脳情報の成分分析とヒューマノイドへの情報伝達

(15300077)

平成 15 年度～平成 16 年度科学研究費補助金（基盤研究(B)(2)）研究成果報告書

平成 17 年 4 月

研究代表者　松山 泰男

(早稲田大学理工学術院教授)

はしがき

この報告書は，平成15年度～平成16年度まで，科学研究費補助金・基盤研究(B)(2)として，「脳情報の成分分析とヒューマノイドへの情報伝達」を行った成果をまとめたものである．得られた成果は，下記のように項目化できる．

[人体モーション・アニメーション・ヒューマノイドのネットワーク上での同一化]

(1) 人体，そのアニメーション，そしてヒューマノイドの動作を三位一体とするには，人体モーションを認識する機構を用意して認識結果を言語表現のレベルに抽象化すればそれが可能となることを見いだした．

(2) 認識機構としては，まず隠れマルコフモデル（HMM, Hidden Markov Models）を用いてこれを実現し，人体モーション・アニメーション・ヒューマノイドの同一化を世界に先がけて実現した．この論文は，APNNA Best Paper Award for Application Oriented Research を受賞した．

(3) 認識機構の改良版として，ベイジアンネットワークを用いる方式を実現した．

(4) ネットワーク環境におけるヒューマノイド制御の時代を見越して，FINALE（Framework for Intelligent Network Agents Looking at the Environment）に基づくエージェント分散システムを構築した．

[独立成分分析による脳情報の推定と利用]

(5) 凸ダイバージェンスの最小化から導かれた独立成分分析（ICA, Independent Component Analysis）のアルゴリズム，すなわちf-ICAを，人間の脳のfMRI画像（磁気共鳴機能画像）に適用し，脳の機能マップを得ることができた．

(6) 生体情報を緊急信号としてヒューマノイドに与えることを検討し，脳情報と筋情報の両者を同時に用いることがよいという結果を得た．

本研究は，以上のような成果を得て終了した．

研究組織

研究代表者： 松山 泰男（早稲田大学理工学術院教授）
研究分担者： 中島 達夫（早稲田大学理工学術院教授）
研究分担者： 勝又 尚人（早稲田大学理工学術院助手）

交付決定額（配分額） （金額単位：千円）

	直接経費	間接経費	合計
平成15年度	9,700	0	9,700
平成16年度	7,100	0	7,100
総計	16,800	0	16,800

研究発表

(Ⅰ) 学会誌等

(1) Yasuo Matsuyama, Ryo Kawamura, and Naoto Katsumata, Independent component analysis with joint speedup and supervisory concept injection: Applications to brain fMRI map distillation, Proceedings of International Workshop on Independent Component Analysis and Blind Signal Separation, pp. 173-178, April 2003.

(2) Yasuo Matsuyama, Independent component analysis minimizing convex divergence, Lecture Notes in Computer Science, No. 2714, pp. 27-34, Springer Verlag, June 2003.

(3) Norio Nishioka, Yasuo Matsuyama, et al, Agent generation and resource allocation in a network computing environment, Proceedings of Asia-Pacific Symposium on Information and Telecommunication Technologies, pp. 63-68, November 2003.

(4) Yasuo Matsuyama, Ryo Kawamura, Hiroaki Kataoka, et al., Image compression based upon independent component analysis: Generation of self-aligned ICA bases, Proceedings of Australian and New Zealand Intelligent Information System Conference, pp.3-8, December 2003.

(5) Yasuo Matsuyama, Hiroaki Kataoka, Naoto Katsumata, and Keita Shimoda, ICA photographic encoding gear: Image bases towards IPEG, Proceedings of International Joint Conference on Neural Networks, vol. 3, pp. 2129-2134, July 2004.

(6) Yasuo Matsuyama and Ryo Kawamura, Promoter recognition for E. coli DNA segments by independent component analysis, Proceedings of Computational Systems Biology, pp. 689-691, August 2004.

(7) Tatsuo Nakajima, Daiki Ueno, Experiences with building middleware infrastructures for home computing on commodity software, Proceedings of 10th International Conference on Real-Time and Embedded Computing, Systems, and Applications, pp246-265, October 2004.

(8) Yasuo Matsuyama, Satoshi Yoshinaga, Hirofumi Okuda, et al, Towards the unification of human movement, animation and humanoid in the network, Lecture Notes in Computer Science, No. 3316, pp. 1135-1141, Springer Verlag, November 2004 (APNNA Best Paper Award for Application Oriented Research) .

(Ⅱ) 口頭発表

(1) Yasuo Matsuyama, Iterative optimization of convex divergence: Applications to independent component analysis, Proceedings of International Symposium on Information Theory, p. 214, June 2004.

研究成果による工業所有権の出願・取得状況

(1) 松山泰男，吉永　聖，奥田裕文，谷川一也，動作伝達システムおよび動作伝達方法，平成 16 年 11 月 17 日出願，特願 2004-333618.

人体モーション，アニメーション，ヒューマノイド動作の一体化に関する研究成果

(APNNA Best Paper Award for Application Oriented Research 受賞論文)

APNNA Best Paper Award for Application Oriented Research

2004

ICONIP 2004
11th International Conference on Neural Information Processing

Paper : Towards the Unification of Human Movement, Animation and Humanoid in the Network

Authors : Y. Matsuyama, S. Yoshinaga, H. Okuda, K. Fukumoto, S. Nagatsuma, K. Tanikawa, H. Hakui, R. Okuhara and N. Katsumata

Presented by : Y. Matsuyama

N. R. Pal
President, APNNA

N. Kasabov
Chair, Award Committee

November 25, 2004

Indian Statistical Institute | Asia Pacific Neural Network Assembly | Jadavpur University

Towards the Unification of Human Movement, Animation and Humanoid in the Network

Yasuo MATSUYAMA[1], Satoshi YOSHINAGA[1,2], Hirofumi OKUDA[1], Keisuke FUKUMOTO[1], Satoshi NAGATSUMA[1], Kazuya TANIKAWA[1], Hiroto HAKUI[1], Ryusuke OKUHARA[1], and Naoto KATSUMATA[1]

[1] Department of Computer Science, Waseda University, Tokyo 169-8555, Japan
[2] IT & Mobile Solutions Network Company, Sony Co. Tokyo 108-6201, Japan

[1] {yasuo, hijiri, hiro, keisuke, nagatyo, tanikawa-k, h891, oku-ryu, katsu}@wizard.elec.waseda.ac.jp
[2] Satoshi.Yoshinaga@jp.sony.com

Abstract. A network environment that unifies the human movement, animation and humanoid is generated. Since the degrees of freedom are different among these entities, raw human movements are recognized and labeled using the hidden Markov model. This is a class of gesture recognition which extracts necessary information transmitted to the animation software and to the humanoid. The total environment enables the surrogate of the human movement by the animation character and the humanoid. Thus, the humanoid can work as a *moving computer* acting as a remotely located human in the ubiquitous computing environment.

1 Introduction

Recent advancement of computing power accompanied by the microminiaturization has promoted sophisticated human interfaces. Another social progress caused by this cost effective enhancement is the networking for the ubiquity. To be compatible with such trends, this paper presents the unification of human movements, animation characters and humanoids in the network computing environment. It is important for this purpose to incorporate various levels of learning algorithms on the machine intelligence.

The degree of the freedom of the human movement is around a few hundred. Humanoids available as contemporary consumer electronics have the freedom of its one tenth. Animation characters as software agents have the order of somewhere in the middle according to the software's sophistication. Because of such differences in the freedom, human movements are modeled first by a Hidden Markov Model (HMM). This problem is a class of gesture recognition which extracts the information transmitted to the animation software and to the humanoid.

The rest of the paper is organized as follows. Chapter 2 is devoted to the generation of the data structure compatible with our purpose. In Chapter 3, an HMM recognizer is designed using the training movements. The learned model is utilized for controlling an animation character and a humanoid called HOAP-2

[1]. Chapter 4 describes the realization of the humanoid movement mimicking the human. Chapter 5 gives concluding remarks including the next step.

2 Data Acquisition and Transformation for Human Body Movement

2.1 Measured Raw Data

Human body's movements are measured in real time by the MotionStar^{TM} [2] which uses the direct-current magnetic field. Eleven sensors are used for our measurement. Each sensor measures a 3×1 position vector and a 3×3 rotation matrix. Therefore, human movements give 11 time series of $3 + 3 \times 3 = 12$-dimensional vector-data as numerals. Such raw data *per se* do not have any spatial structure for the body movement. Therefore, we have to specify relationships among these time series.

2.2 Bone Frame Expression

Bones are connected. This connection can be expressed precisely by the tree structure in Figure 1. The root element is selected to represent the Hips. Sub-elements are LeftHip, RightHip, Chest, each of which has further sub-elements. These data are expressed by the BVH format (Bio Vision Hierarchical data) [3].

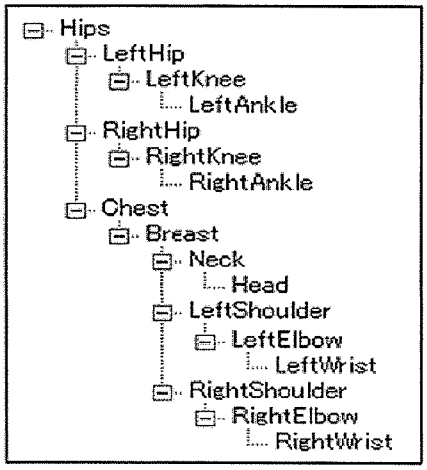

Fig. 1. Tree structure of bones.

It is important to maintain the independence of the personal physique. For the sake of this demand, we give the following comments in advance.

(a) Data from 11 sensors are expanded to 17 time series by the interpolation according to the tree structure of Figure 1. This is because sensors fixed at some joints may irregularly move to create inaccurate data. Therefore, for instance, movements of two elbows and two knees are computed by using nearby sensors' data and normalized bones. Such a process gives $17 \times 3 = 54$-row data.
(b) Relative rotation angles is found better than absolute ones for the portability to a wide range of humans.
(c) As will be explained in the experiment in Section 3.4, the original data set is further expanded to 69-row data.
(d) The time-frame is selected to be 50 ms. Because of the network communication and the humanoid movement computation, the time-frame needs to be long enough. But, this can not be too much. Thus, the time-frame of 50 ms was selected so that the movement can be tracked and reproduced as an animation smoothly enough based upon the experience of the speech recognition whose typical case is 20 ms.
(e) In the case of the speech recognition, each time-frame is expressed by only 25 rows or so. Therefore, the body movement recognition, or the gesture recognition, can not be a direct adaptation of the well-established speech recognition.

3 Recognizer Design by HMM Learning

3.1 Recognition System

Given the input of the 69-row data stream, it is necessary for the recognizer to categorize human body movement. The Hidden Markov Model (HMM) is a viable learning algorithm for this purpose. HMM's transitions correspond to the labels. The HMM software can be anything if it has a flexible input/output interface. We chose the HTK (Hidden Markov Model Toolkit) [4] since the modification of the I/O style matching with it does not require heavy tasks. Figure 2 illustrates our configuration of the total gesture recognition system. As is usual in learning systems, the model is fixed after the training.

3.2 Tasks and Associated Data Preparation

As is expressed by the tree structure of Figure 1, movements of four limbs besides the hips are the most important for the gesture recognition. Therefore, we prepared eight labels for the movements as in Table 1. For this experiment, we prepared training and test data sets as follows.

(a) 10 sets of 8 patterns generated by a single person for training the recognizer ($10 \times 8 = 80$ patterns).
(b) 80 patterns by the same person in a different environment for testing.
(c) Different 8 persons' patterns for testing ($8 \times 8 = 64$ patterns).

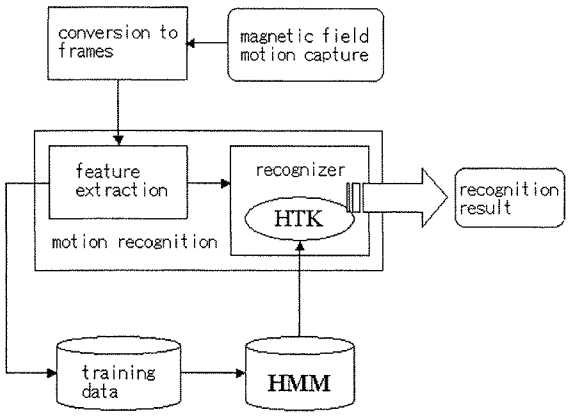

Fig. 2. Recognition system.

Table 1. Recognition labels for movements

	ArmUp	ArmDown	LegUp	LegDown
Left	LAU	LAD	LLU	LLD
Right	RAU	RAD	RLU	RLD

3.3 Number of States

The first step is to identify an appropriate number of states. There are theoretical criteria for this purpose such as the MDL (Minimum Description Length), however, repeated experiments on real data are essential to decide the actual best number. Therefore, we have to test various number of states by measuring the recognition performance. Figure 3 compares the average log-likelihood which reflects the performance of the recognition by the HMM models[1]. By this test, the number of states was judged to be 5 or 6.

The next test is to see the difference between the best model and its runner-up. Table 2 shows the difference in the log-likelihood. This result indicates that the more the number of states is, the larger the difference is. Therefore, we chose the number of states to be 6.

3.4 Selected Features

In parallel to the state number selection, we checked to see which form of the input data is best for the recognition task. We prepared four types of input data:

(A) Use $17 \times 3 = 51$-row rotation data,
(B) Use 51-row *rotation difference* data and 3-row root position (Hips),

[1] These data correspond to the case (D) of Section 3.4.

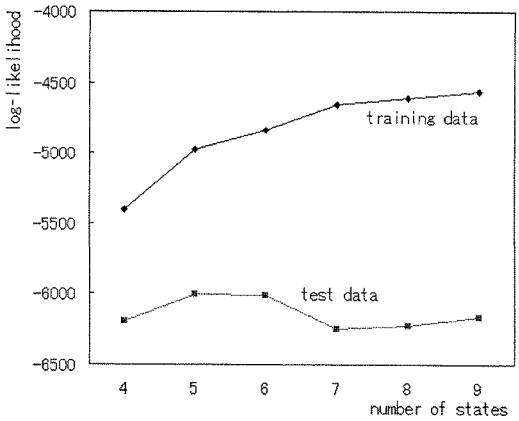

Fig. 3. Average log-likelihood.

Table 2. Difference to the Runner-up

	number of states					
	4	5	6	7	8	9
First	-6192.33	-6004.40	-6011.06	-6250.39	-6226.51	-6162.67
Second	-8838.18	-9888.39	-10993.00	-13262.22	-14196.33	-15780.30
difference	2645.86	3884.00	4981.94	7011.83	7969.82	9617.63

(C) Use 105-row data by adding (A) and (B),
(D) Use 69-row data by adding (B) and 5 leaves.

Table 3 summarizes the results of the recognition on the data outside the training data. By this result, the difference of the rotation angle is better than the rotation angle *per se*.

Table 3. Recognition performance

	method A	method B	method C	**method D**
subject 1 (48 patterns)	25%(12)	100%(48)	27%(13)	**98%(47)**
subject 2 (46 patterns)	37%(17)	87%(40)	43%(20)	**98%(45)**
total	31%(29)	94%(88)	35%(33)	**98%(92)**

3.5 Animation for Monitoring

There are a few commercially available animation tools for BVH data. But, we had to develop our own display tool. This is because, as in Figure 2, our system needs to be designed including the recognizer and the controller for the succeeding system, the humanoid.

Figure 4 illustrates the course of the LeftLegUp. Thus, the label of LeftLegUp stands for such series of motions, not a still pose of the left-leg-up. This will be related to the humanoid motion of Section 4.

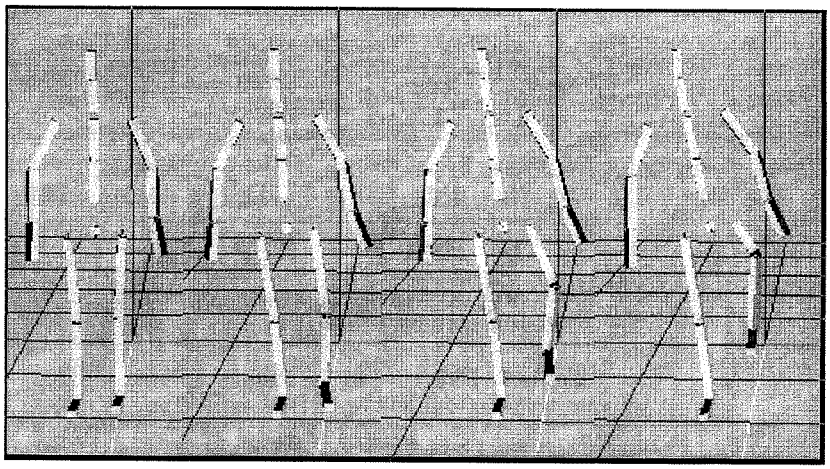

Fig. 4. A series of movements for LeftLegUp.

4 Humanoid Motion

4.1 Transmission of Recognized Label

Characters in the animation can behave more sophisticatedly according to the level of the software. But, humanoids can behave only less flexible. Contemporary humanoids, even though they have made a great advance, are mostly composed of metallic materials and powered by motors. Body balances of humans and humanoids are very different. HOAP-2 appearing in this paper has 12 joints with 21 degrees of the movement freedom. Considering this ability, we transmit the recognition results as commands. Transmitting the BVH data directly leads to malfunctioning of humanoid motions. Imagine standing on one leg as is illustrated in Figure 4. This is possible by HOAP-2, however, its duration needs to be shorter than actual human movement.

4.2 Execution of Transmitted Labels

The recognition and the labeling of human motions given in Section 3 have the role of ameliorating the discrepancy between the differences of the freedom and *muscle* powers. Thus, the obtained labels for the motion can be used as commands to the humanoid. The humanoid is controlled by the built-in real-time Linux. Figure 5-left shows LeftLegUp by the humanoid. Figure 5-right illustrates LeftLegUp by the animation character, which is closer to actual human motion.

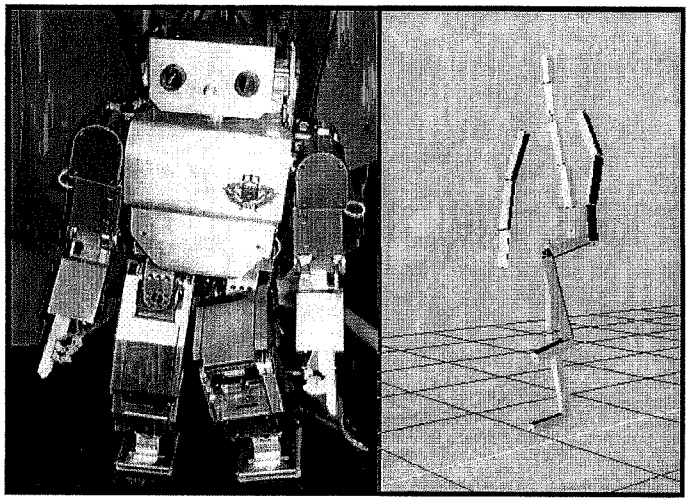

Fig. 5. LeftLegUp by HOAP-2 and the animation character.

5 Concluding Remarks

The technical purpose of this paper included

(a) the recognition of human motions,
(b) the utilization of the recognition results for controlling the humanoid,
(c) imbedding the humanoid to a network environment as a movable computing node.

As the initiative attempt, these items were satisfied. The use of the recognized label together with lower level data, including biological ones, can enhance the sophistication of the role of the humanoid in the network. This includes the surrogate of a remote human. This is the step connected to this paper's study.

Acknowledgment

This work was supported in part by the Grant-in-Aid for Scientific Research by MEXT, #15300077. The authors are grateful to Mr. Goro Kazama for his early contributions to the motion capturing and animation coding.

References

[1] Fujitsu Automation Co.: HOAP-2 Reference Manual (2003)
[2] Ascention Technology Co.: http://www.ascention-tech.com/
[3] Meredith, M. and Maddock S.: Motion Capture File Formats Explained, Department of Computer Science, University of Sheffield (2001)
[4] Young, S., et al.: The HTK Book, Cambridge University Engineering Department, Speech Group and Entropic Research Laboratory Inc. (1989)

独立成分分析と脳情報に関する研究成果

INDEPENDENT COMPONENT ANALYSIS WITH JOINT SPEEDUP AND SUPERVISORY CONCEPT INJECTION: APPLICATIONS TO BRAIN fMRI MAP DISTILLATION

Yasuo Matsuyama, Ryo Kawamura, and Naoto Katsumata

Department of Computer Science, School of Science and Engineering,
Waseda University, Tokyo 169-8555 Japan
yasuo2@waseda.jp, ryo@wizard.elec.waseda.ac.jp, katsu@ruri.waseda.jp

ABSTRACT

Methods to combine speedup terms and supervisory concept injection are presented. The speedup is based upon iterative optimization of the convex divergence. The injection of supervisory information is realized by adding a term which reduces an additional cost for a specified concept. Since the convex divergence includes usual logarithmic information measures, its direct application gives faster algorithms than existing logarithmic methods. This paper first shows a list of newly obtained general properties of the convex divergence. Then, these properties are used to derive faster algorithms for the independent component analysis. Then, an additional term for incorporating supervisory information is introduced. The efficiency of the total algorithm is tested using a set of real data - - brain fMRI time series. Successful results in view of convergence speed, software complexity, and extracted brain maps are reported. Finally, another class of the convex divergence optimization, the α-EM algorithm, is commented upon.

1. INTRODUCTION

Computing and optimizing information measures comprise many important problems both in theory and in applications. Independent component analysis (ICA) [1] is has dual aspects: It is theoretically interesting due to its semi-parametric nature, and it is rich in applications due to its independence of physical entity. This paper covers both of such aspects.

Usually, the target information measure for optimization is based upon logarithm [2] and [3]. But, the information measure to be optimized in this paper is the convex divergence [4]. Since the convex divergence includes usual logarithmic information measures as special cases, we can expect better performance than the logarithmic ones. In the problem of the ICA, the merit appears in convergence speed without losing the algorithm's flexibility. This is a featuring aspect of this paper in theory.

The other side of a coin, the application aspect, is related to the brain functional MRI analysis (fMRI) [5]. Since the derived algorithm using the convex divergence maintains flexibility to create variants, an injection of supervisory information [6] is possible. Therefore, the organization of this paper becomes as follows. Section II gives basic properties of the convex divergence and their relationships to the extended class of logarithm. Section III gives a formulation of the independent component analysis as a minimization of the convex divergence. Then, concrete algorithms are derived. On the convergence speed, the proposed method is faster than traditional or logarithmic methods. This is examined in Section IV through brain fMRI map distillation. The separation of brain map's active areas is quite successful using the supervisory information. Section V gives general remarks on the use of the convex divergence. Other problems coined into the convex divergence minimization, e.g., expectation-maximization are commented on.

2. PROPERTIES OF THE CONVEX DIVERGENCE

2.1. Definition and Differential Properties

The convex divergence, or f-divergence [4] (as its forerunner, Eq. (4.20) of [7]), is defined as follows. Let ψ and φ be generic parameters for probability density functions. The convex divergence between two probability densities p_ψ and p_φ is defined by the following equation.

$$\begin{aligned} D_f(\psi\|\varphi) &= \int_{\mathcal{Y}} p_\varphi(y) f(p_\psi(y)/p_\varphi(y)) dy \\ &= \int_{\mathcal{Y}} p_\psi(y) g(p_\varphi(y)/p_\psi(y)) dy \\ &= D_g(p_\varphi\|p_\psi) \geq g(1) = f(1). \end{aligned} \quad (1)$$

Here, $\mathcal{Y}$ is chosen to be a N-dimensional Euclidian space. The function $f(r)$ is convex for $r \in (0, \infty)$. Its dual func-

This work was supported by the Productive ICT Academia Program in the 21st Century COE Programs, and by the Grant-in-Aid for Scientific Research.

tion $g(r)$ is defined by

$$g(r) = rf(1/r), \tag{2}$$

which is also convex for $r \in (0, \infty)$. We normalize the constant $f(1) = 0$. Then, the convex divergence is zero if and only if $p_\psi(y) = p_\varphi(y)$, y-a.e.

We consider the case that $f(r)$ is twice continuously differentiable. Let ∂^{ij} mean that i-times partial differentiation with respect to ψ and j-times partial differentiation with respect to φ. Then, we have the following relationships.

$$D_f(\varphi\|\varphi) = 0, \tag{3}$$
$$\partial^{10} D_f(\varphi\|\varphi) = 0, \tag{4}$$
$$\partial^{20} D_f(\varphi\|\varphi) = f''(1) F_Y(\varphi). \tag{5}$$

Here, $F_Y(\varphi)$ is the Fisher information matrix. Because of the relationship (5), the convex divergence can be regarded as a fundamental amount of information. Then, we pay attention to the following ratio.

$$c \stackrel{\text{def}}{=} \frac{f''(1)}{f'(1)} = -\frac{g''(1)}{g'(1)} \in \boldsymbol{R} \tag{6}$$

By using this constant, the following expansions can be obtained:

$$\frac{f''(r)}{f'(1)} = \frac{1}{c(1-c)}(r - r^c) + o(1), \tag{7}$$
$$\frac{g''(r)}{g'(1)} = \frac{-1}{c(1-c)}(r^{1-c} - 1) + o(1). \tag{8}$$

From equations (7) and (8), we find that

$$L^{(c)}(r) = \frac{1}{1-c}(r^{1-c} - 1) \tag{9}$$

can be regarded as an extended class of the logarithm. We call this the c-logarithm. In fact, we have

$$L^{(1)}(r) = \log r. \tag{10}$$

If we add a set of assumptions that

$$f(xy) = kf(x)f(y) \tag{11}$$

and

$$f''(1) = g''(1) = 1, \tag{12}$$

then the α-divergence [8], [9] is obtained. In this case, the constants c and α have the following relationships:

$$c = \frac{1-\alpha}{2} \tag{13}$$

and

$$1 - c = \frac{1+\alpha}{2}. \tag{14}$$

2.2. Information Matrix and Cramér-Rao Bound

By using the c-logarithm, we have the following equality on the information matrices.

$$M^{(c)}(\varphi) \stackrel{\text{def}}{=} E_p\left[c\, p^{-2(1-c)} \left(\frac{\partial L_c}{\partial \varphi}\right)\left(\frac{\partial L_c}{\partial \varphi^T}\right)\right] \tag{15}$$
$$= -E_p\left[p^{-(1-c)} \left(\frac{\partial^2 L_c}{\partial \varphi \partial \varphi^T}\right)\right] = cF_Y(\varphi), \tag{16}$$

whose early versions are found in [10], [11], [12]. We consider the case that the information matrices are positive definite, i.e., $F_Y(\varphi) > 0$, $c > 0$, and hence $M_Y^{(c)}(\varphi) > 0$. Because of Equations (9), (15) and (16), we have that the Cramér-Rao bound is independent of c. This means that the general convex divergence can be used in estimation problems without sacrificing the performance in comparison with the logarithmic methods. Guaranteed by this fact, we discuss iterative minimizations of the convex divergence for the independent component analysis.

3. INDEPENDENT COMPONENT ANALYSIS USING CONVEX DIVERGENCE

3.1. Problem Formulation

In the problem of the convex divergence, a set of vector random data is given.

$$x(n) = [x_1(n), \cdots, x_K(n)]^T = As(n), \quad (n = 1, \cdots, N). \tag{17}$$

Here, the K by K matrix A and the source vector

$$s(n) = [s_1(n), \cdots, s_K(n)]^T \tag{18}$$

are unknown. Additional assumptions are the following.

1. The components $s_i(n)$ and $s_j(n)$ are independent each other for $i \neq j$.
2. The unknown components $s_i(n)$, $(i = 1, \cdots, K)$, are non-Gaussian except for at most one specific i.

