

複数の動機によるロボットの制御 ～ R-BIOS の実装 ～

○ 松田 昌史 宮下 敬宏 細田 耕 浅田 稔
大阪大学大学院 工学研究科 知能機能創成工学専攻

The Robot Basic Input Output System for multi motivated robots

○ Masashi Matsuda Takahiro Miyashita Koh Hosoda Minoru Asada

Department of Adaptive Machine Systems, Graduate School of Engineering, Osaka University

Abstract : *In this paper, we propose R-BIOS. R-BIOS is useful for a servo system based on multi motivations. R-BIOS is made up of a imaginary sensor/actuator module and the anti-interference servo module. Using R-BIOS, each servoing module does not need to consider other modules. We show a four legged robot system based on the R-BIOS.*

KeyWords : robot architecture, motivation, R-BIOS

1 はじめに

多自由度ロボットの制御システムは複雑になりがちである。ロボットの自由度の増加に従い、制御システムを行う処理が飛躍的に増大するからである。複雑な多自由度ロボットの制御を取り扱いやすくする為に、従来からいくつかの手法が提案されている。制御単位をモジュール化してそれぞれ独立したプロセスで動作させること、モジュールに対しロボットを仮想化することなどである。一般にモジュール化を行った場合、制御システムの設計開発や改良をモジュール毎に行うことができる。そのため制作者の負担を軽減できる。しかし制御単位をモジュール化し実行を独立にすると出力の干渉等が起こり易くなる。仮想的なロボットを用いることは、従来ロボット毎に固有だった制御プログラムを他のロボットにも再利用しやすくする。しかし汎用の仮想ロボットを設定するのは難しい。

従来提案された手法の例としては、岡らが提案した BeNet¹⁾がある。BeNet はビヘイビアをモジュール化し、非同期並列動作させることによりロボットの動作制御を実現している。しかしモジュールの出力が干渉した場合、優先度の高いものしか出力ができない。また Fujita et al. が提案した OPEN-R²⁾は、ロボットのボディに対応した多数の動作モジュールが結合し、入出力を仮想化するレイヤを介して動作する。この手法では設計段階でモジュールの出力を干渉しない様にしている。これらのようにモジュール間干渉は起らない様に設計されるか、または干渉発生時に優先度の低いものが無条件に切り捨てられる仕組みを用いているものが多い。しかし設計段階で干渉を全て考慮するのは困難である。また優先度の低いモジュールを切り捨てる場合、条件によっては優先度が下位の動機は実現されなくなる。

本稿で述べるシステム R-BIOS³⁾では、上記の問題を解決する。以下では姿勢制御や運脚といった動作規範を動機と呼ぶが、動機をそれぞれモジュール化して並列動作させる。そして動機が占有する自由度を用いて出力の非干渉化を行う。そのため優先度の低い動機も自由度に余裕があれば動作を実現することができる。同時にロボットの入出力を仮想化する。R-BIOSのこれらの機能によって、各モジュールは他を気にする必要がない。このシステムはリアルタイム OS 上に構築され、独立プロセスで動作する。動機モジュールも独立したプロセスで実行し、動的に制御を開始や終了する

ことができる。次節で R-BIOS の詳細を示し、この実装例を示す。

2 R-BIOS

Fig.1に R-BIOS を用いたロボットシステムの構成例を示す。ホストコンピュータ内で実行されるのは、OS を除くと R-BIOS と動機モジュールである。R-BIOS 内には共有メモリ (SHARED MEMORY) 上にある入出力の仮想化部分 (VIRTUAL INPUT/VIRTUAL OUTPUT) と非干渉化を行う部分 (VO-COMPOSER) がある。

