計測自動制御学会東北支部 第208回研究集会 (2003.5.22)
資料番号 208-1

ロボットアームの姿勢学習と制御

Attitude Learning and Control of Robot Arm

○落宰 公志*, 西山 清*

○ Koushi Ochisai*, Kiyoshi Nishiyama*

*岩手大学工学部情報システム工学科

*Dep. of Computer & Infomation Science, Iwate University

キーワード： ニューラルネットワーク (neural network), ロボットアーム (robot arm),
姿勢学習 (attitude learning), H_2学習 (H_2-learning), TCP/IP 通信 (TCP/IP communication)

連絡先： 〒020-8551 盛岡市上田 4-3-5 岩手大学 工学部 情報システム工学科
西山 清, Tel.: 019-621-6475，Fax.: 019-621-6475，E-mail: nisiyama@cis.iwate-u.ac.jp

1. はじめに

ロボット研究において名工たちの卓越した技能を如何に抽出し、コンピュータ上に実装するかは重要なテーマであり、近年多くの研究がなされている。

本研究では、筆跡学習のためのロボットアーム制御システムの開発を目的としている。その際、ロボットの腕先の姿勢が重要となる。今回は学習時に教師データから教示者の姿勢 (くせ) をニューラルネットワークによって学習し、動作時には直交座標データから姿勢角を想起してロボットアームを制御するシステムを考案した。これより、マウスによってロボットアームを遠隔から自由に操作できることを示す。

2. ロボットアームの仕様

本研究では 5 つの関節を持つ自由度 5 の産業用ロボットを使用する。ロボットの直交座標軸と姿勢角および各関節軸の名称、位置変数を図 1に示す。

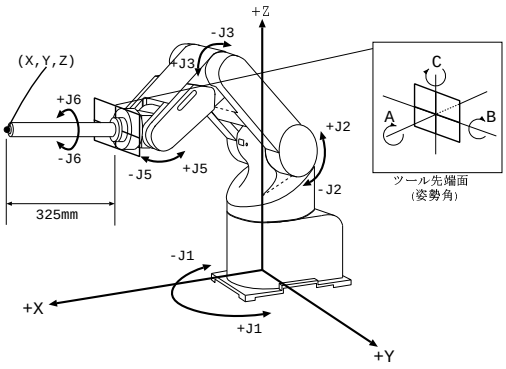


Fig. 1 ロボットの直交軸及び各関節;
ロボット本体：RV-2AJ(5 軸垂直多関節形)
コントローラ：CR1-571
関節自由度：5

ここで、姿勢角とは図 1のツール先端面の向きであり、A、B、C の角度で表す。なお、C は 6 軸ロボット用の姿勢データであり、図 1のような自由度が 1 つ少ない 5 軸のロボットでは C は 0 となる。

また、描画時において姿勢角 A の変化による影響は非常に小さいものとし、A は 0 とした。

これらの姿勢角 (A,B,C) と直交座標 (X,Y,Z) を要素とする位置変数を指定し、コントローラボックスにおいて BASIC で記述された動作プログラムを実行することによって、ロボットアームの先端を指定した位置へ移動させることができる。例えば、以下のような BASIC プログラムをコントローラボックスで実行すればロボットアームの先端を位置変数"P1"で指定された位置へ移動させることができる。

```
10   MOV   P1        '位置"P1"へ移動
20   END
P1 = (400,100,300,0,90,0)
```

ただし、ロボットの制御点は初期の設定では図 2(a) のように姿勢角によって大きく先端の位置が変化してしまう。よって、姿勢角があまり大きな影響を与えないようにツールの先端に制御点を変更している。

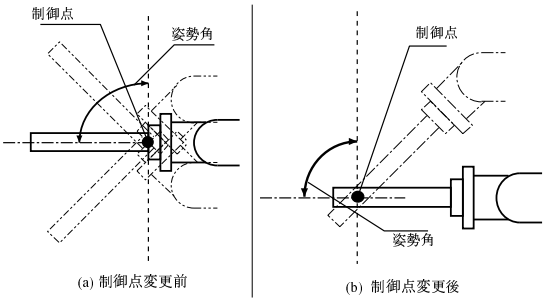


Fig. 2　制御点と姿勢角

図 2からも分るように、1 つの座標に対する姿勢のとり方は複数存在する。しかし、どんな姿勢でも動作が可能なわけではなく、設定された動作範囲を越えたり、無理な姿勢をとらせようとした場合、ロボットはエラーを返して緊急停止する。

