計測自動制御学会東北支部 第242回研究集会 (2008.5.13)
資料番号 242-1

ロボットアームシミュレーションの実時間実行のための設計と実装の検討

Design and Implimentation of Robotic Arm Simulation for Real Time Execution

○齊藤圭*，佐藤宏明*，高津戸稔**，田山典男*，

○Kei Saitou*, Hiroaki Satou*, Minoru Takatuto** , Norio Tayama*,

*岩手大学 , **三菱電機

*Iwate University , **Mitsubishi Electric

キーワード： シミュレーション (Simulation)，ロボットアーム(Robottic Arm)，実時間実行 (Real Time Execution)，オブザーバモデル(Observer Model) HILS (Hardware in the Loop Simulation),

連絡先： 〒020-8551　盛岡市上田四丁目3–5 岩手大学 工学部 電気電子工学科 田山・佐藤研究室
佐藤宏明，Tel.: (019)621-6392，Fax.: (019)621-6392，E-mail: hsato@iwate-u.ac.jp

1. はじめに

様々な制御システムにおいて，電子制御は必須となっている．その設計と開発において，システム中の機能が複雑に連携しているために，機能要素ごとの開発が非常に困難かつ複雑化している．そのような複雑な制御システムを効率的かつ信頼性のあるものとして開発するため，システムを動作させながら常にターゲットの状況を把握し，リアルタイムに修正が可能な開発環境が要求されている．

従来，シミュレーション環境による装置開発は，宇宙空間のように装置の開発段階で実機を使用環境へ持ち込むことが困難である場合などで用いられている．近年では制御対象とそのシミュレーションモデルを同時に動かして観測するHILS (Hardware in the Loop Simulation)という技術が発展し，自動車業界がこれを積極的に導入している．本研究室でもHILSを用いて，よりスムーズかつ確実に要素ごとの機能を考えた開発を行っていくためのシミュレーション環境の構築を目指している[1)]．

本稿では，本研究室で開発したHILS装置の概要と，このシステム上に多関節ロボットアームをターゲットとして作成したシミュレーションモデル，及び現在考案中のシミュレーションモデルの高速化の手法について述べる．

2. HILS実験環境とターゲットシステム

2.1 HILS実験環境

本研究のHILS実験環境について説明する．計測制御環境にはdSPACE社のCPUボードDS1103と，計測用ソフトウェアControl Deskのセットで構成さ

Fig. 1　DS1103

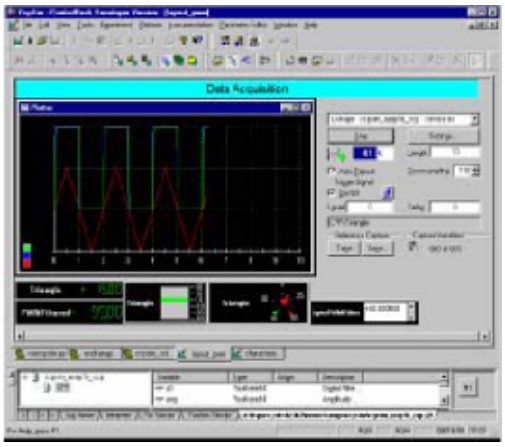

Fig. 2　Control Desk

れる統合計測環境を用いている．dSPACEはMatlabで作成したモデルをReal-Time Workshop(Simlinkで作成したモデルをＣコードなどに変換できるMatlab内のツール)によってネイティブなプログラムコードに変換し，CPUボード上のプロセッサのプログラムとして実装し，計測制御を行うことができるハードウェア環境である．DS1103はFig. 1に示すようなdSPACE社製のシングルボードコンピュータの一つで，演算用CPUや信号入出力用AD/DAコンバータ，周波数発信機などを備えている．Control DeskはFig. 2のようなユーザーが任意に計測用GUIを設計できるソフトウェアである．

2.2　シミュレーションターゲット

シミュレーションターゲットをFig. 3に示す．三菱電機製MOVE MASTER RV-M1(５自由度多関節型ロボットアーム)を用いている．このターゲットは５つの関節それぞれが相互に影響しあっているため，複雑な動作になるのでシミュレーション環境を実証するのに適していると判断した．駆動方式はDCサーボモータによる電気サーボ駆動，制御方式はPTP位置制御方式で動作している．

