The Robotics Toolbox for MATLAB

藤田研究室　　尹　磊々

平成 18 年 6 月 5 日

1　はじめに

文献 [1] には，ビジュアルフィードバック制御 (Visual Feedback Control) は 3 次元で出来ているが, シミュレーションや実験は水平 2 自由度のマニピュレータでしかやっていなかった．そして，ビジュアルフィードバック制御を三次元マニピュレータで検証したい．さらに，これからモデル予測制御，ビジュアルフィードバック制御とマニピュレータの Synchronization 問題の融合するためにも，三次元でのシミュレーションをやりたい．本レポートでは，参考文献 [2] を基づいて，Matlab 上で利用可能な 3 次元マニピュレータ (3-Dimensional manipulator)PUMA560 のシミュレーションモデル，Robotics Toolbox for MATLAB を紹介することは目的となる．

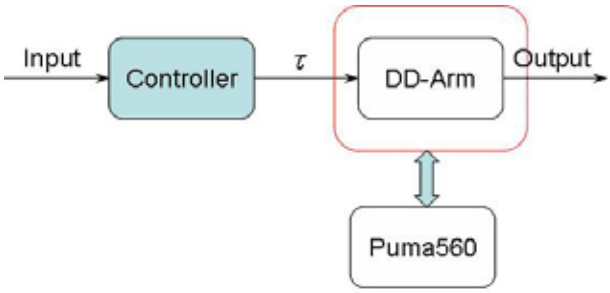

Fig.1: Layout

Fig. 1 はシステムの概略図である．今までシミュレーションや実験では二次元のロボットアーム DD-Arm を利用した．そして，DD-Arm に替わって，三次元の 6 自由度マニピュレータ (PUMA560) を入れたい．DD-Arm と入れ替えって，マニピュレータを動かすためには，まずマニピュレータのいろいろな基本情報が必要となる．そして，その情報に基づいて，コントローラを再構築しなければいけない．このレポートでは，第 2 章にはこれから 3 次元マニピュレータシステム構築し，コントローラを作るための動力学の紹介をメインとして，Toolbox の機能を紹介する．そして，第 3 章では，マニピュレータを動かすために必要な情報を 2 章で紹介する動力学の関数など利用して，マニピュレータの基本情報を計算し，新たにコントローラを作って，マニピュレータを動かすのは目的である．
PUMA560 のモデル図は Fig. 1 で表す．また PUMA560 の各関節のパラメータは Table1 に示す．し

2　Robotics Toolbox について

この Robotics Toolbox を利用して、自由にパラメータを設定して多リングマニピュレータ (serial-link manipulator) を作れる．そして，Toolbox 中にもよく使われている PUMA560 や Stanford Arm といったマニピュレータのモデルが予め用意されている．そして，Toolbox には同次変換 (homogeneous transformations)，Quaternions，軌道の生成 (Trajectory Generation) といったロボット制御に必要な情報を簡単に作成や計算できる関数を用意されている。また、マニピュレータの解析 (analyze) に必要な順運動学（forward kinematics)，逆運動学（inverse kinematics)，順動力学 (forward dynamics)，逆動力学 (inverse dynamics) なども簡単に算出できる．この章では，主にマニピュ

Table 1: Parameter of PUMA560

Link	α	A	θ	D	range of θ
1	90 °	0	θ_1	0	-160 °〜160 °
2	0	0.4318	θ_3	0	-225 °〜45 °
3	-90 °	0.0203	θ_3	0.15	-45 °〜225 °
4	90 °	0	θ_4	0.4318	-110 °〜170 °
5	-90 °	0	θ_5	0	-100 °〜100 °
6	0	0	θ_6	0	-266 °〜266 °

