

# リアルタイム産業機械向けエミュレータの開発

## - 個別ユニットエミュレータの機構計算の実装と並列化 -

横山 哲也    田畑 克彦    山田 俊郎    稲葉 昭夫

### Development of Real-time Emulator for Industrial Machine - Implementation and Parallel Calculation of Emulator for Arm Robot -

Tetsuya YOKOYAMA    Katsuhiko TABATA    Toshio YAMADA    Akio INABA

**あらまし** 産業機械の組込みソフトウェアの開発には、実機の動作を模擬するエミュレータの活用が有効である。我々は平成20年度に、産業機械の個別ユニットを制御する組込みソフトウェアの検証用エミュレータとして、組込みソフトウェアと論理時間上で同期して実機の動作を模擬する「個別ユニットエミュレータ」の試作を行った。今年度はアームロボットのエミュレータを新たにパソコンで実現するため、機構の動きを算出する機構計算の実装を行い、モーションコントローラを用いてエミュレート出来ることを確認した。また機構計算の並列化を行い、シングルスレッドでの実行に比べて計算時間が短縮することがわかった。

**キーワード** 組込みソフトウェア, エミュレータ, 数値計算

#### 1. はじめに

近年、産業機械の開発期間は短縮傾向となっている。また、産業機械における組込みソフトウェアへの機能要求が高度化している。このため、組込みソフトウェアの検証時間が十分に確保できずに、機能安全性の低下を招いている。実機を動かす組込みソフトウェアと実機との統合検証は、両者の完成を待って行われる。多くの場合、組込みソフトウェアが先に完成することから、実機が完成するまでの間に手待ち状態が生じ、開発効率が低下する。また、複数の部署が開発を行う場合、実機が部署数分ないと並行して検証を行うことができないため、手待ち状態が増加し更に開発効率が低下する。この解決策として、数値計算に基づき実機と同様に振舞うシステム(以下、エミュレータ)を用いて、実機の完成を待たずに検証を行う手法、HIL(Hardware In the Loop)<sup>[1]</sup>があり、メカトロニクス製品メーカーを中心に活用されている。

HILは専用電子基板を用いて、機械の機構をエミュレートする。組込みソフトウェアが実装されたコントローラにHILを接続すれば、コントローラからは実機が接続されているように見える装置である。ただし、HILは専用基板で構成されていることから一般的に高価であり、開発担当者数確保するのは難しい。

上記課題に対して、我々は今年度、共同研究事業者と共同して安価に提供できるエミュレータとして、汎用

FPGAを用いた「個別ユニットエミュレータ」を開発した。ただし、汎用FPGAはリソースの制約で記述できるモデルに制限があり、また浮動小数計算の実装が難しい。一方、最近のパソコンはCPUの性能向上が著しく、エミュレータをパソコン内部で実現したい要望もある。パソコンであれば汎用FPGAが抱える上述の問題をクリアすることができる。

そこで本研究では、パソコンでの実装を目的とした個別ユニットエミュレータの機構計算を開発し、パソコンで実装を行った。さらに、機構計算の計算時間を短縮するための並列計算も行ったので、その内容について記述する。

#### 2. 機構計算の実装

##### 2. 1 個別ユニットエミュレータ

個別ユニットエミュレータは、産業機械を構成する個別ユニットの実機動作を模擬するエミュレータで、エミュレータ部、デバッグインターフェースとデータコンバータで構成され、組込みコントローラで制御される(図1)。エミュレータ部は複数の部品で構成される機械の動きを、運動方程式に基づいて数値積分する機構計算で構成される。以下ではこの機構計算について記載する。デバッグインターフェースとデータコンバータに関しては、平成20年度の報告書<sup>[2]</sup>を参照されたい。

