論　文

FPGAによる確率ニューラルネットワークのハードウェア実現

卜　楠[†a)]　濱本　泰治[†]　福田　修[††]　辻　敏夫[†]

FPGA Implementation of a Probabilistic Neural Network

Nan BU[†a)], Taiji HAMAMOTO[†], Osamu FUKUDA[††], and Toshio TSUJI[†]

あらまし　統計的構造を組み込んだ確率ニューラルネットワークは，生体信号などの複雑な信号のパターン識別問題に有効である．しかしながら，確率ニューラルネットワークの実装には通常，処理計算用のコンピュータが不可欠で，小型の応用システムを実現することは困難であった．そこで本論文では，確率ニューラルネットワークを組み込んだ実用的なシステム機器を実現することを目的として，ハードウェア化によるシステムの1チップ化を試みた．本システムはFPGAを利用しているため，確率ニューラルネットワークとそれ以外の回路が再構成可能であり，問題ごとに適切な回路構成を実現することができる．本論文では，まず，確率ニューラルネットワークのハードウェア実現法を示し，その精度確認実験を行った．次に，実システムの一例として，筋電位信号を用いたポインティングデバイスのハードウェア化を行い，動作検証を行った．検証実験の結果，ソフトウェアでシステムを構成した場合と同等の識別精度で計算することができ，本システムの有効性が明らかになった．

キーワード　確率ニューラルネットワーク，FPGA，パターン識別，EMG信号

1. まえがき

近年，パターン識別問題などに関して確率ニューラルネットワーク（PNN）を用いた研究が数多く行われている[1]〜[3]．一般にPNNは非線形であるため，ネットワークの構造や重みを適切な値に調節することにより，学習パターンから真の確率分布を精度良く近似することができる．そのPNNの一つとして，Gaussian Mixture Model（GMM）を対数線形化したLog-Linearized Gaussian Mixture Network（LLGMN）が辻らにより提案されている[4]．このLLGMNは入力パターンに対する事後確率が計算可能であり，更に確率変数による重みの制限がないことから，ニューラルネットワーク（NN）としての表現能力が高いという特徴をもつ．そして辻らはLLGMNを利用し，筋表面の微弱な電気信号である筋電位（EMG）信号を用いて手先動作のパターン識別を行い，高い識別結果を得ている．更にEMG信号の識別結果を利用して電動義手の制御，ポインティングデバイス（EMGマウス）の開発などを行ってきた[5]〜[7]．しかしながら，LLGMNの実装には処理計算用のコンピュータが不可欠なため小型の応用システムの実現が困難であった．もし上記システムのハードウェア化による1チップ化を行うことができれば，筋電義手などの携帯型デバイスやウェアラブルシステムへの組込みを行うことができ，様々な環境下で使用することができる．またEMGマウスでは，問題点となるCPUへの負荷を取り除いたデバイスの開発が期待できる．

一方，計算機ハードウェアの発展に伴い，NNなどのソフトウェアアルゴリズムのハードウェア化を行い，実用的なシステムへ応用する研究が数多く報告されている[2], [8]．そのハードウェア化には，Field Programmable Gate Array（FPGA）を用いたものやApplication Specific Integrated Circuit（ASIC）による方法が一般的である．FPGAは回路を再構成可能であるため，チップを変更することなく問題ごとに適切な回路を構成でき，設計期間が短いといった利点もある．ASICでの実現ではFPGAに比べ高速な回路が可能であるが，回路構成を一度決定してしまうと

† 広島大学大学院工学研究科，東広島市
Department of Artificial Complex Systems Engineering, Hiroshima University, 1–4–1 Kagamiyama, Higashi-hiroshima-shi, 739–8527 Japan
†† 独立行政法人産業技術総合研究所，つくば市
National Institute of Advanced Industrial Science and Technology, 1–2–1 Namiki, Tsukuba-shi, 305–8564 Japan
a) E-mail: bu@ieee.org

変更することができない．

そこで本論文では，FPGA を用いた LLGMN の 1 チップ化を行い，実用的な組込み型ハードウェアの試作を行う．そして，開発したハードウェアを用いて，EMG マウス [5] を実現することを試みる．

以下，**2.** では LLGMN の構成について説明する．**3.** では，NN のハードウェア化手法について説明を行い，設計したハードウェアをソフトウェアと比較する．**4.** では，EMG マウスのシステム構成，及び FPGA 実装後の操作実験について示す．

2. LLGMN

LLGMN [4] は GMM を対数線形化した PNN で，ネットワークの出力である事後確率をもとに情報論的学習が可能である．以下，LLGMN の構造と学習則について説明する．