Therefore, we want to find a demixing matrix

$$W = \Lambda \Pi A^{-1} \tag{19}$$

so that the components of

$$Wx(n) \stackrel{\text{def}}{=} y(n) = [y_1(n), \cdots, y_K(n)]^T \tag{20}$$

are independent each other for every n. Here, Λ is a nonsingular diagonal matrix and Π is a permutation matrix. These two matrices are also unknown.

Using the convex divergence D_f, this ICA problem is formulated as a minimization of the following cost function.

$$\begin{aligned} I_f(\bigwedge_{i=1}^K Y_i) &\stackrel{\text{def}}{=} D_f\left(p(y_1,\cdots,y_K)\|\prod_{i=1}^K q_i(y_i)\right) \\ &\stackrel{\text{def}}{=} D_f(p(y)\|q(y)) \\ &= D_g(q(y)\|p(y)) \\ &= I_g(\bigwedge_{i=1}^K Y_i) \\ &= \int_{\mathcal{X}} p(x) g\left(\frac{|W|q(y)}{p(x)}\right) dx. \end{aligned} \tag{21}$$

3.2. Update Equations

The generalized gradient [13], relative gradient [14], or natural gradient [15] denoted by $\tilde{\nabla}$ is obtained by multiplying cW^TW after partially differentiating (21) with respect to W. Then, we have the following equality.

$$\begin{aligned} &-\tilde{\nabla} I_g(\bigwedge_{i=1}^K Y_i) \stackrel{\text{def}}{=} -\frac{\partial I_g(\bigwedge_{i=1}^K Y_i)}{\partial W}(cW^TW) \\ &= -c\int_{\mathcal{X}} q(y) g'\left(\frac{|W|q(y)}{p(x)}\right)\left\{I - \vartheta(y)x^TW^T\right\}|W|dx\, W \\ &= -c\int_{\mathcal{Y}} q(y) g'\left(\frac{q(y)}{p(y)}\right)\left\{I - \vartheta(y)y^T\right\} dy\, W. \end{aligned} \tag{22}$$

Here, $\vartheta(y)$ is a vector

$$-\vartheta(y) = \text{col}\left[\left\{\frac{q_i'(y_i)}{q_i(y_i)}\right\}_{i=1}^K\right]. \tag{23}$$

We assume that $\vartheta(y_i)$ be an odd function such as y_i^3 and $tanh(y_i)$. Note that $\nabla I_g = \tilde{\nabla} I_f$. Equation (22) is not yet in a realizable form as a concrete algorithm since it contains an unknown probability density $q(y)$. Therefore, the next step is to find a realizable approximation to (22). Since

$$qg'(q/p) = -g''(1)p + \{g'(1) + g''(1)\}q + o(1) \tag{24}$$

holds around $p \approx q$, we have the following update value for an iterative minimization.

$$\begin{aligned} &-\frac{\partial I_f}{\partial W}(cW^TW) \\ &= -\frac{\partial I_g}{\partial W}(cW^TW) \\ &= f''(1)\Big[c\left\{I - E_{p(y)}[\vartheta(y)y^T]\right\}W \\ &\quad +(1-c)\left\{I - E_{q(y)}[\vartheta(y)y^T]\right\}W\Big] + o(1), \end{aligned} \tag{25}$$

and

$$\tilde{\Delta}_f W = -\rho_t \frac{\partial I_f}{\partial W} W^TW \tag{26}$$

Here, ρ_t is a small positive number called the learning rate. Thus, $0 < c \leq 1$ is a region for faster convergence with the ratio of

$$r = 1 + (\frac{1-c}{c})\frac{q}{p}. \tag{27}$$

Note that $c = 1$ is the case of the minimum mutual information ICA because of (10).

3.3. Utilization of Past and Future Information

Equation (25) still requires the unknown probability density function $q(y)$. Therefore, we need to give an interpretation of $q(y)$ in iterative updates. Since $p(y)$ is expected to converge to $q(y)$ as the matrix W is updated, we interpret $p(y)$ and $q(y)$ as follows.

1. [Use of the past information]

   For the current iteration index t, the interpretation is $p^{(t-\tau)}(y) := p(y)$ and $p^{(t)}(y) := q(y)$.

2. [Use of future estimation]

   In this case, the interpretation is $p^{(t)}(y) := p(y)$ and $p^{(t+\tau)}(y) := q(y)$.

Here, τ is a natural number.

3.4. Algorithms

The first version utilizes a set of past update information.

[Momentum f-ICA]

If we use $p(y)$ as $p^{(t-\tau)}(y)$ and $q(y)$ as $p^{(t)}(y)$ at the t-th iteration, then the sample-based learning is as follows.

$$\begin{aligned} &\tilde{\Delta}_f W(t) = \tilde{\Delta} W(t) + \mu_f \tilde{\Delta} W(t-\tau) \\ &= \rho_t\Big[\{I - \varphi(y(t))y(t)^T\}W(t) \\ &\quad +\mu_f\{I - \varphi(y(t-\tau))y(t-\tau)^T\}W(t-\tau)\Big] \end{aligned} \tag{28}$$

Here, $\mu_f = \frac{c}{1-c}$. Thus, we added a momentum term $\tilde{\Delta} W(t-\tau)$ weighted by μ_f. Note that the case of $\mu_f = \frac{1-\alpha}{1+\alpha}$ corresponds to the α-ICA [16]. Further special case of $\alpha = 1$, i.e., $\mu_f = 0$ is the plain minimum mutual information method of [2], [3].

The second version utilizes estimation of a future value.

[Turbo (Look-ahead) f-ICA]

$$\begin{aligned} &\tilde{\Delta}_f W(t) = \tilde{\Delta} W(t) + \nu_f \tilde{\Delta} W(t+\tau) \\ &= \rho_t\Big[\{I - \varphi(y(t))y(t)^T\}W(t) \\ &\quad +\nu_f(t)\{I - \varphi(\hat{y}(t+\tau))\hat{y}(t+\tau)^T\}\hat{W}(t+\tau)\Big] \end{aligned} \tag{29}$$

Here, $\nu_f = \frac{1}{\mu_f} = \frac{1-c}{c}$.

The look-ahead terms $\hat{W}(t+\tau)$ and $\hat{y}(t+\tau)$ are estimations of $W(t+\tau)$ and $y(t+\tau)$ using the usual log-version. Thus, we added a predicted term $\tilde{\Delta}\hat{W}(t+\tau)$ weighted by ν_f.

We give the following comments on the above two update methods.

1. Equation (28) is the result of a weighted superposition of convex functions: Positively weighted superposition of convex functions gives another convex function.

2. There is a duality between Equations (28) and (29).

3. $\tau = 1$ works effectively enough for both anticipatory and non-anticipatory methods.

4. On the use of the look-ahead method, a semi-batch mode is recommended to show the merit of speedup.

3.5. Partial Supervision

Because of the unknown permutation matrix Π, the resulting matrix W forces users to identify which source is which. This enhances undesirable off-line nature of the algorithm. Therefore, we consider injection of partially supervisory information so that the target information is recovered as the top source.

From Equation (20), the observed signal $x(n)$ is expressed by a mixture of $y(n)$ by

$$x(n) = W^{-1}y(n) \stackrel{\text{def}}{=} Uy(n). \tag{30}$$

Let

$$U \stackrel{\text{def}}{=} [u_1, \cdots, u_K]. \tag{31}$$

Here,

$$u_j = [u_{1j}, \cdots, u_{Kj}]^T. \tag{32}$$

Then,

$$x(n) = u_1 y_1(n) + \cdots + u_K y_K(n). \tag{33}$$

Thus, the vector u_k possesses the information on the mixture. Therefore, we consider to control the ordering of $\{u_k\}_{k=1}^K$ and each vector's components. Suppose we have a set of teacher signals or a target pattern, say $\hat{R}$. Then, this teacher information can be incorporated into the iterative minimization by adding a descent cost term obtained from

$$F(U, \hat{R}) = tr\{(\hat{R} - U)^T(\hat{R} - U)\}. \tag{34}$$

For this cost function, the gradient descent term is

$$\Delta U = \lambda(\hat{R} - U), \tag{35}$$

where λ is a small positive constant. If $\hat{R}$ is nonsingular, the following approximation can be used

$$\Delta U = \lambda\hat{R}\{I - (W\hat{R}^{-1})\} \approx \lambda\hat{R}(W\hat{R} - I). \tag{36}$$

Since we have to use the effect of ΔU with the increment ΔW, the following transformed version is used.

$$\Delta V = -W\{\Delta U\}W. \tag{37}$$

This equation comes from an expansion of an the update matrix U^{-1} [17], [18].

4. APPLICATIONS TO BRAIN MAP ESTIMATION

4.1. Assigned Task and Teacher Pattern

Experiments in this section is used to evaluate convergence speed and accuracy of extracted brain maps. An important feature here is that the test data is a real world one - - not a simulation.

As was explained in the previous section, the supervisory information is injected to the matrix U by specifying the task pattern $\hat{R}$. This supervision is column-wise. Let a column vector

$$\hat{a}_1 = \text{col}[0, 0, 0, 1, 1, 1, 0, 0, 0, \cdots, 1, 1, 1, 0, 0, 0], \tag{38}$$

be an on-off pattern of the assigned task to the subject. Then, we compute its power-matched version $\hat{r}_1$ where the column sum is zero and the variance is the same as u_1. Then, Δu_1 was computed by using Equation (33). Since the rest vectors $\{\hat{r}_j\}$, $j > 1$, are arbitrary, i.e., unsupervised, this freedom was interpreted as $\Delta u_j = 0$ for $j > 1$. Note that

(i) Selecting $k = 1$ is a process of finding an appropriate permutation.

(ii) The power matching reduces the amplitude's uncertainty in Λ.

4.2. Experiments

The presented algorithm was tested for visual area separation experiments on human brain fMRI data. Time and spatial axes are transposed so that independent areas are obtained [5]. Figure 1 illustrates the comparison of the convergence speed. This figure shows that

[Presented method with a constant learning rate]
$\vee$
[Presented method with
Hestenes-Stiefel type learning rate adjustment]
$\vee$
[Minimum mutual information method].

Figure 2 illustrates an obtained activation pattern. Because of the partially supervised learning, this pattern is obtained as the first column of the matrix U. Figure 3 is the extracted brain map which gives separation of V1 and V2 areas. This result is compatible with the one obtained from the t-test.

5. CONCLUDING REMARKS

In this paper, we discussed the utilization of the convex divergence for iterative optimization. Besides the theoretical interest as a generalization, there is a concrete merit of speedup of convergence in comparison with usual optimization of logarithmic information measures. In the problem

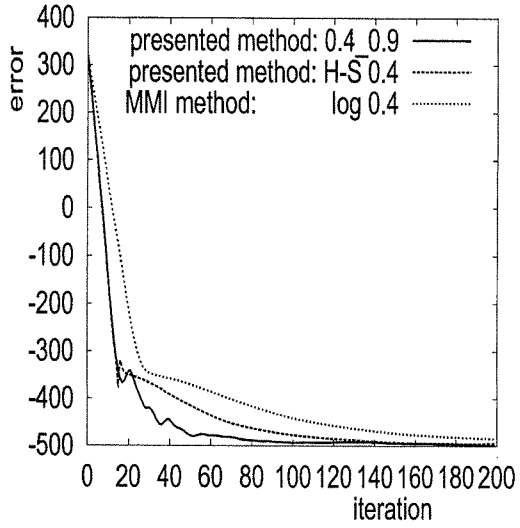

Figure 1: Speed comparison of the presented methods and the traditional method.

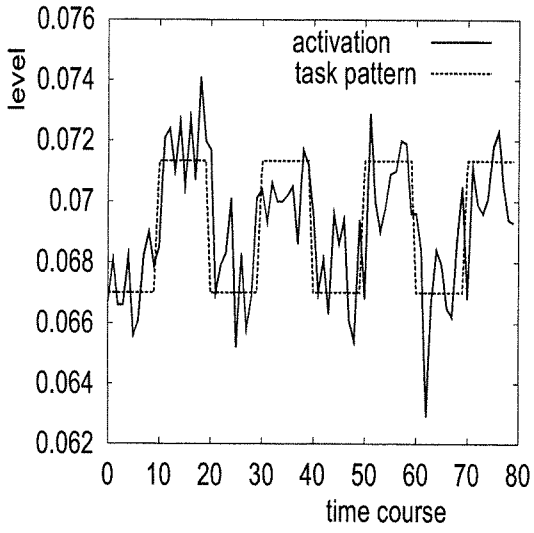

Figure 2: Estimated activation pattern u_1.

of ICA, the convex function $f(r)$ was directly used for the derivation of concrete algorithms. This is because, in the problem of ICA, the effect of the convex divergence is concentrated on the number c of Eq. (6). In a different class of convex divergence minimizations, however, a restriction on the function f is necessary so that a closed form update can be obtained. The α-EM algorithm is such a case, where the α-divergence is used. Interested readers are requested to refer to [11], [12].

ACKNOWLEDGMENT

The authors are grateful to Dr. K. Tanaka and Dr. R.A. Waggoner of RIKEN Brain science Institute for permitting them to try out the test data set. The authors are also grateful to Mr. Shuichiro Imahara (now, with Toshiba), for his discussions and experiments.

6. REFERENCES

[1] C. Jutten and J. Herault, Blind separation of sources, Part I: An adaptive algorithm based on neuromimetic architecture, Signal Processing, vol. 24, pp. 1-20, 1991.

[2] A.J. Bell and T.J. Sejnowski, An information-maximization approach to blind separation and blind deconvolution, Neural Computation, vol. 7, pp. 1129-1159, 1995.

[3] H.H. Yang and S. Amari, Adaptive online learning algorithm for blind separation: Maximum entropy and minimum mutual information, Neural Computation, vol. 9, pp. 1457-1482, 1997.

[4] I. Csiszár, Information-type measures of difference of probability distributions and indirect observations, Studia Sci. Math. Hungarica, vol. 2, pp. 299-318, 1967.

[5] M.J. McKeown, T-P. Jung, S. Makeig, G. Brown, S.S. Kindermann, T-W. Lee and T.J. Sejnowski, Spatially independent activity patterns in functional MRI data during the stroop color-naming task, Proc. National Academy of Sci. USA, vol. 95, pp. 803-810, 1998.

[6] Y. Matsuyama and S. Imahara, The α-ICA algorithm and brain map distillation from fMRI images, Proc. Int. Conf. on Neural Information Processing, vol. 2, pp. 708-713, 2000.

[7] A. Rényi, On measures of entropy and information, Proc. 4th Berkeley Symp. Math. Stat. and Pr., vol. 1, pp. 547-561, 1960.

[8] S. Amari, Differential geometry of statistics, Institute of Mathematical Statistics Lecture Notes, vol. 10, pp. 21-94, 1985.

[9] S. Amari and H. Nagaoka, *Methods of Information Geometry*, Iwanami, 1993 (Translation by D. Harada, AMS, 2000).

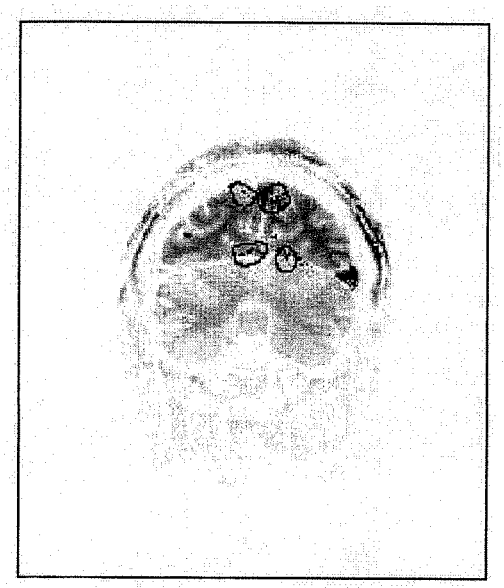

Figure 3: Obtained brain map.

[10] Y. Matsuyama, The α-EM algorithm: A block connectable generalized learning tool for neural networks, Lecture Notes in Computer Science, No. 1240, pp. 483-492, Berlin, Germany: Springer-Verlag, June, 1997.

[11] Y. Matsuyama, The α-EM algorithm and its basic properties, Transactions of Institute of Electronics, Information and Communications Engineers, vol. J82-D-I, pp. 1347-1358, 1999.

[12] Y. Matsuyama, The α-EM algorithm: Surrogate likelihood optimization using α-logarithmic information measures, IEEE Trans. on Information Theory, vol. 49, pp. x-y, 2003.

[13] M. Jamshidian and R.I. Jennrich, Conjugate gradient acceleration of the EM algorithm, J. ASA, vol. 88, pp. 221-228, 1993.

[14] J.-F. Cardoso and B.H. Laheld, Equivariant adaptive source separation, IEEE Trans. on SP, vol. 44, pp. 3017-3030, 1996.

[15] S. Amari, Natural gradient works efficiently in learning, Neural Computation, vol. 10, pp. 252-276, 1998.

[16] Y. Matsuyama, N. Katsumata, Y. Suzuki and S. Imahara, The α-ICA algorithm, Proc. Int. Workshop on Independent Component Analysis, pp. 297-302, 2000.

[17] Y. Matsuyama, S. Imahara and N. Katsumata, Optimization transfer for computational learning, Proc. Int. Joint Conf. on Neural Networks, vol. 3, pp. 1883-1888, 2002.

[18] Y. Matsuyama and R. Kawamura, Supervised map ICA: Applications to brain functional MRI, Proc. Int. Conf. on Neural Information Processing, vol. 5, pp. 2259-2263, 2002.

[19] T. Kailath, The divergence and Bhattacharyya distance measures in signal selection, IEEE Tranc. Comm., vol. COM-15, pp. 52-60, 1967.

[20] R. Beran, Minimum Hellinger distance estimates for parametric models, Annnals of Statistice, vol. 5, pp. 445-463, 1977.

[21] S. Amari, A. Cichocki and H.H. Yang, A new learning algorithm for blind signal separation, In: Advences in Neural Infromation Processing Systems, MIT Press, pp. 757-763, 1996.

ADDITIONAL REMARKS

Reviewers gave comments on this paper. The authors are quite thankful to them. Since the comments were diverse in their contents, the authors decided to summarize their replies here so that the page space can be effectively used.

1. "Speedup" in this paper is used to indicate the comparison of this paper's method and its subclass, the minimum mutual information.

2. The method of ICA is different from EM. The ICA is semi-parametric.

3. Experiments using simulated data are givn in [16]. It is observed that the speedup is the effect beyond the increase of the learning rate.

4. For the momentum ICA, $c = 0.7$ is a recommended rule-of-thumb. Note that $c = 0.5$, or $\alpha = 0$, is the case of the Bhattacharyya distance [19], and equivalently the Hellinger distance [20]. Thus, properties obtaind therein will be beneficial to readers.

5. The vertical axis of Figure 1 shows

$$D(W) + H(X) - \frac{n}{2}\log(2\pi e)$$

[21]. Therefore, the value can be a negative number.

6. Additive regularization term to the main function can be used in a wide variety of gradient-style ICA algorithms. It is necassary to decrease this effect as the iteration proceeds so that the independence of estimated components is the main target. The term used in this paper worked effectively beuause of its simplicity.

Independent Component Analysis Minimizing Convex Divergence

Yasuo Matsuyama, Naoto Katsumata, and Ryo Kawamura

Department of Computer Science, Waseda University,
Tokyo 169-8555, Japan
yasuo2@waseda.jp, {katsu,ryo}@wizard.elec.waseda.ac.jp

Abstract. A new class of learning algorithms for independent component analysis (ICA) is presented. Starting from theoretical discussions on convex divergence, this information measure is minimized to derive new ICA algorithms. Since the convex divergence includes logarithmic information measures as special cases, the presented method comprises faster algorithms than existing logarithmic ones. Another important feature of this paper's ICA algorithm is to accept supervisory information. This ability is utilized to reduce the permutation indeterminacy which is inherent in usual ICA. By this method, the most important activation pattern can be found as the top one. The total algorithm is tested through applications to brain map distillation from functional MRI data. The derived algorithm is faster than logarithmic ones with little additional memory requirement, and can find task related brain maps successfully via conventional personal computer.

1 Introduction

Optimization of information measures is a rich resource of learning algorithms. This is mainly because observed data are often probabilistic in nature. Independent component analysis (ICA) [1] is a typical case obtained from such optimization. Usually, the performance measure for the optimization is based upon logarithmic information measures [1], [2], [3]. But, there is a wider class of information measure called the convex divergence or the f-divergence [4][1].

Starting from discussions on the basic properties of the f-divergence, we derive a new class of ICA algorithms called the f-ICA by minimizing this information measure. Contribution of this paper can be previewed as follows.

(i) New properties of the f-divergence and related information measures are presented.
(ii) The f-ICA contains usual logarithmic ICA as a special case. Convergence speed is faster than the logarithmic one.
(iii) Obtained algorithms are modifiable to be partially supervised learning.
(iv) Corresponding software is executable on a personal computer. Applications to human brain map distillation from functional Magnetic Resonance Imaging (fMRI) are successfully made.

[1] Equation (4.20) of [5] is a forerunner of the f-divergence.

2 Convex Divergence and New Properties

2.1 Definition and Properties

Convex divergence is a measure of information which gives a directed distance between two probability densities p_ψ and p_φ by using an adjustable convexity. Here, ψ and φ are generic parameters. Let $f(r)$ be convex on $r \in (0, \infty)$, and let $g(r) \stackrel{\text{def}}{=} rf(1/r)$ be its dual convex function. Then, the convex divergence, or f-divergence, is defined as follows [4].

$$\begin{aligned} D_f(\psi\|\varphi) &\stackrel{\text{def}}{=} \int_{\mathcal{Y}} p_\varphi(\boldsymbol{y}) f(p_\psi(\boldsymbol{y})/p_\varphi(\boldsymbol{y}))d\boldsymbol{y} \\ &= \int_{\mathcal{Y}} p_\psi(\boldsymbol{y}) g(p_\varphi(\boldsymbol{y})/p_\psi(\boldsymbol{y}))d\boldsymbol{y} \stackrel{\text{def}}{=} D_g(\varphi\|\psi) \geq g(1) = f(1) \stackrel{\text{def}}{=} 0 \quad (1) \end{aligned}$$

We are interested in the case that $f(r)$ is twice continuously differentiable. This assumption makes it possible to discuss information matrices and gradient style learning. Differential properties are as follows.

$$D_f(\varphi\|\varphi) = D_g(\varphi\|\varphi) = 0 \quad (2)$$

$$\partial^{10} D_f(\varphi\|\varphi) = \partial^{10} D_g(\varphi\|\varphi) = 0 \quad (3)$$

$$\partial^{20} D_f(\varphi\|\varphi) = f''(1) F_Y(\varphi) = g''(1) F_Y(\varphi) = \partial^{20} D_g(\varphi\|\varphi) \quad (4)$$

Here, ∂^{ij} stands for i and j times partial differentiation with respect to ψ and φ, respectively. $F_Y(\varphi)$ is the Fisher information matrix.

Next, we define the following constant[2].

$$c \stackrel{\text{def}}{=} \frac{f''(1)}{f'(1)} = -\frac{g''(1)}{g'(1)} \in (-\infty, \infty). \quad (5)$$

Then, the following expansion holds around $r = 1$.

$$\frac{f(r)}{f''(1)} = \left(\frac{1}{c} r^c\right) \left\{ \frac{1}{1-c}(r^{1-c} - 1) \right\} + o(1) \quad (6)$$

$$\frac{g(r)}{g''(1)} = \left(\frac{-1}{c}\right) \left\{ \frac{1}{1-c}(r^{1-c} - 1) \right\} + o(1) \quad (7)$$

Here, $o(1)$ is the higher order term. From Equations (6) and (7), we find that

$$L^{(c)}(r) = \frac{1}{1-c}(r^{1-c} - 1) \quad (8)$$

is regarded as an extended class of logarithm. In fact, $L^{(1)}(r) = \log r$ in the limit. This "c-logarithm" has relationships to the Fisher information matrix and the Cramér-Rao bound. Let L_c be an abbreviated notation of $L^{(c)}(p_\varphi)$. Then, we have

$$M_Y^{(c)}(\varphi) \stackrel{\text{def}}{=} E_{p_\varphi}\left[c p_\varphi^{-2(1-c)} \left(\frac{\partial L_c}{\partial \varphi}\right) \left(\frac{\partial L_c}{\partial \varphi^T}\right) \right] = -E_{p_\varphi}\left[p_\varphi^{-(1-c)} \left(\frac{\partial^2 L_c}{\partial \varphi \partial \varphi^T}\right) \right]. \quad (9)$$

[2] If we add a set of assumptions that $f(xy) = kf(x)f(y)$ and $f''(1) = g''(1) = 1$, then the α-divergence [6], [7] is obtained. In this case, $c = \frac{1-\alpha}{2}$ holds. The symbol $o(1)$ in (6) and (7) becomes unnecessary.

The case of $c = 1$ is reduced to the Fisher information matrix $F_Y(\varphi)$:

$$M_Y^{(c)}(\varphi) = cM_Y^{(1)}(\varphi) = cF_Y(\varphi). \tag{10}$$

Because of Equations (10), the use of the information matrix $M_Y^{(c)}(\varphi)$ does not deteriorate the Cramér-Rao bound [8], [9], [10]. We assume that underlying problems are regular, so that $M^{(c)}(\varphi) > 0$, $F(\varphi) > 0$, and $c > 0$.

2.2 Optimization Transfer

Equations (4), (8), (9) and (10) mean that the f-divergence and the c-logarithm can be used as targets of optimizations instead of logarithmic information measures. That is, optimizations can be transferred to the f-divergence and/or to the c-logarithm [10], [11]. From the next section, independent component analysis is discussed through the minimization of the f-divergence between the observed joint probability density p and the independent probability density q.

3 The f-ICA Algorithm

3.1 Derivation of the Algorithm

In the problem of ICA, we are given a set of vector random variables.

$$\boldsymbol{x}(n) = [x_1(n), \cdots, x_K(n)]^T = A\boldsymbol{s}(n), \ (n = 1, \cdots, N). \tag{11}$$

Here, the matrix A and the vector

$$\boldsymbol{s}(n) = [s_1(n), \cdots, s_K(n)]^T \tag{12}$$

are all unknown but the following: (i) The components $s_i(n)$, $(i = 1, \cdots, K)$, are non-Gaussian except for at most one i. (ii) The components $s_i(n)$ and $s_j(n)$ are independent each other for $i \neq j$.

Under the above conditions, we want to estimate a demixing matrix

$$W = \Lambda\Pi A^{-1} \tag{13}$$

so that the components of

$$W\boldsymbol{x}(n) \stackrel{\text{def}}{=} \boldsymbol{y}(n) = [y_1(n), \cdots, y_K(n)]^T \tag{14}$$

are independent each other for every n. Here, Λ is a nonsingular diagonal matrix and Π is a permutation matrix, both of which are unknown too.

For the independent component analysis of this paper, we minimize the following f-divergence.

$$\begin{aligned} I_f(\bigwedge_{i=1}^K Y_i) &\stackrel{\text{def}}{=} D_f\left(p(y_1, \cdots, y_K) \| \prod_{i=1}^K q_i(y_i)\right) \\ &= D_g\left(\prod_{i=1}^K q_i(y_i) \| p(y_1, \cdots, y_K)\right) \stackrel{\text{def}}{=} I_g(\bigwedge_{i=1}^K Y_i) \end{aligned} \tag{15}$$

This quantity counts how the joint probability density $p(y_1, \cdots, y_K)$ is close to $\prod_{i=1}^{K} q_i(y_i)$. Traditional methods [1], [2], [3] minimize the mutual information or maximize the differential entropy, which corresponds to $c = 1$.

