図 2)にあたる[4]．しかし，UMRS-2010 は新たなテーマとして防爆機能を搭載することを目的としているため，中身は火花が発生しないブラシレスモータを使用するなど，外見は酷似しているもののまったく別のロボットシステムとなっている．

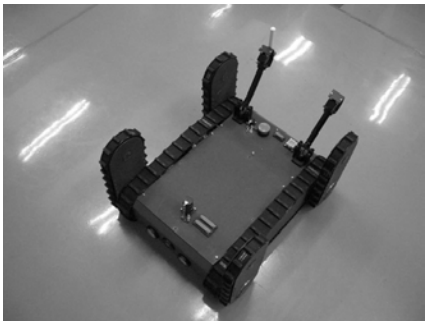

**図１**　UMRS-2010 の概観図

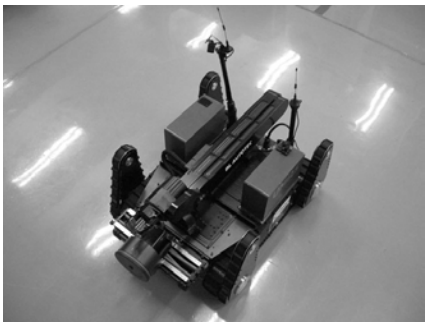

**図２**　UMRS-2009 の概観図

図 3 に UMRS-2010 の内部構造を示す．図 3 において，全てのブラシレスモータ(マッスル株式会社 COOL MUSCLE 2 CM2-C-56B20A-K ，CM2-C-60A10A-R)は，デイジーチェーンによって結合され，シリアル(D-Sub9pin)によってに制御用 PC(ICOP I.T.G. 株式会社 VDX-6314-512)接続される．また，4 つのカメラ(MAXWIN 小型カラーバックカメラ CAM11A)はビデオサーバ(ACTi ACD-2000QT)を介して，LAN により接続される．同様に，光ファイバージャイロ(日立電線株式会社 HOFG-OLC-1)についても，組み込み用超小型デバイスサーバ(Lantronix 株式会社 XPort)を用いてシリアルを LAN に変換した後接続される．さらに，加速度センサ(クロスボー株式会社 CXL04GP3)および LED(三菱電機オスラム株式会社 DP03A-W4-754)については，それぞれ CR フィルタと LED ドライバを介した後，COOL MUSCLE 2(以下, CM2)経由で VDX-6314-512 に接続される．なお，操作卓には NEC Corporation ShieldPRO FC-N22A/BX6SS1B を，無線通信には株式会社バッファロー WLI-UC-GNHP を使用している．

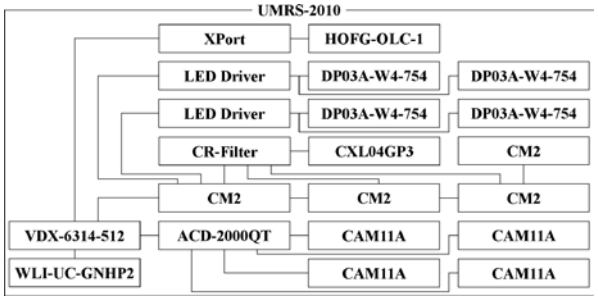


**図３**　UMRS-2010 の内部構造

### ２．１　一体型サーボシステム

CM2 は，一体型サーボシステムであり，モータ・ドライバ・エンコーダ・コントローラ・電源ユニット・PLC 機能の全てを内蔵している（図 4）．これにより，モビリティロボットに必要とされるモータ制御を全て CM2 に委ねることができる．結果，制御用 PC である VDX-6314-512 はシリアルを介して CM2 に目標指令を送るだけとなる．また，A/D 変換や I/O といったロボットに必要とされる機能も搭載されているため，ハードウェアに依存する(特殊デバイスへのアクセス)部分については，全てを CM2 に委ねることができる．これにより，ロボット製作においてハードウェアとのやり取りをモジュールという形で外部に設けることで，ロボット内部におけるシステムを簡略化することができる．

**図４**　COOL　MUSCLE 2 概観図

### ２．２　制御用 PC

制御用 PC である VDX-6314-512 は，CPU として DM&P Vortex86DX 800 Mhz SoC を搭載し

ており，メモリが 512 MB，IDE/ATA フラッシュストレージモジュールとして記憶容量が 8 GB と組み込み用ボードとしては十分なスペックを要している（図 5）．そのため，本ロボットでは Linux（Ubuntu 10.04 LTS）をインストールして使用する．また，基板のサイズも 100×66 mm とコンパクトである．さらに，I/O 関係も充実しているが先に述べたように，CM2 に外部制御を任せているため，USB およびシリアル以外は使用しない．