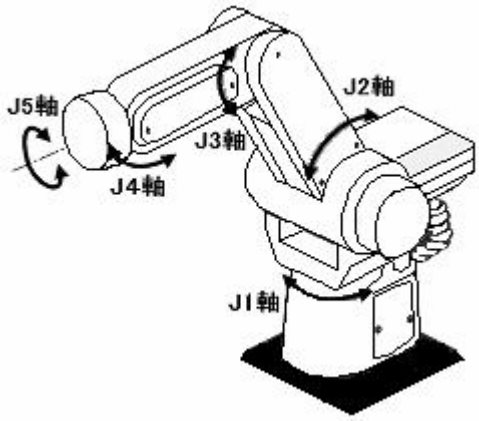

Fig. 3　MOVE MASTER RV-M1

3.　シミュレーションモデルの構成

3.1　シミュレーションモデルの概要

Fig. 4にシミュレーションモデルを示す．シミュレーションモデルの作成にはMath Works社のMatlab/Simulink環境にシームレスに統合されているSimMechanics ToolBoxを用いている．Simulinkは Matlab内でシステムのモデリングや数値計算をブロック図表現で行うことができる対話型設計支援ツールである．SimMechanicsはSimulinkの中でも，ボディやジョイントなどのメカニカルなコンポー

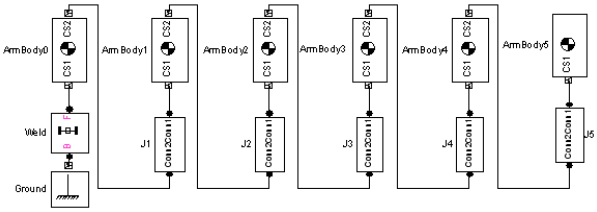

Fig. 4　シミュレーションモデル

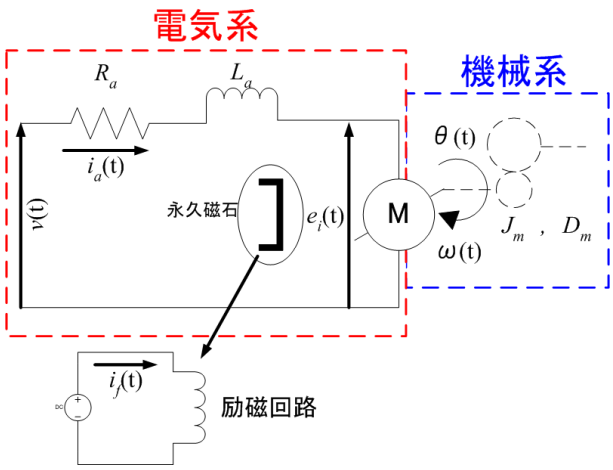

Fig. 5　DCサーボモータの等価回路図

ネントをブロック図表現したツールボックスである．シミュレーションモデルはロボットアームをジョイントと剛体リンクの要素の集合とみなし，ジョイントと剛体リンクが一組になって構成されているマルチボディモデル表現[2]として構築されている．

3.2　機械系のシステムブロック

まずDCサーボモータの運動方程式を導出する．DCサーボモータの等価回路をFig. 5に示す．機械系においてはニュートンの法則に従って，以下の式で表される．

$$\begin{cases} \dfrac{d\theta(t)}{dt} = \omega(t) \\ J_m \dfrac{d\omega(t)}{dt} + D_m\omega(t) = T_g(t) - T_L(t) \end{cases} \tag{1}$$

$$T_g(t) = pMi_f(t)i_a(t) = K_t'\phi(t)i_a(t) = K_t i_a(t) \tag{2}$$

(永久磁石モータのため界磁一定：$i_f = I_f$ =一定)

ここで，K_t：モータトルク定数，$\theta(t)$：回転角度，$\omega(t)$：回転角速度，$T_L(t)$：負荷トルク，J_m：慣性モーメント，$i_a(t)$：電気子電流，$i_f(t)$：界磁電流，D_m：モータ粘性抵抗，p：極対数，$\phi(t)$：界磁磁束，M：電気子巻線と界磁巻線間の相互インダクタンスである．