3.　姿勢学習と制御

3.1　全体の処理手順

人間が手で様々な作業を行えるのは、人間の脳の中に"間接空間"から"作業空間" への写像の逆写像が生成されていると考えることができる。しかし、人間の腕、手、指には冗長性があり、この逆写像は一意ではない。同様に、人間の腕を模したロボットアームもある位置に対しての姿勢は一意ではなく、この動作を制御するには 3 次元の直交座標とその位置に対応した姿勢角の値を組み合わせ、位置変数としてロボットに伝える必要がある。しかし、直交座標は任意に決定できても、姿勢角は複数の関節によって決定されるため、計算で算出することは困難である。しかも、前節で述べたとおり、不正な姿勢が入力された場合、ロボットは停止してしまう。そこで、ニューラルネットワークを用いて直交座標に対する姿勢角を学習し、得られたネットワークから姿勢角を想起させることによってロボットアームを制御するための位置変数を得る方法を考案した。図 3には姿勢の学習から想起までの流れを示す。

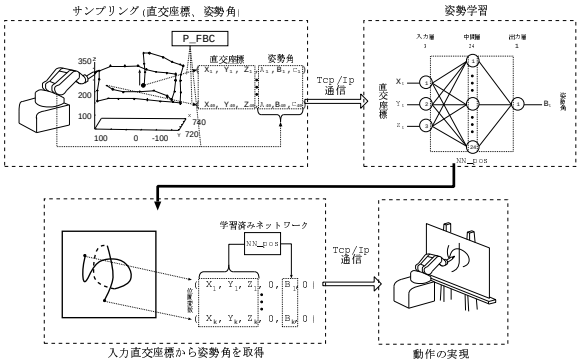


Fig. 3　姿勢学習の流れ

- ロボットの関節を柔らかい状態にして直接人間が手で動かし、直交座標と姿勢角のデータを取得する。

↓

- 得られた直交座標を入力データ、姿勢角を教師データとして Linux 上でニューラルネットワークの学習を行う。

↓

- 学習済のネットワークを用いて、任意の直交座標に対する姿勢角を想起し、位置変数を生

成する。

↓

- Linux 上の C プログラムでロボット動作用BASICプログラムと位置変数を編集し、TCP/IP 通信を用いてコントローラボックスへ送信後、ロボットアームに所望の動作を実行させる。

以上より、複雑な計算を必要とせずに姿勢角を定めることができるだけでなく、筆跡学習において重要な教示者の姿勢 (くせ) を学習によって獲得することができる。また、ロボットのコントローラボックスでは動作の時系列データの実行のみを行い、時系列データの生成は Linux 上で行っている。また、時系列データは TCP/IP 通信で送られているため遠隔からもロボットアームを操作することができる。

3.2 ニューラルネットワークの構造

図 4に、3 次元直交座標 (X,Y,Z) の入力に対して姿勢角 B を出力する 3 層ネットワークの構造を示す。すなわち、ニューラルネットワークはデータ中のサンプリング点の"位置"に対する"姿勢"を学習している。

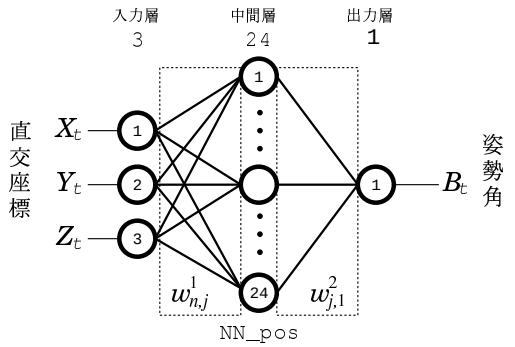


Fig. 4 ニューラルネットワークの構造

t は教師データのサンプル数 (パターン数) を表し、本研究では 40 とする。中間層のニューロン数は 24 とした。

このネットワークの出力は次式で表される。

$$z_1^3[p] = f\left(\sum_{j=1}^{N_2} w_{j,1}^2 f\left(\sum_{i=1}^{N_1} w_{i,j}^1 z_i^1[p] + \theta_j^1\right) + \theta_1^2\right), \quad p = 1, 2, \cdots, N_p \tag{1}$$