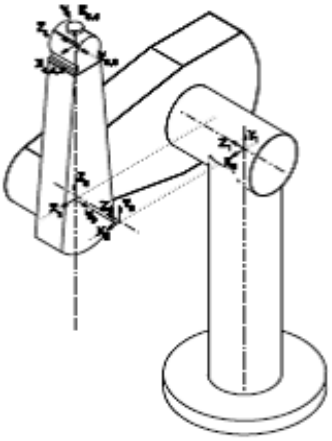

Fig.2: PUMA560 Robot

レータの運動解析やそれを制御するためのコントローラ設計に必要となるマニピュレータ動力学 (dynamics) を説明する．そして，今回はマニピュレータ制御に直接関わらないが，マニピュレータ運動学 (kinematics) について軽く説明する．また，多リンクマニピュレータの設計や解析に必要となる座標変換 (Coordinate Transformations)(同次変換 (homogeneous transformations) や Quaternions，Trajectory) については今回のレポートに直接に関わらないので，付録で説明する．

2.1 運動学 (Kinematics)

Toolbox には順運動学 (forward kinematics)，逆運動学 (inverse kinematics) 及びヤコビアン (Jacobian) などの計算関数が入っている。マニピュレータ運動学に関する関数は Table 2 に表す。順運動学解法はマニピュレータの角関節の変位 (Translation) と回転角度 (Roatation) を与えて，エンドエフェクタの位置，姿勢を求める方法 (式 1) である． *fkine* 関数は与えられた関節座標 (joint coordinate)q から end-effector の位置座標の同次表現 T を順運動学解法で計算する．書き方は $T = fkine(robot, q)$.

$$ {}^0T_n = {}^0A_1^1A_2 \cdots {}^{n-1}A_n = K(\underline{q}) \tag{1} $$

順運動学解法よりもっとよく使われているのは，逆運動学解法，制御したい手先 (例えばエンドエフェクタの中心点) の位置座標は (x, y, z)，姿勢角度 (α, β, θ) を与えて，角関節の角度，変位 $(q_1, q_2, \cdots q_6)$ を算出する方法. 式 2 で表す．逆運動学は $q = ikine(robot, T)$ で計算する。*ikine* は同次座標 T を与えられた end-effector からマニ

Table 2: Kinematics

Kinematics	
diff2tr	differential motion vector to transform
fkine	compute forward kinematics
ikine	compute inverse kinematics
ikine560	compute inverse kinematics for Puma 560 like arm
jacob0	compute Jacobian in base coordinate frame
jacobn	compute Jacobian in end-effector coordinate frame
tr2diff	homogeneous transform to differential motion vector
tr2jac	homogeneous transform to Jacobian

ピュレータ *robot* の各関節の座標 q を求める．

$$\underline{q} = K^{-1}(T) \tag{2}$$

マニピュレータの end-effector の位置や姿勢の微小変位と関節の微小変位の関係を表すヤコビアン行列 J_q である。n 自由度マニピュレータの end-effector の座標は

$${}^0\underline{\dot{x}}_n = {}^0J_q(\underline{q})\underline{\dot{q}} \tag{3}$$

$${}^{T_n}\underline{\dot{x}}_n = {}^{T_n}J_q(\underline{q})\underline{\dot{q}} \tag{4}$$

で表す．この二つのヤコビアンは Toolbox では ${}^0J_q = jacob0(robot, q), {}^{T_n}J_q = jacobn(robot, q)$ といった関数がある．

2.2 動力学 (Dynamics)

マニピュレータを素早く目標に達する為に，リンクを高速で動かす必要がある．また，三次元でリンクの重量，や各関節の粘性摩擦力，動摩擦力を考慮しなくてはならない．前節で説明したマニピュレータ運動学の解法を使っても，ゆっくりマニピュレータを動かすことが出来るが，トルクなど全く考えてないので，実際では使えない．そのため，マニピュレータを制御するため，マニピュレータの運動に影響及ぼす各種力を考慮した方法を使う必要がある．そして，マニピュレータ動力学は運動方程式を基づいて，アクチュエータなどから受けた外力でマニピュレータを動かす．多関節マニピュレータの運動方程式は式 5 で表す．

$$\underline{Q} = M(\underline{q})\underline{\ddot{q}} + C(\underline{q}, \underline{\dot{q}})\underline{\dot{q}} + F(\underline{\dot{q}}) + G(\underline{q}) \tag{5}$$