2.1 構　　造

LLGMN [4] のネットワーク構造を図 1 に示す．まず，入力ベクトル $\boldsymbol{x} \in \Re^d$ を次式のように非線形変換する．

$$\boldsymbol{X} = \left[1, \boldsymbol{x}^{\mathrm{T}}, x_1^2, x_1x_2, \cdots, x_1x_d, x_2^2, x_2x_3, \cdots, x_2x_d, \cdots, x_d^2\right]^{\mathrm{T}} \tag{1}$$

この変換は，混合正規分布を構成する正規分布（以後コンポーネントと呼ぶ）の確率計算を，新たな入力ベクトル $\boldsymbol{X}$ と NN の重み係数による線形計算で表現するために行う．

ネットワークの第 1 層は，$\boldsymbol{X}$ の次元数 $H = 1 + d(d+3)/2$ に合わせて H 個のユニットから構成される．各ユニットは恒等関数を入出力関数として用いている．第 2 層は，第 1 層の出力に重み $w_h^{(c,m)}$ $(c = 1, 2, \cdots, C;\ m = 1, 2, \cdots, M_c)$ が掛け合わされて伝えられる．ただし，C は対象とするクラス数，M_c はクラス c に属するコンポーネント数を表し，$w_h^{(C,M_C)} = 0$ $(h = 1, 2, \cdots, H)$ である．今，第 1 層の出力を ${}^{(1)}O_h$，第 2 層のユニット $\{c, m\}$ への入力を ${}^{(2)}I_{c,m}$，出力を ${}^{(2)}O_{c,m}$ とすると，

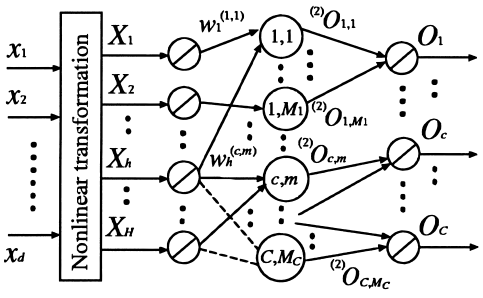

図 1　LLGMN の構造
Fig. 1　The structure of LLGMN.

$$ {}^{(2)}I_{c,m} = \sum_{h=1}^{H} {}^{(1)}O_h w_h^{(c,m)} \tag{2}$$

$$ {}^{(2)}O_{c,m} = \frac{\exp[{}^{(2)}I_{c,m}]}{\sum_{c'=1}^{C}\sum_{m'=1}^{M_{c'}} \exp[{}^{(2)}I_{c',m'}]} \tag{3}$$

となる．

そして，第 3 層は ${}^{(2)}O_{c,m}$ の各クラスでのコンポーネントの総和が入力となり，各クラスに対する事後確率を出力する．3 層の入出力をそれぞれ ${}^{(3)}I_c$，${}^{(3)}O_c$ とすると

$$ {}^{(3)}O_c = {}^{(3)}I_c = \sum_{m=1}^{M_c} {}^{(2)}O_{c,m} \tag{4}$$

となる．

2.2 学　習　則

N 個のサンプルデータ $\boldsymbol{x}^{(n)}$ $(n = 1, \cdots, N)$ を用いて学習を行う場合を考える．ここで，$\boldsymbol{T}^{(n)} = [T_1^{(n)}, \cdots, T_c^{(n)}, \cdots, T_C^{(n)}]$ は教師信号であり，観測した事象がクラス c であるときは $T_c^{(n)} = 1$ とし，それ以外は $T_c^{(n)} = 0$ とする離散値である．

学習における評価関数 J は，

$$J = -\sum_{n=1}^{N}\sum_{c=1}^{C} T_c^{(n)} \log {}^{(3)}O_c^{(n)} \tag{5}$$

と定義し，これを最小化するように学習を行うことにより対数ゆう度を最大にする．また，教師信号が $[0, 1]$ の連続値である場合は Kullback の情報量を用いて評価関数を

$$J' = -\sum_{n=1}^{N}\sum_{c=1}^{C} T_c^{(n)} (\log {}^{(3)}O_c^{(n)} - \log T_c^{(n)}) \tag{6}$$

のように定義し，これを最小化するように重みを修正して学習する [4].

