ユーザの操作履歴に基づくマルチウインドウレイアウトシステム

A Multiwindow Layout System Based on User Operation History

○福田 洋平（立命館大学大学院理工学研究科）　星野 孝総（立命館大学理工学部）　亀井 且有（立命館大学理工学部）

Yohei FUKUDA, Graduate School of Computer Science and Engineering, Ritsumeikan University
Yukinobu HOSHINO, Computer Science, Ritsumeikan University
Katsuari KAMEI, Computer Science, Ritsumeikan University

About multiwindow on Computer. Moving a window carries the risk of overlapping or completely hiding other windows. It is difficult to determines the best position for the windows automatically in a system because of users preferences and application objectives. This paper proposes a multiwindow layout system that determines the best window positions for individual users based on user operation history. This paper proposes a multiwindow layout system that determines the best positions based on user operation history. This paper shows the effectiveness of the multiwindow layout system through experiments comparing this system with current multiwindow systems.

Key Words: Multiwindow，LVQ

1. は　じ　め　に

グラフィカルユーザインタフェース（Graphical User Interface：GUI）の環境では，複数のウインドウを同時に画面に表示するマルチウインドウシステムが実現されている．これにより，ユーザは一画面上で複数のタスクを同時に行うことが可能である．しかしながら，マルチウインドウシステムでは，ウインドウの重なり合いが可能なため，同時に複数のウインドウを開くと，必要な情報を含むウインドウが，他のウインドウの下に隠れる場合がある．そのため，ユーザはウインドウの位置や大きさを頻繁に変更することで，必要なウインドウを探したり，ウインドウを作業しやすい配置にする．しかしこれには，ユーザの作業効率を低下させるという問題点がある．

この問題を解決するために，様々な手法が提案されている(1)が，隠れているウインドウの扱いのみに注目したものが多い．これらは，ウインドウを整理することで，ユーザにとって必要なウインドウを探し出すという要求は満たしているが，作業する上で最適な配置を実現しているとは言い難い．また，最適なウインドウの配置は，その時のウインドウの状態や，ユーザの好みにより異なる．そのため，システムで最適な配置を一意的に決定するのは困難である．

そこで本研究では，ユーザの操作履歴に基づくマルチウインドウレイアウトシステムを構築した．このシステムでは，ユーザのウインドウ配置の履歴を用いることで，最適な配置がウインドウの状態やユーザの好みにより異なるという問題を解決する．さらに，システムがユーザとのインタラクションにより学習していくことで，配置の履歴の中から，ユーザにとって最適な配置の選択を可能にする．学習アルゴリズムには，学習ベクトル量子化（Learning Vector Quantization：LVQ）を用いる．LVQ は 2 層構造のネットワークで構成され，パターン分類に優れた教師あり学習である(2)．したがって，第 1 層に現在のウインドウの状態，第 2 層にウインドウの配置履歴を適用することで，現在のウインドウの状態を，履歴内のウインドウの配置にパターン分類できる．また，LVQ は教師あり学習であるため，ユーザに配置を選択させ，その結果を教師信号として，システムに学習させることで，ユーザとのインタラクションを可能にする．

本稿では，従来のマルチウインドウシステムと比較実験を行い，本システムの有効性を検証する．

2. マルチウインドウシステムの問題点

マルチウインドウシステムでは，複数のウインドウを同時に画面に表示することができ，ウインドウの位置，サイズなども自由に変更することができる．これによって，ユーザは一画面上で複数のタスクを同時に行うことが可能である．マルチウインドウシステムにはタイル型，オーバーラッピング型の 2 種類がある．タイル型はウインドウをタイル状に並べる方式で，初期の GUI の OS で実装されていた．これはすべてのウインドウを一覧できるが，ウインドウが相互に重なり合うことを許可していないため，同時に多数のウインドウを開くことができなかった．オーバーラッピング型はウインドウ重ね方式とも呼ばれており，複数のウインドウを相互に重なり合いを許して表示する方式である．現在のマルチウインドウシステムはオーバーラッピング型を意味する．オーバーラッピング型のマルチウインドウシステムでは，ウインドウの重なり合いが可能であるため，同時に複数のウインドウを開くと，必要な情報を含んだウインドウが隠れてしまう場合がある．そのため，ユーザはウインドウの位置や大きさを頻繁に操作することで，必要なウインドウを探したり，ウインドウを作業しやすい配置にしなければならない．これは無駄な操作であると共に，ユーザがストレスを感じ，作業効率が低下するという問題点がある(3)．