ここで

- $\underline{q}$ マニピュレータの姿勢 (pose) を表す関節一般化座標 (generalized joint coordinates)Vector
- $\underline{\dot{q}}$ 関節速度 (velocity)Vector
- $\underline{\ddot{q}}$ 関節加速度 (acceleration)Vector
- M マニピュレータの慣性 (inertia) tensor
- C Coriolis 力及び遠心 (centripetal) 力項
- F (粘性，動) 摩擦項
- G 重力項
- Q 一般化座標での $\underline{q}$ に対する一般化力 Vector

Table 3: Dynamics

Dynamics	
accel	compute forward dynamics
cinertia	compute Cartesian manipulator inertia matrix
coriolis	compute centripetal/coriolis torque
friction	joint friction
ftrans	transform force/moment
gravload	compute gravity loading
inertia	compute manipulator inertia matrix
itorque	compute inertia torque
nofriction	remove friction from a robot object
rne	inverse dynamics

この Toolbox に用意された動力学関数は Table 3 で表す.

動力学は関節トルクを計算する逆動力学とシミュレーションでよく使われるトルクから関節加速度を出す順動力学がある.

順動力学 (forward dynamics) は関節トルクが与えられたとき，関節から発生する加速度を求める方法である．マニピュレータシミュレーションによく使われている．順動力学は Toolbox では *accel* 関数を利用して解く.

$$\ddot{\underline{q}} = M(\underline{q})^{-1}\underline{\tau} - C(\underline{q}, \dot{\underline{q}})\dot{\underline{q}} - G(\underline{q} - F(\dot{\underline{q}}) \tag{6}$$

しかし，式 6 に表しているように，加速度 $\ddot{\underline{q}}$ を求めるには，入力 τ 以外，関節速度 $\dot{\underline{q}}$ 及び関節回転角 $\underline{q}$ が必要となる．そして，順動力学から得られた加速度 $\ddot{q}$ を積分して，関節速度 $\dot{q}$ 及び関節変位 q を算出しなければいけない．順動力学の積分計算は Toolbox の $fdyn()$ 関数 (式 7) が利用できる.

$$\begin{aligned}
[t\ q\ qd] &= fdyn(robot, t0, t1) \\
[t\ q\ qd] &= fdyn(robot, t0, t1, torqfun) \\
[t\ q\ qd] &= fdyn(robot, t0, t1, torqfun, q0, qd0) \\
[t\ q\ qd] &= fdyn(robot, t0, t1, torqfun, q0, qd0, arg1, arg2, \cdots)
\end{aligned} \tag{7}$$

7 式の $fdyn$ 関数はマニピュレータの運動方程式 (式 6) を時間 $t0$ から $t1$ にかけて Mallab の $ode45$ 関数で積分し，行は時系列的に列は関節毎に，速度と回転角度を出力させる関数である.

逆動力学 (inverse dynamics) は与えられた関節角度，速度，加速度を実現するための関節トルクを計算する方法である．逆動力学の計算はニュートン－オイラ法 (Newton-Eular Formulation)[1]がよく使われている．Toolbox の逆動力学計算関数 $\tau = rne(robot, q, qd, qdd)$ はこの方法を利用している．Puma560 マニピュレータのコントローラの設計に必要となるロボットの慣性行列やコリオリ力，遠心力及び重力トルクなどを導出すためこの逆動力学関数がよく使われる．次章は PUMA560 のコントローラを作るのに，この逆動力学関数 $rne()$ を利用する.