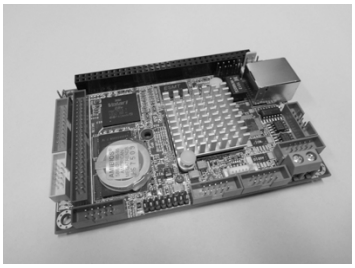

**図 5**　VDX-6314-512 の概観図

## 3．ソフトウェア

本ロボットシステムのプログラム構成の概略を図 6 に示す．ソフトウェアの開発には，操作卓，ロボット本体ともに Ubuntu 10.04 LTS 上で動作するものとし，開発言語には C 言語を用いる．なお，操作卓側については，共有メモリを用いた分散処理システム Marionette を使用している．

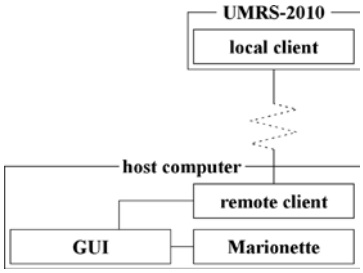


**図 6**　ロボットシステムプログラムの概略図

以下にロボット本体プログラムについて述べる．その後，Marionette と操作卓プログラムについても述べる．

### 3．1　ロボット本体プログラム

本ロボットは台車（ベース）と呼ばれている部分に該当し，現場調査において必要とされる必要最低限の機動力を確保するものである．以下に搭載機能を示す．

・操作卓 4 カメラ画像表示
　カメラ画像を操作卓に表示する機能

・ロボット本体の姿勢検知
　加速度を基にロボット本体の姿勢を算出する機能

・ロボット本体の速度，角速度
　ロボット本体への指令として，スティック（任天堂株式会社 WiiRemote）から速度，角速度を算出する機能

・モータ制御
　CM2 による PID 制御機能

・電波強度検出
　操作卓とロボット間の電波強度を Linux 機能を用いて算出する機能

・バッテリ残量検出
　CM2 によるバッテリ残量検出機能

・自己位置推定
　光ファイバージャイロとモータの回転数からジャイロオドメトリにより，自己位置推定する機能

・LED 点滅制御
　CM2 と LED ドライバによる LED 点滅制御機能

・現在位置表示（未実装）
　自己位置推定によって得られたロボットの座標を地図に表示する機能

・通信切断時の対処
　Linux 機能により接続状況を随時確認し，切断時に緊急停止する機能

このことから，図 7 に示すようにロボット本体プログラムは制御用プログラム control と画像処理用プログラム server_connection の 2 つで構成される．

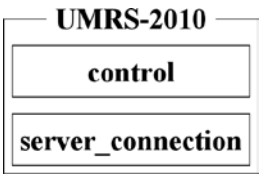


**図7**　ロボット本体プログラムの概観図

2つのプログラムは独立に動作し，それぞれが干渉することはない．このことから，プログラム間でのデータ通信によるバグの発生をなくすとともに，システムを簡略化することが可能となる．なお，それぞれのプログラムについては以下の通りの役割を果たす．

・control
操作卓からの指令を基にモータを介してモータ制御およびLED制御行う．また，モータを経由して受け取ったバッテリ残量データと，加速度データを基に算出したロボット本体の姿勢を操作卓へ送信する．さらに，モータの回転数と光ファイバージャイロのデータからジャイロオドメトリを算出し，ロボットの現在位置を操作卓へ送信する．なお，安全対策として，操作卓・ロボット間の通信が途絶えタイムアウトした場合のロボット緊急停止機能も搭載している．

・server_connection
ACD-2000QT とのやり取りを行い，4 つのカメラ画像を取得する機能を搭載している．また，無線LANによる画像送信に対応するため，カメラ画像(JPEG 画像)を分割しヘッダ情報を付加することで，通信時のデータ欠落を防止する機能も搭載している．

### ３．２　分散処理システム

分散処理システム Marionette は，糸を使った操り人形を語源としている．操り人形の糸は，各部位を繋ぐ横糸と人形全体を操るための縦糸に分かれる(図 8)．また，縦糸は人形を操作するために一本もしくは複数の棒によって接続されている．そのため，棒を操作することで人形全体を操ることが可能となる．