この問題を解決するために様々な手法が提案されている．Mac OS X v10.3 ではショートカットキーを押すことで画面上のすべてのウインドウ，または特定のアプリケーションのウインドウがタイル状に並んで表示される．これにより，目的のウインドウを容易に見つけることが可能である．しかしこれは，作業する上での最適な配置については考慮されていない．最適な配置は，そのとき使っているアプリケーションの種類や，行っているタスクなどの状態に依存する．また，ユーザの好みが反映されるため，システムで最適なウインドウの配置を一意的に決定することは困難である．

3. 学習ベクトル量子化

学習ベクトル量子化（Learning Vector Quantization：LVQ）は，入力データのパターンを分類する．LVQ は，自己組織化マップ（Self-organizing feature map：SOM）(4)を教師あり学習に改良したものである．

Fig.1 に示すように，LVQ は，SOM と同様に 2 層構造のネットワークで構成され，第 1 層を入力層，第 2 層を出力層という．入力層の入力ベクトルの各要素は，すべての出力層のニューロンと結合している．また，同じ層のニューロン間に結合はない．LVQ では，入力ベクトルは教師信号として，自身の属するクラスを持っている．出力層のそれぞれのニューロンも自分のクラスを持っている．入力ベクトルは入力層と出力層の間の重みによって，出力層のいずれかのクラスに分類される(5)．この時，入力ベクトルが正しく分類された場合と，誤って分類された場合で異なる式を用い，重みを更新する．

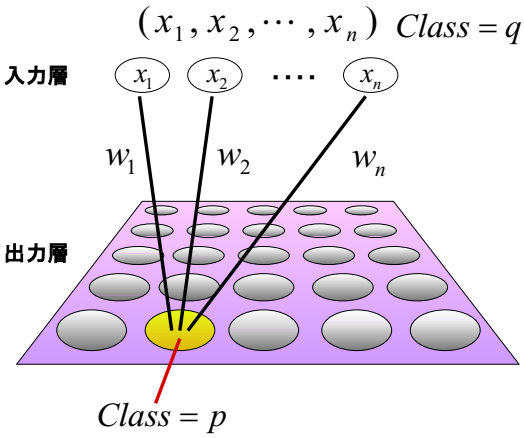

Fig. 1　LVQ

4. ユーザの操作履歴に基づくマルチウインドウレイアウトシステム

4.1 システムの概要

本研究では，ユーザの操作履歴に基づくマルチウインドウレイアウトシステムを構築する．

本システムは，基本的には，Fig.2 に示すように，3 つのウインドウで構成され，一つのウインドウを動かした時，残り 2 つのウインドウの配置候補を表示するものである．

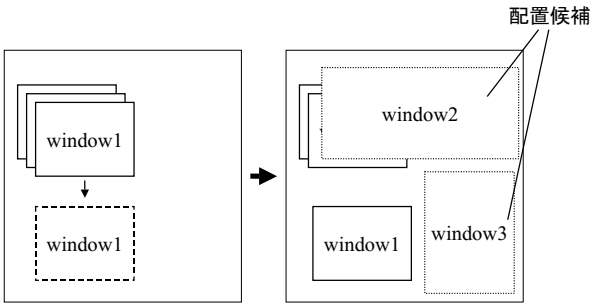

Fig. 2　The screen of a multiwindow layout system

ユーザは画面に表示されている配置候補をキーボードの「Ctrl」+「Shift」+「A」を押すことで，他の候補に切り替えることができる．その中から 1 つを選択し，「Ctrl」+「Shift」+「Z」を押して決定すると，ウインドウが配置候補の位置に移動する．また，候補の中に好ましい配置が無かった場合，「Ctrl」+「Shift」+「X」を押すことで，配置候補を消去し，ユーザが任意の配置にウインドウを移動する．この時，「Ctrl」+「Shift」+「Z」を押すことで，この配置を操作履歴に追加できる．Table.1 にコマンドを示す．