3.3　電気系のシステムブロック

(3.2)同様に界磁を一定とすると，キルヒホッフの法則より次式の関係を得る．

$$L_a \frac{di_a(t)}{dt} + R_a i_a(t) = v(t) - e_i(t) \tag{3}$$

ここで，$v(t)$：直流印加電圧，R_a,L_a：電気子巻線及び回路の全抵抗と全インダクタンスである．逆起電力は次式で表現される．

$$e_i(t) = pMi_f(t)\omega(t) = K_e\omega(t) \tag{4}$$

ここで，K_e：逆起電力定数である．

4.　シミュレーションモデルのシステム表現

4.1　状態空間表現によるシステム表現

現代制御論の状態空間表現に基づいてロボットアームのシステム表現を行った[3]．状態空間表現は内部信号の微分方程式である状態方程式と出力方程式とで記述される[4]．状態方程式を式(5)に、出力方程式を式(6)に示す。

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{5}$$

$$y(t) = Cx(t) + Du(t) \tag{6}$$

ここで，$x(t)$：システムの内部信号，$u(t)$：操作量，$y(t)$：制御量を表し，$\dot{x}(t) = \dfrac{dx(t)}{dt}$を指す．また，$A$：n×n行列，$B$：n×1ベクトル，$C$：1×nベクトル，$D$：スカラーである．このモデルにお

いては$x(t)$，$\dot{x}(t)$，A，B，C，D，はそれぞれ以下のようになる．

$$x(t) = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} \theta(t) \\ \dot{\theta}(t) \\ \theta_m(t) \\ \dot{\theta}_m(t) \\ i_a(t) \end{bmatrix}$$

$$\dot{x}(t) = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{x}_4 \\ \dot{x}_5 \end{bmatrix} = \begin{bmatrix} \dot{\theta}(t) \\ \ddot{\theta}(t) \\ \dot{\theta}_m(t) \\ \ddot{\theta}_m(t) \\ \dot{i}_a(t) \end{bmatrix}$$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ \frac{-NK_g}{J_1} & \frac{-D_g}{J_1} & \frac{NK_g}{J_1} & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ \frac{NK_g}{j_m} & 0 & \frac{-K_g}{J_m} & \frac{-D_m}{J_m} & \frac{K_t}{J_m} \\ 0 & 00 & \frac{K_e}{R_a T_m} & \frac{-1}{T_m} & \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \frac{1}{R_a T_m} \end{bmatrix}$$

$$C = [1 \quad 1 \quad 1 \quad 1 \quad 0]$$

$$D = 0$$

ここで，N：ギヤ数，k_g：ギヤトルク定数，D_g：ギヤ粘性抵抗，J_l：負荷モーメント，R_a：電気子抵抗，T_m：モータ時定数である．Fig. 6に状態方程式に基づいて作成したJ2軸のジョイントブロックを，Fig. 7に内部のモーターブロックを示す． また、Fif. 8にこのモデルとロボットアームのJ2軸を動作させたときの信号を比較した結果を示す．モデルとロボットアームの動作は一致しなかった.これはロボットアームの動作信号と物理モデル内のパラメータの組み合わせの違いによるものだが，モデル内のパラメータはこれ以上調べることができない．そこで，オブザーバを用いて内部状態を