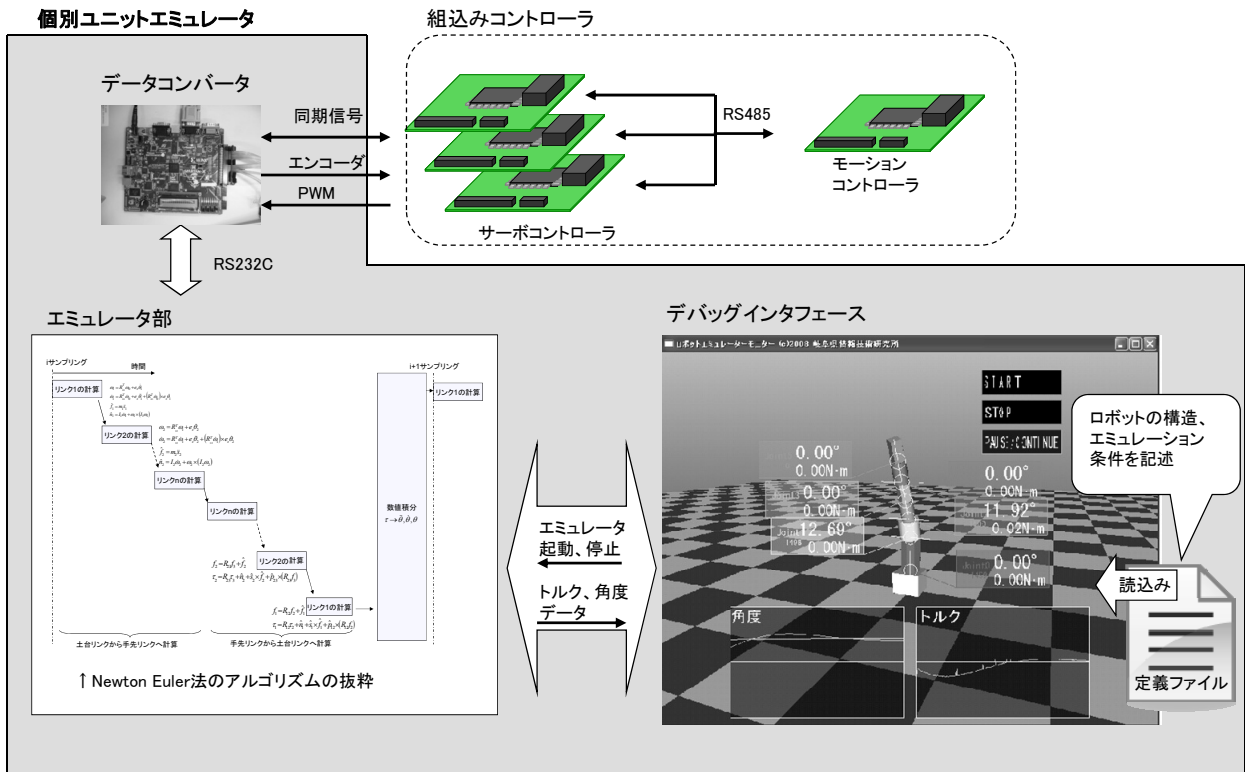


図1 個別ユニットエミュレータおよび組み込みコントローラ

2. 2 機構計算

本研究では対象とする個別ユニットは、産業機械として多用されているシリアルリンクのアームロボットとした。ロボットの機構は、複数のリンクがシリアルに繋がった多関節リンクとその関節に設けられたモータで構成されるため、エミュレータ部は図2に示すモータと多関節リンクのモデル構成とした。モータモデルは組み込みコントローラからの制御電圧と多関節リンクモデルのエミュレーション結果に基づいて、電機子回路と回転軸の挙動をエミュレートする。多関節リンクモデルはモータモデルのエミュレーション結果に基づいて、関節角度の挙動をエミュレートする。以下、モータと多関節リンクの機構計算を記載する。