ここで、$\{w_{i,j}^1\}$、$\{w_{j,1}^2\}$、$\{\theta_j^1\}$、$\{\theta_1^2\}$ はそれぞれ結合重みとしきい値を表す。また $f(x)$ はシグモイド関数と呼ばれ、一般に次の関数が用いられる。

$$f(x) = \frac{1 - \exp(-\alpha x)}{1 + \exp(-\alpha x)} \tag{2}$$

ただし、$\alpha > 0$ は $f(x)$ の傾きを表す。

3.3 H_2学習

本節ではカルマンフィルタとその動作を説明する。その後に、カルマンフィルタを非線形問題に拡張し、ニューラルネットワークに適用することで H_2学習アルゴリズム (g-EKF) を導出する。

カルマンフィルタのアルゴリズムを以下に示す。

[**カルマンフィルタ**]

$$\begin{aligned}
\hat{\boldsymbol{x}}_{k|k} &= \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k[\boldsymbol{y}_k - \boldsymbol{H}_k\hat{\boldsymbol{\Sigma}}_{k|k-1}] \\
\hat{\boldsymbol{x}}_{k+1|k} &= \boldsymbol{F}_k\hat{\boldsymbol{x}}_{k|k} \quad \text{(フィルタ方程式)} \qquad (3) \\
\boldsymbol{K}_k &= \hat{\boldsymbol{\Sigma}}_{k|k-1}\boldsymbol{H}_k{}^T \cdot[\boldsymbol{H}_k\hat{\boldsymbol{\Sigma}}_{k|k-1}\boldsymbol{H}_k{}^T + \boldsymbol{\Sigma}_{v_k}]^{-1} \\
&\qquad \text{(カルマンゲイン)} \qquad (4) \\
\hat{\boldsymbol{\Sigma}}_{k|k} &= \hat{\boldsymbol{\Sigma}}_{k|k-1} - \boldsymbol{K}_k\boldsymbol{H}_k\hat{\boldsymbol{\Sigma}}_{k|k-1} \\
\hat{\boldsymbol{\Sigma}}_{k+1|k} &= \boldsymbol{F}_k\hat{\boldsymbol{\Sigma}}_{k|k}\boldsymbol{F}_k{}^T + \boldsymbol{G}_k\boldsymbol{\Sigma}_{w_k}\boldsymbol{G}_k{}^T \\
&\qquad \text{(誤差の共分散行列方程式)} \qquad (5)
\end{aligned}$$

ただし、

$$\hat{\boldsymbol{x}}_{0|-1} = \bar{\boldsymbol{x}}_0, \quad \hat{\boldsymbol{\Sigma}}_{0|-1} = \boldsymbol{\Sigma}_{x_0} \qquad \text{(初期値)} \qquad (6)$$

このアルゴリズムの動作は次の通りである。

[**カルマンフィルタの動作**]

観測値$\boldsymbol{y}_k$を観測したとき、

1) 一段前の共分散行列の予測値$\hat{\boldsymbol{\Sigma}}_{k|k-1}$を用いてカルマンゲイン$\boldsymbol{K}_k$を計算する。

2) 現在の状態$\hat{\boldsymbol{x}}_{k|k}$とその共分散行列$\hat{\boldsymbol{\Sigma}}_{k|k}$を計算する (推定)。

3) $\hat{\boldsymbol{x}}_{k|k}$、$\hat{\boldsymbol{\Sigma}}_{k|k}$より、それぞれの一段先の予測値$\hat{\boldsymbol{x}}_{k+1|k}$、$\hat{\boldsymbol{\Sigma}}_{k+1|k}$を計算する (予測)。