Table 1　Keyboard operation

「Ctrl」+「Shift」+「A」	表示している配置候補の切り替え
「Ctrl」+「Shift」+「Z」	配置候補の決定 または，データベースに追加
「Ctrl」+「Shift」+「X」	配置候補を消去

学習システムでは，アクティブなウインドウの座標と大きさを入力とし，配置候補を出力する．その中からユーザが選択した配置を教師信号として，LVQ により，配置候補の重みを更新する．Fig.3 に学習システムの概要を示す．

4.2 学習システムの流れ

学習システムでは，最初に，データベースから配置履歴とその時の重みを取り出して，重みベクトルを設定する．次に，入力ベクトルとの距離が小さい重みベクトルを持つ配置履歴を，配置候補として 5 つ選択する．その中で，入力ベクトルとの距離が最小となる重みベクトルを持つ配置候補を画面に表示する．そこで，ユーザが決定した配置を教師信号とし，LVQ により，データベースの重みベクトルを更新する．また，ユーザが配置候補から配置を決定しなかった場合，ユーザが任意にウインドウを動かし，決定した配置を新たにデータベースへ追加する．以下に，本システムのアルゴリズムを示す．

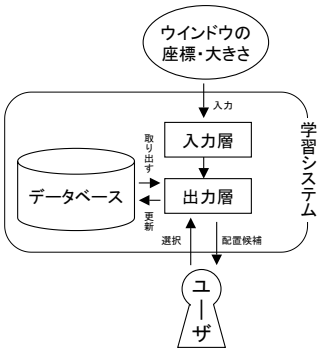

Fig. 3　Overview of the system

Step1：入力ベクトルの入力

ウインドウを一定以上動かしたとき，ウインドウの x 座標，y 座標，横幅，縦幅のピクセル値を要素とする 4 次元ベクトル $\boldsymbol{x}_k$ を入力ベクトルとして，学習システムの入力層に入力する．$\boldsymbol{x}_k$ を式 (1) に示す．この時，k は動かしたウインドウの番号である．

$$\begin{aligned} &\boldsymbol{x}_k = (x_{k0}, x_{k1}, x_{k2}, x_{k3}) \\ &k = 1, 2, 3 \end{aligned} \tag{1}$$

Step2：データベースから配置履歴を取り出す

データベースには，3 つのウインドウの配置履歴とその時の重みが格納されている．配置履歴は，1 つのデータが 3 つのウインドウそれぞれの x 座標，y 座標，横幅，縦幅で表され，単位はピクセルである．

Table 2　Database

		x 座標	y 座標	横幅	縦幅
$Class(1)$	window1	200.0	0.0	500.0	600.0
	window2	50.0	500.0	550.0	900.0
	window3	540.0	700.0	700.0	400.0
重み	$\boldsymbol{w}_{11}$	198.5	5.2	522.0	633.0
	$\boldsymbol{w}_{21}$	50.0	500.0	550.0	900.0
	$\boldsymbol{w}_{31}$	544.0	712.3	697.9	388.1

データベースから配置履歴の重みを取り出し，学習システムの入力層と出力層の間の重みベクトルとして設定する．重みベクトルは $\boldsymbol{w}_{kj}$ とし，式 (2) に示す．また，重みベクトルの初期値は配置履歴と同じ値となる．

$$\boldsymbol{w}_{kj} = (w_{kj0}, w_{kj1}, w_{kj2}, w_{kj3}) \tag{2}$$

Step3：出力層で入力ベクトルと重みベクトルの距離を計算

出力層では，入力ベクトルと重みベクトルの距離 d_{kj} を計算する．d_{kj} を式 (3) に示す．

$$d_{kj} = \sqrt{\sum_{i=0}^{3}(x_{ki} - w_{kji})^2} \tag{3}$$

Step4：重みベクトルの距離が小さい配置履歴の選択

出力層で d_{kj} が小さい配置履歴を配置候補として，5 つ選択する．また，d_{kj} が最小となる配置候補 p の配置を画面に表示する．

Step5：配置の決定

ユーザが配置候補の中から配置を決定した場合，その配置にウインドウを移動する．この時，選択された配置候補を q とする．また，ユーザが配置候補の中から配置を決定しなかった場合，ユーザは任意の配置にウインドウを移動し，その時の配置をデータベースに追加する．