モータ電機子回路に関する運動方程式は、式(1)として記述できる。

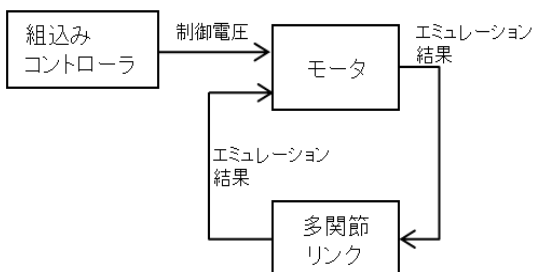


図2 エミュレータ部のモータと多関節リンクとの関係

$$V_{in} - K_b \dot{\theta}_m = R i + L \dot{i} \tag{1}$$

ここで  $V_{in}$  はモータの印加電圧、 $K_b$  は逆起電力係数、 $\theta_m$  はモータ回転角度、 $R$  は電機子抵抗、 $L$  は電機子インダクタンス、 $i$  は電機子に流れる電流を指す。

また、モータ回転軸に関する運動方程式は、式(2)として記述できる。

$$M_m \ddot{\theta}_m = K_T i - N_g \underbrace{\{K_g(N_g \theta_m - \theta_l) + D_g(N_g \dot{\theta}_m - \dot{\theta}_l)\}}_T \tag{2}$$

ここで  $M_m$  はモータとギア1次側の慣性モーメント、 $K_T$  はトルク定数、 $N_g (< 1)$  は減速比、 $K_g$  はギアの剛性係数、 $D_g$  はギアの減衰係数、 $\theta_l$  はギア2次側(アーム側)の回転角度である。 $T$  はギア2次側に伝達するトルクであり、ギアのねじれを表現している<sup>[3][4]</sup>。

モータの運動方程式(1)(2)は、連立微分方程式として式(3)のとおり表現できる。

$$\frac{d}{dt} \begin{bmatrix} i \\ \theta_m \\ \dot{\theta}_m \end{bmatrix} = \begin{bmatrix} -\frac{R}{L} & 0 & -\frac{K_b}{L} \\ 0 & 0 & 1 \\ \frac{K_t}{M_m} & -\frac{N_g K_g N_g}{M_m} & -\frac{D_g N_g K_g N_g}{M_m} \end{bmatrix} \begin{bmatrix} i \\ \theta_m \\ \dot{\theta}_m \end{bmatrix} + \begin{bmatrix} V_{in} \\ 0 \\ \frac{N_g K_g}{M_g} (\theta_l + D_g \dot{\theta}_l) \end{bmatrix} \quad (3)$$

一方、Nリンクで構成される多関節リンクの運動方程式は、式(4)で記述できる。

$$M_l(\theta_l) \ddot{\theta}_l + C(\dot{\theta}_l, \theta_l) + D_l \dot{\theta}_l = T \quad (4)$$

$M_l \in R^{N \times N}$  は多関節リンクの慣性モーメント係数行列、 $C \in R^N$  は遠心力とコリオリ力の非線形項および重力項をまとめたベクトル、 $D_l$  はギア2次側の減衰係数行列である。ギア2次側の慣性モーメントは多関節リンクの慣性モーメントに比べて小さいことから無視した。

リンク数の増加に伴い、式(4)の係数行列とベクトルを直接算出すると計算量が大きいことが知られている<sup>[5]</sup>。そのため、本研究ではNewton-Euler法<sup>[6] [7]</sup>を用いて係数を算出する。Newton-Euler法はシリアルリンク構造において、各リンクの力とモーメントのつりあいから運動方程式を算出する計算法である。

運動方程式(3)と(4)を時間歴で数値積分をするにあたり、モータの時間刻み(サンプル時間)と多関節リンクの時間刻みを同一にするのが望ましい。ただし、モータの電流時定数は多関節リンクの時定数に比べて小さいため、多関節リンクの時間刻みをモータの時間刻みに合わせることは計算の効率が悪い。またモータの時間刻みを多関節リンクに合わせることは、数値計算の安定性上難しい。そこで本研究では、モータには計算コストは高いが安定性の優れた後退オイラー法を、アームロボットには計算コストが低い前進オイラー法の数値計算法を用いることで、両者の時間刻みをアームロボットの時間刻みに揃えた。