3. LLGMN のハードウェア設計

LLGMN は $[0, 1]$ の連続値で確率を表現するため，

実数値を用いる必要がある．また，第 2 層のユニットの出力関数（式 (3)）に指数関数が，そして学習時の評価関数には対数関数が含まれることや，ニューロン数の増加に伴う乗算回数やメモリ容量の増加を考慮する必要がある．そこで，本論文では LLGMN のハードウェア設計において以下の対策を行った．

3.1 固定小数点表示の採用

計算速度などを考慮して，実数値は固定小数点表示を用いて表現する．固定小数点表示は，設定するビット幅により回路全体の規模や計算精度に大きな影響を与える．整数部 a，小数部 b の固定小数点表示を考え，ビット幅の同じ 2 変数を掛け合わせた場合の計算誤差を考えると，小数部 $b+1$ [bit] 目の情報から誤差を生じる．そのため乗算時の最大計算誤差は $2^{-(b-a)}$ となる．

3.2 ルックアップテーブルの利用

指数関数，対数関数などの演算は，計算速度を考慮してルックアップテーブル（LUT）を用いて行った．ここでは指数関数を例に説明を行い，LUT を用いた指数関数を図 2 に示す．まず，出力値の変化が小さい部分（図 2 中 (a)）では刻み幅を大きくし，逆に出力値の変化が大きい部分（図 2 中 (c)）に関しては入力の刻み幅を小さくすることで，LUT の面積を減らし回路面積も小さくなるように設計した．そして LUT のアドレス計算に関しては，乗算をシフトレジスタを用いて計算するため 2^n ($n \in \mathbf{Z}$) となるように設計した．今回の設計では $[-10,-5)$，$[-5,5]$，$(5,10]$ の 3 領域に対して刻み幅をそれぞれ 0.5，0.25，0.125 と設定した．

3.3 パイプライン処理による高速化

LLGMN による識別処理に必要な乗算回数は非線形変換による乗算と，式 (2) での乗算との総和になる．非線形変換では $(d(d+1)/2)$ 回，第 1，2 層間では $(1+d(d+3)/2)\sum_{c=1}^{C} M_c$ 回の乗算が行われる．これらの処理を高速化するために単純に並列化してしまうと乗算の数に比例した乗算器が必要となり，回路面積が大きくなってしまう．そこで，各ニューロンにおいての計算を可能な限りパイプライン処理化することで高速化を行う．本論文では各層で重みと入力を掛け合わせ，その後ルックアップテーブルを参照，総和を計算するところまでをパイプライン化した．

その他の高速化として，いくつかの乗算処理をビットシフト処理で再現することや，外部メモリに保存したサンプルパターンを読み出す処理は他の一つのデータを計算処理している間に行うなどといった方法をとっている．

3.4 ハードウェア構成

図 3 に実装したハードウェア構成を示す．図 3 (a) が LLGMN 全体のハードウェア構成を示し，同図 (b) に Neural Network Unit を示す．ハードウェアの構成は重み（w），学習用サンプルデータ（x）を RAM に格納する．そして，入力を非線形変換する Nonlinear Transformation Unit，入力に対する事後確率を計算する Neural Network Unit，学習による重みの修正量を計算する Learning Unit，重みを修正する Weight

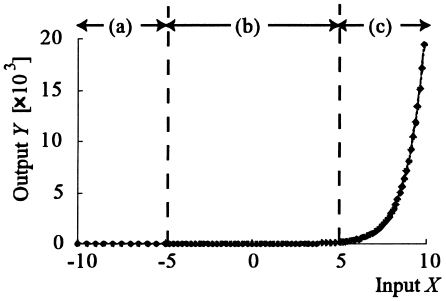

図 2 LUT を用いた指数関数
Fig. 2 The LUT based exponential function.

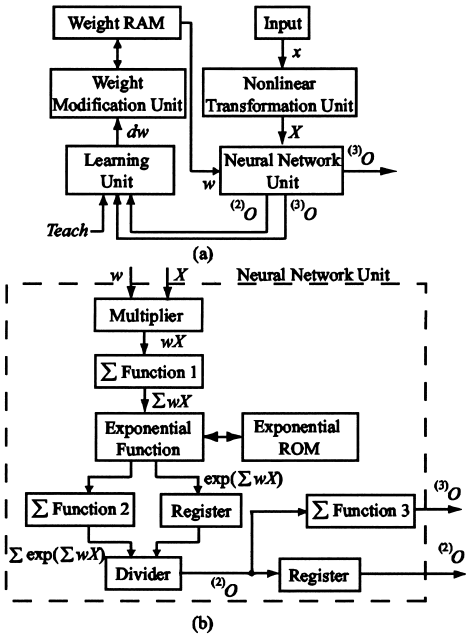

図 3 LLGMN のシステム構成
Fig. 3 Structure of LLGMN system.

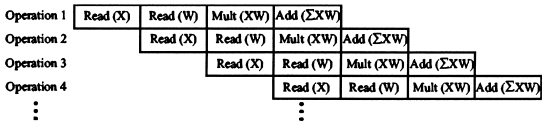

図 4　パイプライン構造
Fig. 4　Structure of pipeline.