For the estimation of the demixing matrix W, we use a gradient descent. In this case, we obtain

$$-\nabla I_g(\bigwedge_{i=1}^{K} Y_i) \stackrel{\text{def}}{=} -\frac{\partial I_g(\bigwedge_{i=1}^{K} Y_i)}{\partial W} = \int_{\mathcal{X}} |W| q(\boldsymbol{y}) g'\left(\frac{|W| q(\boldsymbol{y})}{p(\boldsymbol{x})}\right) \left\{W^{-T} - \boldsymbol{\varphi}(\boldsymbol{y}) \boldsymbol{x}^T\right\} d\boldsymbol{x}. \tag{16}$$

Here,

$$\boldsymbol{\varphi}(\boldsymbol{y}) = [\varphi_1(y_1), \ldots, \varphi_K(y_K)]^T = -\left[\frac{q_1'(y_1)}{q_1(y_1)}, \ldots, \frac{q_K'(y_K)}{q_K(y_K)}\right]^T \tag{17}$$

is a nonlinear function assumed to be such as $\varphi_i(y) = y^3$ or $\tanh(y)$. For the natural gradient [12], [13], [14], we multiply cW^TW. Then, we have

$$\begin{aligned}
&-\tilde{\nabla} I_g(\bigwedge_{i=1}^{K} Y_i) \stackrel{\text{def}}{=} -\frac{\partial I_g(\bigwedge_{i=1}^{K} Y_i)}{\partial W}(cW^TW) \\
&= -c \int_{\mathcal{Y}} q(\boldsymbol{y}) g'\left(\frac{q(\boldsymbol{y})}{p(\boldsymbol{y})}\right) \left\{I - \boldsymbol{\varphi}(\boldsymbol{y}) \boldsymbol{y}^T\right\} d\boldsymbol{y}\, W \\
&= f''(1)\Big[c\left\{I - E_{p(\boldsymbol{y})}[\boldsymbol{\varphi}(\boldsymbol{y})\boldsymbol{y}^T]\right\} W + (1-c)\left\{I - E_{q(\boldsymbol{y})}[\boldsymbol{\varphi}(\boldsymbol{y})\boldsymbol{y}^T]\right\} W\Big] + o(1)
\end{aligned} \tag{18}$$

Here, the last equality is obtained by the expansion of $qg'(q/p)$ around $p \approx q$. Then, the update equation is

$$W(t+1) = W(t) + \tilde{\Delta}_g W(t), \tag{19}$$

with

$$\tilde{\Delta}_g W(t) = \rho(t) \left\{-\tilde{\nabla} I_g(\bigwedge_{i=1}^{K} Y_i)\right\}_{W=W(t)}. \tag{20}$$

Here, $\rho(t)$ is a small positive number called the learning rate. We call the learning algorithm (19) and (20) the f-ICA. Note that $c = 1$ is the case of the minimum mutual information ICA [2], [3]. The region $0 < c < 1$ gives faster convergence with the ratio of $1 + \frac{1-c}{c}\frac{q}{p}$.

3.2 Realization Using Past and Future Information

Equation (18) is a resource of f-ICA algorithms. The next important step is to find effective interpretations of this expression. Since we are given sample observations, the expectation $E_{p(\boldsymbol{y})}$ can be approximated by repeated applications of given data in either a batch or a successive style. But, the expectation $E_{q(\boldsymbol{y})}$ contains an unknown probability density function $q(\boldsymbol{y})$ because of the semi-parametric formulation of ICA. Since "$p(\boldsymbol{y}) \to q(\boldsymbol{y})$" holds as the update is repeated, $p(\boldsymbol{y})$ and $q(\boldsymbol{y})$ can be considered as two time-ordered states extracted from learning iterations [15]. Therefore, we can use a time-shifted version of p for the sake of unknown q. This interpretation leads to the following versions which are readily programmable for computer software.

[Momentum f-ICA]

If we use $p(\boldsymbol{y})$ as $p^{(t-\tau)}(\boldsymbol{y})$ and $q(\boldsymbol{y})$ as $p^{(t)}(\boldsymbol{y})$ at the t-th iteration, then the sample-based learning is realized as follows.

$$\tilde{\Delta}_g W(t) \stackrel{\text{def}}{=} \tilde{\Delta} W(t) + \mu_c \tilde{\Delta} W(t-\tau)$$
$$= \rho(t)\Big[\{I - \boldsymbol{\varphi}(\boldsymbol{y}(t))\boldsymbol{y}(t)^T\}W(t) + \mu_c\{I - \boldsymbol{\varphi}(\boldsymbol{y}(t-\tau))\boldsymbol{y}(t-\tau)^T\}W(t-\tau)\Big] \quad (21)$$

Here, $\mu_f = \frac{c}{1-c}$. Thus, we add a momentum term $\tilde{\Delta} W(t-\tau)$ weighted by μ_c.

[Turbo (Look-ahead) f-ICA]

If we use $p(\boldsymbol{y})$ as $p^{(t)}(\boldsymbol{y})$ and $q(\boldsymbol{y})$ as $p^{(t+\tau)}(\boldsymbol{y})$ at the t-th iteration, then the sample-based learning is realized as follows.

$$\tilde{\Delta}_g W(t) \stackrel{\text{def}}{=} \tilde{\Delta} W(t) + \nu_c \tilde{\Delta} \hat{W}(t+\tau)$$
$$= \rho(t)\Big[\{I - \boldsymbol{\varphi}(\boldsymbol{y}(t))\boldsymbol{y}(t)^T\}W(t) + \nu_c\{I - \boldsymbol{\varphi}(\boldsymbol{y}(t+\tau))\boldsymbol{y}(t+\tau)^T\}\hat{W}(t+\tau)\Big] \quad (22)$$

Here, $\nu_c = \frac{1}{\mu_c} = \frac{1-c}{c}$, and $\hat{W}(t+\tau)$ is a predicted future value.

3.3 Batch and Semi-Batch

Since we are given $\{\boldsymbol{x}(n)\}_{n=1}^N$ as a set of source vectors, the expectation $E_p[\cdot]$ is approximated by $\frac{1}{T}\sum_{i=1}^T[\cdot]$, where T is the number of samples in a selected window. The case of $T = N$ is the full batch mode. If we use $T < N$ as a window, it becomes a semi-batch mode. If $T = 1$, the case is an incremental learning. It is possible to choose a window size smaller than N for the look-ahead part so that the computation is alleviated. This style of semi-batch mode is recommended for the turbo f-ICA.

3.4 Partial Supervision

Because of the unknown permutation matrix Π, the resulting matrix W still requires users to identify which source is which. This aggravates undesirable off-line nature of the algorithm. Therefore, we consider to inject partially supervising data so that the target information is recovered as the top source.

From Equation (14), the observed signal $\boldsymbol{x}(n)$ is expressed as a transformation of $\boldsymbol{y}(n)$ by

$$\boldsymbol{x}(n) = W^{-1}\boldsymbol{y}(n) \stackrel{\text{def}}{=} U\boldsymbol{y}(n). \quad (23)$$

Let

$$U \stackrel{\text{def}}{=} [\boldsymbol{u}_1, \cdots, \boldsymbol{u}_K] \quad (24)$$

and

$$\boldsymbol{u}_j = [u_{1j}, \cdots, u_{Kj}]^T. \quad (25)$$

Then,

$$\boldsymbol{x}(n) = \boldsymbol{u}_1 y_1(n) + \cdots + \boldsymbol{u}_K y_K(n). \quad (26)$$

Thus, the vector $\{\boldsymbol{u}_j\}_{j=1}^K$ possesses the information on the mixture. Therefore, we consider to control the ordering of $\boldsymbol{u}_j$. Suppose we have a set of teacher signals or a target pattern, say $\bar{R}$. Then, this teacher signal can be incorporated into the iterative minimization [16]. The method is to add a descent cost term

$$F(U, \bar{R}) = tr\{(\bar{R} - U)^T(\bar{R} - U)\}. \tag{27}$$

For this cost function, the gradient descent term is

$$\Delta U = \gamma(\bar{R} - U), \tag{28}$$

where γ is a small positive constant. If $\bar{R}$ is nonsingular, the following approximation can be used

$$\Delta U = \gamma\bar{R}\{I - (W\bar{R}^{-1})\} \approx \gamma\bar{R}(W\bar{R} - I). \tag{29}$$

Since we have to use the effect of ΔU with the main increment $\tilde{\Delta}_g W$ of (20), the following transformed version is used.

$$\Delta V = -W\{\Delta U\}W. \tag{30}$$

This equation comes from an expansion of an the update matrix U^{-1} [11], [16], [17]. Since we applied the natural gradient to obtain the main update term (20), we need to use the same method to ΔV. But, the natural gradient in this case is the same as ΔV becase of the following equality:

$$\tilde{\Delta} V = -W\{\Delta U\}(U^T U)W(W^T W) = \Delta V. \tag{31}$$

4 Real-World Applications of the f-ICA: Brain Map Distillation

The purpose of this experiment is to find independent spatial patterns in the brain functional magnetic resonance imaging (fMRI) using a conventional personal computer. Since Equation (26) holds, we can regard each column vector of $U = W^{-1}$ as an activation pattern of separated brain maps [18]. The fMRI data are measured by assigning a series of "on-off" stimuli to a tested person.

Figure 1 illustrates convergence speed. The dotted line shows the speed of the usual logarithmic method (minimum mutual information with natural gradient). The solid line is the presented method using the momentum strategy with constant learning rates. Thus, the presented method in this paper is successful. Note that, placed between these two lines is the curve using Hestenes-Stiefel type learning rate adjustment. Because the true cost function is semi-parametric in ICA, time-dependent adjustment of the learning rate may not always be effective. Figure 2 is the extracted activation pattern (a time course) which corresponds to an assigned on-off task to a subject (a young male)[3]. This pattern is the top one, i.e., $\boldsymbol{u}_1$. The prior knowledge injection of Section 2.4 was so successful. Figure 3

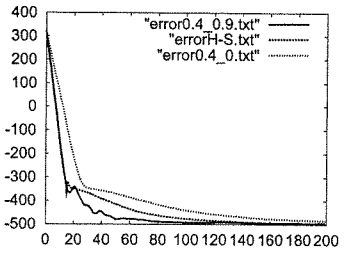

Figure 1. Learning speed.

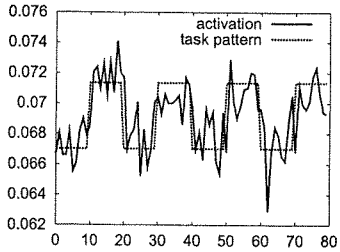

Figure 2. Corresponding activation.

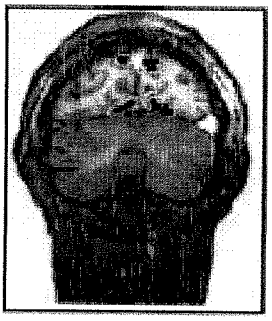

Figure 3. Separation of V1 and V2.

is the resulting brain map. This map clearly separates the edges of visual regions V1 and V2.

Experiments were executable by a conventional personal computer. This is due to the increased speed of the presented algorithm which exploits the second term of Equation (18). A usual memory size is sufficient since the presented algorithm requires very little memory increase.

5 Concluding Remarks

In this paper, the concept of the optimization transfer to an information measure which is more general than the logarithmic one was explained. This paper showed (i) basic properties of the f-divergence and related information measures, (ii) derivation of a general class of ICA algorithms based upon the convex divergence, (iii) reduction of indeterminacy in ICA by using a partially supervised strategy, (iv) applications to human brain's fMRI map distillation. It was shown that human brain's fMRI data can be handled by a conventional personal computer.

In this paper's ICA, the transferred optimization was the *minimization* of the convex divergence. There is an important relative to the optimization transfer. It is the alpha-EM algorithm. In that case, the likelihood ratio of Equation (8) is

[3] The authors are very grateful to Dr. Keiji Tanaka and Dr. R. Allen Waggoner of RIKEN BRI for permitting them to try out their data set.

maximized. This method contains the traditional log-EM algorithm as its special case. Interested readers are requested to refer to [8], [9], [10].

References

1. C. Jutten and J. Herault, Blind separation of sources, Part I: An adaptive algorithm based on neuromimetic architecture, Signal Processing, vol. 24, pp. 1-20, 1991.
2. A.J. Bell and T.J. Sejnowski, An information-maximization approach to blind separation and blind deconvolution, Neural Computation, vol. 7, pp. 1129-1159, 1995.
3. H.H. Yang and S. Amari, Adaptive online learning algorithm for blind separation: Maximum entropy and minimum mutual information, Neural Computation, vol. 9, pp. 1457-1482, 1997.
4. I. Csiszár, Information-type measures of difference of probability distributions and indirect observations, Studia Sci. Math. Hungarica, vol. 2, pp. 299-318, 1967.
5. A. Rényi, On measures of entropy and information, Proc. 4th Berkeley Symp. Math. Stat. and Pr., vol. 1, pp. 547-561, 1960.
6. S. Amari, Differential geometry of statistics, Institute of Mathematical Statistics Lecture Notes, vol. 10, pp. 21-94, 1985.
7. S. Amari and H. Nagaoka, *Methods of Information Geometry*, Iwanami, 1993 (Translation by D. Harada, AMS, 2000).
8. Y. Matsuyama, The α-EM algorithm: A block connectable generalized learning tool for neural networks, Lecture Notes in Computer Science, No. 1240, pp. 483-492, Berlin, Germany: Springer-Verlag, June, 1997.
9. Y. Matsuyama, The α-EM algorithm and its basic properties, Transactions of Institute of Electronics, Information and Communications Engineers, vol. J82-D-I, pp. 1347-1358, 1999.
10. Y. Matsuyama, The α-EM algorithm: Surrogate likelihood optimization using α-logarithmic information measures, IEEE Trans. on Information Theory, vol. 49, no. 3, pp. 692-706, 2003.
11. Y. Matsuyama, S. Imahara and N. Katsumata, Optimization transfer for computational learning, Proc. Int. Joint Conf. on Neural Networks, vol. 3, pp. 1883-1888, 2002.
12. M. Jamshidian and R.I. Jennrich, Conjugate gradient acceleration of the EM algorithm, J. ASA, vol. 88, pp. 221-228, 1993.
13. J.-F. Cardoso and B.H. Laheld, Equivariant adaptive source separation, IEEE Trans. on SP, vol. 44, pp. 3017-3030, 1996.
14. S. Amari, Natural gradient works efficiently in learning, Neural Computation, vol. 10, pp. 252-276, 1998.
15. Y. Matsuyama, N. Katsumata, Y. Suzuki and S. Imahara, The α-ICA algorithm, Proc. Int. Workshop on Independent Component Analysis, pp. 297-302, 2000.
16. Y. Matsuyama and S. Imahara, The α-ICA algorithm and brain map distillation from fMRI images, Proc. Int. Conf. on Neural Information Processing, vol. 2, pp. 708-713, 2000.
17. Y. Matsuyama and R. Kawamura, Supervised map ICA: Applications to brain functional MRI, Proc. Int. Conf. on Neural Information Processing, vol. 5, pp. 2259-2263, 2002.
18. M.J. McKeown, T-P. Jung, S. Makeig, G. Brown, S.S. Kindermann, T-W. Lee and T.J. Sejnowski, Spatially independent activity patterns in functional MRI data during the stroop color-naming task, Proc. National Academy of Sci. USA, vol. 95, pp. 803-810, 1998.

Iterative Optimization of Convex Divergence: Applications to Independent Component Analysis

Yasuo Matsuyama[1]
Department of Computer Science,
Waseda University,
Tokyo 169-8555, Japan
e-mail: yasuo2@waseda.jp

Abstract — **Iterative optimization of convex divergence is discussed. The convex divergence is used as a measure of independence for ICA algorithms. An additional method to incorporate supervisory information to reduce the ICA's permutation indeterminacy is also given. Speed of the algorithm is examined using a set of simulated data and brain fMRI data.**

I. Introduction

Computing and utilizing information measures have been placed at the center of information theory. In this paper, such a measure is the convex divergence. Following a list of new basic properties of the convex divergence, this measure is iteratively minimized for Independent Component Analysis (ICA). The speed is faster than logarithmic methods. Flexibility to accept supervisory information is maintained. The obtained algorithms are tested using both simulated data and brain functional Magnetic Resonance Imaging data (fMRI).

II. Convex Divergence, Information Matrix and Cramér-Rao Bound

Let ψ and φ be generic parameters for probability densities. The convex divergence, or f-divergence [1], between p_ψ and p_φ is defined by the following equation.

$$D_f(\psi\|\varphi) = \int_{\mathcal{Y}} p_\varphi(y) f(p_\psi(y)/p_\varphi(y))dy \geq f(1) \stackrel{\text{def}}{=} 0 \quad (1)$$

The function $f(r)$ is convex for $r \in (0,\infty)$. For $g(r) = rf(1/r)$, the duality $D_f(\psi\|\varphi) = D_g(\varphi\|\psi)$ holds. We consider the case that $f(r)$ is twice continuously differentiable. Then, the following differential equalities hold:

$$\partial^{10} D_f(\varphi\|\varphi) = 0, \quad \partial^{20} D_f(\varphi\|\varphi) = f''(1) F_Y(\varphi). \quad (2)$$

$F_Y(\varphi)$ is the Fisher information matrix assumed to be positive. Next, define $c \stackrel{\text{def}}{=} f''(1)/f'(1) = -g''(1)/g'(1) \in \mathbb{R}$. Then, the following expansions are obtained:

$$f''(r)/f'(1) = (r - r^c)/\{c(1-c)\} + o(1), \quad (3)$$

$$g''(r)/g'(1) = -(r^{1-c} - 1)/\{c(1-c)\} + o(1). \quad (4)$$

Equations (3) and (4) indicate that $L^{(c)}(r) = (r^{1-c}-1)/(1-c)$, with $L^{(1)}(r) = \log r$, can be regarded as an extended class of the logarithm. This is called the c-logarithm. Then, the following equality can be obtained:

$$M^{(c)}(\varphi) \stackrel{\text{def}}{=} E_p\left[c p^{-2(1-c)} \left(\frac{\partial L_c}{\partial \varphi}\right)\left(\frac{\partial L_c}{\partial \varphi^T}\right)\right]$$
$$= -E_p\left[p^{-(1-c)}\left(\frac{\partial^2 L_c}{\partial\varphi\partial\varphi^T}\right)\right] = cF_Y(\varphi). \quad (5)$$

Because of Equation (5), the Cramér-Rao bound is independent of c. Therefore, the convex divergence can be used in estimation problems without sacrificing the performance.

[1]This work was supported by the Productive ICT Academia Program in the 21st Century COE Programs, and by the Grant-in-Aid for Scientific Research.

III. Independent Component Analysis

In the ICA, observed vector data is assumed to satisfy

$$\boldsymbol{x}(n) = [x_1(n), \cdots, x_K(n)]^T = A\boldsymbol{s}(n), \quad (n = 1, \cdots, N). \quad (6)$$

The mixing matrix A and the source vector $\boldsymbol{s}(n)$ are unknown except for the following: (i) *The components $s_i(n)$ and $s_j(n)$ are independent each other for $i \neq j$.* (ii) *The unknown components $s_i(n)$, $(i = 1, \cdots, K)$, are non-Gaussian except for at most one specific i.* Therefore, we want to find a de-mixing matrix $W = \Lambda\Pi A^{-1}$ so that the components of $W\boldsymbol{x}(n) \stackrel{\text{def}}{=} \boldsymbol{y}(n)$ are independent each other for every n. Here, the nonsingular diagonal matrix Λ and the permutation matrix Π are also unknown. The de-mixing matrix W is iteratively estimated by adding a decent cost fraction of

$$-\{\partial D_g(\textstyle\prod_{i=1}^K q_i(y_i)\|p(y_1, \cdots, y_K)/\partial W\}(cW^TW)$$
$$= f''(1)\left[c\{I - E_{p(\boldsymbol{y})}[\boldsymbol{\vartheta}(\boldsymbol{y})\boldsymbol{y}^T]\}W\right.$$
$$\left. +(1-c)\{I - E_{q(\boldsymbol{y})}[\boldsymbol{\vartheta}(\boldsymbol{y})\boldsymbol{y}^T]\}W\right] + o(1). \quad (7)$$

Here, $\vartheta_i(r)$ is assumed to be such as r^3 or $\tanh(y)$. Equation (7) can be realized as (i) *utilization of the past update (momentum ICA)*, and/or (ii) *utilization of the prediction (look-ahead ICA)*. Plain logarithmic methods [2] do not have the property (i) nor (ii). Figure 1 shows momentum ICA's speed and the resulting brain map separating V1 and V2 areas. In this experiment, an additive regularization term which reflects designed time course is incorporated [3].

IV. Concluding Remarks

Transferring optimization to the divergence gives effective tools. Applications to the EM algorithm can be found in [4].

References

[1] I. Csiszár, "Information-type measures of difference of probability distributions and indirect observations," *Studia Sci. Math. Hungarica*, vol. 2, pp. 299-318, 1967.

[2] H.H. Yang and S. Amari, "Adaptive online learning algorithm for blind separation: Maximum entropy and minimum mutual information," *Neural Computation*, vol. 9, pp. 1457-1482, 1997.

[3] Y. Matsuyama, N. Katsumata and R. Kawamura, "Independent component analysis minimizing convex divergence," *Lecture Notes in CS*, Springer-Verlag, 2003.

[4] Y. Matsuyama, "The α-EM algorithm: Surrogate likelihood optimization using α-logarithmic information measures," *IEEE Trans. IT*, Vol. 49, No. 3, 2003.

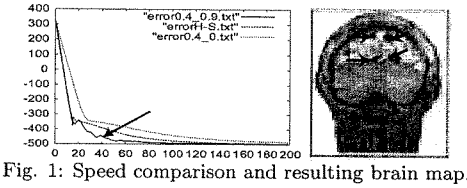

Fig. 1: Speed comparison and resulting brain map.

ネットワーク環境の構築に関する研究成果

Agent Generation and Resource Allocation in a Network Computing Environment

N. Nishioka, Y. Matsuyama, A. Saitoh, Y. Morita, N. Katsumata,
H. Kataoka, R. Mizuta, and S. Yoshika

Department of Computer Science, Waseda University
Tokyo, 169-8555 Japan
{norio-n@ruri., yasuo2@, a_psytoo@suou., morita@moegi., katsu@ruri.,
pp_kataoka@moegi., m-rky.zzr2663@ruri., mizzle@ruri.}waseda.jp

Abstract

Generation and allocation of mobile agents that can convey various contents are discussed. Mobile agents can move around the networked computing environment autonomously. Therefore, by plain strategies, traditional agents often concentrate themselves in a higher performance computer. This creates undesirable monopoly or oligopoly by a few computers which leads to degradation of the total network performance. This paper presents a method to circumvent such concentrations. The presented system realizes a novel agent allocation mechanism over the network where each agent can migrate within the network in a performance-increasing manner. This system is built on the middleware called FINALE (Framework for Intelligent Network Agents Looking at their Environment). The FINALE itself is improved so that the total network watch becomes possible. Experiments measuring the total network performance show that the presented agent allocation method is very effective and close to the omniscient situation.

1. Introduction

Recent progress in networking technologies creates a variety of hardware for the network. In spite of limited computational resources, terminal machines for the network, such as cellular phones and PDA's, are versatile in their own sophistications. Then, networked computers need to watch out and control dispatched tasks so that the total network performance is free from undesirable concentrations on specific sub-networks. This is because each computer may have a different computing resource in the network. Therefore, it is necessary for the network computing system to have a resource allocation mechanism for the high performance operation. We regard that this mechanism is desirable if

(i) the allocation reflects the total network performance, and

(ii) terminal users need not consider the network in detail.

Considering these trends, we select the mobile agent strategy and present a novel resource allocation method for such a network.

Mobile agents (often called agents in the text unless confusion exists) can move around the network. This is a fascination property. But, naive strategies may cause undesirable concentration on a few specific computers. Therefore, we provide a middleware for the agents migration, which is called FINALE (Framework for Intelligent Network Agents Looking at their Environment) [1] written in Java. In this paper, the FINALE itself is extended on the agent migration control.

Following this section, this paper is organized as follows. In Section 2, mobile agents and the resource allocation are discussed. Explanations on two types of administrations are given. In Section 3, the structure and ability of the basic FINALE are explained. In Section 4, agents scheduling methods are explained. The original FINALE is extended here so that it matches to the controlled mobile agent migration. Section 5 shows experiments on performance evaluation. Therein, performance near to the omniscience knowledge is reported. Concluding remarks are given in Section 6 including further sophistications.

2. Mobile Agents and Network Resources

One of the main purposes of this paper is to give an effective framework contributing to the ubiquitous computing. The importance is to find a systems technology to utilize the network's computing resources as effectively as possible. There can be a variety of choices on basic strategies. We choose the mobile agent technology since it can support various contemporary information terminals such as cellular phones and PDA's beside traditional computers [2].

2.1. Mobile Agent Technology

The mobile agent technology applied to the network computing is promising due to the possibility to move around and to add various features to agents [3]. This can be easily admitted from theoretical or abstract thinking at the idea level. But, the importance exists how to devise this mechanism. Before giving detailed explanation on this paper's contributions, we list up two salient features of the mobile agents. The mobile agent technology is different from other realizations as follows.

1. **Mobility**:

   Each agent can move in the network. There are two types of agents:

   (i) an agent which can move with its class file [4], [5], and

   (ii) an agent whose class file need to be left at its original environment [6].

   There are another classification on agents:

   (a) an agent which can carry its current status at its migration [4], [7], and

   (b) an agent which need to restart when it migrates [5].

   Our agents are of the type (i)(b). The reason that our type is not (i)(a) is that we are interested in the resource allocation. In addition to this, once our agent finds its environment, the execution can start. In this paper, Java class files will be used.

2. **Flexibility**:

   Besides the class file, an additional set of meta information can be attached to each agent when it migrates. Such a set of meta information can be utilized on the administration of the agent migration and the parameter adjustment.

2.2. Optimization of Resource Allocation to Mobile Agents

If agents could move to anywhere at their uncontrolled *will*, agents would select possibly one computer whose original or starting resources are affluent. Provided that one computer were much more powerful than others, this could be allowed. But, this still wastes other computers' resources. Computer networks holding only a few dominant machines are unrealistic these days, however, oligopoly may still emerge. Therefore, we have to find a mechanism to avoid such concentrations of agents.

The concentration may occur if agents can move without any guidance or with a static or an old one. Thus, it is desirable for agents to have the latest information for its migration and resource acquisition. Figure1 illustrates what should be considered for the

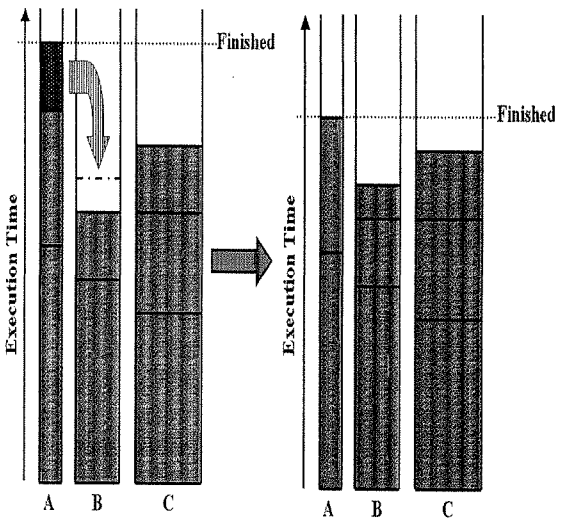

Figure 1: Resource allocation

optimization of resources. The width of machines A, B, and C represents conceptual computational power and resources. If agents in the machine B are completed and yet the rest agents run in machines A and C, the CPU of the machine B becomes idle. Therefore, we want to devise a mechanism to migrates agents from the machine A as is expressed by a thick arrow. The right side of Figure 1 is a resulting allocation while the networked system is working. Here, we would like to notice readers in advance that this figure still conveys static aspect since actual mechanism in later sections provides queues for migrating agents.