R-BIOS の入出力仮想化部分ではロボットとの全ての入出力を行う。ロボットに搭載される装置類には様々な物がある。例えば角度情報を得るために不定時にアクセスするポテンシオメータやモータドライバ、33[ms] 毎に取り込む NTSC カメラ画像等である。アクセス方法がそれぞれ独自のものは当然だが、サンプリングレートの違いが一番大きな問題である。入出力タイミングを正確に行うために、Fig.1の様にリアルタイム OS とハードウェアタイマ割り込みを用いる。入出力で用いる数値形式は大抵が装置固有であるため、R-BIOS では標準の単位系の値に変換する。入出力が終わった後、各装置から得た入力値を次のサンプリングまでホールドする。これによって各モジュールはサンプリングレートを気にせずに値を用いることができる。これらの機能によってロボットのハードウェア構成が異なっても動機側は全く気にする必要が無い。

R-BIOS には最高のプロセス優先度を与え、他のモジュールを実行中でも高速かつ確実に R-BIOS へ処理を切り替える。このため正確な時間間隔で実行可能である。仮想化部によるデータの更新が行われた後、R-BIOS は動機モジュールに実行権を渡す。各動機モジュールには予め OS 上での実行優先度と動機の出力優先度の二つを与える。実行は OS によりプロセス優先度の高いものから順に実行される。各動機はそれぞれ固有の動作規範に基づき演算を行う。演算が終り次第、動機モジュールは次の実行タイミングまで待機状態となり他のモジュールに実行権を移す。各動機モジュールは完全に独立しており、R-BIOS とは規定の手続きさえすればアクセスできるため、任意のタイミングで動的にロードやアンロードが可能である。

R-BIOS の非干渉化部では動機の出力非干渉化に、制御で占有する自由度を用いる。正確には入力値と出力値の変動量の関係を示すヤコビ行列を用いる。出力優

先度順に、動機は全自由度がなす空間から必要な自由度空間を確保していく。上位の優先度を持つ動機が必要な自由度を割り当てた後、より優先度の低い動機は余っている自由度の中で必要な分が割り当てられる。優先度が低い動機では十分な自由度を確保できるとは限らないため、動作を正しく実現する保証がない。しかし限られた自由度を多くの動機で利用できるのは利点だと考えられる。

バッファに保存されている各動機モジュールの出力、入力値と目標値との誤差とヤコビ行列の2つを用いて、非干渉化した最終的な出力を得る。この結果は仮想化部が適切なタイミングでロボットへ出力する。