本 Marionette も操り人形を模試しており，各プロセスと Marionette は縦糸で，各プロセス同士は横糸で接続された形をとる(図 9)．これにより，一つ一つのプロセスは，人形の腕や足のように，それぞれの担当する小規模な機能を実装するだけで済み，プログラムをシンプルに構成することが可能となる．

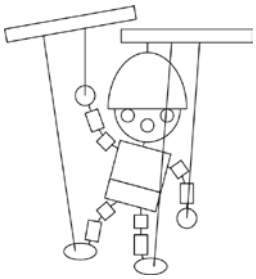

**図8**　糸操り人形の概観図

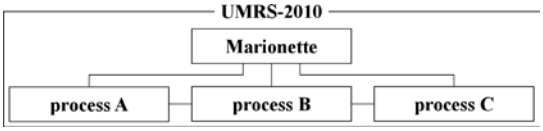


**図9**　Marionette の概略図

実際にプログラムを実装する際は，縦糸と横糸は共有メモリとなり，プロセス間同士，プロセス・Marionette 間はそれぞれ1対1の関係となっている．
そのため，図 10 で示すような 1 対多数や多数対多数という接続は発生しない．もし複数を接続したい場合は，図 11 のような接続方法を用いる．
複数接続の方法として，このような形式を取ることはプログラム全体を見ると PC への負担を多くすることに繋がる．しかし，近年の PC のスペックがかなり高くなった点と，バグが発生した際に原因を早急に発見できる点を考慮にいれてこのような形式を採用している．

なぜ，バグが早急に発見できるかについてであるが，一つの共有メモリに対して複数のプロセスがアクセスする形を取った場合，同期を取る必要が生じプログラムが複雑化するとともに，この部分にバグが発生する恐れがある．また，受け渡しデータの中にバグが生じた際，どのプロセスが原因となっているか追求することが困難だからである．これに対して2つのプロセス間での共有メモリのやり取りであれば，基本的に同期を取る必要はなく，同時読み込みによるクリティカルはセマフォによって回避することが可能で，受け渡しデータにバグが発生した場合であっても2つのプロセスのみをチェックするだけでよいからである．

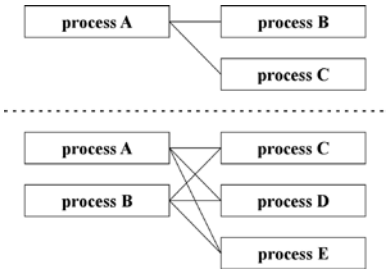


**図１０**　Marionette の禁止事項

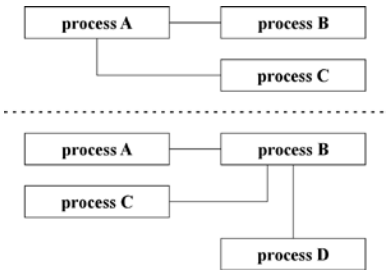


**図１１**　Marionette の接続例

以下に，Marionette における制約を示す．

・プロセスは 1 対 1 の関係を維持する．
・共有メモリはセマフォ使った排他制御によって管理する．
・プロセスの起動および終了は同期する．

これらの制約を実装するために，Marionette 本体では，それぞれのプロセスを同時に起動するとともに，終了の際も同期を取って同時に終了処理を行う．この管理を行うものが，先に示した縦糸に該当する共有メモリである．また，プロセス間のデータ通信に用いる共有メモリについても Marionette 本体が管理する．そのため，Marionette 以外が共有メモリを管理できないよう，Marionette は各セマフォと共有メモリのキーをそれぞれ 2 個ずつ，プロセスの起動時に必要なプロセスのみに受け渡す形を取る（図 12）．これによってシステム全体におけるバグの発生を抑制している．

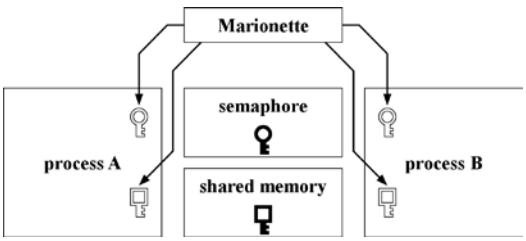


**図１２**　Marionette のキーの受け渡し

## ３．３　操作卓プログラム

Marionette をベースとした操作卓プログラムの構成を図 13 に示す．ロボット本体プログラムと同様，制御系とカメラ画像処理系は，ほぼ独立した形を取り，操縦者との GUI 表示部分において統合される．