2.3. Scheduling of Mobile Agents

Scheduling considered in this paper stands for the direction for agents to move in the network. The network here is *closed* in the sense that agents can move to computers with access rights. LAN is such an example. The network size is independent of this definition.

In order to make each computer's completion time even, there are two types of scheduling:

(a) Scheduling of agents' migrations between computers.

(b) Scheduling of agents' execution order in one computer.

For Item (a), we introduce the concept of the task size (TaskSize). TaskSize is a measure which reflects the heaviness of each agent's task. Thus, the FINALE decides the destination of the migration. Then, the destination computer decides the ordering with the migration information and its own environment.

2.3.1. Destination Decision

Concerning to Item (a), the network system computes the expected completion time (ECT), and then

decides the destination machine. The ECT depends on each computer.

$$\text{ECT [ms]} = \text{TaskSize}/k_j \text{ [ms]} \quad (1)$$

Here, the number k_j stands for the j-th computer's ability. A faster machine with a higher clock frequency has a proportionally larger value.

2.3.2. Execution Order

After the migration of an agent, each computer decides which agents should be processed. The basic strategy is the smallest-task-size-first strategy. That is, an agent with the smallest TaskSize is processed with the highest priority. But, the preemptive control may occur according to the information given at the migration. This part is explained in Sections 3 and 4.

3. FINALE

3.1. Structure of the FINALE

It is necessary to prepare a middleware which gives an environment to support agents' generation, migration and execution. FINALE (Framework for Intelligent Network Agents Looking at their Environment) [1] written in Java is a good candidate for the resource allocation. Figure 2 shows the basic configuration of the FINALE.

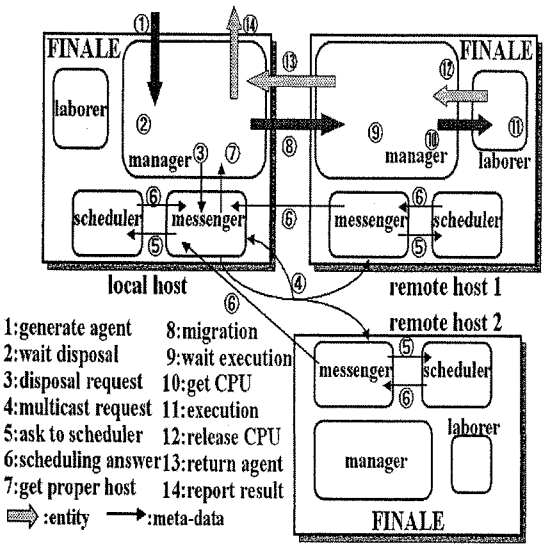

Figure 2: Basic configuration of the FINALE

There are four core parts in each FINALE unit. Their roles are as follows.

1. **Manager** administrates generation and migration of agents. Especially, the manager controls allocation of agents.
2. **Messenger** is used for communications between FINALE units. Note that the control of agents is undertaken by the manager.
3. **Laborer** gives CPU rights to agents with or without the computer priority. So far, each processor in one FINALE unit accepts one agent for execution.
4. **Scheduler** decides the scheduling based on mutual queries between computers.

3.2. Activities in FINALE

The process transition in the FINALE is as follows:

[generation] → [ready] →
[migration (when necessary)] → [ready] →
[run (if migration occurs)] → [report]

"Generation," "Migration," "Execution," "Report," are explained in more detail as follows.

1. **Generation**:

   This is a process of turning a Java class file to an agent via the FINALE. Meta information containing

   (a) the generated time,
   (b) the generated position (a machine),
   (c) the task size,
   (d) the deadline,
   (e) the file information

   are needed for execution. Here, the deadline is a maximum duration for the preemption.

2. **Migration**:

   When an agent becomes running, it may look for an executable machine. Any destination of the agent is not pre-assigned, but is found by the network scheduling algorithm. Agents are serialized before they migrate. Since each agent possesses a class file and meta information, it becomes active in the new environment. Such dynamic loading is possible because of Java's ability. Figure 3 illustrates the migration from a computer to another.

3. **Execution**:

   When an agent migrates to the destination, it is placed in the target machine's queue. The position in this queue need not be at the end, but can be in its middle by reflecting TaskSize. The execution occurs by its class method's invocation.

4. **Report**:

   When an agent completes its execution, this status is reported to the computer which generated this agent.

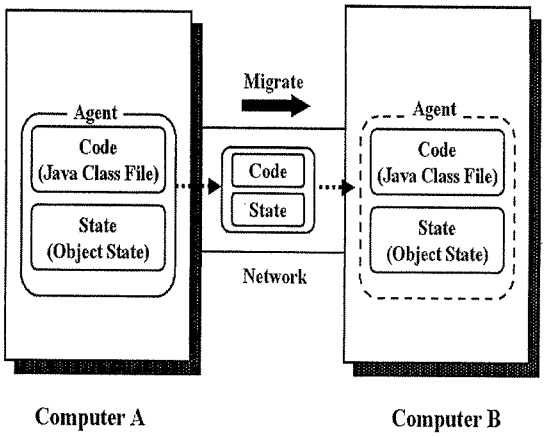

Figure 3: Migration of mobile agents

4. Process Scheduling over the Network

The network computing via agents requires an integrated scheduling both in one computer and inter-computers. We explain such a total scheduling methods step by step.

4.1. Default Model (Level 1)

The default model stands for a naive system which assigns an equal number of agents to each machine according to their generated order. The default model uses the original FINALE which was explained in Section 3. Note that this basic system will be extended in Sections 4.2 and 4.3, which is this paper's main contribution.

Figure 4 illustrates such an unsophisticated scheduling. Since this assignment does not reflect

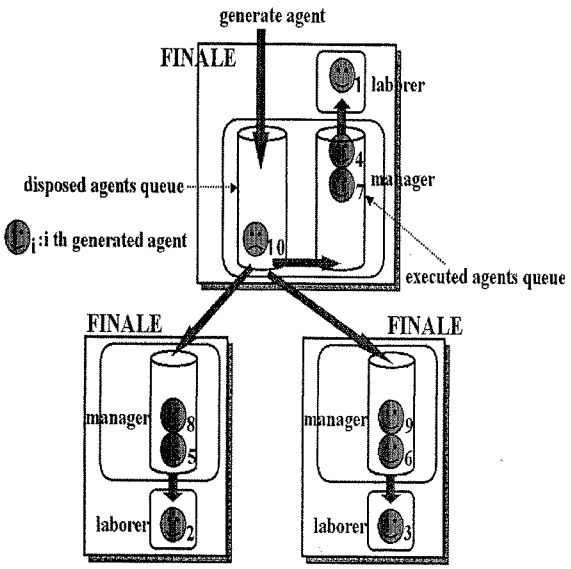

Figure 4: Default model

each computer's CPU power nor any resources, completion times of agents become very uneven.

4.2. Information Administration Server (Level 2)

The next step is to relocate agents, i.e., to cause forced migration. Repeated migrations of agents may degrade the network performance because of excessive communications. Therefore, at Level 2, we design a mechanism which does not transfer agents but only their essential information. For this purpose, the FINALE was extended so that it can behave as an administration server for this information. The

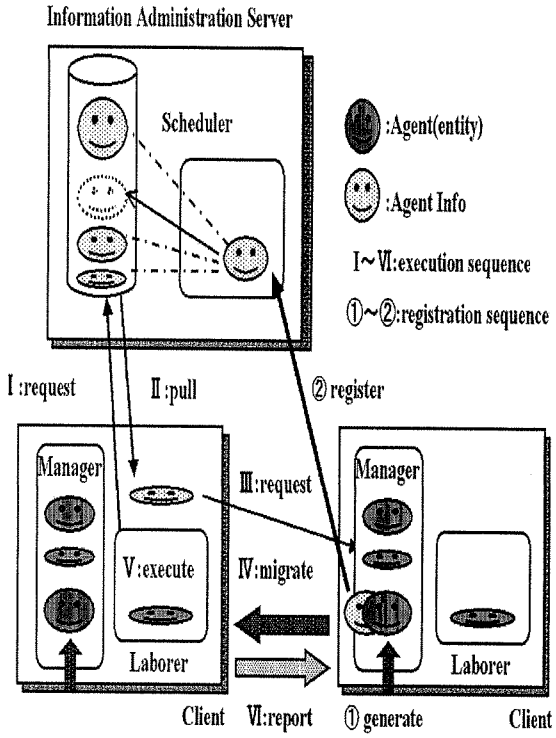

Figure 5: Information administration server

administration server has a queue for generated information corresponding to each agent. This information contains the name and the generated position of the agent. There is another queue which contains only over-deadline agents' information. Figure 5 illustrates this mechanism. Such a mechanism can improve Level 1, however, it does not reflect each computer's ability yet.

4.3. Inside Scheduling (Level 3)

Computers in the network are usually different in their CPU powers. Therefore, powerful computers can accept many agents. Inferior computers can accept few. Yet, inferior computers need to be utilized as members of the network. Therefore, Level 3 is provided to realize the mechanism which can adjust termination times of computers as even as possible. Figure 6 illustrates this machanism of Level 3.

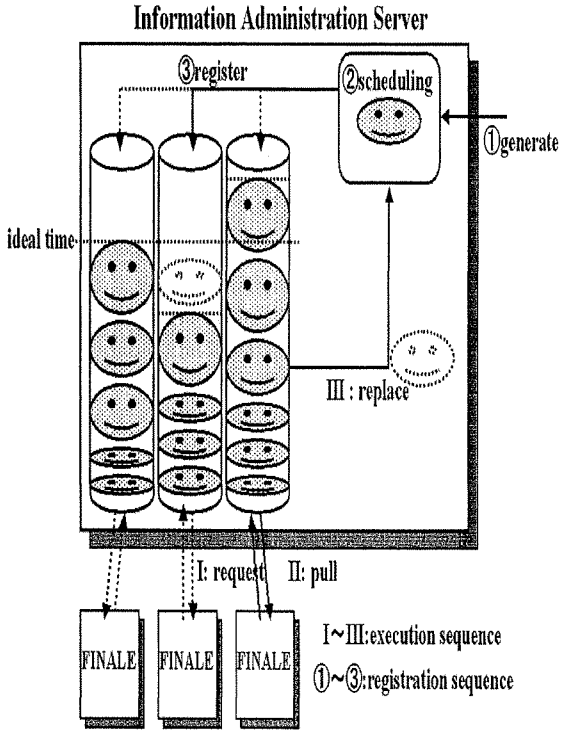

Figure 6: Scheduling via server

At Level 3, we prepare an information queue for each computer. This mechanism is heavier than that of Level 2, however better performance can be obtained. This is because each agent's information is stored in a selected queue by predicting termination time.

One may have a feeling that the environment for each agent varies with time. Pre-assignment or prediction may fail. Therefore, Level 3 contains an information feedback mechanism. Each agent's actual termination time is reported so that the selection of the best queue can reflect actual results. If a big discrepancy is detected between actual and previous scheduling, corresponding agent's information is exchanged for the dynamic model improvement.

5. Performance Measurement

In this section, we present experimental results on the performances of the Levels 1, 2, and 3, as well as the omniscient performance. Here, "omniscient" means that perfect oracle with no misplacement nor division into agents. This ideal situation can never be attained in agent computing, so that our purpose is how to make our system performance as close as possible. Measurements are performed in the following way.

In our experiments, we checked to see the following.

(i) How the presented methods can perform in the sense of the execution time.

(ii) How the latency time is increased or decreased.

The agent set is generated by a benchmarking determinant computation. Table 1 describes a set of resulting performances. Figures 7 and 8 are their graphical illustrations. X-time in Table 1 stands for the execution time, and L-time means the latency time. OMNI stands for the omniscience which can never be achieved by human knowledge. The X-time is normalized by OMNI=100 which is also a normalized value of the JVM execution time without agents. Table 1 tells the following.

Table 1: Execution time and latency time

strategy	X-Time	L-Time	presenter
Level 1	249.4	51893.3	anybody
Level 2	110.4	21572.7	this paper
Level 3	108.0	24799.7	this paper
OMNI	100.0	-	the omniscient

(i) The execution times of Levels 2 and 3 are much better than Level 1. This is because of the existence of the information server which enables each computer to possess its execution status. Level 3 is better than Levels 1 and 2, and is nearer to the unrealizable omniscient performance. This is due to a kind of *conscience mechanism* for arranging the termination time by each machine.

(ii) On the latency time, Levels 2 and 3 are similar. But, Level 2 is better than Level 3. This is natural since Level 2 issues via the small-first principle which has low overhead. On the other hand, Level 3 reallocates agents according to their information.

(iii) It is important emphasize that X-time includes L-time for all agents in each machine. Therefore, X-time is authoritative on measuring the system performance.

6. Concluding Remarks

The main purpose of this paper was to show that resources in the computer network can fully utilized via mobile agent strategy. It is expected that the mobile agent method can contribute to the realization of ubiquitous computing. Therefore, our study was focused on the network where mobile agents exist. In this paper, we developed network adiministration mechanisms where total system can process agents in a shorter time. It is important to note here that each machine are heterogeneous in its power. We started from the basic FINALE.

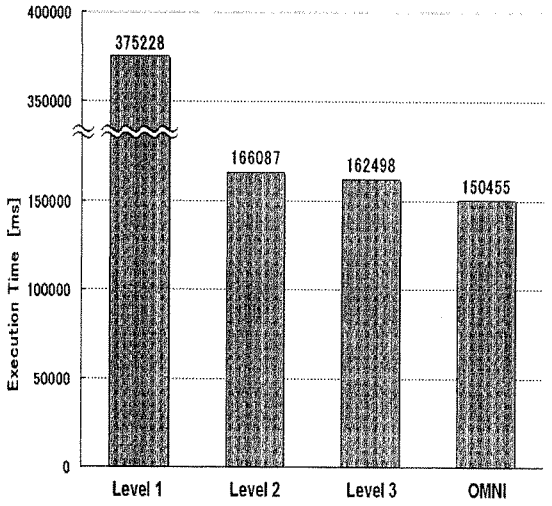

Figure 7: Execution time of each strategy

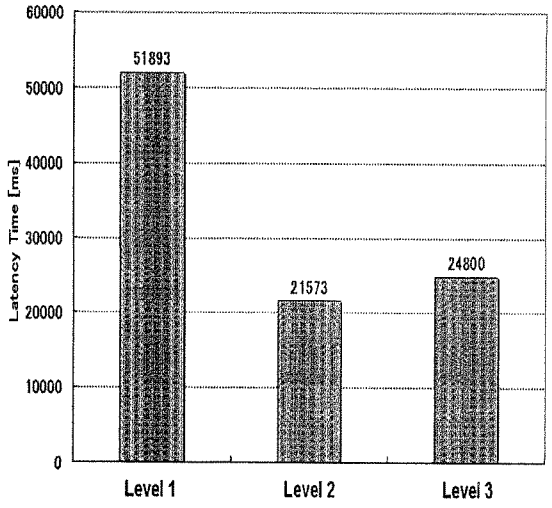

Figure 8: Latency time of each strategy

The FINALE-based systems presented in this paper showed the performances summarized in Table 1. This is a creditable result of mobile-agent-based systems. Besides the speed, any user can exploit our system only by understanding the communication mechanism. Therefore, developers of application software can use our system without knowing the inside of the network. They need to generate Java programs with usual class files.

Our system has further possibility of improvements. By listing up such possibility, we complete this paper.

(a) When we turn one big Java program into a set of mobile agents, task sizes of resulting agents can not be predicted with a good guess in advance. It is desirable to have a good prediction method on this information.

(b) Methods in this paper do not have its own security mechanism. Although this theme is not the main focus of this paper, it is worthy to pursue towards applications with security demands.

Acknowledgment

This study was supported by the Grant-in-Aid for Scientific Research #15300077 and by the Productive ICT Academia of the 21st Century COE Program granted to Waseda University. The authors are grateful to Mr. Y. Aoki of SONY for his early contributions.

References

[1] Y. Aoki, "Construction of the FINALE with a Three-Dimensional Library for Distributed Cooperative Application" Master's Thesis, Department of Electrical, Electronics and Computer Engineering, Waseda University, 2002.

[2] K. Hiroshige, K. Kawakami, H. Sasaki, Y. Okataku, and S. Honiden, "Agent Migration Control for Mobile Environment," *IPSJ SIG Notes on Mobile Computing and Wireless Communications*, 2001.

[3] A. Fuggetta, G.P. Picco, G. Vigna, Understanding Code Mobility, *IEEE Trans. Soft. Eng.*, Vol. 24, No. 5, pp. 342-361, 1998.

[4] B.D. Lange and M. Oshima, "Programming and Deploying Java Mobile Agents with Aglets," Addison-Wesley, 1998.

[5] I. Satoh, "A Mobile Agent-Based Framework for Active Networks," *Proceedings of IEEE Systems, Man, and Cybernetics Conference (SMC'99)*, pp. 71-76, IEEE, 1999.

[6] T. Kawamura, T. Hasegawa, A. Ohsuga, S. Honiden, "Bee-gent: Bonding and Encapsulation Enhancement Agent Framework for Development of Distributed Systems," *Systems and Computers in Japan*, John Wiley & Sons, Inc., Vol. 31, No. 13, pp. 42-56, 2000.

[7] E. Truyen, B. Robben, B. Vanhaute, T. Coninx, W. Joosen and P. Verbaeten, "Portable Support for Transparent Thread Migration in Java," Lecture Notes in Computer Science, Springer-Verlag, September, 2000.

Middleware Design Issues for Ubiquitous Computing

Tatsuo Nakajima, Kaori Fujinami, Eiji Tokunaga, Hiroo Ishikawa
Department of Computer Science
Waseda University
tatsuo@dcl.info.waseda.ac.jp

ABSTRACT

Our daily lives will be dramatically changed by embedded small computers in our environments. The environments are called *ubiquitous computing environments*. To realize the environments, it is important to reduce the cost to develop ubiquitous computing applications by encapsulating complex issues in middleware infrastructures that are shared by various applications.

In this paper, we describe three middleware infrastructures for supporting ubiquitous computing, that have developed in our projects. Our infrastructures have tried to hide some complexities to make it easy to develop ubiquitous computing applications in an easy way. We also show some lessons learned in our projects.

Keywords

Ubiquitous Computing, Middleware Design

1. INTRODUCTION

Our daily lives become more and more complex every day. Information technologies have been increasing these complexities, because a large proportion of our daily lives is currently spent in analyzing various sorts of information. Ironically, present ubiquitous computing technologies will increase the amount of such information dramatically, and increase complexities in our daily lives. A variety of appliances surrounding us rapidly become commodities. Today, it is very difficult to create an appliance that offers special, distinctive features. For example, we cannot distinguish among different vendor's televisions. Therefore, it is important to take into account pleasurable experiences when a user uses the appliances[14].

These devices and appliances should be integrated to work together, and it is important to develop many attractive services and applications. However, it is not easy to develop ubiquitous computing applications on exisiting software infrastructures currently since we need a variety of knowledge to develop them. We believe that middleware infrastructures for ubiquitous computing that hide a variety of complexities such as distribution and context-awareness are important to make it easy to develop ubiquitous computing applications.

In this paper, we present overviews of three middleware infrastructures that we have developed. We have considered the following issues during the design and implementation of our systems.

- Which abstraction is appropriate ?
- How to hide complexities in ubiquitous computing environments?
- How to reduce the development cost of middleware ?

Our middleware infrastructures offer high level abstraction for building specific application domains to hide complexities such as distribution and context-awareness. We report how to offer high level abstraction and to implement non functional properties hidden in the middleware infrastructures. We also discuss how to implement them on standard infrastructure software and protocols to make it easy to develop our systems. Finally, we show what we have learned during their design and implementation.

The remaining of the paper is structured as follows. Section 2 describes three middleware infrastructures for ubiquitous computing. In Section 3, we show six lessons learned for building our middleware, and Section 4 concludes the paper.

2. MIDDLEWARE INFRASTRUCTURE FOR UBIQUITOUS COMPUTING

This section describes three middleware infrastructures that have developed in our project. These middleware infrastructures do not offer generic services for building ubiquitous computing applications. They support to develop applications for specific domains to realize ubiquitous computing visions.

2.1 Middleware for Mixed Reality

2.1.1 Design Issues

The middleware infrastructure described in this section allows us to build distributed mixed reality applications in an easy way. When designing the middleware, we take into account the following issues.

MUM '2004 Octorber 27-29, College Park, Maryland, USA

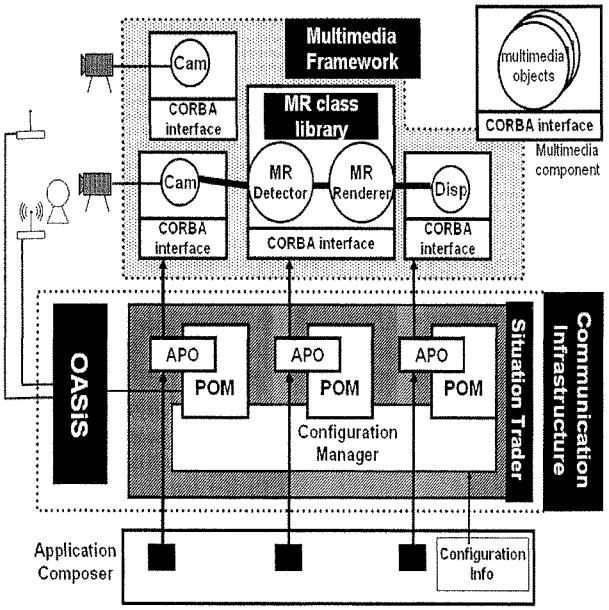

Figure 1: Overview of MiRAGe Architecture

- An application programmer should not take into account complex algorithms for implementing mixed reality applications.
- Distribution should be hidden in the middleware.
- Dynamic reconfiguration according to the current situation should be hidden from a programmer.

Our middleware infrastructure makes it dramatically easy to develop mixed reality applications for ubiquitous computing by composing existing multimedia components, and the connections among the components are reconfigured when the current situation is changed, but the change is not aware from programmers.

2.1.2 *Basic Architecture*

Our middleware infrastructure called MiRAGe[16] consists of the *multimedia framework*, the *communication infrastructure* and the *application composer*, as shown in Figure 1. The multimedia framework is a CORBA(Common Object Request Broker Architecture)-based component framework for processing continuous media streams. The framework defines CORBA interfaces to configure multimedia components and connections among the components. In the figure, each circle means a multimedia component that can be controlled through the CORBA interface.

Multimedia components supporting mixed reality can be created from the MR class library. The library contains several classes that are useful to build mixed reality applications. By composing several instances of the classes, mixed reality multimedia components can be constructed without taking into account various complex algorithms realizing mixed reality.

The communication infrastructure based on CORBA consists of the *situation trader* and *OASiS*. The situation trader is a CORBA service that supports automatic reconfiguration, and is colocated with an application program. It contains Adaptive Pseudo Objects(APO), Pseudo Object Managers(POM), and a configuration manager that are used for dynamic configuration of multimedia components, that are described in Section 2.1.4. Its role is to manage the configuration of connections among multimedia components when the current situation is changed. OASiS is a context information database that gathers context information such as location information about objects from sensors. Also, in our framework, OASiS behaves like as a naming and trading service to store objects references. The situation trader communicates with OASiS to detect changes in the current situation.

Finally, the application composer, written by an application programmer, coordinates an entire application. A programmer needs to create several multimedia components and connect these components. He specifies a policy on how to reconfigure these components to reflect situation changes. By using our framework, the programmer does not need to be concerned with detailed algorithms for processing media streams because these algorithms can be encapsulated in existing reusable multimedia components. Also, distribution is hidden by our CORBA-based communication infrastructure, and automatic reconfiguration is hidden by the situation trader service. Therefore, developing mixed reality applications becomes dramatically easy by using our framework.

2.1.3 *Multimedia Framework*

The main building blocks in our multimedia framework are software entities that internally and externally stream multimedia data in order to accomplish a certain task. We call them *multimedia components*.

A multimedia component consists of a CORBA interface and one or more *multimedia objects*. Our framework offers the abstract classes `MSource`, `MFilter` and `MSink`[1]. Developers extend the classes and override the appropriate methods to implement functionality. Multimedia objects need only to be developed once and can be reused in any components. For example, Figure 1 shows three connected components. One component contains a camera source object for capturing video images, one component contains the `MRDetector` and `MRRenderer` filter objects for implementing mixed reality functionality, and one component contains a display sink object for showing the mixed reality video images.

In a typical component configuration, video or audio data are transmitted between multimedia objects, possibly contained by different multimedia components, running on remote machines. Through the CORBA interface defined in MiRAGe connections can be created in order to control the streaming direction of data items between multimedia objects.

2.1.4 *Communication Infrastructure*

A *configuration manager*, owned by the situation trader, manages stream reconfiguration by updating connections between multimedia objects. Complex issues about automatic reconfiguration are handled by the situation trader and they are hidden from application programmers. The situation trader is linked into the application program.

In our framework, a proxy object in an application composer refers to APO, managed by the situation trader. Each

[1]In Figure 1, *Cam* is an instance of *MSource*, *MRDetector* and MRRenderer are instances of MFilter, and *Disp* is an instance of MSink.

APO is managed by exactly one POM that is responsible for the replacement of object references by receiving a notification message from OASiS upon a situation change.

A reconfiguration policy needs to be set for each POM. The policy is passed to OASiS through the POM, and OASiS selects the most appropriate target object according to the policy. In the current design, we can specify a location parameter as a reconfiguration policy. The policy is used to select the most suitable multimedia component according to the current location of a user.

A configuration manager controls the connections among multimedia components. Upon situation change, a callback handler in the configuration manager is invoked in order to reconfigure affected streams by reconnecting appropriate multimedia components.

2.1.5 *Mixed Reality Class Library*

The *MR class library* is a part of the MiRAGe framework. The library defines *multimedia mixed reality objects* for detecting visual markers in video frames and superimposing graphical images on visual markers in video frames. These mixed reality multimedia objects are for a large part implemented using the ARToolkit[2]. Application programmers can build mixed reality applications by configuring multimedia components with the mixed reality objects and stream data between them. In addition, the library defines data classes for the video frames that are streamed through the mixed reality objects.

`MRFilter` is a subclass of `MFilter` and is used as a base class for all mixed reality classes. The class `MVideoData` encapsulates raw video data. The `MRVideoData` class is a specialization `MVideoData` and contains a `MRMarkerInfo` object for storing information about visual markers in its video frame. Since different types of markers will be available in our framework, the format of marker information must be defined in a uniform way.

The class `MRDetector` is a mixed reality class and inherits from `MRFilter`. The class expects a `MVideoData` object as input and detects video markers in the `MVideoData` object. The class creates a `MRVideoData` object and adds information about detected markers in the video frame. The `MRVideoData` object is transmitted as output. The class `ARTkDetector` is a subclass of `MRDetector` that implements the marker detection algorithm using the ARToolkit.

The `MRRenderer` class is another mixed reality class derived from `MRFilter`. The class expects an `MRVideoData` as input and superimposes graphical images at positions specified in the `MRMarkerInfo` object. The superimposed image is transmitted as output. The `OpenGLRenderer` is a specialization of `MRRenderer` and superimposes graphical images generated by an OpenGL program.

2.1.6 *An Application Scenario*

In a typical mobile mixed reality application, our real-world is augmented with virtual information. For example, a door of a classroom might have a visual tag attached to it. If a PDA or a cellular phone, equipped with a camera and an application program for capturing visual tags, the tags are superimposed by a schedule of today's lecture.

We assume that in the future our environment will deploy many mixed reality servers. In the example, the nearest server stores information about today's lecture schedule and provides a service for detecting visual tags and superimposing them by the information about the schedule. Other mixed reality servers, located on a street, might contain information about what shops or restaurants can be found on the street and until how late they are open.