### 2. 3 モーションコントロールを用いた機構計算の動作確認

モーションコントロールは、直交座標系におけるロボットの手先位置を目標位置に追従させる。図1ではモーションコントローラをマイコンで実装しているが、ここでは浮動小数計算を行うためパソコンに置き換えた。モーションコントローラで現在の手先位置から目標位置を実現するための目標関節角度を算出し、その値をサーボコントローラにRS485経由で送る。サーボコントローラからアームロボットの現在の関節角度をパソコンにRS485経由で送り、モーションコントローラで現在のアームロボットの手先位置を計算する。

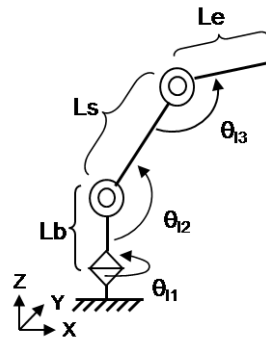


図3 3軸シリアルリンクのアームロボット

制御対象のアームロボットのモデルを図3に示す。3関節のシリアルリンク型で、関節にはギアを有したモータが備わっている。

モーションコントローラからサーボコントローラに送る目標関節角度は、Jacobi行列を用いて算出する。アームロボットの手先速度  $\dot{X}(\theta_l) \in R^3$  と関節角速度  $\dot{\theta}_l \in R^3$  の関係は、Jacobi行列  $J(\theta_l) = \frac{\partial X}{\partial \theta_l} \in R^{3 \times 3}$  を用いて式(5)で記述できる。

$$\dot{\theta}_l = J(\theta_l)^{-1} \dot{X}(\theta_l) \quad (5)$$

ここで  $\dot{\theta}_l = (\theta_{ld} - \theta_l) / \Delta t$ 、 $\dot{X}(\theta_l) = (X_d - X(\theta_l)) / \Delta t$  とおくことで、アームロボットの手先の目標位置  $X_d \in R^3$  を実現するための目標関節角度  $\theta_{ld} \in R^3$  を、式(6)のとおり与えた。

$$\theta_{ld} = wJ(\theta_l)^{-1} \{X_d - X(\theta_l)\} + \theta_l \quad (6)$$

ここで  $w$  は重み係数である。式(6)の  $\theta_{ld}$  はモーションコントローラの制御周期により随時更新し、サーボコントローラに送信される。サーボコントローラは受信した目標関節角度を追従することになる。なお、この動作確認では重力は作用しないとした。

図4にロボットの手先位置と目標位置の偏差を示す。横軸はモーションコントローラの指示回数、縦軸は目標値に対する偏差である。目標位置は、y軸のみに10mm、x

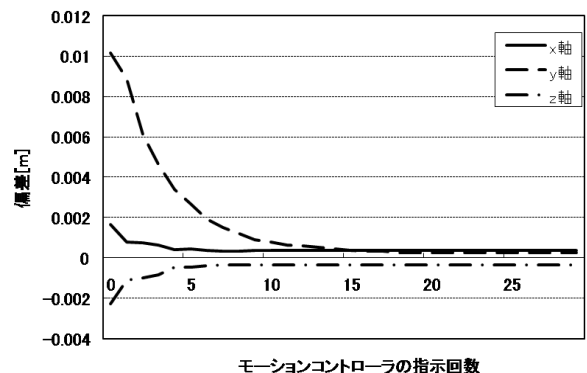


図4 ロボット手先位置の目標位置の偏差

軸, z軸に関しては0mmを与えた. 図から指示回数の増加に伴い, 偏差が小さくなっていることがわかる. 偏差が0にならない理由としては, エンコーダの分解能等が考えられる.

### 3. 機構計算の並列化の検討

多関節リンクに係る数値計算はモータに比べて計算量が多い. そこで多関節リンクの数値計算で使用しているNewton-Euler法を並列化することで, 計算にかかる時間を抑えることを試みた.