Step6：重みの学習

ユーザが配置候補の中から，配置を決定した場合と，決定しなかった場合で，重みベクトルの更新式が異なる．また，η は [0,1] の定数である．

ユーザが配置を決定した場合

$$\Delta \boldsymbol{w}_{kp} = \begin{cases} +\eta(\boldsymbol{x}_k - \boldsymbol{w}_{kp}) : p = q \\ -\eta(\boldsymbol{x}_k - \boldsymbol{w}_{kp}) : p \neq q \end{cases} \tag{4}$$

$$\Delta \boldsymbol{w}_{kq} = +\eta(\boldsymbol{x}_k - \boldsymbol{w}_{kq}) : p \neq q \tag{5}$$

ユーザが配置を決定しなかった場合

$$\Delta \boldsymbol{w}_{kp} = -\eta(\boldsymbol{x}_k - \boldsymbol{w}_{kp}) \tag{6}$$

Step1 から **Step6** の動作を繰り返して，重みを学習する．

5. 実　　験

本システムの有効性を検証するために，本システムと従来のマルチウインドウシステムで，それぞれ，簡単なクイズ形式のタスクを用いて，タスク達成時間に関する比較実験を行った．

タスクを Table.3 に示す．タスクの Q1，Q3 の問題はそれぞれ 20 問の中から，および Q2 の画像は 15 枚の中から，毎回ランダムに 1 問出題される．これらのタスクはすべて，一定以上のウインドウサイズと，複数のウインドウの参照が必要という特徴がある．よって，被験者がタスクを効率よく行うためには，ウインドウの位置や大きさを変更しなければならない．なお，地図や画像など，タスクに必要な情報はすべてアイコン化してデスクトップ上に配置している．被験者は，アイコンをウインドウにドラッグアンドドロップすることで，これらの内容を確認する．これは，ウインドウの下に隠れているアイコンをクリックするという作業を想定している．

Table 3　List of the task

Q1	地図 1 を見て，以下の問題に答えよ
Q2	画像 x と画像 y を見比べ，違いを以下の中から選べ
Q3	文 1 と文 2 は同じ内容の文章であるが，文 1 は方言で書かれている．これらを見比べて，以下の方言の意味を答えよ

実験は，20 代の男性 9 名，女性 3 名の合計 12 名を被験者として，Table.4 に示す順序で行った．被験者は各システムで，最初に予備実験として，Q1，Q2，Q3 のすべてのタスク達成に要した時間を計測する．これを繰り返し，タスク達成時間が一定の値に収束するまで行う．その後，本実験を行う．本実験は予備実験と同様の方法で，5 回タスク達成時間を計測し，その平均を計算する．

各被験者は，両方のシステムで予備実験，および本実験を行ったが，順序による慣れの影響を避けるため，被験者 1〜被験者 6 は，本システム，従来のマルチウインドウシステムの順に，被験者 7〜被験者 12 は逆に，従来のマルチウインドウシステム，本システムの順に実験を行った．また，$\eta = 0.2$ に設定した．

Table 4　experiment procedure

実験順序	被験者 1〜6	被験者 7〜12
1	本システムの予備実験	従来のシステムの予備実験
2	本システムの本実験	従来のシステムの本実験
3	従来のシステムの予備実験	本システムの予備実験
4	従来のシステムの本実験	本システムの本実験

6. 結 果 と 考 察

Fig.4 に予備実験における，全被験者のタスク達成時間の平均の推移を示す．本システムのタスク達成時間は，実験回数を重ねるごとに，減少している．これは，本システムの操作に対する被験者の学習が良好に行われたことを示している．