ニューラルネットワークの学習は多入力多出力の非線形システムの推定問題となる。よって、学習アルゴリズムとしてカルマンフィルタを適用することができる。すなわち、線形近似したニューラルネットワークにカルマンフィルタを適用すれば次の学習アルゴリズム (g-EKF) が得られる [1]。

$$\hat{\boldsymbol{w}}_{k+1|k+1} = \hat{\boldsymbol{w}}_{k|k} + \boldsymbol{K}_{k+1} \cdot (\boldsymbol{y}_{k+1} - \boldsymbol{h}_{k+1}(\hat{\boldsymbol{w}}_{k|k})) \quad (7)$$

$$\boldsymbol{K}_k = \hat{\boldsymbol{P}}_{k|k-1}\boldsymbol{H}_k^T \cdot (\boldsymbol{H}_k \hat{\boldsymbol{P}}_{k|k-1}\boldsymbol{H}_k^T + \boldsymbol{I})^{-1} \quad (8)$$

$$\hat{\boldsymbol{P}}_{k+1|k} = \hat{\boldsymbol{P}}_{k|k-1} - \boldsymbol{K}_k\boldsymbol{H}_k\hat{\boldsymbol{P}}_{k|k-1} \quad (9)$$

ただし、

$$\boldsymbol{H}_{k+1} = \left.\frac{\partial h_{k+1}(\boldsymbol{w})}{\partial \boldsymbol{w}^T}\right|_{w=\hat{w}_k} = \left[\frac{\partial h_{k+1}(\boldsymbol{w})}{\partial \theta_1^1}, \frac{\partial h_{k+1}(\boldsymbol{w})}{\partial w_{1,1}^1}, \cdots, \frac{\partial h_{k+1}(\boldsymbol{w})}{\partial w_{N_1,1}^1}\right]_{w=\hat{w}_k} \quad (10)$$

ここで、$\boldsymbol{H}_{k+1}$は$\hat{\boldsymbol{w}}_{k|k}$の関数であるため通常のカルマンフィルタのように$\hat{\boldsymbol{P}}_{k+1|k}$、$\boldsymbol{K}_k$をオフラインで計算することはできない。また、システムの部分的な線形化を行っているので線形近似誤差が観測雑音$\boldsymbol{v}_k$に加わることに注意されたい。