To build the application, an application composer uses components for capturing video data, detecting visual markers, superimposing information on video frames and displaying them. The application composer contacts a situation trader service to retrieve a reference to a POM managing references to the nearest mixed reality server to a user. When he moves, a location sensor component notifies sensed location information to OASiS, and OASiS notifies the situation trader to replace the current object reference to the reference of the nearest mixed reality server currently. In this way, the nearest mixed reality server can be selected dynamically according to his location, but the automatic reconfiguration is hidden from an application programmer.

2.2 Middleware for Interaction Devices

2.2.1 *Design Issues*

Future ubiquitous computing applications will use a variety of interaction devices to control them. These devices are distributed and are changed according to a user's current situation. Since we already have many interactive applications that adopt existing GUI toolkits, we like to reuse these applications in ubiquitous computing environments. To realize the goal, we take into account the following issues when designing the second middleware.

- Existing interactive applications can be controlled by various interaction devices.
- Interaction devices can be changed according to a user's current situation.

An application programmer needs not to consider which interaction device is appropriate by hiding the complex decision into the middleware infrastructure, and he can use existing GUI toolkits to develop ubiquitous computing applications by adopting our middleware.

2.2.2 *Basic Architecture*

Figure 2 shows an overview of the architecture of our middleware infrastructure that is called Unit[11]. In the architecture, an application generates bitmap images containing information such as control panels, photo images and video images. These applications can receive keyboard and mouse events to be controlled. The user interface middleware receives bitmap images from applications and transmits keyboard and mouse events to the applications. The role of the middleware is to select appropriate interaction devices by using context information. Input/output events and mouse/keyboard events are converted according to the characteristics of respective interaction devices.

The application implements graphical user interface by using a traditional GUI toolkit such as the GTK+ or Qt. The bitmap images generated by the user interface system are transmitted to our middleware. On the other hand, mouse and keyboard events captured by the middleware are forwarded to the toolkit. The protocol between the middleware and the user interface system are specified as a standard protocol called a *universal interaction protocol.*

Our system consists of the following four components.

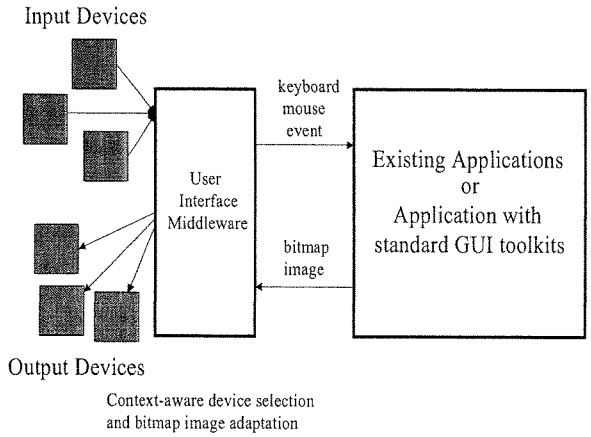

Figure 2: Basic Architecture

- Interactive Application
- Unit Server
- Unit Proxy
- Input/Output Interaction Devices

Interactive applications generate graphical user interface written by using traditional GUI toolkits. In our system, we can use any existing GUI based interaction applications, and they are controlled by a variety of interaction devices that are suitable for a user's current situation.

The *Unit server* transmits bitmap images generated by a GUI toolkit using the universal interaction protocol to a Unit proxy. It forwards mouse and keyboard events received from a Unit proxy to the GUI toolkit.

The *Unit proxy* is the most important component in our system. The Unit proxy converts bitmap images received from a Unit server according to the characteristics of output devices. The Unit proxy converts events received from input devices to mouse or keyboard events that are compliant to the universal interaction protocol. The Unit proxy chooses a currently appropriate input and output interaction devices for controlling appliances. To convert interaction events according to the characteristics of interaction devices, the selected input device transmits an input specification, and the selected output device transmits an output specification to the Unit proxy. These specifications contain information that allows a Unit proxy to convert input and output events.

The last component is *input and output interaction devices*. An input device supports the interaction with a user. The role of an input device is to deliver commands issued by a user to control interactive applications. An output device has a display device to show graphical user interface to control the interactive applications.

In our approach, the Unit proxy plays a role to deal with the heterogeneity of interaction devices. Also, it can switch interaction devices according to a user's situation or preference. This makes it possible to personalize the interaction between a user and appliances.

2.2.3 Unit Proxy

The current version of Unit proxy is written in Java, and the implementation contains four modules. The first module is the universal interaction module that executes the universal interaction protocol to communicate with a Unit server. The second module is the plug and play management module. The module collects currently available interaction devices, and builds a database containing information about respective interaction devices. The third module is the input management module. The module selects a suitable input interaction device by using the database contained in the plug and play management module. The last module is an output management module. The module also selects a suitable output interaction device. Also, the module converts bitmap images received from the universal interaction module according to the output specification of the currently selected output interaction device.

2.2.3.1 Management of Available Interaction Devices:

The plug and play management module detects currently available input and output devices according to context information. The module implements UPnP(Universal Plug and Play) to detect currently available interaction devices. An interaction device transmits advertisement messages using SSDP(Simple Service Discovery Protocol). When a Unit proxy detects the messages, it knows the IP address of the interaction device. Then, the Unit proxy transmits an HTTP (HyperText Transfer Protocol) GET request to the interaction device. We assume that each interaction device contains a small Web server, and returns an XML(eXtensible Markup Language) document.

The XML document contains information about the interaction devices. If the interaction device is an input device, the document contains various attributes about the device, which are used for the selection of the most suitable device. For an output device, the document contains information about the display size and the attributes for the device. The plug and play management module maintains a database containing all information about currently detected interaction devices.

2.2.3.2 Adaptation of Input and Output Events: .

The role of the input management module and the output management module is to determine the policies for selecting interaction devices. As described in the previous paragraph, all information about currently available interaction devices are stored in a database of the plug and play management module. The database provides a query interface to retrieve information about interaction devices. Each entry in the database contains a pair of an IP address and a list of attributes for each interaction device, then the entry whose attributes are matched to a keyword provided in a query is returned.

The output management module converts bitmap images received from the universal interaction module according to the display size of an output device. The size is stored in the database of the plug and play management module. When an output device is selected, the display size is retrieved from the database. The bitmap image is converted according to the retrieved information, then it is transmitted to the selected output device.

2.2.4 An Application Scenario

An example described in this section is a ubiquitous video phone that enables us to use a video phone in various ways. In this example, we assume that a user speaks with his friend by using a telephone like a broadband phone developed by AT&T Laboratories, Cambridge. The phone has a receiver like traditional phones, but it also has a small display. When the phone is used as a video phone, the small display renders video streams transmitted from other phones. The display is also able to show various information such as photos, pictures, and HTML(Hypertext Markup Language) documents that are shared by speakers. Our user interface system makes the phone more attractive, and we believe that the extension is a attractive application in ubiquitous computing environments.

When a user needs to start to make a dinner, he will go to his kitchen, but he likes to keep to talk with his friend. The traditional phone receiver is not appropriate to continue the conversation with his friend in the kitchen because his both hands may be busy for cooking. In this case, we use a microphone and a speaker in the kitchen so that he can use both hands for making the dinner while talking with his friend. In the future, various home appliances such as a refrigerator and a microwave have displays. Also, a kitchen table may have a display to show a recipe. These displays can be used by the video phone to show a video stream. In a similar way, a video phone can use various interaction devices for interacting with a user. The approach enables us to use a telephone in a more seamless way.

Our system allows us to use a standard VoIP application running on Linux. The application provides a graphical user interface on the X window system, but our system allows a user to be able to choose various interaction styles according to his/her situation. If his/her situation is changed, the current interaction style is changed according to his preference.

2.3 Middleware for Home Computing

2.3.1 Design Issues

In the future, there are many home applications in our home environments. It is important to control these appliances in an easy way. The third middleware infrastructure called a *personal home server*[12] allows us to aggregate them by using a personal device. While designing the middleware, we take into account the following issues.

- We like to control home appliances from various presentation documents such as HTML and Flash.
- A way to control home appliances is changed according to a user's preference.

Our middleware offers high level abstraction to specify appliances that a user likes to control, and each user's personal device contains rules for personalizing the control of appliances. Since a personal home server can be carried with a user, he can aggregate home appliances by using the same preferences in a seamless way anytime anywhere.

2.3.2 Basic Architecture

A personal home server that is carried by a user is implemented in a personal device like a cellular phone, a wrist watch, or a jacket. Thus, the server can be carried by a user anytime anywhere. The personal home server collects information about home appliances near a user, and creates a database storing information about these appliances. Then, it creates an HTML-based presentation document containing the attributes of appliances and the commands to control them. A display near the user also detects the personal home server, and retrieves the presentation document containing the automatically generated user interface. The display shows the presentation document on the display. The document contains URLs(Uniform Resource Locator) embedding the attributes of appliances and their commands specified by using our URL-based naming scheme. Also, the presentation document is customized according to the user's preference. When the user touches the display, a URL containing the attributes of an appliance and its command is transmitted to his/her personal home server via the HTTP protocol. The server translates the URL to a SOAP command by accessing a database containing information about the appliance that s/he likes to control. Finally, the SOAP(Simple Object Access Protocol) command is forwarded to the target appliance.

2.3.3 Spontaneous Appliance Detection

2.3.3.1 URL-based Naming Scheme: .

Our framework allows a user to access one or more appliances through a personal home server. We introduce a URL-based naming convention for specifying and controlling appliances. In our approach, by embedding the attributes of appliances and commands in URLs, an HTML-based presentation document can be used to control home appliances. The convention is defined within the standard URL but the path elements of the URL form can contain some additional information.

The URL definition is very flexible because we can specify various attributes to identify a target home appliance. We can also use attributes that represent context information such as location. A personal home server can select an appliance in a context-aware way.

2.3.3.2 Service Management: .

In our system, the service management module in a personal home server knows respective appliances via SSDP, and retrieves service specification documents represented as RDF(Resource Description Framework).

The service database in a personal home server contains all service specification documents detected currently. It contains a link to a WSDL(Web Service Description Language) document identifying commands that can be accepted. If an appliance contains several functionalities, its specification document may contain several links to WSDL documents. Also, attributes of the document are used to identify a target appliance.

2.3.3.3 Personalization Management: .

A personal home server customizes a presentation document according to a user's preference encoded in a preference rule. Now, a personal home server detects several types of light appliances. We assume that a rule to filter light alliances whose type is not a ceiling light is stored in the personal home server. The presentation document contains only ceiling lights that reside in a room where a user is. A personal home server omits information about other

types of light appliances. The preference rule is encoded in a tag, and it can be registred in a personal home server by closing the tag to a user's personal device[13].

2.3.4 An Application Scenario

In this scenario, we assume that surrounding various objects will embed tags. Since these tags contain different preference rules, the customization is changed according to objects near a user. For example, if a child holds a stuffed animal that contains tags, the rules encoded in the tags are registered in the child's personal coordination server. S/he can customize how to control information appliances by changing stuffed animals that s/he holds currently.

For example, we assume that a child is holding a stuffed toy dog. The dog contains a tag including a rule for selecting televisions because the child believes that the dog likes to watch televisions. Thus, a display shows a user interface to control televisions. On the other hand, when the child holds a stuffed toy rabbit, the display shows a user interface for music players because she believes that the rabbit likes to listen to music.

The tags can also be embedded in our daily goods like clothes, accessories, and shoes. Especially, a young person usually wants to put on these goods according to his/her feeling or emotion everyday. The goods reflect his/her preferences and current mental condition either consciously or unconsciously. Customizing services depending on what a user puts on today makes his/her daily lives more pleasurable.

3. DISCUSSIONS

The section describes some experiences while building our middleware infrastructures, and identifies six lessons learned from the experiences.

3.1 High Level Abstraction and Middleware Design

One of the most important design issues for building middleware infrastructures for ubiquitous computing is what properties the middleware infrastructures should hide. In our first middleware, distribution and automatic reconfiguration are hidden from a programmer. In the second middleware, the automatic selection of interaction devices is hidden from an application programmer. The last middleware hides device discovery and personalization from a programmer. Our experiences show that hiding these complex issues makes application programming dramatically easy. However, achieving complete distribution transparency is very hard to be implemented because different abstractions require different ways to hide the distribution. Each abstraction may also have different assumptions and constraints to hide dynamic reconfiguration. It is not easy to hide these properties in a common infrastructure that can be shared from various middleware for ubiquitous computing, and we need to carefully consider how to hide the properties in each middleware infrastructure.

High level abstraction supporting a specialized application domain is very useful to develop ubiquitous computing applications easily. The similar conclusion is discussed in a middleware infrastructure for supporting synchronous collaboration in an office environment[15], and a middleware infrastructure for building location-aware applications[6]. In our approach, the first middleware focuses on supporting continuous media applications. The second middleware supports interactive applications that are controlled by a variety of interaction devices. The third middleware makes it easy to aggregate home appliances in a spontaneous way. These middleware supports only specialized application domains, and their functionalities do not overlap each other. Thus, they can coexist to develop one application. For example, we can use the second middleware to control an mixed reality application implemented on the first middleware. Also, the third middleware can be used to discover multimedia components implemented on the first middleware dynamically.

Lesson 1: It is important to develop specialized high level abstraction for supporting one specific domain, but the abstraction should be generic to cover a wide range of applications.

Middleware infrastructures for ubiquitous computing need to offer various non-functional properties such as context-awareness, timeliness, reliability and security. For example, the first and second middleware infrastructures automatically change the configurations according to a user's situation, and the third middleware infrastructure generates a graphical user interface automatically according to the currently available appliances. However, it is not easy to offer a common service for supporting context-awareness because modeling our real world for building any ubiquitous computing applications require to define ontologies in a complete way. Therefore, we suspect that we can implement a generic and reusable high level service to support context-awareness that is used in various middleware for ubiquitous computing. On the other hand, we believe that a low level support for handling context information like [4] can be used uniformly in many middleware infrastructures. This means that it is desirable to hide the details of the behavior of sensors in a component to develop middleware infrastructures in an easy way, but the middleware should not interpret the meaning of the value retrieved from sensors because there is no common consensus how to model our world in a standard way.

Also, we found that it is not easy to offer a common service for adding security in each middleware infrastructure. In the second system, a system needs to register their interaction devices before using them manually, but after registering them, the devices can be changed according to a user's situation automatically. In the third system, a very light-weighted security support is implemented to make the system simple[12]. We found that each middleware infrastructure requires a customized security support because each application domain may have different requirements for supporting security.

We believe that future middleware infrastructures should offer a variety of non-functional concerns such as security, timeliness, privacy protection, trust relationship among people, and reliability. The generic support of the concerns may make the infrastructures too big and complex. Especially, when multiple middleware infrastructures are integrated, the concerns are cross-cut across them. Therefore, it is important to support only minimum supports, and customized supports for non-functional properties should be implemented respectively using software structuring techniques like an aspect-oriented programming technique[9] on the minimum supports.

Lesson 2: Common generic services that offer non-functional concerns may make it difficult to develop practical ubiquitous computing applications. The service should be customized in respective middleware infrastructures.

3.2 Development of Middleware Infrastructures

The third middleware has adopted standard protocols such as SOAP as underlying protocols. This makes it easy to adopt commercial products in our experiments, and the approach is very important for incremental evolution of ubiquitous computing environments. However, devices and appliances in smart environments may want to change their interfaces independently. We need a more spontaneous approach to collaborate them. We believe that it is desirable not to fix interface between appliances, but to determine the message format to communicate with each other. The format may adopt the XML-based representation. In the approach, each appliance may add extra tags that offer additional functionalities. Let us assume that an appliance receives the message. If the message contains some tags that cannot be understood by the appliance, the tags can simply be ignored. Therefore, each appliance can extend the message format independently. We believe that the approach is desirable in ubiquitous computing environments to support communication among appliances.

In the first middleware, we have adopted CORBA to compose multimedia components. The middleware offers multiple interfaces to communicate between an application composer and components. The approach is useful to support multiple versions of the interfaces, and components can extend its functionalities by adding new interface.

Lesson 3: Each program should extend its interface independently without considering other programs if they are loosely coupled.

Our project has adopted Linux, CORBA, OSGi and Java as underlying infrastructures, and they reduce the development cost dramatically. We think that it is not a good approach to extend existing commodity software because this makes it difficult to replace software platforms. Our middleware exploits to use existing software and appliances. For example, in the first middleware, we use a CORBA system as an underlying infrastructure. However, we have not modified the existing CORBA runtime to support dynamic configuration. Therefore, we can use any commercial CORBA runtimes for executing our middleware. The second middleware allows us to use existing GUI-based applications as ubiquitous computing applications without modifying them. Thus, the approach allows us to use exisiting interactive applications in ubiquitous computing environments. Also, the third middleware adopts OSGi[3] as its component framework. Therefore, we can use standard services provided by OSGi, and we can use any OSGi components to develop new services on personal home servers.

We believe that the approach to use traditional commodity software without modifying them is very desirable to migrate to new platforms easily when old platforms will become obsolete. If we adopt a modified version of commodity software, it is difficult to promote our middleware on the new platforms. The approach is very desirable to migrate from the current environments to ubiquitous computing environments in a seamless and incremental way. For example, in [10], we have extended CORBA to support dynamic transport protocol selection. However, we need to rewrite applications to select the most appropriate transport protocol. The approach is very useful to optimize the performance of applications on a specific environment, but the modification cost is not cheap to use existing large applications.

Lesson 4: We should not extend traditional standard middleware infrastructures to support advanced ubiquitous computing services if possible.

3.3 Human Factors

In our middleware infrastructures described in the paper, dynamic reconfiguration is hidden from a user. However, these properties are closely related to human factors. For example, if an interaction device is switched in an unexpected way, a user may surprise the context change. This means that middleware infrastructures that hide dynamic reconfiguration will need to take into account human factors when designing middleware[5]. Our experiences show that automatic selection of interaction devices is not good approach. Instead, we use a token to choose the most suitable interaction device in an explicit way. For example, let us assume that a user is using a telephone in a living room. When s/he moves to a kitchen, s/he may use a speaker and a microphone in the kitchen. In this case, the user brings a token attached to the telephone, and put it into a base in the kitchen. Our system detects the event, and changes the interaction devices for the user.

However, implicit changes may be attractive for realizing pleasurable services. For example, if an environment detects that a user and his girl friend are together in a room, it is desirable to make the room's lighting strategy more romantic automatically. Also, implicit changes are desirable if a user utilizes services without being aware. In this case, the services should not interrupt the user's current activity that he is focusing on. We believe that it is better to control the strategies for dynamic reconfiguration should be customized in each application. The programming interface to control dynamic reconfiguration should be clearly separated from other programming interface to make the structure of an application clear.

When representing personal information on a display, we need to take into account how to protect privacy information of a user. However, the information is useful to offer better services customized for the user. We found that it is important to take into account the tradeoff between the quality of services and the amount of privacy information. When a user cannot trust an application, s/he will not offer his/her personal information, but if s/he wants better services and trusts the services, s/he can offer more his/her personal information. Also, when multiple users share a display, it is desirable to abstract information represented on the display to protect privacy information. The abstraction level of the representation is determined according to how the information is secret. Ambient displays or informative art[7] is a first step towards information abstraction that allows us to protect privacy information and to reduce information overload in our society.

Lesson 5: Middleware infrastructures should be flexible to implement dynamic reconfiguration and information representation according to the requirements of respective applications.

In the near future, designers for ubiquitous computing middleware should learn psychology, and we need to consider that the adaptation of software should not contradict our mental model. We believe that how to implement the real world model and the mental model in middleware infrastructures is an important research topic for building practical middleware for ubiquitous computing. For example, in our second middleware, if choosing a suitable interaction device is not consistent with a user' s mental model, the user will confuse which interaction device s/he should use. In the first and second middleware, if the dynamic configuration is not consistent with a user's mental model, the user may surprise the dynamic changes. However, the implementation of a real world model and mental model requires to represent ontologies in a standard way to access them from a variety of middleware for ubiquitous computing.

We also need to consider social aspects and cultural aspects when designing applications interacting with the real world. For example, it is important to take into account trust and privacy in future ubiquitous computing environments, but we need to learn sociology and anthropology to know whether our understanding is enough or not. We believe that it is important to consider how to model psychological, social, and anthropological concepts into our programs to interact with the real world properly when designing middleware infrastructures for ubiquitous computing. For example, in our middleware infrastructures, we use a user's location to offer context-aware services, but it depends on a user's situation to offer location information to our middleware. These information related to privacy should be treated very carefully, and traditional concepts about privacy and trust are very naive to offer practical ubiquitous computing services.

We also believe that the designers for ubiquitous computing middleware should know aesthetics to provide pleasurable services[8] or to abstract information. To develop pleasurable services, we may need to take into account emotion, peak experience, and unconsciousness to develop software. For example, our third middleware supports a mechanism to design pleasurable experiences by encoding preference rules in RF tags. Our system infrastructure allows us to embed tags into various objects and places, and controls our experiences by changing the behavior of applications[13], and we found that the approach is very useful to offer pleasurable services.

Lesson 6: Middleware infrastructures should incorporate a model for psychological, sociological and anthropological concepts explicitly.

4. CONCLUSION AND FUTURE DIRECTION

This paper has described three middleware infrastructures that have developed in our project. We have also presented several experiences and future directions for building middleware for ubiquitous computing. We believe that there are new requirements to develop the middleware infrastructures for ubiquitous computing. Especially, we believe that it is important to take into account human factors to develop them.

One of the most important future topics in our project is to develop a pattern language[1] for building middleware infrastructures for ubiquitous computing. The language will support to consider what abstraction should export, which properties should be hidden and how to offer non-functional properties. Also, the language will help to consider how to implement middleware infrastructures in an easy way and how to use legacy software.

5. REFERENCES

[1] C.Alexander, "A Pattern Language: Town, Building, Construction", Oxford Press, 1977.

[2] ARToolkit, http://www.hitl.washington.edu/research/shared_space/download/.

[3] K.Chen, L.Gong, "Programming Open Service Gateways with Java Embedded Server", Addison-Wesley, 2001.

[4] A.Dey, G.D. Abowd, D.Salber, "A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications", Human-Computer Interaction, Vol.16, No.2-4, 2001.

[5] Edwards, K., Bellotti, V., Dey, A.K., Newman, M. "Stuck in the Middle: The Challenges of User-Centered Design and Evaluation for Middleware", In the Proceedings of CHI 2003, 2003.

[6] A.Harter, A.Hopper, P.Steggles, A.Ward, P.Webster, "The Anatomy of a Context-Aware Application", In Proceedings of Mobicom 2000, 2000.

[7] L.E.Holmquist, T. Skog, "Informative Art: Information Visualization in Every day Environments", In Proceedings of GRAPHITE 2003, 2003.

[8] P.W.Jordan, "Designing Pleasurable Products", CTI, 2000.

[9] G.Kiczales, J.Lamping, A.Memdheker, C.Maeda, C.V. LOpes, J-M. Loingtier, J.Irwin, "Aspect-Oriented Programming", In Proceedings of ECOOP'97, 1997.

[10] T.Nakajima, "Practical Explicit Binding Interface for supporting Multiple Transport Protocols in a CORBA system", In Proceedings of ICNP'00, 2000.

[11] Tatsuo Nakajima, et. al., "Making Existing Interactive Applications Context-Aware", In Proceedings of Europar 2003, 2003.

[12] T.Nakajima, I.Satoh, "Personal Home Server: Enabling Personalized and Seamless Ubiquitous Computing Environments", In Proceedings of Percom2004, 2004.

[13] T.Nakajima, "A Personalization Framework in a Personal Home Server: System Infrastructure for Designing Pleasurable Experiences", To be submitted, 2004.

[14] B.J.Pine II, J.H. Gilmore, "The Experience Economy", High Bridge Company, 1999.

[15] P.Tandler, "The BEACH Application Model and Software Framework for Synchronous Collaboration in Ubiquitous Computing Environments", Journal of Systems and Software, October, 2003.

[16] Eiji Tokunaga, Tatsuo Nakajima, et. al., "A Middleware Infrastracture for Building Mixed Reality Applications in Ubiquitous Computing Environments", In the Proceedings of Mobiquitous2004, 2004.

独立成分分析に関するその他の研究成果

Image Compression Based Upon Independent Component Analysis:
Generation of Self-Aligned ICA Bases

Yasuo MATSUYAMA, Ryo KAWAMURA, Hiroaki KATAOKA, Naoto KATSUMATA,
Kenzo TOJIMA, Hideaki ISHIJIMA, and Keita SHIMODA

Department of Computer Science, Waseda University
Tokyo, 169-8555 Japan

{yasuo2, ryo@asagi, pp_kataoka@moegi, katsu@ruri,
k-tojima@toki, suzaku@toki, kei215@asagi}.waseda.jp

Abstract

Generation of the ordered set of ICA bases (Independent Component Analysis bases) and its applications to image compression are discussed. The ICA bases have similar properties to existing orthogonal bases. Orthogonal bases generate uncorrelated coefficients, while, the ICA bases bring about independent coefficients. The independence is stronger than the uncorrelatedness. Therefore, the ICA bases can extract source information better. One difficulty using ICA is the permutation indeterminacy among these bases. This paper presents partially supervised learning for generating self-aligned ICA bases. It is observed that: (i) Each basis reflects edges and textures like the early vision. (ii) Bases can be self-aligned in the sense of spatial frequency. (iii) Coefficients of the bases can be used for image compression. Experiments show that (iv) the set of ICA image bases is a well-qualified alternative to existing orthogonal ones.

1. Introduction

Independent Component Analysis (ICA) [1] is a method of multivariate analysis to decompose measured data into independent components. It is a class of learning algorithms from data. Its application is wide including images, speech, music signals and so on. Therefore, ICA has received much attention from communities of adaptive learning and multimedia processing. This paper contributes to these fields by showing a new method to obtain ICA image bases and novel applications to image compression.

Organization of this paper is as follows. In Section 2, the ICA problem is formulated. The role of the ICA basis set is elucidated. The permutation indeterminacy, which essentially exists in the ordinary ICA, is explained. Presentation of the ICA model for image construction is also given. Then, in Section 3, preprocessing, orthonormalization, and ordinary ICA algorithms are explained to assist later explanations on improved methods. In Section 4, The ICA learning with weak guidance is presented. This partial supervision is effective to the reduction of the permutation indeterminacy, whose step is necessary for the application of ICA bases. In Section 5, experiments on digital images are executed. The ICA image bases are successfully aligned by reflecting spatial frequencies. Experiments show that the ICA bases are promising in the image compression as the theory predicts. Section 6 gives concluding remarks with prospects of future studies.

2. Problem Formulation of ICA

2.1. Mixture of Independent Components

In the problem of ICA, a vector random variable

$$\boldsymbol{x} = [x_1, \cdots, x_n]^T \tag{1}$$

is assumed to be generated by another random variable

$$\boldsymbol{s} = [s_1, \cdots, s_n]^T \tag{2}$$

by the following mixture.

$$\boldsymbol{x} = \boldsymbol{A}\boldsymbol{s} = [\boldsymbol{a}_1 \cdots, \boldsymbol{a}_n]\boldsymbol{s} = \textstyle\sum_{i=1}^{n} \boldsymbol{a}_i s_i \tag{3}$$

The matrix $\boldsymbol{A}$ and the vector $\boldsymbol{s}$ are both unknown except for the following information.

(a) The components s_i and s_j are independent each other for $i \neq j$.

(b) The components s_i, $(i = 1, \cdots, n)$, are non-Gaussian except for at most one i.