従来からあるNewton-Euler法を用いた並列化に関する研究として, Newton-Euler法がリンク単位での計算であることに着目して, 関節毎にCPUを配置して計算時間を減らす試みがある<sup>[8]</sup>. しかし, この手法は関節数のCPUを用意することから並列演算機の使用が前提となる. 本研究では, 一般的なスペックのパソコンでの実装を目標としているため, 関節毎に処理を並列化するのではなく, Newton-Euler法の一連のシーケンス処理を分割して並列計算を行う. 分割数は昨今のパソコンのCPUがデュアルコアを備えていることから, 2つとした.

図5にNewton-Euler法の一連のシーケンスを示す. シーケンスは大別して①運動学計算と順動力学計算, ②リンク両端間における力/トルクの変換, ③数値積分の3つに分けられ, シングルスレッドで実行される. 本研究での並列化では, ①~③の処理を図6に示す2つのスレッドに

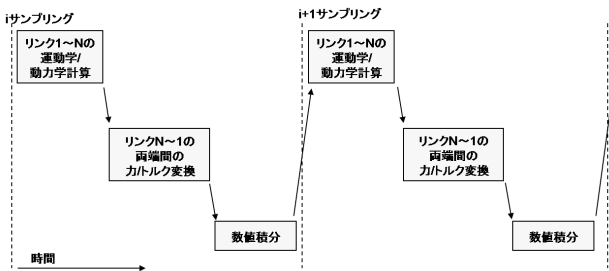


図5 Newton-Euler法のシーケンス

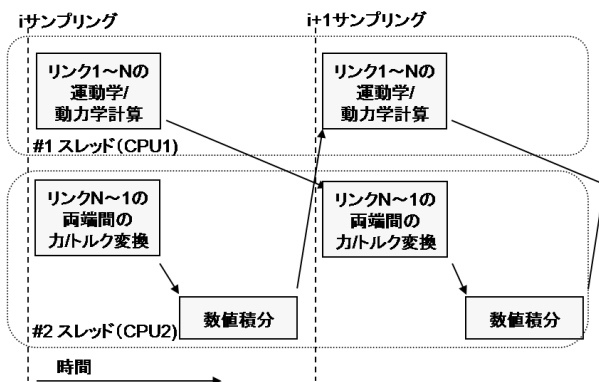


図6 Newton-Euler法の並列シーケンス

割り当てる. なお, スレッド間は同期を保つためにフラグを活用した.

図6に示す並列化においては, 並列化に伴う計算誤差が生じる. 図5のシングルスレッドでの実行は, ③数値積分で角度を更新してから①運動学計算を行うが, 図6の並列計算では1サンプル前の角度情報を使用して運動学を計算することになる. これは関節角度  $\theta_l$  に誤差  $\Delta\theta$ , 関節角速度  $\dot{\theta}_l$  に誤差  $\Delta\dot{\theta}$  が含まれることに相当し, その影響は式(4)の係数  $M_l(\theta_l), C(\dot{\theta}_l, \theta_l)$  中の三角関数および係数  $C(\dot{\theta}_l, \theta_l)$  中の  $\dot{\theta}_l$  に表れる.

そこで, 並列化に伴い生じる三角関数の誤差の影響を考える.  $|\Delta\theta| \ll 1$  の条件のもとでは, 三角関数はテイラー展開を用いて以下のように近似できる.

$$\begin{aligned} \sin(\theta_l + \Delta\theta) &= \sin(\theta_l) \cos(\Delta\theta) + \cos(\theta_l) \sin(\Delta\theta) \\ &= \sin(\theta_l) \left(1 - \frac{\Delta\theta^2}{2} + \dots\right) + \cos(\theta_l) \left(\Delta\theta - \frac{\Delta\theta^3}{2} + \dots\right) \\ &\approx \sin(\theta_l) \end{aligned}$$

$$\begin{aligned} \cos(\theta_l + \Delta\theta) &= \cos(\theta_l) \cos(\Delta\theta) - \sin(\theta_l) \sin(\Delta\theta) \\ &= \cos(\theta_l) \left(1 - \frac{\Delta\theta^2}{2} + \dots\right) - \sin(\theta_l) \left(\Delta\theta - \frac{\Delta\theta^3}{2} + \dots\right) \\ &\approx \cos(\theta_l) \end{aligned}$$

これにより, 刻み時間を小さくすれば  $|\Delta\theta| \ll 1$  が成立し,  $\Delta\theta$  を含む項がなくなることから, 三角関数の誤差の影響は無視できる.