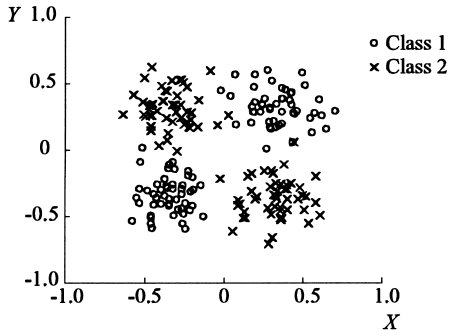

図 5　識別データ散布図（100 データ/クラス）
Fig. 5　Scatter graph of discrimination data (100 data/class).

Modification Unit で構成している．

Neural Network Unit では各 $\sum$ Function までをパイプライン化する．例えば，図 4 で示すように Nonlinear Transformation Unit より値（X）の読込み，重み（w）の読込み，乗算（Xw），$\sum$ Function (2): $\sum Xw$) までの計算をパイプライン化する．これらの計算処理をクラス c，コンポーネント m のそれぞれで計算し高速化を行う．

そして，ここで得られた ${}^{(2)}O$，${}^{(3)}O$ を用いて学習を行い，重みを更新し RAM へ保存する．

3.5　精度確認実験

設計した LLGMN についてハードウェア化による計算誤差を，評価ボード（セロックシカ社製 RC1000）を用いて評価した．評価ボードには FPGA チップ（Xilinx 社製 XCV2000E-6BG560：最大搭載ゲート数約 200 万ゲート）が搭載されている．ハードウェア設計用言語は Handel-C を用い，回路シミュレータ DK2 による動作確認と FPGA 実装を行った．

ハードウェア化した LLGMN の精度を確認するため，図 5 で示すデータの識別実験を行った．識別するクラス数は 2 クラス（$C = 2$）とし，それぞれを構成するデータの分布は二次元上の二つの正規分布（$M_c = 2$）とした．正規分布の属するクラス c，コンポーネント m の中心を $\mu_{c,m}$ とした場合，$\mu_{1,1} = (0.35, 0.35)$，$\mu_{1,2} = (-0.35, -0.35)$，$\mu_{2,1} = (0.35, -0.35)$，$\mu_{2,2} = (-0.35, 0.35)$ であり，すべての分布は同じ分散の値 $\sigma^2 = 0.0225$ とした．図 5 は同じ分布で各クラス 100 個のデータをプロットしたもので，図に示すようにデータには重なりが存在する．データ生成に用いた分布情報に基づくベイズ推定による理論的な識別精度は 98.00%である．ここではデータを十分に表現するため固定小数点のビット幅は，整数部 $a = 8$，小数部 $b = 16$ で構成した．ただし指数関数計算部分では値の範囲が大きいため $a = 16$，$b = 16$ とし計算を行っている．

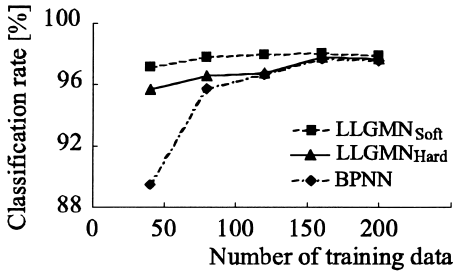

図 6　学習データ数による識別率の変化
Fig. 6　Effect of the number of training data on classification ability.

結果を図 6 に示す．識別には 4000 個のデータを用いた．図は学習データ数に対する識別率の変化で，比較のため，ハードウェア化した LLGMN，ソフトウェア（C 言語）で実現した LLGMN，ソフトウェアで実現した誤差逆伝搬型 NN [9]（BPNN）の結果を示す．ここで BPNN は十分な学習能力と識別能力を実現するニューロン数を選択する必要があるため，従来研究 [4] をもとに構造を決定する．BPNN は 4 層の階層型 NN とし，各階層のニューロン数をそれぞれ，入力層 2，隠れ層第 1 層 10，隠れ層第 2 層 10，出力層 2 とした．図から，ハードウェア化した LLGMN はソフトウェアで実現した LLGMN とほぼ同等の識別能力を有していることが分かる．また，特に学習データが少ない場合には，BPNN の識別率は大きく低下している．

表 1 に学習データ数が 200 の場合の識別率を示す．いずれも，5 パターンの平均値，標準偏差，最大値，最小値である．表 1 より，ハードウェア化した LLGMN の能力が十分良いことが分かる．

次に，ソフトウェアと FPGA の計算誤差を検証するため，初期重みや学習率，学習回数を同じにし，ソフトウェアと FPGA それぞれを用いて学習を行った．

ここで，学習用データは各クラスに 100 個のデータを用意し，100×2 とした．識別には各クラスに 4000 個のデータを用いた．

精度評価は次式を用いて行った．

$$err(n) = \frac{1}{C}\sum_{c=1}^{C}\sqrt{(P_{\mathrm{S}}(c|\boldsymbol{x}(n)) - P_{\mathrm{H}}(c|\boldsymbol{x}(n)))^2} \tag{7}$$