4. マウスによるロボットアームの操作

ロボットアームをマウスによって自由に操作するための方法を説明する。本研究ではLinux上で

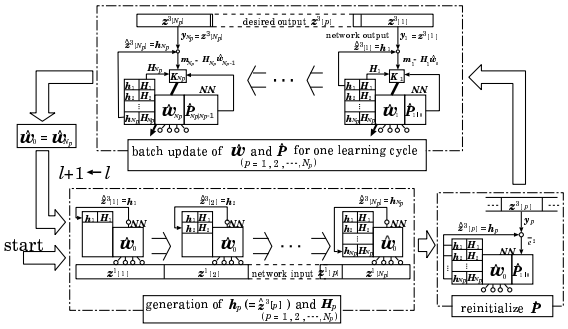


Fig. 5 g-EKF 学習アルゴリズムの動作

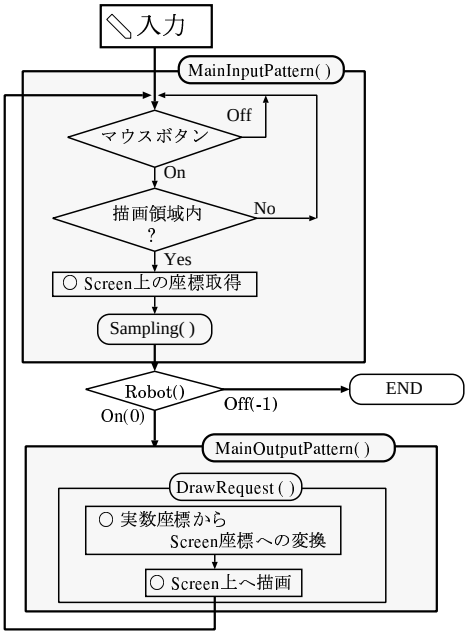


Fig. 6 マウスでロボットアームを操作するための処理の流れ

Xlibを用いて開発したマウスで任意の図形を描くツールの中にロボットアームの制御システムを組み込んだ。

図6に全体の処理の流れを示す。この図からわかるように、マウスによる描画の処理を行う部分とロボットの制御を行う部分がある。

4.1 C言語による実装

マウスでウィンドウ上に図形を描きその軌跡をサンプリングすることによってX、Y、Z座標の時系列データを得る。ただし、軌跡のサンプリング

数は25点とし、ウィンドウ上におけるZ座標はペンの上げ下げの2つの場合しかないため、0か1の値しかとらない。

次に、このウィンドウ上の直交座標値を実際にロボットを制御するための関数”Robot()”に渡す。

図7に示す関数”Robot()”の詳細な処理を以下に示す。

1) ソケットを作成し、コントローラボックスと通信を試みる。

2) **関数”GenerateTrajectory()”**
サンプリングされたウィンドウ上の直交座標データをロボットの描画領域に合わせてスケーリングし、学習済みのネットワークを用いて姿勢角を想起していき、位置変数の時系列データを生成する。

3) **関数”ControlRobot()”**
位置変数に従ってロボットアームを動作させるためのBASICプログラムを編集する。そしてTCP/IP通信を用いてコントローラボックスへプログラムと位置変数を一行ずつ送信していき、編集が終了したら後に実行する。

4) ロボットの動作が終了したら通信を終了し、処理をマウスによる入力待ち状態へ戻す。

4.2 学習実験

[教師データのサンプリング]

教師データは図8のようにロボットアームの各関節をできる限り柔らかくした状態で手動で得た。その際、ツール先端の筆が垂直な紙面上を図形を描くように動かし、0.5秒間隔で計40点の直交座標と姿勢角のデータを取得していった。このような操作で得られた時系列の教師データを図9とその一部を表1に示す。

図9と表1に示すような教師データと、前節で述べたH_2学習アルゴリズム(g-EKF)を用いてネットワークの学習を行った。また、学習の終了判定は40点各パターンの2乗誤差の総和が10^{-6}以下とした。学習結果を以下に示す。

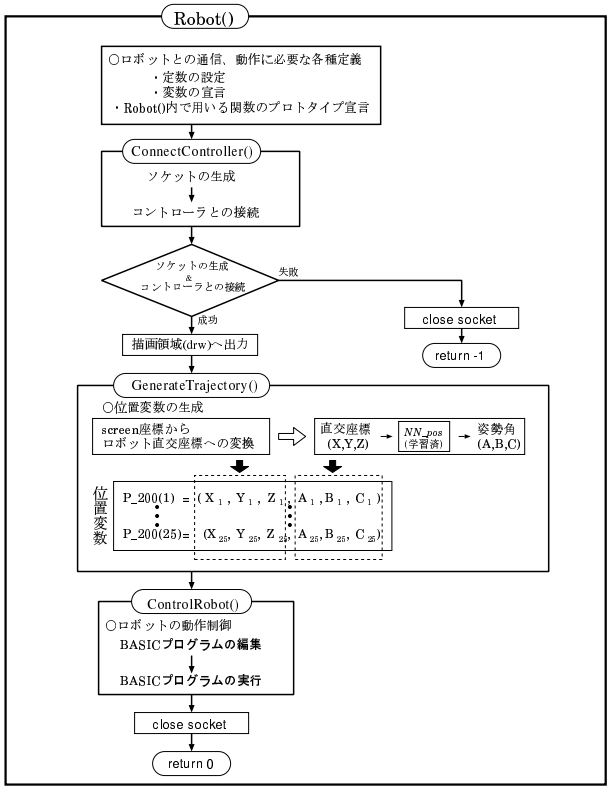


Fig. 7　関数”Robot()”フローチャート

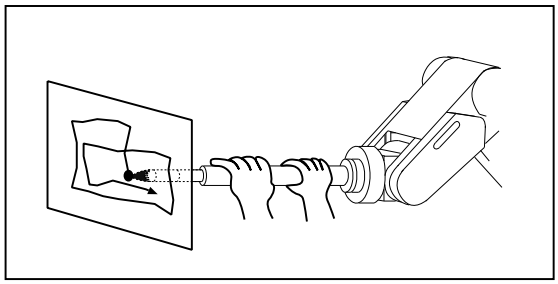


Fig. 8　教示の様子

- 学習回数：85
- 出力誤差：9.98×10^{-7}

この学習が終了したニューラルネットワークをテストするため、図10(a)のような正確な円の直交座標データをネットワークに入力して、ロボットアームに描画を行わせた。その結果、自然な動作で図10(b)のような円をロボットアームで描くことができた。