Table.5，Table.6 に，本実験における，被験者ごとの平均タスク達成時間を示す．被験者 1〜被験者 6 は，本システム，従来のシステムの順で実験を行ったため，タスクに対する慣れを考慮すると，後に行った従来のシステムの方が，タスク達成時間が短くなると予想できる．しかし，Table.5 に示すように，被験者 2 と被験者 4 は，本システムの方が，従来のシステムより，平均タスク達成時間が短い．これは本システムが，今回のタスクに対して，最適な配置を学習したことを示している．

Table.7 に，本実験における，全被験者のタスク達成時間の平均を示す．両システム間でタスク達成時間に大きな差はなかった．被験者に今回の実験に関する感想を聞き取り調査したところ，原因として，多くの被験者が今回のタスクを，ウインドウの配置をほとんど変更せずに行ったことが挙げられる．これは，被験者が今回のタスクに対して，ウインドウを作業しやすい配置に変更することよりも，作業しにくいにもかかわらず，ウインドウの配置を変更しないことを優先したと考えられる．

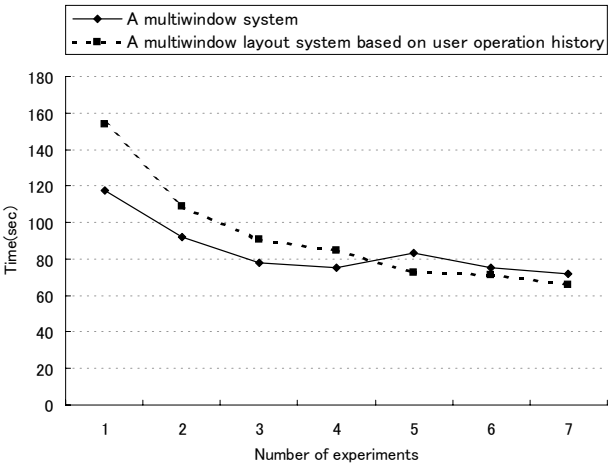

Fig. 4　The time it take to complete the task in pilot study

Table 5　Experimental subjects' 1〜6 results

被験者	1	2	3	4	5	6
本システム	59.6	50.7	71.2	65.5	71.2	80.2
従来のシステム	49.4	63.2	65.5	74.3	51.2	57.4

Table 6　Experimental subjects' 7〜12 results

被験者	7	8	9	10	11	12
従来のシステム	55.4	75.5	93.7	66.0	66.2	64.0
本システム	50.0	62.8	85.8	55.2	60.2	51.6

Table 7　All experimental subjects' results

	平均タスク達成時間
従来のシステム	65.1
本システム	63.7

7. お　わ　り　に

本研究では，ユーザの操作履歴から LVQ を用いて学習することで，ユーザにとって最適なウインドウの配置候補を出力するマルチウインドウレイアウトシステムを構築した．また，従来のマルチウインドウシステムとのタスク達成時間に関する比較実験を行い，その有効性を検証した．その結果，一部の被験者で作業効率の向上が確認できた．

しかし，その他の被験者においては，本システムの有効性は検証できなかった．その原因として，被験者がウインドウの配置を変更せずにタスクを行ったこと，インタフェースのユーザビリティ評価をしなかったことが挙げられる．今後は，ウインドウの配置を変更しなければ，作業がより困難となるタスクでの実験，およびユーザビリティ評価の方法の検討を行う．

参 考 文 献

(1) 津原進：“ファジイルールを用いたウインドウの自動配置”，電気学会論文誌 C，vol.112–C, no.1, pp.10–18，1992.

(2) 萩原将文：“ニューロ・ファジィ・遺伝的アルゴリズム”，pp.70–72，産業図書株式会社，1994.

(3) 有澤誠：“ヒューマンインタフェイス”，実教出版株式会社，1995.

(4) 徳高平蔵，岸田悟，藤村喜久郎：“自己組織化マップの応用”，海文堂出版株式会社，1999.

(5) 馬場則夫，小島史男，小澤誠一：“ニューラルネットの基礎と応用”，pp.83–85，共立出版株式会社，1994.

[連絡先]

住所　：　〒 525-8577 滋賀県草津市野路東 1-1-1
立命館大学理工学部情報学科亀井研究室

Tel　：　077-561-2807

Fax　：　077-561-2861

e-mail　：　fukuda@spice.cs.ritsumei.ac.jp