ただし，ソフトウェアと FPGA の事後確率をそれぞれ $P_{\mathrm{S}}(c|\boldsymbol{x}(n))$，$P_{\mathrm{H}}(c|\boldsymbol{x}(n))$ とする．識別率は C 言語が 97.8%であり，FPGA が 97.9%であった．

評価結果を表 2 に示す．誤差の原因は FPGA の指数関数の精度や固定小数点のビット幅による制限のためと考えられるが，ほぼソフトウェアと同等の結果が得られており，精度良く学習，識別が実現できたことが分かる．

なお，今回作成した回路において 1 データを学習するのに要するクロック数は 517 [clock] であり，一括学習法 [9] を用いて 1 セット（200 データ）を学習するのに必要なクロック数は 103488 [clock] であった．また，1 データを識別するのに要するクロック数は 247 [clock] であった．FPGA の動作周波数は 20 MHz まで動作確認した．今回は FPGA と Ultra SparcII 360 MHz 上でのソフトウェア（C 言語）で速度比較を行った．学習データ数 200，学習回数 1000 回の学習時間は FPGA，ソフトウェア，それぞれ，11504 ms，4669 ms であった．FPGA の計算速度はソフトウェアに比べ 41%程度となっている．また，データ数 4000 の識別に要した計算時間は FPGA，ソフトウェア，それぞれ 729 ms，332 ms であった．こちらも FPGA の計算速度はソフトウェアに比べ 45%程度となっている．また，ここで使用ゲート数は 212,197 であり，スライス数は 10,335 となり全体の 53%を使用している．

表 1　識別率の比較結果
Table 1　Comparision of the classification rates.

	Mean	S. D.	Maximum	Minimum
$\mathrm{LLGMN_{Soft}}$	97.89	1.28×10^{-1}	98.10	97.78
$\mathrm{LLGMN_{Hard}}$	97.66	1.33×10^{-1}	97.83	97.48
BPNN	97.53	4.58×10^{-1}	97.88	96.98

表 2　誤差の評価結果
Table 2　Evaluation results for the error.

Mean	S. D.	Maximum	Minimum
8.31×10^{-3}	1.18×10^{-2}	8.26×10^{-2}	0.70×10^{-5}

4. EMG マウスのハードウェア実現

本論文で LLGMN を用いたシステムの一例として，EMG マウス [5] をハードウェア化した．EMG マウスは使用者の手首の動き方向を EMG 信号から推定し，それを用いてマウスポインタの移動方向を決定，操作するシステムである．

4.1 システム構成

図 7 にシステムの構成図を示す．ハードウェア化したシステムは，(1) EMG 信号処理部，(2) NN 部，(3) ポインタ制御部からなる．以下に詳細を説明する．

(1) EMG 信号処理部：まず，特徴量として複数の筋における筋収縮のパターンを EMG 信号から計算する．計測した d チャネル分の EMG 信号を A–D 変換し，チャネルごとに全波整流した後，二次のディジタルバタワースフィルタを用いて平滑化する．この信号を $EMG_l(t)$ $(l = 1, \cdots, d)$ とする．

しかし，チャネルによっては EMG 信号の値が小さく特徴が現れにくいことがある．そこで，それぞれの EMG 信号の最大値を規格化する．更に $EMG_l(t)$

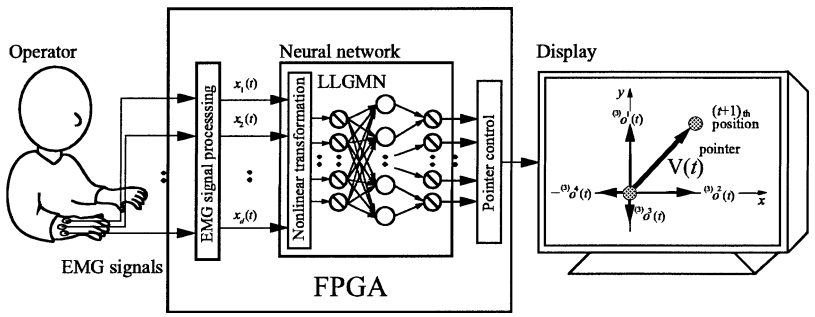

図 7　システム構成
Fig. 7　Structure of the proposed system.