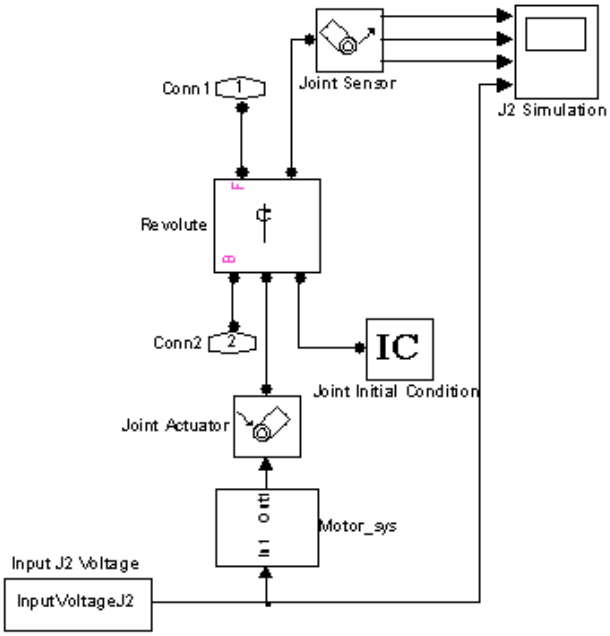

Fig. 6　J2軸のジョイントブロック

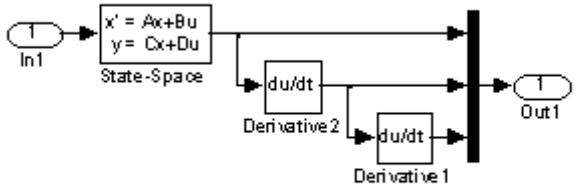

Fig. 7　モータの内部ブロック図

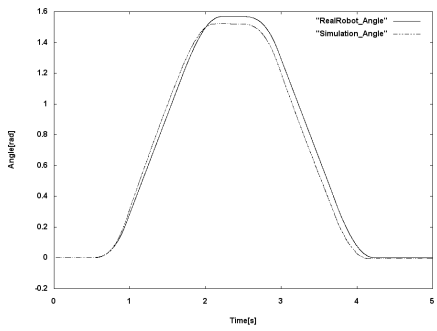

Fig. 8　状態空間表現モデルの動作結果

推定することでモデルの精度を向上させることを試みた．

4.2 全状態観測器

式(5)のような状態方程式の状態量をすべて推定する観測器を状態観測器と呼ぶ．全状態観測器の式を求める．まず，式(5)と同じ特性を持つ物理モデルをつくり，それを操作信号$u(t)$で動作させたとする．

$$\frac{d\hat{x}(t)}{dt} = A\hat{x}(t) + Bu(t) \tag{7}$$

このとき推定状態量を$\hat{x}$として$x(t)$に一致するような特性を持つか考える．それには，誤差

$$e(t) = \hat{x}(t) - x(t) \tag{8}$$

が0に収束するかどうかを考えればよい．そこで，式(5)から式(7)を引くことで次のような誤差方程式を得る．

$$\begin{aligned}\frac{de(t)}{dt} &= \frac{d\hat{x}(t)}{dt} - \frac{dx(t)}{dt} \\ &= A\hat{x}(t) + Bu(t) - Ax(t) - Bu(t) \\ &= Ae(t)\end{aligned} \tag{9}$$

この推定誤差の動特性は，元のシステムの動特性と同じ振る舞いをする．ここで，オブザーバの動特性を改善し推定精度を向上するために，オブザーバの出力$\hat{y}$と系の出力$y(t)$の差をフィードバックすることを考える．

$$\begin{aligned}\frac{d\hat{x}(t)}{dt} &= A\hat{x}(t) + Bu(t) - L(C\hat{x}(t) - y(t)) \\ &= (A - LC)\hat{x}(t) + Ly(t) + Bu(t)\end{aligned} \tag{10}$$

ここで，L：n×m行列はオブザーバのフィードバックゲイン行列である．式(8)，式(9)で求めたように，誤差の動特性を求める．

$$\begin{aligned}\frac{de(t)}{dt} &= \dot{\hat{x}}(t) - \dot{x}(t) \\ &= (A - LC)\hat{x}(t) + LCx(t) \\ &\quad + Bu(t) - Ax(t) - Bu(t) \\ &= (A - LC)[\hat{x}(t) - x(t)]\end{aligned} \tag{11}$$