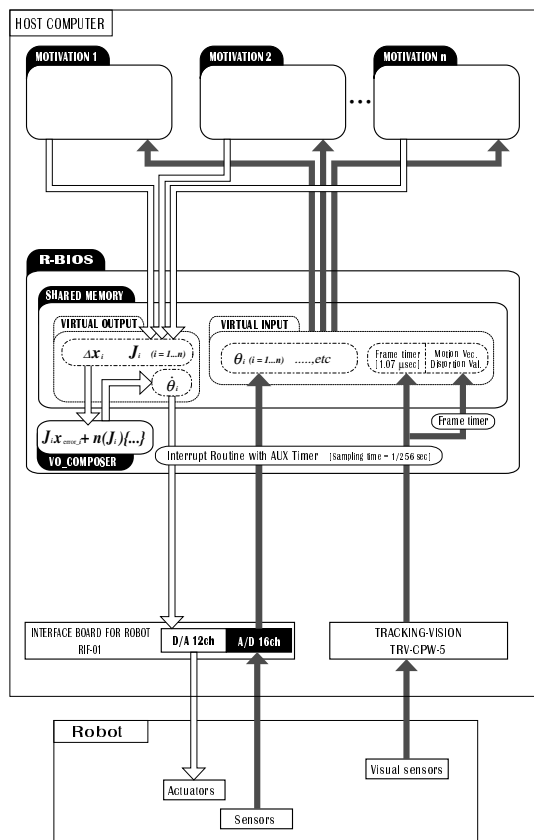


Fig.1 Implementation of R-BIOS

3 実装例

次に複数のモジュールが正確に動作を切り替えさせている例を示す。

3.1 実装するシステム

本実装例ではリアルタイム OS として VxWorks ver. 5.3.1 (Wind River Systems, Inc.)、コンピュータとして AT 互換 PC を用いた。ロボットのベースには 12 自由度 4 脚ロボット、TITAN-VIII⁴⁾ を用いる。これに各関節のポテンシオメータ、カラー CCD カメラ、各脚先に 1 軸力覚センサを搭載している。センサデータの入出力は AD/DA ボード、RIF-01 (富士通製) を用いる。カメラ画像の処理には画像相関演算処理ボード、TRV-CPW-5 (or CPD-6: 富士通製) を用いる。ポテンシオメータと力覚センサ、モータ出力のサンプリング周期は全て $1/256[s] = 3.906[ms]$ である。カラー CCD カメラは $1/29.97[s] = 33.37[ms]$ 毎にサンプリングし、その後 3 回ずつ画像相関演算を行っている。

動機モジュールは 3 つ用意する。出力優先度・実行優

先度順に 1 つは脚先間の距離を一定に保持する動機、2 つ目は視覚目標の見える位置を一定に保持する為に姿勢を変化させる動機、3 つ目はロボットの重心が常に支持多角形の中に入っている様に脚の踏みかえを行う動機である。これらの動機の組み合わせで、ロボットは視覚目標に従って歩行する。

3.2 結果

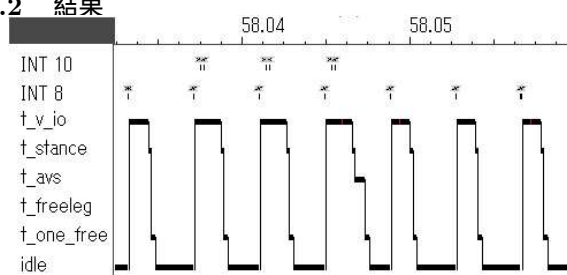


Fig.2 Process Switching

R-BIOS が動作中のプロセス遷移を Fig.2 に示す。図中の t_{v_io} は R-BIOS 本体である。INT8 はハードウェアタイマ割り込みであり、これを利用して R-BIOS が動作する。INT10 は画像取り込みボードからの処理完了割り込みである。 t_{stance} は脚先間隔保持の動機、 t_{avs} は視覚追従、 $t_{freeleg}$ と t_{one_free} は脚の踏みかえの動機である。R-BIOS と脚先間保持制御、脚の踏みかえの t_{one_free} はセンサ入力と同期して実行されるが、画像追従は処理の必要なときにしか実行されない。 $t_{freeleg}$ は図中では動作の必要がなく、待機状態のままである。Fig.2 に示すように、R-BIOS は正確なタイミングで動作している。また動機がプロセス優先度に従いそれぞれの周期で動作していることがわかる。これより R-BIOS が、複数の動機によるロボット制御に関して有効であることが示された。

4 おわりに

本報告ではリアルタイム OS 上で動作する R-BIOS を提案し、実機への実装によりその有効性を検証した。本手法ではリアルタイム OS によるマルチプロセス処理を最大限に利用することにより、動機に基づくロボット制御に有効だとする結果を得ることができる。また占有自由度を用いた非干渉化を用いることにより、動機を設計する際や実行時に他の動機を考慮する必要がない。本報告では 3 つの動機と 3 種のセンサからなるロボットの実装例を示したが、より多くの動機や多様なセンサがあった場合の優先度の決定が今後の課題である。

参考文献

- 1) 岡哲資, 稲葉雅幸, 井上博允. BeNet: 自律ロボットの情報システム記述のための並列情報処理モデル. 日本ロボット学会誌, Vol. 15, No. 6, pp. 878-885, 1997.
- 2) Masahiro Fujita and Koji Kageyama. An open architecture for robot entertainment. In *Proc. of the First International Conference on Autonomous Agents 97*, pp. 435-442, 1997.
- 3) 宮下敬宏, 細田耕, 浅田稔. 動機に基づくロボットアーキテクチャの実装. 第 5 回ロボティクスシンポジウム, pp. 313-316, 2000.
- 4) K. Arikawa and S. Hirose. Development of quadruped walking robot TITAN-VIII. In *the 1996 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pp. 208-214, 1996.