$(l = 1, \cdots, d)$ を各チャネルの総和が 1 となるように正規化し，NN の入力ベクトルとして，特徴パターン $\boldsymbol{x}(t) = [x_1(t), x_2(t), \cdots, x_d(t)]^{\mathrm{T}}$ を算出する．また，筋力情報 $\alpha(t)$ を

$$\alpha(t) = \sum_{l=1}^{d} EMG_l(t) \tag{8}$$

として定義し，ポインタ制御部へ入力する．

(2) NN 部：NN 部では LLGMN を用いて，EMG パターン $\boldsymbol{x}(t)$ からポインタの各基準方向（$C = 4$：上下左右）への事後確率を推定する．

LLGMN は，K 個の学習方向 $\phi = 2\pi\frac{i}{K}$ $(i = 0, \cdots, K-1)$ のそれぞれに用意された Z 個の信号（$N = Z \times K$）を学習パターンとして学習を行う．ただし，ϕ は図 7 の y 軸正方向を $\phi = 0$ とし，時計回りに増加するよう定義する．

(3) ポインタ制御部：ポインタ制御部では LLGMN の出力，及び推定した筋力情報をもとにポインタの動きを計算する．LLGMN の出力は，図 7 に示すように各基準方向 c $(c = 1, 2, 3, 4)$ へポインタが移動する確率を示している．そこで時刻 t の EMG パターンによるポインタの移動方向ベクトル $\boldsymbol{v}(t) = (v_X(t), v_Y(t))^{\mathrm{T}}$ を，以下のように定義する．

$$v_X(t) = \sum_{c=1}^{C} {}^{(3)}O^c(t)\sin(2\pi(c-1)/C) \tag{9}$$

$$v_Y(t) = \sum_{c=1}^{C} {}^{(3)}O^c(t)\cos(2\pi(c-1)/C) \tag{10}$$

ここで，$v_X(t)$，$v_Y(t)$ は各基準方向へ対応するベクトルの x 成分，y 成分を示す．

更に使用者の力に合ったポインタの操作感覚を実現するために，筋力情報 $\alpha(t)$ と移動方向ベクトル $\boldsymbol{v}(t)$ に基づき，以下のインピーダンスモデルを用いてポインタがどのように移動するかを計算する．

$$M_e\ddot{\boldsymbol{p}}(t) + B_e\dot{\boldsymbol{p}}(t) = \boldsymbol{F}_{\boldsymbol{level}}(t) \tag{11}$$

ここで，M_e，B_e はそれぞれ慣性，粘性パラメータで，$\boldsymbol{p}(t)$ はポインタの位置ベクトルである．また $\boldsymbol{F}_{\boldsymbol{level}}(t)$ はポインタに加える仮想的な力のレベルで，$\alpha(t)$，$\boldsymbol{v}(t)$ から計算する．

$$\boldsymbol{F}_{\boldsymbol{level}}(t) = \begin{cases} g\alpha(t)\boldsymbol{v}(t) & (\alpha(t) \geq \alpha_0) \\ 0 & (\alpha(t) < \alpha_0) \end{cases} \tag{12}$$

ただし，g は力ゲインであり，α_0 はポインタ操作の有無を判定するしきい値である．式 (11) は粘性摩擦 B_e を有する平面に置かれた質量 M_e の物体に力 $\boldsymbol{F}_{\boldsymbol{level}}(t)$ が加わった際の運動方程式を表している．このような物理モデルに従い，数値積分によってポインタの位置，速度を計算することで，操作者の力感覚に対応した自然な操作感が期待できる．

4.2 ハードウェア実装

EMG 信号を用いたポインタ制御システムのハードウェア化及び動作実験を行った．実装環境，設計言語などは **3.5** と同様である．EMG 信号の計測にはマルチテレメータ（Web5000：(株) 日本光電製）を用いた．二次のディジタルバタワースフィルタのカットオフ周波数は 1 [Hz] とし，平滑化後のサンプリング周波数は 250 [Hz] とした．動作実験の EMG を計測した部位は右前腕部に 4ch（橈側手根伸筋，橈側手根屈筋，尺側手根伸筋，尺側手根屈筋）とした．LLGMN のコンポーネント数とクラス数は，それぞれ $M_c = 1$ $(c = 1, \cdots, C)$，$C = 4$，学習方向は，$\phi = 0$ [rad] から $\pi/4$ [rad] 刻みの $K = 8$ 方向（各基準方向角度とその中間角度）と設定し，学習用データは 16×8 動作とした．なお，斜め方向に対する学習は，例えば $\phi = \pi/4$ [rad] の場合，$\phi = 0, \pi/2$ [rad] の 2 クラスの事後確率が 0.5 となるように行った．