Under the above conditions, we want to estimate a demixing matrix

$$\boldsymbol{W} = \boldsymbol{\Lambda}\boldsymbol{\Pi}\boldsymbol{A}^{-1} \tag{4}$$

so that the components y_i, $(i = 1, \cdots, n)$, of

$$\boldsymbol{W}\boldsymbol{x} \stackrel{\text{def}}{=} \boldsymbol{y} = [y_1, \cdots, y_n]^T \tag{5}$$

are independent each other. Here, $\boldsymbol{\Lambda}$ is a nonsingular diagonal matrix which decides components' scale, and $\boldsymbol{\Pi}$ is a permutation matrix. These matrices are unknown too. This property is called the indeterminacy, which essentially exists in the ICA formulation. In this paper, such indeterminacy will be carefully avoided.

2.2. ICA bases

Column vectors of $\boldsymbol{W}^{-1} \stackrel{\text{def}}{=} \boldsymbol{U}$ can be interpreted as ICA bases since the following equality holds for the observed data $\boldsymbol{x}$.

$$\boldsymbol{x} = \boldsymbol{U}\boldsymbol{y} = [\boldsymbol{u}_1, \cdots, \boldsymbol{u}_n]\boldsymbol{y} = \textstyle\sum_{i=1}^{n} \boldsymbol{u}_i y_i \tag{6}$$

In order to save notational alphabets, $\boldsymbol{U}$ is re-expressed by $\boldsymbol{A}$ hereafter, and so is $\boldsymbol{y}$ by $\boldsymbol{s}$. This is applied only if there is no confusion.

When an ICA basis $\boldsymbol{a}_i$ is used in image processing, it is interpreted as a two dimensional patch $\{\{a_i(x, y)\}_{x=1}^{m}\}_{y=1}^{m}$. Then, each pixel is modeled by

$$I(x, y) = \textstyle\sum_{i=1}^{n} a_i(x, y) s_i, \tag{7}$$

where $n = m^2$. Once the ICA bases are learned from data, they are *fixed*. Therefore, $\{s_i\}_{i=1}^{n}$ are subject to coding for image compression.

3. ICA Learning Algorithms

3.1. Preprocessing and Orthonormalization

Observed data are preprocessed in the following way so that the estimate of $\boldsymbol{W}$ converges properly.

1. [Mean and variance normalization] Observed data are normalized to have the zero mean and the unit variance.
2. [Whitening] Observed data are then transformed to $\boldsymbol{z} = \boldsymbol{V}\boldsymbol{x}$ so that $\mathcal{E}[\boldsymbol{z}\boldsymbol{z}^T] = \boldsymbol{I}$. Here, $\mathcal{E}$ stands for the expectation. We use $\boldsymbol{V} = \boldsymbol{D}^{-1/2}\boldsymbol{E}^T$ in our experiments. Here, $\boldsymbol{D}$ is a diagonal matrix whose elements are eigenvalues of $\mathcal{E}[\boldsymbol{x}\boldsymbol{x}^T]$. $\boldsymbol{E}$ is the matrix whose columns are corresponding eigenvectors.
3. [Orthonormalization] Another transformation is the orthonormalization: $\boldsymbol{W} \leftarrow (\boldsymbol{W}\boldsymbol{W}^T)^{-1/2}\boldsymbol{W}$. This is an expensive computation, however, the merits of $\boldsymbol{U} = \boldsymbol{W}^T$ and $\boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}$ are obtained.

3.2. First-Stage Learning Algorithm

Estimation or learning of $\boldsymbol{W}$ from observed data is performed by the following iteration:

$$\boldsymbol{W}^{\text{new}} = f(\boldsymbol{W}^{\text{old}}), \tag{8}$$

or equivalently,

$$\boldsymbol{W}^{\text{new}} = \boldsymbol{W}^{\text{old}} + \Delta\boldsymbol{W}. \tag{9}$$

The updated vector $\boldsymbol{W}^{\text{new}}$ can be obtained by optimizing statistical measures for the independence [2] $\sim$ [7].

We gave necessary explanations on the first-stage algorithm for $\boldsymbol{W}$ except for the following. We are given sample image patches rather than an abstract random variable in a probability space. Therefore, we need to write down these samples in matrix forms: $\boldsymbol{X} = [\boldsymbol{x}(1), \cdots, \boldsymbol{x}(m)]$, $\boldsymbol{S} = [\boldsymbol{s}(1), \cdots, \boldsymbol{s}(m)]$, and $\boldsymbol{Y} = [\boldsymbol{y}(1), \cdots, \boldsymbol{y}(m)]$. Thus, the data generation model is expressed by

$$\boldsymbol{X} = \boldsymbol{A}\boldsymbol{S}. \tag{10}$$

Then, the first-stage learning algorithm becomes as follows.

[First-stage learning algorithm]

[Step 1 (Preprocessing 1)]
Obtain a sample matrix $\boldsymbol{X}$ as a training data set. Normalize each column vector to be zero mean and unit variance.

[Step 2 (Preprocessing 2: Whitening)]
Obtain Whitening Matrix $\boldsymbol{V}$ from $\boldsymbol{X}$, and compute $\boldsymbol{Z} = \boldsymbol{V}\boldsymbol{X}$.

[Step 3 (Initialization)]
Choose an orthonormalized initial value for $\boldsymbol{W}$.

[Step 4 (Update 1)]
Update $\boldsymbol{W}$ by (8) or (9).

[Step 5 (Update 2)]
Orthonormalize the matrix $\boldsymbol{W}$.

[Step 6 (Convergence check)]

Check to see if convergence is achieved. Otherwise repeat Steps 4 and 5.

[Step 7 (Resulting matrices)]

Resulting matrices are obtained by

$$\boldsymbol{W}_{\text{stage1}} = \boldsymbol{W}\boldsymbol{V}, \tag{11}$$

and

$$\boldsymbol{A}_{\text{stage1}} = (\boldsymbol{W}\boldsymbol{V})^{-1} = \boldsymbol{V}^{-1}\boldsymbol{W}^{T}. \tag{12}$$

It is necessary to comment here that:

(i) The first-stage algorithm still inherits the permutation indeterminacy. We need further learning algorithms which does not suffer from this indeterminacy.

(ii) In the image compression, the matrix

$$\boldsymbol{Y}_{\text{data}} = \boldsymbol{W}_{\text{stage1}}\boldsymbol{X}_{\text{data}} \tag{13}$$

is encoded to $\hat{\boldsymbol{Y}}_{\text{data}}$. Decoded is then

$$\hat{\boldsymbol{X}}_{\text{data}} = \boldsymbol{A}_{\text{stage1}}\hat{\boldsymbol{Y}}_{\text{data}}. \tag{14}$$

4. Learning Under Weak Guidance

4.1. Indeterminacy Reduction I: Topographic Alignment of ICA Bases

The above $\boldsymbol{A}_{\text{stage1}}$ could be used as a set of image compression bases, if one would dare to check manually the whole matrix pattern, and if high performance is not required. Thus, the bases are more suitable for the image compression if they have ordered by spatial frequencies precisely. Therefore, we consider to use the resulting image bases as an initial set for further learning modification. This is allowed since the image bases need not be computed on-line but to be stored in the encoder-decoder pair. There is one more evidence to support this: All computation in this paper can be carried out by a conventional personal computer, which will be understood in Section 5.

The first step to obtain an aligned image basis set is to modify the matrix $\boldsymbol{W}_{\text{stage1}}$ by using the topographic ICA [8]. In this case, (9) is used with the following computation:

$$\Delta \boldsymbol{w}_i = \eta E[\boldsymbol{z}(\boldsymbol{w}_i^T\boldsymbol{z})r_i], \tag{15}$$

$$r_i = \textstyle\sum_{k=1}^{n} h(i,k)G'(\sum_{j=1}^{n} h(k,j)(\boldsymbol{w}_j^T\boldsymbol{z}^2)). \tag{16}$$

On the choices of $G(y)$ and $h(i,j)$, readers are requested refer to [8]. Hereafter, the update matrix by (15) is denoted by $\Delta\boldsymbol{W}_{\text{tp}}$.

4.2. Indeterminacy Reduction II: Weak Guidance

Resulting ICA bases as a topographic map show an intriguing visual pattern. But, a very important indeterminacy is not yet resolved. A human can instantly find the position of the central basis corresponding to the lowest spatial frequency, however, machines can not do so instantly. Therefore, we need a further important mechanism to reduce such indeterminacy. This is the method of *weak guidance* as a partially supervised learning. Such a method was first used in the distillation of brain maps from fMRI data [9], [10].

[Weak Guidance]

First, we prepare a teacher signal, or a reference pattern, as a matrix $\bar{\boldsymbol{R}}$. Then, we compute $\boldsymbol{U} = \boldsymbol{V}^{-1}\boldsymbol{W}^{T}$. The increment by the teacher signal is

$$\Delta\boldsymbol{U} = \boldsymbol{V}\{\lambda(\bar{\boldsymbol{R}} - \boldsymbol{U})\}. \tag{17}$$

Here, λ is a learning parameter. Then, the update term for the weak guidance is computed by

$$\Delta\boldsymbol{W}_{\text{wg}} = -\boldsymbol{W}\Delta\boldsymbol{U}\boldsymbol{W}. \tag{18}$$

Readers are requested refer to [9] or [10] for the derivation of (17) and (18).

4.3. Total Learning Algorithm

By the preceding preparations, the total algorithm to obtain the ICA bases can be described as follows.

[Step 1 (Learning parameters)]

Control rules of the small learning parameters $\eta > 0$ and $\lambda > 0$ are decided. The rules can be arbitrary as long as (i) η increases and saturates. (ii) λ decreases.

[Step 2 (Weak guidance)]

Compute the updated matrix with the weak guidance

$$\boldsymbol{W} \leftarrow \boldsymbol{W} + \Delta\boldsymbol{W}_{\text{wg}}. \tag{19}$$

[Step 3 (Topographical map)]

Compute the updated matrix with the topographic constraint

$$\boldsymbol{W} \leftarrow \boldsymbol{W} + \Delta\boldsymbol{W}_{\text{tp}}. \tag{20}$$

[Step 4 (Convergence check)]

Check to see if the matrix update is converged. If not, then the iteration is repeated on Steps 2 and 3 after the update of λ, η, and $\boldsymbol{W}$.

5. Experimental Results

5.1. ICA Image Bases with Self-Alignment

All necessary tools were given in the preceding sections. We can now apply them to real images. Training data for the ICA bases contain many images such as natural images, screen text images, and animations. Figure 1 illustrates the ICA bases obtained by the topographic method alone, i.e., without the weak guidance. Each basis is of 8×8 pixels so that the size is compatible with usual JPEG and JPEG2000. As can be observed, the basis of the lowest spatial frequency is located off-centered in the two dimensional array. This position can not be specified in advance. Therefore, the human perception is still necessary to identify where the exact center is. Besides, the obtained ICA bases are inefficient since the center is near the corner of the array.

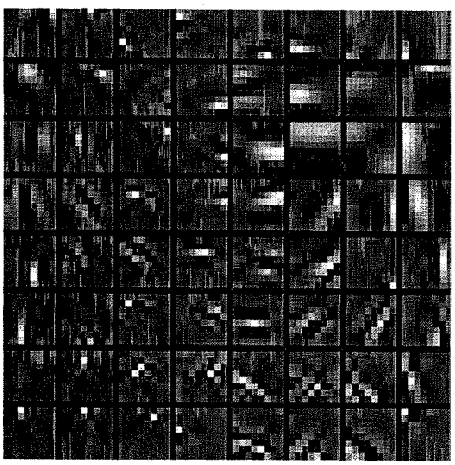

Figure 1. Image bases only by the topographic method.

Figure 2 shows the resulting self-aligned ICA bases by our weak guidance method. The first basis is located at the north west of the four central bases. Low-frequency bases are concentrated around the center of the two-dimensional array. High-frequency bases are located at the corners. This was specified to be so by virtue of the weak guidance. We call such a class of bases the *ICA ripplet set*, or simply the *ripplet set*. The ripplet set is readily applicable to the image compression due to the following properties.

(a) Ordering from low to high spatial frequencies is completed.

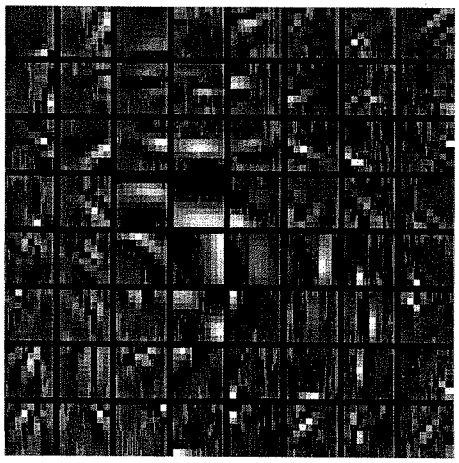

Figure 2. Self-aligned image bases by the weak guidance.

(b) Bases are balanced because of the centering by the weak guidance.

Item (a) can be observed from Figure 3 which was obtained from Figure 2 by the clockwise spiral sorting started from the origin, the north west of the four central bases. This figure clearly shows a self-aligned ordering from low to high spatial frequencies which has the following merit:

(c) Users of this set can understand the role of each basis in a linearly ordered sense. High-frequency bases may correspond to noisy patterns. Such a merit will be utilized in Section 5.3.

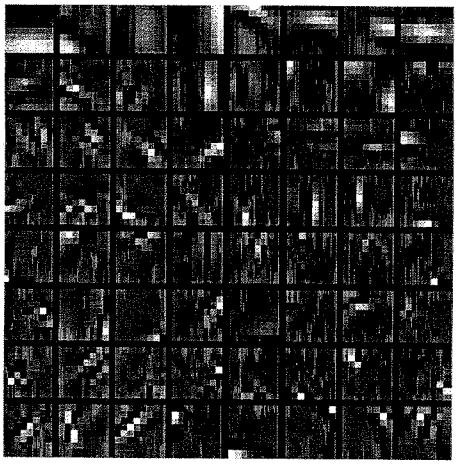

Figure 3. Aligned image bases.

Figures 1 to 3 can claim the similarity to the receptive field properties [11]. Not only appreciating such an intriguing similarity, but we pay attention to the self-aligned ICA bases of Figures 2 and 3 on the use for the image compression.

5.2. Distribution of Coefficients

Source images are reconstructed by using Equation (7). We remind readers here that the relationship on the ICA image bases: $\boldsymbol{A}^{-1} \leftarrow \boldsymbol{W}$. Since the image bases are not altered any more, they are memorized in the encoder and the decoder. In the encoder, coefficients s_i are obtained by $\boldsymbol{S} = \boldsymbol{A}^{-1}\boldsymbol{X}$. Here, each column vector of $\boldsymbol{X}$ corresponds to an image patch. Then, encoded values of elements s_i in $\boldsymbol{S}$ are transmitted (in a two dimensional form, $s_{i,j}$).

The distribution of $\boldsymbol{s}$ is $\{8 \times 8 = 64\}$-dimensional which is unable to illustrate visually. But, the flatness of the distribution can be estimated from the histogram of $s_{i,j}$. If the distribution of $s_{i,j}$ were nearly flat, there would be very little possibility for data compression because of high entropy. Therefore, we have to examine the distribution on real images. Figure 4 is the result-

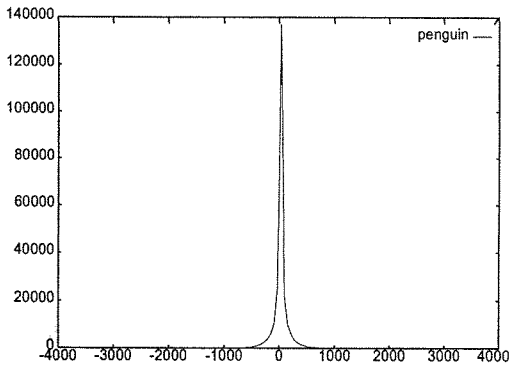

Figure 4. Distribution of coefficients to be encoded.

ing distribution obtained from an image outside of the training data. The horizontal axis shows values of s_i. The vertical axis shows the number of appearances, i.e., the frequency. As can be observed, this distribution is far from flat. It is highly super-Gaussian reflecting the nature of the ICA transformation of images. This means that most of s_i are centered around zero. Few important numbers are distant from zero. Therefore, we can judge that the distribution of $\boldsymbol{s}$ is very sparse. This property encourages us with the anticipation that the encoding for data compression will be effective.

5.3. Image Compression

Here, we discuss the case of variable-length coding based upon the run-length and Huffman coding. The source to be compressed is the matrix

$$\boldsymbol{S} = [\boldsymbol{s}_1, \cdots, \boldsymbol{s}_M]. \tag{21}$$

Here, $\boldsymbol{s}_i$ is the vector coefficient for one patch in the source image. Therefore, $M = 3750$ for a 600×400 pixel image since $(600/8) \times (400/8) = 3750$. The vector $\boldsymbol{s}_i$ is quantized in group. The quantization is set to be granular if a coefficient is for a low spatial frequency. On the other hand, the quantization is rough if the coefficient is for higher frequencies. Quantized zeroes appear frequently because of the sparseness explained in Section 5.2. Then, we denote the resulting coefficient matrix by $\tilde{\boldsymbol{S}}$. We found that quantized zeroes run consecutively if we raster scan this $\tilde{\boldsymbol{S}}$ *vertically* because of the property explained in Item (c) of Section 5.1: High-frequency bases correspond to noisy patterns. Therefore, run-length coding is effective. Huffman coding is used for non-zeroes.

Figure 5. Uncompressed image.

Figure 6. Compressed image 1.

Figures 5 is a source image selected from [12]. Figure 6 is a compressed image by this paper's method. The compressed image has the performance of $\mathrm{SNR}_{\mathrm{pp}} =$

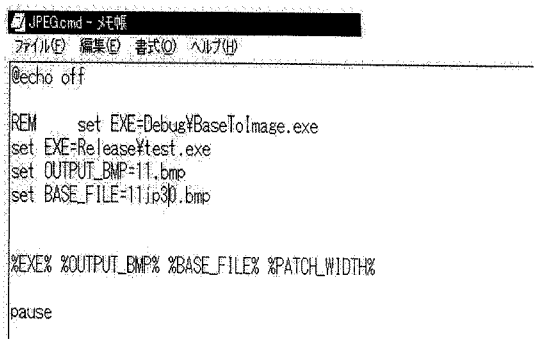

Figure 7. Compressed image 2.

35.2 dB at 1.24 bit/pixel. Figure 7 is another compressed image containing charecters. This image is not a set of outline fonts but is obtained from a computer display. This image has the performance of $\mathrm{SNR}_{\mathrm{pp}} = 34.8$ dB at 1.28 bit/pixel.

More experiments besides Figures 6 and 7 were tried. We can conclude that the image compression based upon the ICA bases designed by this paper's method shows the excellent performance.

6. Concluding Remarks

The main purpose of this paper was to show that

(i) The permutation indeterminacy of the ICA can be avoided. The resulting bases can be used in engineering applications, particularlly for image compression.

(ii) The ICA bases learned from images extract important information. Such bases can be applied to reconstruct unlearned images.

(iii) Coefficients for the reconstruction can be used for the image compression.

It was possible to show that the image compression based upon the ICA bases is promising. We shall have the immediate sophistication of this paper's study as follows:

(a) Incorporation of better lossless coding on coefficients.

(b) Applications to color image compression.

Acknowledgment

This study was supported by the Grant-in-Aid for Scientific Research #15300077 and by the Productive ICT Academia of the 21st Century COE Program granted to Waseda University. The authors are grateful to Mr. S. Imahara of Toshiba Co. for his early contributions.

References

[1] C. Jutten and J. Herault, "Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture," Signal Processing, vol. 24, pp. 1-10, 1991.

[2] J-F. Cardoso and A. Souloumiac, "Blind beamforming for non Gaussian signals," IEE Proceedings F, vol. 140, pp. 362-370, 1993.

[3] A. J. Bell and T. J. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution," Neural Computation, vol. 7, pp. 1129-1159, 1995.

[4] H. H. Yang and S. Amari, "Adaptive online learning algorithm for blind separation: Maximum entropy and minimum mutual information," Neural Computation, vol. 9, pp. 1457-1482, 1997.

[5] A. Hyvärinen "Fast and robust fixed-point algorithms for independent component analysis," IEEE Trans. Neural Networks, vol. 10, pp. 626-639, 1999.

[6] Y. Matsuyama, N. Katsumata, Y. Suzuki and S. Imahara, "The α-ICA algorithm," Proc. Int. Workshop on Independent Component Analysis, pp. 297-302, 2000.

[7] Y. Matsuyama, S. Imahara and N. Katsumata, "Optimization transfer for computational learning," Proc. Int. Joint Conf. on Neural Networks, vol. 3, pp. 1883-1888, 2002.

[8] A. Hyvärinen, P.O. Hoyer and M. Inki, "Topographic independent component analysis," Neural Computation, vol. 13, pp. 1527-1558, 2001.

[9] Y. Matsuyama and S. Imahara, "The α-ICA algorithm and brain map distillation from fMRI images," Proc. Int. Conf. Neural Info. Processing, vol. 2, pp. 708-713, 2000.

[10] Y. Matsuyama, N. Katsumata anr R. Kawamura, "Independent component analysis minimizing convex divergence," Proc. ICANN/ICONIP03, Lecture Notes in Computer Science, Springer-Verlag, 2003.

[11] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," Nature, vol. 381, pp. 607-609, 1996.

[12] "Master Photo 75,000," H^2 Soft Co., 1999.

ICA Photographic Encoding Gear: Image Bases Towards IPEG

Yasuo MATSUYAMA, Hiroaki KATAOKA, Naoto KATSUMATA, and Keita SHIMODA
Department of Computer Science, Waseda University,
Tokyo, 169-8555 Japan
E-mail: yasuo2@waseda.jp, pp_kataoka@moegi.waseda.jp, naoto@katsumata.name, kei215@asagi.waseda.jp

Abstract— **Independent component analysis (ICA) is applied to image coding. There, new design methods for ICA bases are presented. The new feature of this learning algorithm includes the weak guidance, or decreasing supervisory information. The weak guidance reduces the permutation indeterminacy which is unavoidable in usual ICA algorithms. In view of the image compression, this effect corresponds to the generation of image bases honoring the space frequency's neighborhood and 2-D ordering. Following the presentation of this learning algorithm, experiments are performed to obtain serviceable ICA bases. Finally, image compression and restoration are demonstrated to show the eligibility for "image. ipeg." Other applications such as image retrieval are also commented.**

I. INTRODUCTION

Independent Component Analysis (ICA) [1] finds generic components which can reproduce source data. ICA algorithms find such components as unsupervised iterative learning. Applications can be quite wide including images, speech, music, fMRI data and so on. Since physical entities of data can be versatile, ICA has received growing attention from the communities of multimedia processing, communications, biomedical engineering and others. This paper contributes to these fields, especially to the areas handling images. More precisely, we present a novel method to obtain ICA image bases and their applications to image compression.

For the above purpose, the text of this paper is organized as follows. Section II gives the formalization of ICA. The role of ICA bases is explained. There, the indeterminacy of the permutation and the amplitude, which essentially exist in the ordinary ICA, is explained.

In Section III, pre-processing and orthonormalization are introduced. Then, ordinary ICA algorithms are explained as the first step to assist later sophistication on improved methods.

In Section IV, the result of the ordinary ICA bases is used as an initial value for this learning step. Then, the ICA learning algorithm with *weak guidance* is presented. The weak guidance is a supervisory mechanism which can inject designer's specification to usual unsupervised learning. Such partial supervision is effective to the reduction of the permutation indeterminacy. This step is necessary for the application of ICA bases.

In Section V, smoothing of the obtained ICA bases using the Gabor function is tried. This is motivated by the desire to understand the relationship between compression and feature extraction.

In Section VI, experiments on digital images are executed. The ICA image bases are successfully generated in a 2-D aligned manner by reflecting spatial frequencies. Experiments show that the ICA bases are promising in the image compression as the theory predicts. Results are expected to lead to "*image.ipeg*[1]." On the other hand, smoothed bases using the Gabor function sometimes fails in image restoration. This is compatible with our anticipation that this basis set would rather fit to feature extraction than compression.

Section VII gives concluding remarks for further steps. The concept of "*image.ipeg*" is not only for data compression but also for data retrieval. This issue is also commented.

II. PROBLEM FORMULATION OF ICA

Here, the formulation of ICA is given to explain what the basis set and the indeterminacy are.

A. Mixture of Independent Components

In the problem of ICA, a vector random variable

$$\boldsymbol{x} = [x_1, \cdots, x_n]^T \tag{1}$$

is assumed to be generated by another random variable

$$\boldsymbol{s} = [s_1, \cdots, s_n]^T \tag{2}$$

by the following mixture.

$$\boldsymbol{x} = \boldsymbol{A}\boldsymbol{s} = [\boldsymbol{a}_1 \cdots, \boldsymbol{a}_n]\boldsymbol{s} = \textstyle\sum_{i=1}^{n} \boldsymbol{a}_i s_i \tag{3}$$

The matrix $\boldsymbol{A}$ and the vector $\boldsymbol{s}$ are both unknown except for the following information.

(a) The components s_i and s_j are independent each other for $i \neq j$.
(b) The components s_i, $(i = 1, \cdots, n)$, are non-Gaussian except for at most one i.

Under the above conditions, we want to estimate a demixing matrix

$$\boldsymbol{W} = \boldsymbol{\Lambda}\boldsymbol{\Pi}\boldsymbol{A}^{-1} \tag{4}$$

so that the components y_i, $(i = 1, \cdots, n)$, of

$$\boldsymbol{W}\boldsymbol{x} \stackrel{\text{def}}{=} \boldsymbol{y} = [y_1, \cdots, y_n]^T \tag{5}$$

[1]The file extension "*ipeg*" may be abbreviated to "*ipe*" in case that the usage of "*ipg*" conflicts with other existing formats. This truncation is the same as "*htm*" for '*html*."

are independent each other. Here, $\mathbf{\Lambda}$ is a nonsingular diagonal matrix which decides components' scaling, and $\mathbf{\Pi}$ is a permutation matrix. These matrices are unknown too. This property is called the indeterminacy, which essentially exists in the ICA formulation. One purpose in this paper is to show a learning mechanism to reduce such indeterminacy in view of image compression.

B. ICA bases

A set of vectors which decompose and recover the source vector is regarded as the bases. In the problem of ICA, column vectors of $\boldsymbol{W}^{-1} \stackrel{\text{def}}{=} \boldsymbol{U}$ can be interpreted as bases since the following equality holds for the observed data $\boldsymbol{x}$.

$$\boldsymbol{x} = \boldsymbol{U}\boldsymbol{y} = [\boldsymbol{u}_1, \cdots, \boldsymbol{u}_n]\boldsymbol{y} = \sum_{i=1}^{n} \boldsymbol{u}_i y_i \tag{6}$$

Hereafter, $\boldsymbol{U}$ is re-expressed by $\boldsymbol{A}$ in order to save notational alphabets.

$$\boldsymbol{x} = \boldsymbol{A}\boldsymbol{y} = [\boldsymbol{a}_1, \cdots, \boldsymbol{a}_n]\boldsymbol{y} = \sum_{i=1}^{n} \boldsymbol{a}_i y_i \tag{7}$$

We carefully apply this convention so that there is no confusion.

When an ICA basis $\boldsymbol{a}_i$ appears in image processing, it goes as a two dimensional patch: $\{\{a_i(x,y)\}_{x=1}^{m}\}_{y=1}^{m}$. That is, each pixel at (x,y) is modeled by

$$I(x,y) = \sum_{i=1}^{n} a_i(x,y) y_i, \tag{8}$$

where $n = m^2$ for a square region. Once the ICA bases are learned from training data, they are fixed. Such a superposition is illustrated in Figure 1. It is important comment here that $\{y_i\}_{i=1}^{n}$ are encoded for image compression, and *the coding method affects the designer's specification on the ICA bases.* As can be observed from Equation (8) and Figure 1, the

Fig. 1. Image Restoration by ICA-Basis Superposition

idea of the ICA restoration and compression of source data is fundamental [2]. But, this paper's sophisticated generation and utilization of the two-dimensional ICA basis array are original.