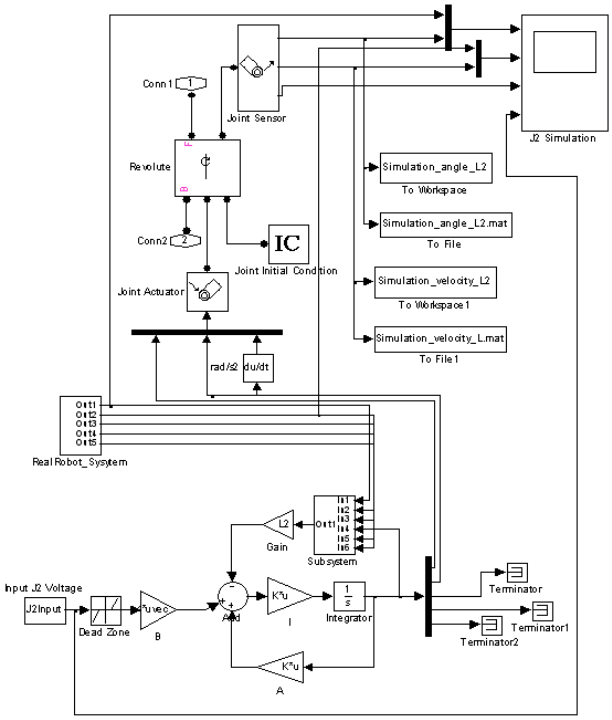

Fig. 9　オブザーバモデルを実装した軸のシステムブロック

よって、行列$(A - LC)$によって誤差が収束することがわかる．(C, A)が可制御であれば，Lの固有値は誤差方程式

$$e(t) = \exp(A - LC)e(0) \tag{12}$$

の解として求めることができる[5]ここで行列$(A - LC)$の固有値の実部がすべて負であれば，この誤差方程式は任意の初期状態に対して誤差は0に収束する[6]．こうして求めたオブザーバゲインLを実装した物理モデルの軸の部分をFig. 9に示す．また，このモデルを用いて実機のロボットアームとの信号比較した結果をFig. 10〜Fig. 13に示す．

物理モデルとロボットアームの信号がほぼ一致していることがわかる．

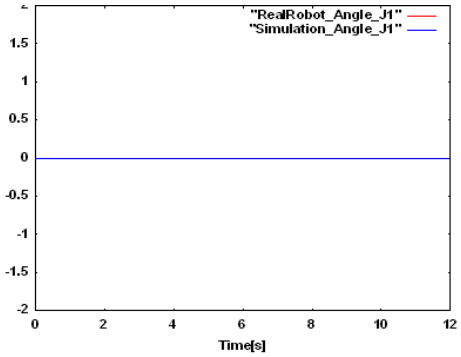

Fig. 10　J1軸0[rad]動作時の変位]

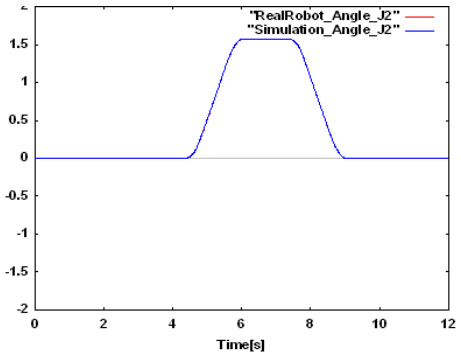

Fig. 11　J2軸π/2[rad]動作時の変位

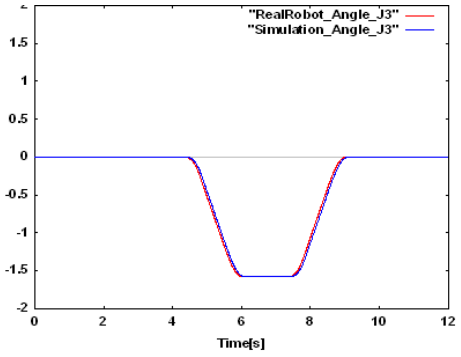

Fig. 12　J3軸π/2[rad]動作時の変位

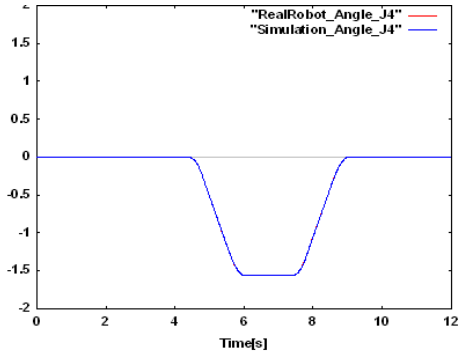

Fig. 13　J4軸π/2[rad]動作時の変位

5.　シミュレーションモデルの高速化の手法

5.1　シミュレーションモデルの高速化

オブザーバを用いることでシミュレーションの動作が誤差の少ないものになった．しかし，現在のモデルを動かすサンプリング周期は0.01[sec]であり，HILSの目的である実機では再現困難な現象の再現及びその解析をするにはサンプリング速度が遅い．具体的には，数百ヘルツの運動の動作解析を目指しているので，サンプリング周期が0.001[sec]程度になるようにしたいと考えている．そこで，サンプリング速度を高速化したシミュレーションモデルの開発を現在行っている．