次に, 式(4)の  $C(\dot{\theta}_l, \theta_l)$  の  $\dot{\theta}_l$  の誤差の影響を考える.  $C$  は遠心力やコリオリ力をまとめたベクトルであり, 角速度の2乗の項を含むことから,  $|\Delta\dot{\theta}| \ll 1$  であれば誤差の影響は無視できると考える. こちらも刻み時間を小さくすることで  $|\Delta\dot{\theta}| \ll 1$  は成立すると考えられる.

同一条件のもとで3関節リンクの機構計算を行い, 1秒経過後のシングルスレッドと並列計算の計算結果の差を表1に示す. 各軸とも刻み時間を小さくとれば, 差は小さくなることがわかる. これより, 刻み時間を小さくとれば誤差の影響は無視できることがわかる.

表1 計算結果の差の比較

刻み時間 (sec)	$\Delta\theta_{l1}$ [rad]	$\Delta\theta_{l2}$ [rad]	$\Delta\theta_{l3}$ [rad]
1.0E-3	0.018696	0.083323	0.109300
1.0E-4	0.004967	0.006647	0.013919
1.0E-5	0.000523	0.000650	0.001419

次に3関節リンクの機構計算において, Newton-Euler法の1ループに係る平均計算時間を測定した. シングルスレッドの場合の計算時間を1とした場合, 並列計算は0.61となった. 今回, 計算に使用したパソコンはHP社 xw6400 (Dual CPU Xeon@3GHz) で4つのCPUコアを有している. 計算に使用するCPUコアの組み合わせによっては, スレッド間の同期に時間が浪費され, 計算時間が1

を超えるケースもあった。

#### 4. まとめ

本研究では、産業機械を構成する個別ユニットのエミュレータ「個別ユニットエミュレータ」の機構計算をパソコンに実装し、モーションコントローラを用いてその動作を確認した。

機構計算の並列化については、Newton-Euler法を2スレッドで実行し、刻み時間を小さくすることで並列化に伴う計算誤差が小さくなることがわかった。計算時間に関しては、CPUコアの組み合わせによってはシングルスレッドに比べ計算時間が減少することを確認したが、引き続き検証する必要がある。

今後は共同研究事業者と共同して、個別ユニットエミュレータの事業化に向けた改良を行いたい。

#### 謝 辞

本研究は、経済産業省戦略的基盤技術高度化支援事業の一環として実施しました。

#### 文 献

- [1] dSPACE社, “Solutions for Control 日本語版”, 2007.
- [2] 横山哲也 他, “リアルタイム産業機械向けエミュレータの開発”, 岐阜県情報技術研究所研究報告, 第10号, pp.19-22, 2009.
- [3] 西田吉春 他, “柔軟関節を持つマニピュレータのロボスタ制御”, 日本ロボット学会誌, Vol.12, No.3, pp.466-471, 1994.
- [4] 大明準治 他, “シリアル2慣性系の非干渉化同定”, 電気学会論文誌D, Vol.129, No.12, pp.1169-1177, 2009.
- [5] 吉川恒夫, “ロボット制御基礎論”, コロナ社, 1995.
- [6] J.Y.S.Luh 他, “On-line Computational Scheme for Mechanical Manipulators”, ASME Journal of Dynamic Systems, Measurement and Control, 102, pp.69-76, 1980.
- [7] M.W.Walker 他, “Efficient Dynamic Computer Simulation of Robot Mechanisms”, ASME Journal of Dynamic Systems, Measurement and Control, 104, pp.205-211, 1982.
- [8] J.Y.S.Luh 他, “Scheduling of Parallel Computation for a Computer-Controlled Mechanical Manipulator”, IEEE Trans, SMC-12, 2, pp.214-234, 1982.