被験者（男子大学生，23 才）には，ポインタの移動方向を右手首の曲げ方向に対応させて NN を学習した後，三角形，四角形の描画を行わせた．動作実験中のポインタ軌跡の一例を図 8 に，同図 (a) に対応する FPGA の入出力結果を図 9 に示す．図 8 ではディスプレイ上で三角形と四角形を図に示す番号の順に描いている．図 8 よりほぼ目的とした図形が描けていることが分かる．一方，図 9 は上から，動作を行った方向，EMG 信号，LLGMN の事後確率（上，右，下，左の 4 方向），ポインタの移動方向，しきい値を超えたときの力を示している．横軸は時間で，灰色で覆っている部分はしきい値により無動作と判定した．図より斜め方向へポインタを移動させる際には各方向への確率が約 0.5 となっていることが確認できる．この結果より，基準の 4 方向の確率を合成することで 8 方向のポインタ移動を実現できており，EMG 信号を用いたポインタ制御システムが FPGA 上で動作していることを確認した．

今回作成した回路において 1 データを学習するのに要するクロック数は 1026 [clock] であり，ターミナル

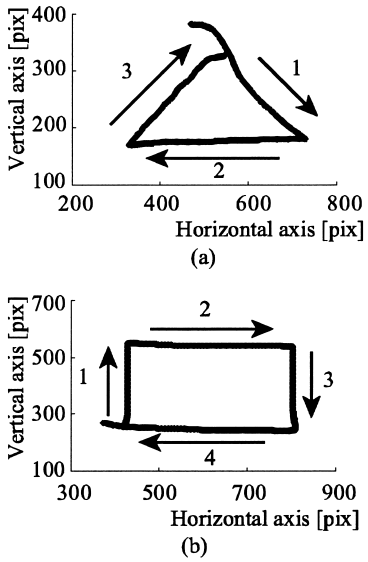

図 8　ポインタの軌跡
Fig. 8　Trajectories of the pointer.

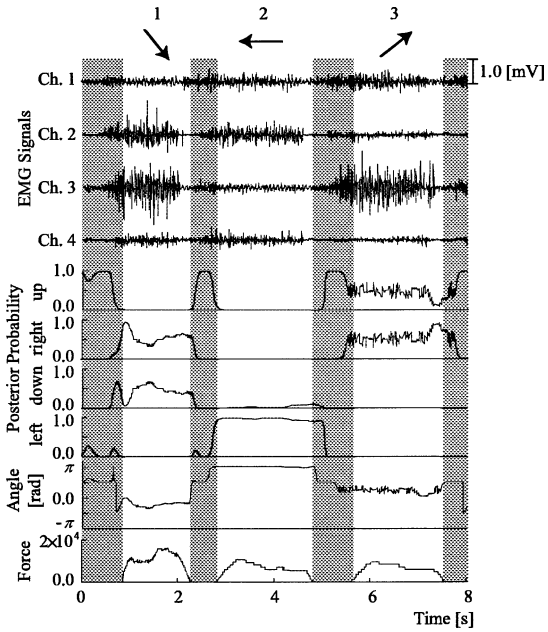

図 9　三角形描画のポインタ操作例
Fig. 9　An example of pointer control for drawing-a-triangle task.

ラーニング法 [10], [11] を用いて 1 セット（128 データ）を学習するのに必要なクロック数は 131397 [clock] であった．そして，1 データを識別するのに要するクロック数は 553 [clock] であった．FPGA の動作周波数は 20 MHz まで動作確認した．実際に学習データ数 128，学習回数 1000 回の学習に要した時間は FPGA，ソフトウェア（Ultra SparcII 360 MHz 上での C 言語），それぞれ 16771 ms，13614 ms であり，FPGA の計算速度はソフトウェアに比べ 81%程度である．また，データ数 20000 の計算に要した計算時間は FPGA，ソフトウェア，それぞれ 11781 ms，4837 ms であり，FPGA の計算速度はソフトウェアに比べ 41%程度である．ここで使用ゲート数は 414,637 であり，スライス数は 18,208 となり全体の 94%を使用している．

5. むすび

本論文では，LLGMN を FPGA に実装し，その動作と精度確認を行った．更に，EMG マウスシステムを FPGA に実装することにより，これまで PC 上で動作を行っていたソフトウェア制御部分を FPGA チップとして実現した．また，本システムがソフトウェア（C 言語）と同等の計算精度を実現していることを確認した．

本論文で取り上げたアプリケーションではサンプリング周波数は 250 Hz と設定している．そのためマウス操作時の FPGA が 1 データを処理する速度は 4 ms 以内であれば問題ない．そして，今回作成したハードウェアシステムでは 1 データを処理する時間は 0.589 ms であり，実装したポインタ制御システムを動作するにあたり十分な速度で動作していることを確認した．

しかし，今回は 1 チップ化を目標としたためニューラルネットだけではなく周辺の処理回路も実装する必要があり，ゲート数の制限により十分な並列計算を行うことができなかった．また，今回は設計言語として Handel-C を用いて実装したため，内部回路の十分な高速化を考慮できていない．今後は回路の高速化などを考え HDL を用いて回路を構成し，動作速度の向上，回路の小型化を行っていく予定である．