III. ICA Learning Algorithms

A. Preprocessing and Orthonormalization

Preprocessing of source data is very effective for later learning phase. Since we use a fixed set of ICA image patch, preprocessing has no effect on coding delay. Computationally expensive methods can be applied as long as they fall in the realm of contemporary PC power.

Observed data are preprocessed in the following way which helps the estimation of $\boldsymbol{W}$ to converge properly.

1) [Mean and variance normalization] Observed data are normalized to have the zero mean and the unit variance.
2) [Whitening] Observed data are then transformed to $\boldsymbol{z} = \boldsymbol{V}\boldsymbol{x}$ so that $\mathcal{E}[\boldsymbol{z}\boldsymbol{z}^T] = \boldsymbol{I}$. Here, $\mathcal{E}$ stands for the expectation. We use $\boldsymbol{V} = \boldsymbol{D}^{-1/2}\boldsymbol{E}^T$ in our experiments. Here, $\boldsymbol{D}$ is a diagonal matrix whose elements are eigenvalues of $\mathcal{E}[\boldsymbol{x}\boldsymbol{x}^T]$. $\boldsymbol{E}$ is the matrix whose columns are corresponding eigenvectors.
3) [Orthonormalization] Another transformation is the orthonormalization: $\boldsymbol{W} \leftarrow (\boldsymbol{W}\boldsymbol{W}^T)^{-1/2}\boldsymbol{W}$. This is an expensive computation, however, the merits of $\boldsymbol{U} = \boldsymbol{W}^T$ and $\boldsymbol{W}^T\boldsymbol{W} = \boldsymbol{I}$ are obtained.

B. First-Stage Learning Algorithm

We use ordinary or unsupervised ICA learning as a process to generate a viable set of ICA bases which can be *exported* to later ICA phases.

Any estimation or learning of $\boldsymbol{W}$ from observed data is summarized by the following iteration:

$$\boldsymbol{W}^{\text{new}} = \boldsymbol{f}(\boldsymbol{W}^{\text{old}}), \tag{9}$$

or equivalently,

$$\boldsymbol{W}^{\text{new}} = \boldsymbol{W}^{\text{old}} + \Delta\boldsymbol{W}. \tag{10}$$

The updated vector $\boldsymbol{W}^{\text{new}}$ can be obtained by optimizing statistical measures for the independence [3] $\sim$ [8]. For the generation of the *initial* ICA bases, any of these methods are equally viable. Therefore, we omit rehashing this subject, but give a bridge to image processing.

We are given M sample image patches rather than an abstract random variable in a probability space. Therefore, we need to write down these samples in matrix forms:

$$\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_M], \tag{11}$$

$$\boldsymbol{S} = [\boldsymbol{s}_1, \cdots, \boldsymbol{s}_M], \tag{12}$$

$$\boldsymbol{Y} = [\boldsymbol{y}_1, \cdots, \boldsymbol{y}_M]. \tag{13}$$

Thus, the data generation model is expressed by

$$\boldsymbol{X} = \boldsymbol{A}\boldsymbol{S}. \tag{14}$$

Then, the first-stage learning algorithm becomes as follows.

[First-stage learning algorithm]

[Step 1 (Preprocessing 1)]
Obtain a sample matrix $\boldsymbol{X}$ as a training data set. Normalize each column vector to be zero mean and unit variance.

[Step 2 (Preprocessing 2: Whitening)]
Obtain Whitening Matrix $\boldsymbol{V}$ from $\boldsymbol{X}$, and compute $\boldsymbol{Z} = \boldsymbol{V}\boldsymbol{X}$.

[Step 3 (Initialization)]
Choose an orthonormalized initial value for $\boldsymbol{W}$.

[Step 4 (Update 1)]
Update $\boldsymbol{W}$ by (9) or (10).

[Step 5 (Update 2)]
Orthonormalize the matrix $\boldsymbol{W}$.

[Step 6 (Convergence check)]

Check to see if convergence is achieved. Otherwise repeat Steps 4 and 5.

[Step 7 (Resulting matrices)]
Resulting matrices are obtained by

$$\boldsymbol{W}_{\text{stage1}} = \boldsymbol{W}\boldsymbol{V}, \tag{15}$$

and

$$\boldsymbol{A}_{\text{stage1}} = (\boldsymbol{W}\boldsymbol{V})^{-1} = \boldsymbol{V}^{-1}\boldsymbol{W}^{T}. \tag{16}$$

We commnent here again that

(i) Resulting ICA bases at this first-stage are exported to the next ICA phase.
(ii) The resulting ICA bases here still inherits severe permutation indeterminacy. We need further learning algorithms which do not suffer from this indeterminacy.
(iii) If we dare to use the ICA bases here for the image compression, the matrix

$$\boldsymbol{Y}_{\text{data}} = \boldsymbol{W}_{\text{stage1}}\boldsymbol{X}_{\text{data}} \tag{17}$$

is encoded to $\hat{\boldsymbol{Y}}_{\text{data}}$. Decoded is then

$$\hat{\boldsymbol{X}}_{\text{data}} = \boldsymbol{A}_{\text{stage1}}\hat{\boldsymbol{Y}}_{\text{data}}. \tag{18}$$

Equations (17) and (18) remain the same for the compression and restoration in later ICA bases except for the suffix *stage1*.

IV. Learning Under Weak Guidance

A. Indeterminacy Reduction I

The above $\boldsymbol{A}_{\text{stage1}}$ could be used as a set of image compression bases, if one would untiringly check manually the whole matrix pattern, and if low performance is bearable. But, the bases are more suitable for the image compression if they reflect spatial frequencies in an aligned manner. Therefore, we consider to use the resulting image bases as an initial set for further learning modification. This is allowed since the image bases need *not* be computed on-line but to be stored in the encoder-decoder pair. There is one more evidence to support this: All computation in this paper can be carried out by a conventional personal computer, which will be understood in Section VI.

The first step to obtain an aligned image basis set is to modify the matrix $\boldsymbol{W}_{\text{stage1}}$ by using the topographic ICA [9]. In this case, (10) is used with the following computation:

$$\Delta \boldsymbol{w}_i = \eta E[\boldsymbol{z}(\boldsymbol{w}_i^T\boldsymbol{z})r_i], \tag{19}$$

$$r_i = \textstyle\sum_{k=1}^{n} h(i,k)G'(\sum_{j=1}^{n} h(k,j)(\boldsymbol{w}_j^T\boldsymbol{z}^2)). \tag{20}$$

On the choices of $G(y)$ and $h(i,j)$, readers are requested refer to [9]. Hereafter, the update matrix by (19) is denoted by $\Delta\boldsymbol{W}_{\text{tp}}$.

It is important to comment here that $\Delta\boldsymbol{W}_{\text{tp}}$ is an update term by the unsupervised learning. There is no explicit engineering specification incorporated in this term. This is a neat learning device, however, we need further ability:

1) More separation of high frequency and low frequency in bases is necessary.
2) The non-floating center of the frequency pattern is necessary.

Readers will understand these items from experiments in Section VI.

B. Indeterminacy Reduction II: Weak Guidance

Resulting ICA bases by the Indeterminacy Reduction I show an intriguing visual pattern as a topographic map. But, a very important indeterminacy is not yet resolved.

1) Positions of low frequency bases are often separated. This leads to information loss.
2) The center of the low frequency area is still floating. A human can instantly find the position of the central basis corresponding to the lowest spatial frequency, however, machines can not do so instantly. Besides, this property could lead to another information loss if the central area happens to be off-centered in the topographic map.

Thus, we need further important mechanisms to reduce such indeterminacy. This is the method of *weak guidance* as a partially supervised learning. Such a method was first used in the distillation of brain maps from fMRI data [10], [11], and then in the precursor of this paper [12].

[Weak Guidance]

The weak guidance[2] is a method to inject supervisory information to unsupervised or self-organizing mechanisms. First, we prepare a teacher signal, or a reference pattern, as a matrix $\bar{\boldsymbol{R}}$. Then, we compute $\boldsymbol{U} = \boldsymbol{V}^{-1}\boldsymbol{W}^{T}$. The increment by the teacher signal is

$$\Delta\boldsymbol{U} = \boldsymbol{V}\{\lambda(\bar{\boldsymbol{R}} - \boldsymbol{U})\}. \tag{21}$$

Here, λ is a learning parameter. Then, the update term for the weak guidance is computed by

$$\Delta\boldsymbol{W}_{\text{wg}} = -\boldsymbol{W}\Delta\boldsymbol{U}\boldsymbol{W}. \tag{22}$$

Readers are requested refer to [10] or [11] for the derivation of (21) and (22).

C. Total Learning Algorithm

By the preceding preparations, the total learning algorithm to obtain the ICA bases can be described as follows.

[Total learning algorithm]

[Step 1 (Learning parameters)]
Control rules of the small learning parameters $\eta > 0$ and $\lambda > 0$ are specified. The rules can be arbitrary as long as these parameters decrease to zero. In the experiments, they are set to decrease monotonically.

[Step 2 (Weak guidance)]

[2]This terminology was named after Richard P. Feynman's informal lecture in early 80's on his anticipation of weak force in the quantum computing mechanism.

Compute the updated matrix with the weak guidance

$$\boldsymbol{W} \leftarrow \boldsymbol{W} + \Delta \boldsymbol{W}_{\mathrm{wg}}. \tag{23}$$

[Step 3 (Topographical map)]
Compute the updated matrix with the topographic constraint

$$\boldsymbol{W} \leftarrow \boldsymbol{W} + \Delta \boldsymbol{W}_{\mathrm{tp}}. \tag{24}$$

[Step 4 (Convergence check)]
Check to see if the matrix update is converged. If not, then the iteration is repeated on Steps 2 and 3 after the update of λ, η, and $\boldsymbol{W}$.

In the above description of the algorithm, Steps 2 and 3 are separated. But, their increments can be added to update $\boldsymbol{W}$.

V. Fitting to Gabor Function

As was stated in Section I, the main purpose of this paper is set to data compression on images. But, as can be observed from later experiments, ICA image bases have similarities to receptive field properties too [13]. Therefore, we try a little bit of digression to compute a set of smoothed ICA bases using the Gabor function [14]. This is motivated by our desire to understand the relationship between compression and feature extraction.

Let $g(x, y|\Phi)$ be a Gabor function with the parameter set Φ. Then, smoothed ICA bases are obtained by the minimization of the cost function $E_i(\Phi)$:

$$E_i(\Phi) = \sum_x \sum_y |a(x, y) - g(x, y|\Phi)|^2. \tag{25}$$

This will give a set of smoother bases. Therefore, some of busy patterns may not be restored by such bases. Experiments will give evidences on such trend.

VI. Experimental Results

All necessary tools were given in the preceding sections. We can now apply them to real images.

A. ICA Image Bases with Self-Alignment for Image Coding

1) Training Data: Source data of 10,000 training samples from 10 typical images were prepared. Thus, the training data contain patches from natural images, screen text images, and animations. Each patch consists of 8×8 pixels so that the size is compatible with usual JPEG.

2) Bases by Plain ICA: Figure 2 illustrates the ICA bases obtained by usual unsupervised ICA. It can be observed that there is no consistent relationships among image bases. By inspection, we can find that low frequencies and high frequencies are covered luxuriously. On the other hand, middle frequencies which are responsible for "twilight zones" need more bases. Precise ordering of the bases by inspection is difficult.

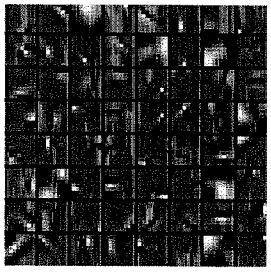

Fig. 2. Image Bases by Plain ICA

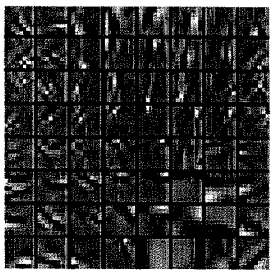

Fig. 3. Image Bases by Topographic ICA

3) Bases by Topographic ICA: Figure 3 illustrates ICA bases obtained by the topographic method alone, i.e., without the weak guidance. As can be observed, the basis of the lowest spatial frequency is located off-centered in the two dimensional array. This position is floating. Therefore, the human perception is still necessary to identify where the exact center is. Besides, the obtained ICA bases are inefficient since the center is near the corner of the array. The floating center property remains even if we use a 2-D torus,

4) Bases by ICA with Weak Guidance: The weak guidance is imposed to the ICA learning through the patterns illustrated in Figure 4. The location of the first one is at the northwest of the center. The second one is placed at the southeast of the center. Then, we obtain the self-aligned ICA bases by our

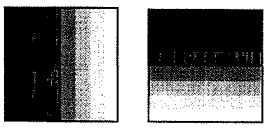

Fig. 4. Suprtvisory Pattern

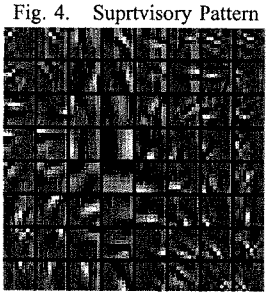

Fig. 5. Self-Aligned Image Bases by the Weak Guidance

weak guidance method as in Figure 5. The first basis is located at the northwest of the four central bases as we specified.

Low-frequency bases are packed around the center of the two-dimensional array. High-frequency bases are relegated to the frames and corners. This was specified to be so by virtue of the weak guidance. We call such a class of bases the *ICA ripplet set*, or simply the *ripplet basis*. The ripplet basis is readily applicable to the image compression due to the following properties.

(a) Bases are well-balanced in a 2-D array because of the centering by the weak guidance. Such a class of harmony will contribute to the efficiency of coding appearing in later sections.

(b) Ordering from low to high spatial frequencies is clear. This ordering starts from the center and ends at the frames. High-frequency bases corresponding to noisy patterns are located in the rings around the frames. This is the very property the 2-D ripplet basis. Linear sorting of plain ICA bases do not lead to such balanced rings. The merit of the ring pattern will be exploited in Section VI-C.

B. Distribution of Decomposed Signals

We use the self-aligned ICA bases of Figures 5 for the image compression. Source images are reconstructed by using Equation (18). We remind readers here that the relationship on the ICA image bases: $\boldsymbol{A}^{-1} \leftarrow \boldsymbol{W}$. Since the image bases are not altered any more, they are stored in the encoder and the decoder. In the encoder, coefficients y_i are obtained by

$$\boldsymbol{Y} = \boldsymbol{A}^{-1}\boldsymbol{X}. \tag{26}$$

Here, each column vector of $\boldsymbol{X}$ corresponds to an image patch. Then, encoded values of elements y_i in $\boldsymbol{Y}$ are transmitted . If some pdf's of y_i's are inappropriate for data compression, the total coding performance will not be effective. Therefore, we have to examine their distribution on real images. Figures 6 and 7 are resulting distributions obtained from image patches outside of the training data. The horizontal axis shows values of y_i. The vertical axis shows the number of appearances, i.e., the frequency. As can be observed, these distributions are highly super-Gaussian reflecting the nature of the ICA transformation of images. Other y_i's have the same trend. This means that most of y_i are centered around zero. Few important numbers are distant from zero. Therefore, we can judge that the distribution of $\boldsymbol{y}$ is very sparse. This property encourages us with the anticipation that the encoding for data compression will be effective.

C. Image Compression

Here, we discuss the case of variable-length coding based upon the run-length and Huffman coding. The source to be compressed is the matrix

$$\boldsymbol{Y} = [\boldsymbol{y}_1, \cdots, \boldsymbol{y}_M]. \tag{27}$$

Here, $\boldsymbol{y}_i$ is the vector representing one sub-patch in the source image. Here, $M = 3750$ for a 600×400 pixel image since $(600/8) \times (400/8) = 3750$. Each component of the vector $\boldsymbol{y}_i$ is scalar-quantized in group. The quantization is set to be

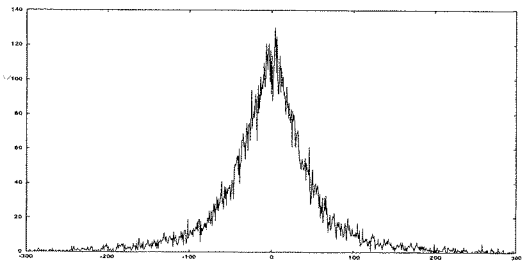

Fig. 6. Distribution of Component y_5

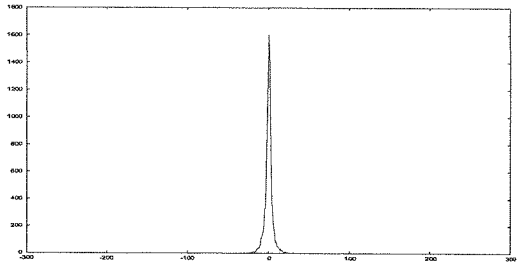

Fig. 7. Distribution of Component y_{20}

granular if a coefficient is for a low spatial frequency. On the other hand, the quantization is rougher if the coefficient is for a high-frequency basis located in the rings near the frames of the ripplet basis. Quantized zeroes appear frequently because of the sparseness explained in Section VI-B. Then, we denote the resulting coefficient matrix by $\tilde{\boldsymbol{Y}}$. We found that quantized zeroes run consecutively if we raster-scan this $\tilde{\boldsymbol{Y}}$ horizontally. This is due to the property explained in Item (c) of VI-A.4 which tells that high-frequency bases correspond to noisy patterns. Therefore, run-length coding is effective. Huffman coding is used for non-zeroes.

Figure 8 is a source image selected from [15]. Figure 9 is a compressed image by this paper's method which is a class of variable-length coding. The compressed image has the performance of $\mathrm{SNR}_{\mathrm{pp}} = 30.0$ dB at 1.00 bit/pixel. Figure 10 is another compressed image. This image has the performance of $\mathrm{SNR}_{\mathrm{pp}} = 30.0$ dB at 1.42 bit/pixel.

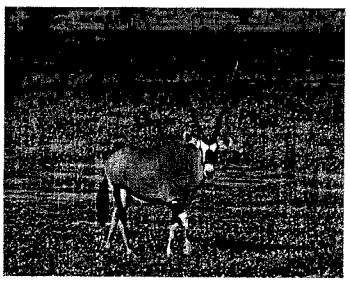

Fig. 8. Uncompressed Image

Numerous experiments were tried besides Figures 9 and 10. We always compared the IPEG performance with the JPEG. Since the JPEG is also a variable length coding with three-level specifications, {*high, medium, low*}-quality, simple distortion-

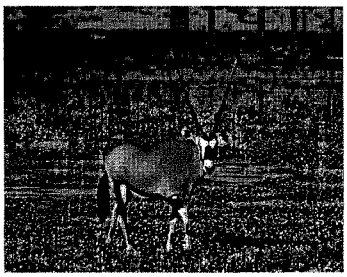

Fig. 9. Compressed Image 1

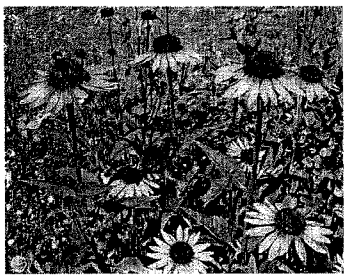

Fig. 10. Compressed Image 2

rate comparisons in SNR often fail to reflect the compression performance appropriately. But, the firm quality was found that the image compression based upon this paper's ICA bases is more effective than the JPEG under the medium-quality button which is the default selection of the JPEG coding.

D. Smoothed Ripplet Basis by Gabor Function

We re-computed the ripplet basis of Figure 5 by minimizing Equation (25). This is to modify the ripplet so that the bases approach to Gabor functions. The resulting ripplet basis is illustrated in Figure 11. As we expected, this ripplet basis is smoother and cleaner than Figure 5. But, such clean smoothness leads to lack of restoration ability for image coding. On the other hand, the Gabor-smoothed ripplet of Figure 11 shows a set of motifs which can decompose and reconstruct image patterns. In the light of such pattern handling, this *motif ripplet basis* can find its own application areas. Such applications include image retrieval and feature extraction.

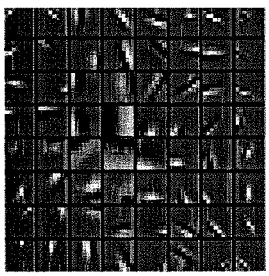

Fig. 11. ICA Bases Fitted to Gabor Function

VII. Concluding Remarks

The main objective of this paper was to show the following:

(a) To construct the total ICA learning algorithm which generates a set of 2-D aligned ICA bases effective to image compression.
(b) To demonstrate this algorithm's ability to generate a desirable basis set, i.e., the ripplet basis.
(c) To show this ripplet's excellent data compression performance on images.

As was presented in the text, all were satisfied. That is, it was possible to show that the image compression based upon this paper's ICA bases is promising. Furthermore, there can be a variety of directions to the next step for this paper's method.

(1) Design of joint ripplet and encoding.
(2) Image retrieval and feature extraction using the motif ripplet.
(3) Applications to other sophisticated image handling.

Acknowledgment

This study was supported in part by the Grant-in-Aid for Scientific Research #15300077 and by the Productive ICT Academia of the 21st Century COE Program of the MEXT (Ministry of Education) granted to Waseda University. The authors are grateful to referees who gave valuable comments for preparing the final version.

References

[1] C. Jutten and J. Herault, "Blind separation of sources, part I: An adaptive algorithm based on neuromimetic architecture," *Signal Processing*, vol. 24, pp. 1-10, 1991.
[2] A. Hyvärinen, "Survey on independent component analysis," *Neural Computing Surveys*, vol. 2, pp. 94-128, 1999.
[3] J-F. Cardoso and A. Souloumiac, "Blind beamforming for non Gaussian signals," *IEE Proceedings F*, vol. 140, pp. 362-370, 1993.
[4] A. J. Bell and T. J. Sejnowski, "An information-maximization approach to blind separation and blind deconvolution," Neural Computation, vol. 7, pp. 1129-1159, 1995.
[5] H. H. Yang and S. Amari, "Adaptive online learning algorithm for blind separation: Maximum entropy and minimum mutual information," Neural Computation, vol. 9, pp. 1457-1482, 1997.
[6] A. Hyvärinen, "Fast and robust fixed-point algorithms for independent component analysis," IEEE Trans. Neural Networks, vol. 10, pp. 626-639, 1999.
[7] Y. Matsuyama, N. Katsumata, Y. Suzuki and S. Imahara, "The α-ICA algorithm," Proc. Int. Workshop on Independent Component Analysis, Espoo, 2000, pp. 297-302.
[8] Y. Matsuyama, S. Imahara and N. Katsumata, "Optimization transfer for computational learning," *Proc. Int. Joint Conf. on Neural Networks*, Honolulu, 2002, vol. 3, pp. 1883-1888.
[9] A. Hyvärinen, P.O. Hoyer and M. Inki, "Topographic independent component analysis," Neural Computation, vol. 13, pp. 1527-1558, 2001.
[10] Y. Matsuyama and S. Imahara, "The α-ICA algorithm and brain map distillation from fMRI images," *Proc. Int. Conf. Neural Info. Processing*, Taejon, 2000, vol. 2, pp. 708-713.
[11] Y. Matsuyama, N. Katsumata and R. Kawamura, "Independent component analysis minimizing convex divergence," *Lecture Notes in Computer Science*, No. 2714, pp. 27-34, Springer-Verlag, 2003.
[12] Y. Matsuyama, R. Kawamura, H. Kataoka, N. Katsumata, K. Tojima, H. Ishijima and K. Shimoda, "Image compression based upon independent component analysis: Generation of self-aligned ICA bases," *Proc. 8th Australian and New Zealand Intelligent Information Systems Conference*, Sidney, 2003, pp. 1-6.
[13] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," Nature, vol. 381, pp. 607-609, 1996.
[14] M. S. Lewicki amd B. A. Olshausen, "Probablistic framework for adaptation and compression of image codes," J. Optical Soc. of America, vol. 16, pp. 1587-1601, 1999.
[15] "Master Photo 75,000," H^2 Soft Co., 1999.

Promoter Recognition for E. coli DNA Segments by Independent Component Analysis

Yasuo MATSUYAMA
Department of Computer Science,
Waseda University, Tokyo, 169-8555 Japan
yasuo2@waseda.jp

Ryo KAWAMURA
Sony-Kihara Research Center Inc.,
Tokyo, 141-0022 Japan
ryo@asagi.waseda.jp

Abstract

A new method for E. coli DNA segment classification on promoters and non-promoters is presented. The algorithm is based on the Independent Component Analysis (ICA). Since the DNA segments are composed of discrete symbols, this paper contains two major steps: (1) Position-dependent transformation of DNA segments to real number sequences, and (2) Applications of the ICA to the E. coli promoter recognition. These steps are related to each other. Therefore, algorithmic explanations are given in detail while referring mutually. The automatic precision of 93.7% is obtained. Since the presented method allows threshold adjustments, twilight-zone data can be further cross-checked individually so that false negatives are reduced.

1. Introduction

DNA sequences contain portions of special functions [1], [2]. The promoter is one of such an important structure which works as a polymerase binding site. Recognition of promoter patterns keeps its importance because of the relationship to the transcription [3]. In this paper, the recognition of E. coli promoter segments is addressed. Existing recognition methods use artificial neural networks [4] and their combination with the expectation maximization algorithm [5]. The new method in this paper, however, is based on the Independent Component Analysis (ICA). On the ICA, we will use our own generalized algorithm [6], [7] derived by the minimization of the convex divergence which is the ultimately general version of the entropy.

The ICA is a statistical learning algorithm for numerical data. On the other hand, DNA sequences are composed of four symbols $\{A, T, G, C\} \stackrel{\text{def}}{=} \mathcal{D}$. Thus, consistent conversions from symbol sequences to real number series are necessary. Therefore, this paper includes a position-dependent conversion based on symbol frequencies. By the ICA with such numerical conversions, the resulting automatic precision of 93.7% is obtained. Since the presented method allows threshold adjustments, twilight-zone data can be further cross-checked individually so that false negatives are reduced.

2. E. coli Promoter Recognition

2.1. Structure and Function of E. coli Promoters

Figure 1 is a conceptual illustration explaining the structure of the E. coli promoter. There are specific sub-

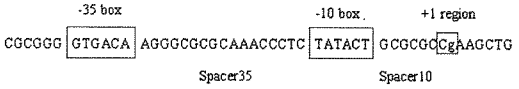

Figure 1. Conserved regions in the E. coli promoter.

configurations in the E. coli promoter. They are called the -35 box, the -10 box, and the +1 region. The transcription starts from the +1 region. The region between the -35 box and the -10 box is called the Spacer35. The region between the -10 box and the +1 region is called the Spacer10. The -35 box and the -10 box have the fixed length of 6 nt[1]. But, their contents may vary. The length of the Spacer35 may vary from 15 to 21 nt. The Spacer10 may vary from 3 to 11 nt length. Thus, there are various promoter patterns for E. coli sequences. Therefore, symbolic pattern matching is not quite appropriate for the promoter analysis, but efficient pattern recognition methods including probabilistic or soft decisions are required.

2.2. Procedure of E. coli Promoter Recognition

Every pattern recognition method contains an off-line training phase (learning phase) and an on-line test phase

1 "nt" stands for nucleotide.

(execution phase). This paper's training procedure for the E. coli promoter recognition is previewed as follows.

[Training Steps: Off-line]

[TR 1] A set of length-adjusted E. coli promoter segments is prepared.
[TR 2.1] The set of -35 boxes is changed to a real valued matrix.
[TR 2.3] A random matrix for the -35 box is generated and juxtaposed to the -35 box matrix.
[TR 2.3] From the total -35 box matrix, the feature of the -35 box is learned by the ICA.
[TR 3.1] The set of -10 boxes is changed to a numerical matrix.
[TR 3.2] A random matrix for the -10 box is generated and juxtaposed to the -10 box matrix.
[TR 3.3] From the total -10 box matrix, the feature of the -10 box is learned by the ICA.
[TR 4.1] The set of the length-adjusted promoters are changed to a real number matrix.
[TR 4.2] A random matrix for the promoters is generated using the ICA results of TR 2.3 and TR 3.3. This random matrix is juxtaposed to the promoter matrix.
[TR 4.3] From the total data matrix, the promoter structure is learned by the ICA.