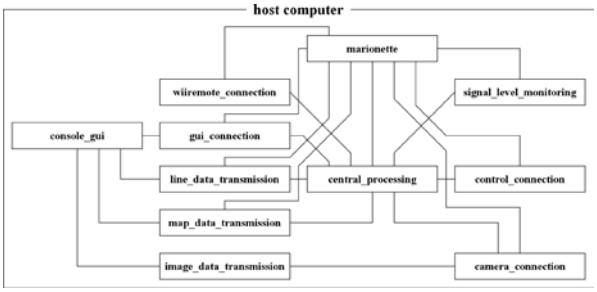


**図１３**　操作卓プログラムの概観図

以下にそれぞれのプログラムについての役割を示す．

・camera_connection
ロボット本体からの画像データを受信し，結合させる機能

・central_processing
集められたデータを基に一括管理を行う機能

・control_connection
ロボット本体・操作卓間のデータ通信（画像以外）を行う機能

・gui_connection
操縦者に必要な情報を GUI として表示する機能

・image_data_transmission
画像データに JPEG ヘッダ情報を付加する機能

・line_data_transmission
操作用ガイドラインを表示するために必要な位置情報を算出する機能

・map_data_transmission
地図データに自己位置をプロットし，送信する機能（未実装）

・signal_level_monitoring
電波強度を検出する機能

・wiiremote_connection
WiiRemote に接続し，デジタルデータを検出する機能

## 4．UMRS-2010 の性能

Marionette を使って作成した UMRS-2010 の操作卓画面（GUI 画面）を図 14 に示す. この操作卓を用いて，UMRS-2010 を制御した結果，3.1 で示した機能とは別に，機体性能として時速 5 km/h の走行，階段走行の限界角度として 45 °を実現した（図 15）. なお，時速 5 km/h については，操縦者が通信の遅延を考慮した上で，遠隔操縦を行った場合でも二次災害（被災者・危険物への衝突など）を起こさない限界速度であり，ロボット自身が持つ限界性能ではない.


**図１４**　操作卓画面

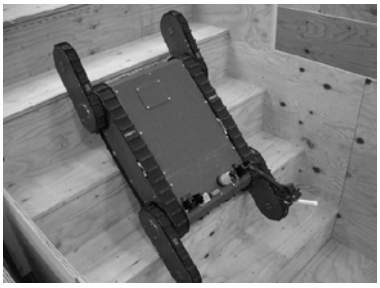

**図１５**　UMRS-2010 階段走行

## 5．結　　言

今回我々は，分散処理システム Marionette の内部構造について述べた．また，実装例として UMRS-2010 の内部構造についても触れた.

しかし，今回製作した Marionette は，UMRS-2010 に特化した，いわば専用ソフトとなっている．今後我々は，この Marionette をソースレベルで一般化し，他のシステムにも容易に適用できる状態を構築する予定である．また，Marionette 実装までの時間がかなりのネックであることから，将来的にはソースを自動生成する形式に移行していく予定である.

## 謝　　辞

本研究は，独立行政法人新エネルギー・産業技術総合開発機構（NEDO）の「戦略的先端ロボット要素技術開発プロジェクト」の一環として実施されたものである.

## 参考文献

[1]　吉田 智章，小柳 栄次，入江 清，原 祥尭，岡田 佳都，永谷 圭司：クローラ型移動ロボット Kenaf を使用した屋外自律走行システム，第 8 回システムインテグレーション部門講演会（SI2007），pp.955～956，（2007）

[2]　永谷 圭司，岡田 佳都，徳永 直木，桐林 星河，小柳 栄次，吉田 智章，油田 信一，久武 経夫：火山探索を目的としたクローラ型移動ロボット Kenaf による桜島での遠隔操作実験，第 10 回システムインテグレーション部門講演会（SI2009），pp.1943～1946，（2009）

[3]　前田 弘文, 五百井 清, 大坪 義一, 小林 滋, 高森 年: レスキューロボットにおけるデバイス管理を容易にするためのミドルウェア開発，日本機械学会講演論文集 No.115-1，p.123～124，（2011）

[4]　石井 良典, 大坪 義一, 小林 滋, 小林 泰弘, 山本 祥弘, 梅田 栄, 海藻 敬之, 前田 弘文, 高森 年，田所 諭：閉鎖空間内探索ロボットのための遠隔操縦システムの開発，第 11 回システムインテグレーション部門講演会（SI2010），pp.1238～1241，（2010）