謝辞　本研究の一部は NEDO 産業技術研究事業費助成金（01A17001b）によるものでここに改めて謝意を表します．

また，本研究は東京大学大規模集積システム設計教育研究センターを通し，セロックシカ（株）の協力で行われたものでここに改めて謝意を表します．

文　献

[1] D.F. Specht, “Probabilistic neural networks,” Neural Netw., vol.3, no.1, pp.109–118, 1990.
[2] 相部範之，安永守利，“確率的ニューラルネットワーク計算

の並列高速化アーキテクチャとその画像認識システムへの適用,” 情処学論, vol.43, no.SIG 6 (HPS 5), pp.206–218, 2002.
[3] G.D. Zhang, “Neural network for classification: A survey,” IEEE Trans. Syst. Man Cybern. C, Appl. Rev., vol.30, no.4, pp.451–462, 2000.
[4] T. Tsuji, O. Fukuda, H. Ichinobe, and M. Kaneko, “A log-linearized Gaussian mixture network and its application to EEG pattern classification,” IEEE Trans. Syst. Man Cybern. C, Appl. Rev., vol.29, no.1, pp.60–72, 1999.
[5] 辻 敏夫, 福田 修, 村上 満, 金子 真, “ニューラルネットを利用した EMG 制御型ポインティングデバイス,” 計測自動制御学会論文集, vol.37, no.5, pp.425–431, 2001.
[6] O. Fukuda, T. Tsuji, and M. Kaneko, “An EMG controlled robotic manipulator using neural networks,” Proc. IEEE International Workshop on Robot and Human Communication, pp.442–447, 1997.
[7] 辻 敏夫, 重吉宏樹, 福田 修, 金子 真, “EMG 信号に基づく前腕動力義手のバイオミメティック制御,” 日本機械学会論文集, vol.66, no.648 (C), pp.2764–2771, 2000.
[8] 渡邊 実, 小林史典, “ニューラルネットワーク VLSI,” 計測制御, vol.40, no.12, pp.893–896, 2001.
[9] D.E. Rumelhart and J.L. McClelland, “Learning internal representations by error propagation,” in Parallel Distributed Processing, Explorations in the Microstructure of Cognition, vol.1, ed. D.E. Rumelhart, J.L. McClelland, and the PDP Research Group, pp.318–362, MIT Press, 1986.
[10] O. Fukuda, T. Tsuji, and M. Kaneko, “Pettern classification of EEG signals using a log-linearized Gaussian mixture neural network,” Proc. IEEE International Conference on Neural Networks, pp.2479–2484, Perth, 1995.
[11] T. Tsuji, O. Fukuda, M. Kaneko, and K. Ito, “Pattern classification of time-series EMG signals using neural networks,” Int. J. Adaptive Control and Signal Processing, vol.14, no.8, pp.829–848, Dec. 2000.

（平成 16 年 4 月 22 日受付，7 月 25 日再受付）

ト　楠

1998 中国大連理工大学･工･機械卒. 2001 同大大学院機械工学専攻修士課程了. 現在, 広島大大学院工学研究科博士課程後期在学中. ニューラルネット, パターン識別, 生体信号解析, ニューロチップなどの研究に従事. IEEE 学生会員.

濱本　泰治　（学生員）

2003 広島大・工・電気卒. 現在, 同大大学院工学研究科博士課程前期複雑システム工学専攻在学中. ニューラルネットワークのハードウェア化に関する研究に従事.

福田　　修　（正員）

2000 広島大大学院工学研究科博士課程後期了. 1997 年 4 月〜1999 年 3 月の期間, 日本学術振興会特別研究員（DC1）. 2000 通商産業省工業技術院機械技術研究所入所. 2001（独）産業技術総合研究所へ転任. 2004 広島大大学院工学研究科複雑システム工学専攻客員助教授, 現在に至る. 博士（工学）. ニューラルネット, 電動動力義手, 超音波画像計測などの研究に従事. 日本機械学会, 日本人間工学会, 日本ロボット学会等各会員.

辻　　敏夫　（正員）

1985 広島大大学院工学研究科博士課程前期了. 同年広島大学工学部助手. 1994 同助教授を経て, 2002 より同大学大学院工学研究科教授, 現在に至る. 工博. 主として, 人間とロボットの運動制御, 生体信号解析, ニューラルネット, ヒューマン・マシンシステムなどに関する研究に従事. 計測自動制御学会学術奨励賞（1986）･論文賞（2002）, バイオメカニズム学会論文賞（1990）, 日本義肢装具学会論文賞（2000）, 日本医科器械学会論文賞（2003）, IEEE 2003 King-Sun Fu Memorial Best Transactions Paper Award（2004）, 日本機械学会ロボティクス・メカトロニクス部門学術業績賞（2004）などを受賞.