[1]Robot Modeling And Control の 7．6 節 p.271-p.282 に Newton-Eular Formulation による動力学の詳しい説明が乗っている

3　6 自由度マニピュレータ PUMA ５６０の制御

2 章では，Toolbox の運動特性や動力特性をメインで説明した．そして，この章では Puma560 のモデル用のコントローラを作成し，Puma560 を利用したシミュレーションを説明する．

- 制御対象　－－　PUMA560 のシミュレーションモデル
- 制御則　－－ Paden and Panja の制御則

$$\begin{cases} \tau &= M(q)\ddot{q_d} + C(q,\dot{q})\dot{q_d} + G(q) - K_p e + \gamma \\ \gamma &= -K_d \dot{e} \end{cases} \tag{8}$$

- 制御目的 －－ 与えられた目標角度へ追従する．

制御則の式 8 では，τ はトルク入力で，

$$\tau = M(q)\ddot{q_d} + C(q,\dot{q})\dot{q_d} + G(q) - K_p e - K_d \dot{e}$$

と書くことが出来る．ここでは，$M(q)$ は慣性行列，$C(q,\dot{q})$ はコリオリ力及び遠心力項，$G(q)$ は重力項，K_p と K_d はゲイン，$e = q - q_d$ $\dot{e} = \dot{q} - \dot{q_d}$ は位置と速度偏差を表す．

$q, \dot{q}, \ddot{q}$ が分かれば，マニピュレータ Puma560 のモデルにおける $M(q)$ ，$C(q,\dot{q})\dot{q_d}$，$G(q)$ は 2 節に説明した動力学の関数 (table 3) を利用して計算できる．この Toolbox では，これらを計算する関数が用意してある．

慣性行列	$M(q) \;=\; inertia(robot,\; q)$
コリオリ力，遠心力トルク	$C(q,\dot{q})\dot{q_d} \;=\; coriolis(robot,\; q,\; qd)$
重力トルク	$G(q) \;=\; gravload(robot,\; q)$

この 3 つの関数の計算方法は 2.2 節で説明した逆動力学関数 $rne()$ で計算する．

式 5 で表したマニピュレータの運動方程式に，摩擦力をゼロと考えて，トルクは τ で書き直すと $\tau = M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q)$ である．

そして，慣性行列 M の計算式は $M = rne(robot, ones(n,1) * q, zeros(n,n), eye(n), [0;0;0])$ 重力項をゼロにして，速度 $\dot{q}$ もゼロにすると，$\tau = M(q)\ddot{q}$ だけになるので，M を求めることが出来る．

コリオリ力・遠心力項 $C(q,\dot{q})$ は式　$tau_c = rne(robot, q, qd, zeros(size(q)), [0;0;0])$ で書いて，重力と加速度をゼロと指定すれば $\tau = C(q,\dot{q})\dot{q}$ だけが残る．$\dot{q}$ は入力なので，$\dot{q}$ の疑似逆行列を計算すれば，$C(q,\dot{q})$ を計算することが出来る．

重力トルクの $G(q)$ は速度，加速度ともにゼロにすれば，つまりマニピュレータにある姿勢に与えて，動かなければ $\tau \;=\; G(q)$ となる．ですので，逆動力学の計算式を利用して $tg = rne(robot, q, zeros(size(q)), zeros(size(q)))$ で $G(q)$ を導出することが出来る．

そして，コントローラを作成するために必要となる M, C, G が分かったら，Paden and Panja の制御則を組み込んだコントローラを作った．このコントローラを利用して，モデル PUMA560 のシミュレーションを行った．全部 6 軸でも動けるけど，計算量が非常に多くなるので，6 軸の中の 1–3 軸を動かして，シミュレーションを行った．Sin 波の目標信号を入れて，シミュレーションの結果は Fig. 3 に示す．

- 結論

  結果のグラフから見てわかるように，各関節に与えた Sin 波の目標信号に対して，制御偏差 $e \to 0$ になることが分かった．また，速度偏差はまだ完全に $\dot{e} \to 0$ と言えないが，こうなる傾向が分かる．