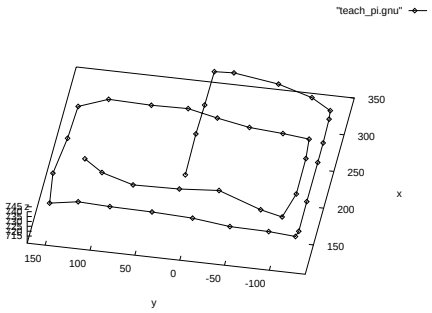
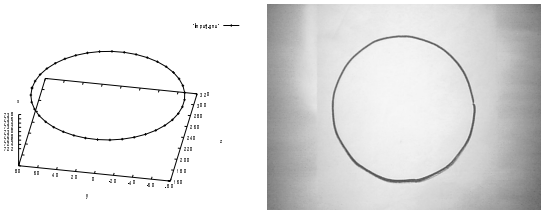


Fig. 9　教師データ

Table 1　教師データの入力と出力 (40 点)

	入力			出力
t	X_t	Y_t	Z_t	B_t
1	737.330	10.910	185.400	96.760
2	740.060	10.940	237.540	89.220
3	737.330	10.920	280.560	83.020
⋮	⋮	⋮	⋮	⋮
39	728.340	104.390	187.750	96.200
40	729.510	126.630	201.790	94.560



(a) テスト入力データ　(b) ロボットの描画図形

Fig. 10　ロボットアームによる描画結果

4.3　マウスからの操作実験

マウスによってロボットアームを操作するシステムを用いて、マウスによる 3 次元線図形の描画実験を行った。図 11に示すようにマウスでひらがなの”ゆ”をウィンドウ上に書いた結果を示す。これに対応して、ロボットは滑らかな動きで図 12に示すような”ゆ”をキャンバス上に書くことができた。

これにより、誰でも簡単にマウスでロボットアームを自由に操作することが可能となった。

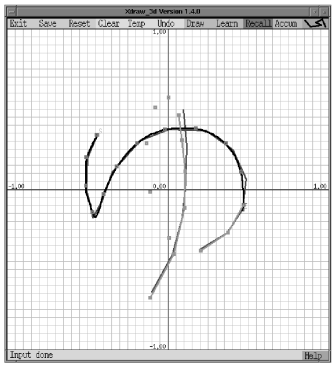


Fig. 11　ウィンドウ上にマウスで描いた”ゆ”

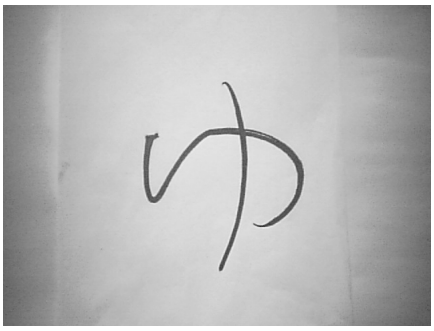


Fig. 12　ロボットが実際に描いた”ゆ”

5.　結論

本研究で開発したシステムでは、姿勢学習によって本来複数の関節軸によって決定される無数の姿勢の組合せの中から複雑な計算を行うことなく 3 次元空間上の位置に対応した姿勢を得ることを可能にし、ロボットアームの動作制御を実現することができた。また、Linux 上で Xlib を用いて開発したウィンドウ上にマウスで図形を描くツールと組み合わせることによって、誰でも簡単にマウスでロボットアームを自由に操作することが可能となった。

今後は、ペンタブレットなどを用いてロボットアームを手動で教示するよりもさらに微妙な人間の姿勢などのくせをロボットに学習させることを考えている。

参考文献

1) 西山 清: 最適フィルタリング, 培風館, 2001.

2) 猪岡 光, 石原 正, 池浦 良淳: 大学院情報理工学 5 知能制御, 講談社, 2000.