5.2　タスク分割

モデルを高速化する手段としてタスク分割によるサイクルタイムの短縮を考えた．dSPACE環境では，複数タスク間における割り込み処理を解決する方法として，タスクをサイクルタイムによって分割して優先度の高いタスクを先に行い，優先度の高いタスクの処理し終わった情報を優先度の低い後のタスクでバッファして用いるということができるようになっている．このことを利用して，Fig. 14に示すように現在のモデルのジョイント部分J1～J5のタスクを分割し，そのタスクを並列に

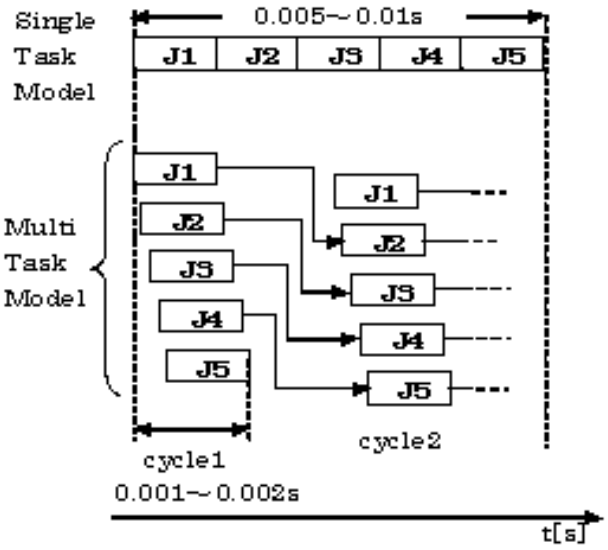

Fig. 14　タスク分割のイメージ

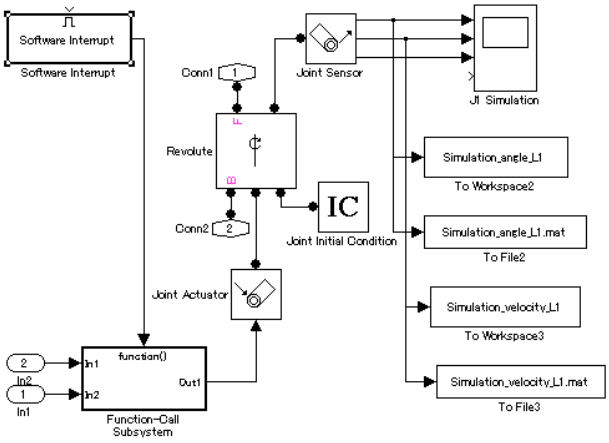

Fig. 15　開発したモデルのJ1軸のジョイントブロック

並べる事でサイクルタイムを高速化することを考えた．このマルチボディモデル表現のモデルではJ2～J5はそれぞれ前のジョイントから位置や角度，角速度などの情報を継承しなければならないが，直前のジョイントの１サイクル前の情報をバッファして用いることで並列計算が可能にできると考えた．開発したモデルのJ1軸をFig. 15に示す．

6.　まとめと今後の課題

本研究ではシステム開発のためのシミュレーション環境を構築するために，シミュレーションモデルを状態空間表現用いて構築し，オブザーバを利用することでその精度を向上させた．

今後の課題としては，シミュレーションモデルの高速化の手法として現在試しているタスク分割の方法をモデルに繰み込み，実装できるようにすること．また，モデルを他の制御方法で構築することで高速化できないか検討することなどが挙げられる．

参考文献

1) 池田，高津戸他:シミュレーションによるシステム制御装置の最適設計，電気関係学会東北支部連合大会，1F4 (2003)

2) Giles D. Wood他:Simulating Mechanical Systems in Simulink with SimMechanics, http://www.mathworks.com/matlabcentral/ (2003)

3) 佐藤，高津戸他　:マルチボディモデルによる多関節ロボットアームのシミュレーション設計法，情報科学技術フォーラム，C-023，233/236，(2005)

4) 井上和夫: MATLAB/Simulinkによるわかりやすい制御工学，159/165，森北出版 (2001)

5) 野波建造：ＭＡＴＬＡＢによる制御理論の基礎，175/187，東京電機大学出版局 (1998)

6) 高津戸　稔:多関節ロボットアームのシミュレーションのための計測制御装置に関する研究，修士学位論文，74/77 (2006)