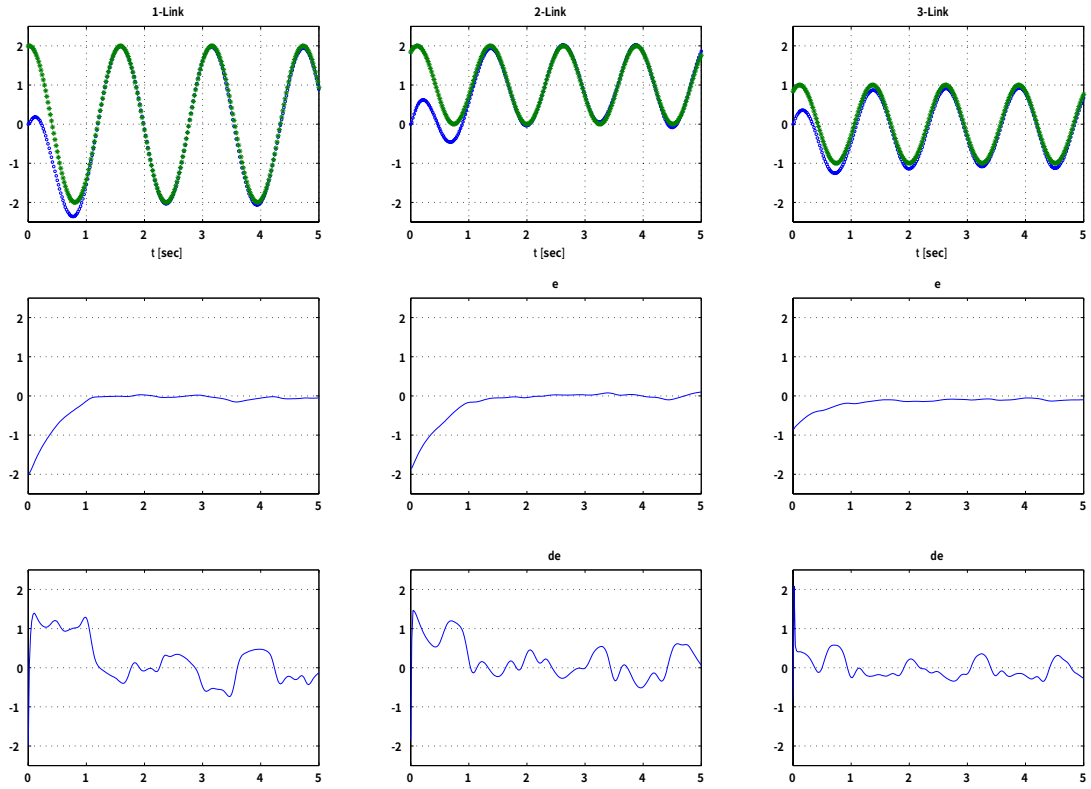

Fig.3: Result of Simulation

4　おわりに

本レポートは Matlab で利用できる 3 次元的な 6 自由度マニピュレータのモデル PUMA560 をメインで， The Robotics Toolbox for Matlab を紹介した．PUMA560 モデルの慣性行列，遠心力コリオリ力，そして重力トルクを Toolbox 中の関数を利用して出した．これらのデータで Paden and Panja の制御則を組み込んだコントローラを作って，PUMA560 モデル上でシミュレーションと実行した結果，目標値に対して，ほぼ偏差無く追従することが出来た．

これから，今までビジュアルフィードバック制御で 2 自由度の DD-Arm によるシミュレーションに置き換えて，PUMA560 のモデルを導入して，三次元でのシミュレーションを行うと考えている．そして，ビジュアルフィードバック制御だけではなくて，モデル予測制御，マニピュレータの Synchronization 及びそれらを融合した制御のシミュレーションモデルとして持って行きたいと考えている．

付録 A　座標変換 (Coordinate Transformations)