It is important to note here that the ICA is used three times; on the -35 box, on the -10 box, and on the total segment.

The test phase is previewed as follows.

[Test Steps: On-line]

[TS 1] A segment to be tested is given. This may be a set of segments.
[TS 2] The given segment is length-adjusted[2] by using the ICA matrices for the boxes obtained in the training steps.
[TS 3] The length-adjusted segment is transformed to a real number sequence.
[TS 4] Using the ICA matrices, the segment is judged to contain an E. coli promoter or not. Estimated boxes are obtained here.

In the above training and testing steps, there are novel features distinctive to this paper.

(a) To all aspects of the training and test steps, the Independent Component Analysis (ICA) is related.

(b) On the conversion of symbols to real numbers, position-dependent base frequencies are used. This is not a naive transformation of symbols to unit vectors.

2 The length adjustment in this paper has a different purpose from ClustalW, BLAST, and so on.

(c) In the training data for boxes and total segments, random segments are juxtaposed to pure data. This is related to the data augmentation or the bootstrap method.

(d) The length-adjustment uses learned ICA matrices.

3. Independent Component Analysis

3.1. Mixture of Independent Components

In the problem of the Independent Component Analysis (ICA), observed or given data are assumed to be an unknown mixture of unknown data. That is, the observed data $\mathbf{x}(t) = [x_1(t), \cdots, x_N(t)]^T$ is generated from unknown source $\mathbf{s}(t) = [s_1(t), \cdots, s_N(t)]^T$ by

$$\mathbf{x}(t) = \mathbf{A}\mathbf{s}(t) = [\mathbf{a}_1, \cdots, \mathbf{a}_N]\mathbf{s}(t) = \sum_{i=1}^{N} \mathbf{a}_i s_i(t), \quad (1)$$

as is illustrated in Figure 2. We want to estimate $\mathbf{s}(t)$ and $\mathbf{A}$ by observing only $\mathbf{x}(t)$. The ICA algorithm gives an estimation of $\mathbf{A}^{-1}$ as a de-mixing matrix $\mathbf{W}$. The vector $\mathbf{y}(t) = \mathbf{W}\mathbf{x}(t)$ has de-mixed, or independent components. In this problem setting, we are allowed to assume that the components of $\mathbf{s}(t)$ are independent each other.

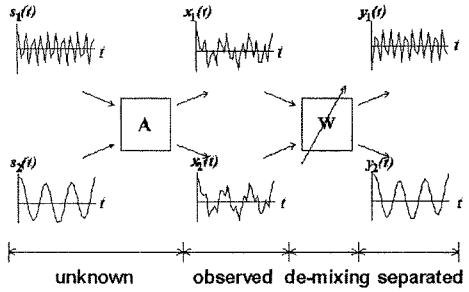

Figure 2. ICA structure ($N = 2$).

In our application of ICA to the promoter recognition, the data $\mathbf{x}(t)$ come from -35 boxes, -10 boxes, and total segments of the E. coli DNA. Therefore, three de-mixing matrices, say $\mathbf{W}_{-35}$, $\mathbf{W}_{-10}$, and $\mathbf{W}_{\text{total}}$ will be learned from given training sets.

To be precise, available information on the mixing structure is only the following: (a) The components s_i and s_j are independent each other if $i \neq j$. (b) The components $s_i(t)$, $(i = 1, \cdots, N)$, are non-Gaussian except for at most one i.

Given such assumptions, we want to estimate a de-mixing matrix $\mathbf{W} = \Lambda\Pi\mathbf{A}^{-1}$ so that the components $y_i(t)$, $(i = 1, \cdots, N)$, of

$$\mathbf{W}\mathbf{x}(t) \stackrel{\text{def}}{=} \mathbf{y}(t) = [y_1(t), \cdots, y_N(t)]^T \quad (2)$$

are independent each other. Here, Λ is a nonsingular diagonal matrix which decides each component's scale, and Π is a permutation matrix. These matrices are unknown too. This property is called the indeterminacy.

3.2. ICA Bases

Column vectors $\mathbf{u}_i$ of $\mathbf{U} \stackrel{\text{def}}{=} \mathbf{W}^{-1}$ are interpreted as ICA bases. This is because the observed data $\mathbf{x}$ is expressed by the weighted summation of the de-mixed components:

$$\mathbf{x}(t) = \mathbf{U}\mathbf{y}(t) = [\mathbf{u}_1, \cdots, \mathbf{u}_N]\mathbf{y}(t) = \sum_{i=1}^{N} \mathbf{u}_i y_i(t). \quad (3)$$

The terminologies *"ICA bases"* and *"DNA bases"* should not be confused. They are totally different concepts. The ICA basis is the very one which represents the promoter structure. This will be illustrated as an experimental result in Section 6.2 (see Figure 3).

3.3. Derivation of the ICA Algorithm

Let $p(\mathbf{y}) = p(y_1, \cdots, y_N)$ be a joint probability density, and $q(\mathbf{y}) = \prod_{i=1}^{N} q_i(y_i)$ be a product probability density. Then, the independence is obtained by the minimization of the following cost function.

$$\begin{aligned} I_f(\bigwedge_{i=1}^{N} Y_i) &\stackrel{\text{def}}{=} D_f\left(p(y_1, \cdots, y_N) \| \prod_{i=1}^{N} q_i(y_i)\right) \\ &\stackrel{\text{def}}{=} D_f(p(\mathbf{y}) \| q(\mathbf{y})) = D_g(q(\mathbf{y}) \| p(\mathbf{y})) \\ &= \int_{\mathcal{X}} p(\mathbf{x})\, g\left(\frac{\det(\mathbf{W}) q(\mathbf{y})}{p(\mathbf{x})}\right) d\mathbf{x} \end{aligned} \quad (4)$$

Here, $D_f(p\|q)$ is the convex divergence [8] between p and q in terms of a twice differentiable convex function $f(r)$ with $f(1) = 0$. The dual function g is defined by $g(r) = rf(1/r)$. Note that a special case of the convex divergence is the Kullback-Leiber divergence or the average mutual information. Further special case is the Shannon's entropy.

By computing the negative gradient of $I_f(\bigwedge_{i=1}^{N} Y_i)$, the increment $\tilde{\Delta}_f \mathbf{W}$ of the following ICA learning equation can be obtained [6], [7]:

$$\mathbf{W}(t+1) = \mathbf{W}(t) + \tilde{\Delta}_f \mathbf{W}, \quad (5)$$

where t is the iteration index for learning[3], and

$$\begin{aligned} \tilde{\Delta}_f \mathbf{W} &= -\rho \frac{\partial I_f}{\partial \mathbf{W}} \mathbf{W}^T \mathbf{W} \\ &= \rho \Big[\left\{ \mathbf{I} - E_{p(\mathbf{y})}[\vec{\varphi}(\mathbf{y})\mathbf{y}^T] \right\} \mathbf{W} \\ &\qquad + \mu \left\{ \mathbf{I} - E_{q(\mathbf{y})}[\vec{\varphi}(\mathbf{y})\mathbf{y}^T] \right\} \mathbf{W} \Big]. \end{aligned} \quad (6)$$

Here, $\vec{\varphi}(\mathbf{y})$ is a nonlinear vector function such as $\varphi_i(\mathbf{y}) = y_i^3$ or $\varphi_i(\mathbf{y}) = \tanh(y_i)$. The coefficient ρ is a small positive number called the learning rate. μ is a non-negative number for the effect of the acceleration on the learning. $E_{p(\mathbf{y})}[\cdot]$ and $E_{q(\mathbf{y})}[\cdot]$ are expectations with respect to the suffixd probability densities. Both are computed from given data. Since $q(\mathbf{y})$ is the unknown target, this quantity is regarded as a time-shifted version of $p(\mathbf{y})$ in programming.

3 This should not be mistaken for the sample data index.

3.4. Sample Data and Pre-processing

In actual data processing, we are given samples of random vectors $\mathbf{x}(t)$, $(t = 1, \cdots, M)$. We write down the set of samples in matrix forms: $\mathbf{X} = [\mathbf{x}(1), \cdots, \mathbf{x}(M)]$, $\mathbf{S} = [\mathbf{s}(1), \cdots, \mathbf{s}(M)]$, $\mathbf{Y} = [\mathbf{y}(1), \cdots, \mathbf{y}(M)]$. Thus, the data generation model (1) is expressed by $\mathbf{X} = \mathbf{AS}$. Also, the de-mixed result is expressed by $\mathbf{Y} = \mathbf{WX}$. Then, the expectation $E_{p(\mathbf{x})}[\cdot]$ is replaced by $\sum_{j=1}^{M}[\cdot]$.

Usually, observed data $\mathbf{X}$ is pre-processed so that the separability into independent components is increased. The first pre-processing is to make the data to be zero-mean. Another important pre-processing is the whitening using the covariance matrix. On these standard pre-processing strategies, readers are requested to refer to ICA literatures, e.g., [7].

4. Parts of Training Steps

4.1. Learning on De-mixing Matrices for Boxes

4.1.1. Numerical Expression of Training Data for Boxes

First, the training set of -35 boxes are changed to a numerical matrix by the following way.

(a) For the training steps, length-adjusted promoters are given. In the later experiment, the total number is $N_p = 106$ drawn from [4]. Each -35 box is 6 nt length.

(b) For each position of the -35 box, normalized $\{A, T, G, C\}$-frequencies are count. This position-dependent count generates a 4×6 matrix, which we call the Table$_{-35}$.

(c) From the training data of -35 boxes, we have an $(N_p - k_{-35}) \times 6$ matrix. Here, $N_p - k_{-35}$ means the removal of k_{-35} identical patterns. In the training set, $N_p - k_{-35} = 72$. Then, each entry of the -35 boxes is changed to a numeral by using Table$_{-35}$. The resulting matrix is called $\mathbf{B}_{-35}$. The matrix $\mathbf{B}_{-35}$ will be used as a core part for obtaining the ICA de-mixing matrix $\mathbf{W}_{-35}$.

Next, the same procedure is tried for -10 boxes to obtain the ICA de-mixing matrix $\mathbf{W}_{-10}$.

4.1.2. Random Matrix Juxtaposition for Learning on $\mathbf{W}_{-35}$ and $\mathbf{W}_{-10}$ For obtaining the de-mixing matrix $\mathbf{W}_{-35}$, an artificially generated random matrix is juxtaposed to the data matrix $\mathbf{B}_{-35}$. This is a kind of data augmentation.

(a) A random data matrix of $(N_p - k_{-35}) \times 6$ is prepared. Here, each entry is drawn from $\{A, T, G, C\} = \mathcal{D}$.

(b) Using the Table$_{-35}$, each entry is changed to numerals. This matrix is called $\mathbf{C}_{-35}$.

(c) By juxtaposing $\mathbf{B}_{-35}^T$ and $\mathbf{C}_{-35}^T$, we have a data matrix of size $6 \times 2(N_p - k_{-35})$. This is called $\mathbf{X}_{-35}$.

(d) The matrix $\mathbf{X}_{-35}$ is preprocessed for the ICA to be the zero mean and the unit covariance. The resulting matrix is renamed $\mathbf{X}_{-35}$.

(e) Using the data matrix $\mathbf{X}_{-35}$, the ICA leaning is carried out. Then, the de-mixing matrices $\mathbf{W}_{-35}$ and the de-mixed data matrix $\mathbf{Y}_{-35}$ are obtained.

For $\mathbf{W}_{-10}$ and $\mathbf{Y}_{-10}$, the same procedure using $\mathbf{B}^T_{-10}$ and $\mathbf{C}^T_{-10}$ is executed.

4.2. Segment Length Adjustment

On the -35 box and -10 box, length adjustments were not needed since their lengths are fixed to 6 nt. But, on the processing of the total promoter region, appearing later in Section 4.3, a length adjustment will become necessary. This is because the Spacer35 and the Spacer10 are variable-length. The algorithm is based on [4]. But, our method uses ICA results of $\mathbf{W}_{-35}$, $\mathbf{Y}_{-35}$ $\mathbf{W}_{-10}$ and $\mathbf{Y}_{-10}$, which are obtained in advance.

The following steps generate length-adjusted segments.

(a) A segment to be length-adjusted is given with an identified start point.

(b) Looking back from the starting point, find the best ending point of the -35 box in the region $[15\cdots21]$. The best position is identified by the maximum inner product using the column vectors $\mathbf{y}_{-35}(k)$, $k = 1, \cdots, 7$. Here, 7 appears as the zone length of the possible ending point. The mechanism of the maximum inner product will be explained in detail in 4.3.2.

(c) The best ending point of the -10 box is identified in the same way using the column vectors $\mathbf{y}_{-10}(k)$ in the region $[3\cdots11]$.

(d) Gaps are inserted so that the total length becomes 65 nt [4].

4.3. Feature Extraction for Total Promoter Structure by the ICA

4.3.1. Numerical Expression for Promoters First, the training set of promoters is changed to a numerical matrix by the following way: (a) There are $N_p = 106$ training promoters with 65 nt length. (b) For each column, the normalized frequencies on each position in the 65 nt length are obtained. This generates a table of the size 5×65 since the promoter sequence contains $\{A, T, G, C, -\} \stackrel{\text{def}}{=} \mathcal{D}^+$. This frequency table is called the $\text{Table}_{\text{promoter}}$. (c) By using the $\text{Table}_{\text{promoter}}$, a numerical matrix of size $N_p \times 65$ is obtained. This is called $\mathbf{B}_{\text{promoter}}$.

4.3.2. Random Matrix Generation for Promoter Learning Similar to the ICA learning on the -35 box and -10 box, a random matrix, say $\mathbf{C}_{\text{promoter}}$, is generated. This matrix will be juxtaposed to the data matrix $\mathbf{B}_{\text{promoter}}$ later.

(a) Prepare N_p random segments of length 50 nt whose elements contain $\{A, T\}$ and $\{G, C\}$ to be 60% and 40%.

(b) The first A or G from the end is regarded as the starting point in this random sequence [4].

(c) A putative -35 box in each segment is found as follows: (1) Prepare 7 sliding segments of length 6 in the region $[15\cdots21]$. Make a 6×7 matrix. Change each element to numerals using Table_{-35}. This matrix is called $\mathbf{Z}_{-35}$. (2) Compute $\mathbf{W}_{-35}\mathbf{Z}_{-35}$. This de-mixed matrix is called $\mathbf{Y}_{-35,\text{C}}$, whose column vectors are called $\mathbf{y}_{-35,\text{C}}(j)$, $j = 1, \cdots, 7$. (3) Using the column vectors $\mathbf{y}_{-35}(k)$, of $\mathbf{Y}^T_{-35}$, compute the summations of the inner products by $q(j) = \sum_{k=1}^{58} \mathbf{y}^T_{-35,\text{C}}(j)\,\mathbf{y}_{-35}(k)$. (4) Set the putative end position of the -35 box to be $J_{-35} = \arg\max_{1 \le j \le 7} q(j)$.

(d) Find a putative -10 box in the segment in the same way as the -35 box.

(e) Perform the length adjustment to arrange the length to be 65 nt. This generates a random matrix of the size 106×65.

(f) Change the entries of this random matrix to numerical numbers by using $\text{Table}_{\text{promoter}}$. The resulting matrix is named $\mathbf{C}_{\text{promoter}}$.

4.3.3. ICA on the Total Promoter Structure Since the matrices $\mathbf{B}_{\text{promoter}}$ and $\mathbf{C}_{\text{promoter}}$ are prepared, the ICA learning for the total promoter region can be carried out.

(a) Juxtapose the matrices $\mathbf{B}^T_{\text{promoter}}$ and $\mathbf{C}^T_{\text{promoter}}$. The resulting 65×212 matrix is called $\mathbf{X}_{\text{prompter}}$.

(b) Preprocessing for the zero-mean and the whitening is executed on $\mathbf{X}_{\text{prompter}}$. The resulting matrix is renamed $\mathbf{X}_{\text{prompter}}$.

(c) Using the data matrix $\mathbf{X}_{\text{prompter}}$, the de-mixing matrix $\mathbf{W}_{\text{prompter}}$ and the de-mixed matrix $\mathbf{Y}_{\text{prompter}}$ are obtained by the ICA algorithm. This completes the whole training phase.

5. Testing on Given Segments

5.1. Preparation of Test Data

Each test segment is processed in the following way: (a) A test segment with a given starting point is given. (b) The sequence is adjusted to be the length of 65 nt by using $\mathbf{W}_{-35}$, $\mathbf{Y}_{-35}$, $\mathbf{W}_{-10}$ and $\mathbf{Y}_{-10}$. On the estimation of -35 box and -10 box, Table_{-35} and Table_{-10} are used for

the numerical conversion. The mean values and the whitening matrices obtained in the learning phase are also applied. (c) The resulting segment is changed to a 65-row numerical vector using the $\text{Table}_{\text{promoter}}$. Then, the mean value adjustment and the whitening are carried out by using the results obtained in the training phase. This is called $\mathbf{x}_{\text{test}}$.

5.2. Promoter Recognition

Given a vector $\mathbf{x}_{\text{test}}$ to be tested, the following judgment is carried out:
(a) Compute $\mathbf{y}_{\text{test}} = \mathbf{W}_{\text{promoter}}\mathbf{x}_{\text{test}}$. (b) If the first element "$y_{\text{test}}(1)$" is positive, the tested sequence is judged to contain a promoter. Otherwise, it is regarded as a non-promoter. For the positive sequence, gaps are removed to identify the estimated boxes. This completes the testing phase.

6. Experiments on Training and Testing

6.1. Data Preparation

A set of training data of length-adjusted segments were obtained from [4]. From this set, the matrices $\mathbf{B}_{-35}$, $\mathbf{B}_{-10}$, and $\mathbf{B}_{\text{promoter}}$ were generated. Then, they were changed to numerical matrices by the aforementioned methods which reflect position-dependent symbol frequencies.

Next, random matrices $\mathbf{C}_{-35}$ and $\mathbf{C}_{-10}$ were generated and changed to numerical matrices. Then, by the juxtaposition, $\mathbf{X}_{-35}$ and $\mathbf{X}_{-10}$ were generated. Then, by the ICA algorithm, $\mathbf{W}_{-35}$, $\mathbf{Y}_{-35}$, $\mathbf{W}_{-10}$, and $\mathbf{Y}_{-10}$ were obtained.

Next, the random matrix $\mathbf{C}_{\text{promoter}}$ was generated. Then, it was changed to a numerical matrix by the aforementioned method. By the juxtaposition of $\mathbf{C}^T_{\text{promoter}}$ to $\mathbf{B}^T_{\text{promoter}}$ the training matrix $\mathbf{X}_{\text{promoter}}$ was generated.

6.2. Execution of the ICA Algorithm

By the execution of the ICA on $\mathbf{X}_{\text{promoter}}$, the de-mixing matrix $\mathbf{W}_{\text{promoter}}$ was obtained. As was explained in Section 3.2, each column vector of $\mathbf{W}^{-1}_{\text{promoter}}$ works as an ICA basis. The first one, say $\mathbf{u}_1$, represents the major property of the promoter. Figure 3 illustrates the resulting ICA basis. We can observe that there are humps around the positions 27 and 52. They correspond to the -35 box and the -10 box, respectively.

6.3. Promoters and Non-Promoters

For the testing and performance evaluation, we prepared 126 promoter segments and 1,000 non-promoter segments. The set of 126 promoter segments were drawn from [4]. But, all gaps were removed in advance since our length-adjustment uses $\mathbf{W}_{-35}$, $\mathbf{Y}_{-35}$, $\mathbf{W}_{-10}$ and $\mathbf{Y}_{-10}$. Therefore, the length of each segment varies at first.

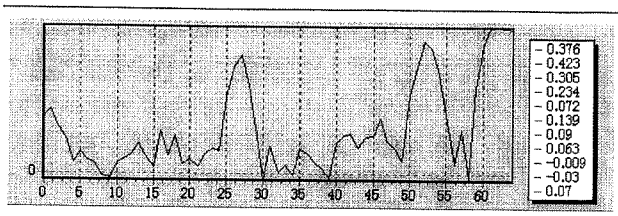

Figure 3. Obtained first ICA basis ($\mathbf{u}_1$).

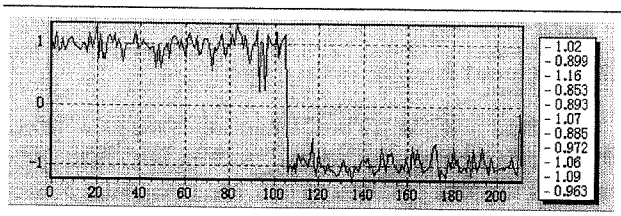

Figure 4. Decision by a threshold.

Generated non-promoter segments contain $\{A, T\}$ and $\{G, C\}$ by the ratio of 60% and 40%. The length of each segment is 50 nt. The first A or G from the end was regarded as the starting point.

On such 1,126 segments, the following test procedure was carried out.

(a) Using $\mathbf{W}_{-35}$, $\mathbf{Y}_{-35}$, $\mathbf{W}_{-10}$ and $\mathbf{Y}_{-10}$, the length-adjustment was made.

(b) Each entry was changed to a numerical value by using $\text{Table}_{\text{promoter}}$. Pre-processing using the learned data was executed.

(c) The resulting matrix $\mathbf{X}_{\text{test}}$ is of 65×1126 in the size.

(d) The de-mixed matrix was computed by $\mathbf{Y}_{\text{test}} = \mathbf{W}_{\text{promoter}}\mathbf{X}_{\text{test}}$.

(e) The first row of $\mathbf{Y}_{\text{test}}$, say "$\mathbf{y}_{\text{test}}(1)$," was taken out. Each element corresponds to each tested segment. The 1,126 elements in "$\mathbf{y}_{\text{test}}(1)$" were judged if they are positive or not. If the k-th element is positive, then a promoter exists in the k-th segment, $k = 1, \cdots, 1126$. Otherwise, the segment was judged to be a non-promoter. Figure 4 illustrates the resulting "$\mathbf{y}_{\text{test}}(1)$." As can be observed, there is a clear separation by the threshold at zero.

6.4. Performance Evaluation

On the recognition of promoters, the performance measures {precision, specificity, sensitivity} were computed. In order to compare the performances with existing studies [4], [5], the performance measures are defined in the usual way.

(a) The precision is computed by $\mathcal{P} = C/N_{\text{total}} \times 100\%$. Here, C is the number of correct judgments, and N_{total} is the total number of tested segments.

(b) The specificity is defined by $\mathcal{S}_p = (1 - N_{\rm fp}/N_{\rm np}) \times 100\%$. Here, $N_{\rm fp}$ is the number of false positives. $N_{\rm np}$ is the number of tested non-promoter segments.

(c) The sensitivity is defined by $\mathcal{S}_n = N_{\rm tp}/N_{\rm p} \times 100\%$. Here, $N_{\rm tp}$ is the number of true positives. $N_{\rm p} = N_{\rm total} - N_{\rm np}$ is the number of tested promoter segments.

Performances by various methods are summarized in Table 1. The first line is the performance of our method. The second line is the performance of [5] which use the unit vector expression of $\{A, T, G, C\}$, the EM algorithm, and artificial neural networks. The third line is the performance of [4] which uses the unit vector expression of $\{A, T, G, C\}$ and artificial neural networks. Thus, the presented method has the best precision of 93.7%. It is important to note that the consensus for the -35 box was TTGACA, and that of the -10 box was TATAAT.

	$\mathcal{P}$	$\mathcal{S}_p$	$\mathcal{S}_n$
This paper	**0.937**	0.934	0.968
Method [5]	0.919	0.918	0.992
Method [4]	0.904	0.902	0.980

Table 1. Performances of various methods

The score of our ICA method in Table I is merely an automatic result via the fixed threshold of $\vartheta = 0.0$. Upon observing Figure 4, however, readers can easily find that there are negative segments having scores only slightly below 0.0 (for instance, the one around the number 210 in Figure 4). They are highly likely to be false negatives. By watching the score, our method accepts additional interactive cross-examinations to reduce the false negatives.

7. Discussions and Concluding Remarks

In this paper, a new statistical method for E. coli promoter recognition was presented. The novelties in the presented method are summarized as follows: (1) The method is based upon the independent component analysis (ICA) which is unsupervised. But, the presented method beat existing supervised learning methods in the precision. (2) The threshold can be adjusted so that false negatives are reduced. (3) The numerical expression of DNA segments reflects position-dependent symbol frequencies.

The presented method can be extended and combined with other methods for further sophistication: (a) In this paper, promoters were recognized by using identified starting points. It is known that the transcription initiation sites may be diverse and can be identified exactly only via wet biological experiments, e.g., [9]. But, posterior probability approaches looking back from the promoter patterns are possible. The ICA promoter recognition method in this paper exists in the realm of this category. Our preliminary study supports this matter. (b) Incorporation of partially supervised mechanism [10] will improve the ability of the ICA. (c) The EM algorithm [11], [12] which contains the Hidden Markov Model as its special class can be combined.

Acknowledgment

The authors are grateful to the referees of the early version for their valuable comments. This study was supported in part by the Grant-in-Aid for Scientific Research #15300077, and by the Productive ICT Academia of the 21st Century COE Program.

References

[1] G.Z. Hertz and G.D. Stormo. Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. *Bioinformatics*, vol. 15, pp. 563-577, 1999.

[2] D.W. Mount. *Bioinformatics: Sequence and Genome Analysis*, Cold Spring Harbor Laboratory Press, 2001.

[3] D.H. Hawley and W.R. McClure. Compilation and analysis of Escherichia coli promoter DNA sequences. *Nucleic Acids Research*, vol. 11, pp. 2237-2255, 1983.

[4] I. Mahadevan and I. Ghosh. Analysis of E. coli promoter structures using neural networks. *Nucleic Acids Research*, vol. 22, pp. 2158-2165, 1994.

[5] Q. Ma, T.L. Wang, D. Shasha, and C.H. Wu. DNA sequence classification via an expectation maximization algorithm and neural networks: A case study. *IEEE Trans. Systems, Man and Cybernetics, Part-C: Applications and Reviews*, vol. 31, pp. 468-475, 2001.

[6] Y. Matsuyama, N. Katsumata, Y. Suzuki and S. Imahara. The α-ICA algorithm. *Proc. Int. Workshop on Independent Component Analysis*, pp. 297-302, 2000.

[7] Y. Matsuyama, N. Katsumata and R. Kawamura. Independent component analysis minimizing convex divergence. *Lecture Notes in Computer Science*, No. 2714, pp. 27-34, Springer-Verlag, 2003.

[8] I. Csiszár. Information-type measures of difference of probability distributions and indirect observations. *Studia Sci. Math Hingarica*, vol. 2, pp. 299-318, 1967.

[9] Y. Suzuki, et al. (15 authors). Diverse transcriptional initiation revealed by fine, large-scale mapping of mRNA start sites. *EMBO reports*, vol. 2, pp. 388-393, 2001.

[10] Y. Matsuyama, H. Kataoka, N. Katsumata and K. Shimoda. ICA photographic encoding gear: Image bases towards IPEG. *Proc. Int. Joint Conf. Neural Networks*, IEEE-INNS, 2004.

[11] A.P. Dempster, N.M. Laird and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm (with discussion). *J. Roy. Statist. Soc.*, B, vol. 39, pp. 1-38, 1977.

[12] Y. Matsuyama. The α-EM algorithm: Surrogate likelihood maximization using α-logarithmic information measures. *IEEE Trans. on Information Theory*, vol. 49, pp. 692-706, 2003.