この Toolbox では，マニピュレータの各関節間の関係は D-H パラメータ (Denavit-Hratenberg parameters)(Table 4 を利用している。

3 次元での多関節マニピュレータを解析するためには，同次変換行列がよく使われている．Table 4 のパラメータ利用して，同次表現は式 9 で表す。

Table 4: Link and Joint Parameters

a_i	link length	the offset distance between the z_{i-1} and z_i axes along the x_i axis
α_i	link twist	the angle from the z_{i-1} axis to the z_i axis about the x_i axis
d_i	link offset	the distance from the origin of frame $i-1$ to the x_i axis along the z_{i-1} axis
θ_i	joint angle	the angle between the x_{i-1} and x_i axes about the z_{i-1} axis

$$ {}^{i-1}A_i = \begin{pmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{9} $$

この Toolbox では同次変換などに関する演算子は Table 5 を参照してください．この Table 5 は Toolbox で予

Table 5: Homogenous Transforms

Homogenous Transforms	
eul2tr	Euler angle to homogeneous transform
oa2tr	orientation and approach vector to homogeneous transform
rotvec	homogeneous transform for rotation about arbitraty vector
rotx	homogeneous transform for rotation about X-axis
roty	homogeneous transform for rotation about Y-axis
rotz	homogeneous transform for rotation about Z-axis
rpy2tr	Roll/pitch/yaw angles to homogeneous transform
tr2eul	homogeneous transform to Euler angles
tr2rot	homogeneous transform to rotation submatrix
tr2rpy	homogeneous transform to Roll/pitch/yaw angles
transl	set or extract the translational component of a homogeneous transform
trnorm	normalize a homogeneous transform

め用意した座標変換の関数 (function) である。これらの関数の回転に関するもの *rotx*，*roty*，*rotz*，及び並進の関数 *transl* を利用すれは，関節の同次表現が簡単に作れる．また，剛体 (rigid body) の 3 D 空間での座標表現が，同次表現以外でもロール・ピーチ・ヨー角 (Roll/pitch/yaw angles) やオイラ角 (Euler angle) の表現法で表せる。Toolbox で用意した *eul2tr*，*tr2eul*，*rpy2tr*，*tr2rpy* などといった関数を利用して，同次表現とオイル角，及びロール・ピーチ・ヨー角による表現と自由に変換することが出来る。

同時表現以外，姿勢と位置を表すには quaternion も用いられる．Quaternion は $q = [s, \underline{v}]$ で表す．s はスカラーで，$\underline{v} \in \Re^3$ である．Table 6 に用意された Quaternion に関する計算関数である．Quaternion は同時変換に比べ，回転は 4 つの数字で表せるので，無駄な計算を減らす，効率よく計算が速くなる利点がある．

参考文献

[1] T. Murao, T. Yamada and M. Fujita, "Predictive Visual Feedback Control with Eye-in-Hand System via Stabilizing Receding Horizon Approach," Proc. of the 45th IEEE Conference on Decision and Control, 2006 (submitted).

Table 6: Quaternion

Quaternion	
/	divide quaternion by quaternion or scalar
***	multiply quaternion by a quaternion or vector
inv	invert a quaternion
norm	norm of a quaternion
plot	display a quaternion as a 3D rotation
q2tr	quaternion to homogeneous transform
qinterp	interpolate quaternions
quaternion	construct a quaternion
unit	unitize a quaternion

[2] P. I. Corke, "A Robotics Toolbox for MATLAB," IEEE Robotics and Automation Magazine, Vol. 3, No.1, pp. 24–32, 1996.

[3] P. I. Corke, "A computer tool for simulation and analysis: the Robotics Toolbox for MATLAB," Proc. of the 1995 National Conference of the Australian Robot Association, pp 319–330, 1995.

[4] B. Panen and R. Panja, "Globally asymptotically stable 'PD+' controller for robot manipulators." International Journal of Control, Vol. 47, No. 6, pp. 1697–1712, 1988.