Quad-Tree based Image Encoding Methods for Data-Adaptive Visual Feature Learning

Cuicui Zhang[1,a)] Xuefeng Liang[1,b)] Takashi Matsuyama[1,c)]

Abstract: Visual feature learning is fundamental to many computer vision tasks. State-of-art methods adopt an image block based multilayer framework to learn hierarchical feature representations. However, the image block is not adaptive for low-level feature extraction and the image pyramid based hierarchical models are neither adaptive nor flexible enough to learn high-level features. To solve these problems, this thesis exploits the image spatial and hierarchical structure using Quad-Trees and employs them for local feature analysis and for hierarchical feature learning. To evaluate the reliability of our methods, we also conduct feature learning in other challenging situations: feature learning with small training data and feature learning in dynamic environments (moving camera videos). Face recognition and motion segmentation are utilized as research backgrounds for algorithm evaluation. Experimental results demonstrate the effectiveness of our methods.

1. Introduction

Visual feature learning refers to the process of learning good feature representations, which can be used for many artificial intelligence (AI) level applications, such as computer vision, statistical pattern recognition, natural language processing, medical imaging, etc.. It acts as the bridge between raw images and learning algorithms, which transforms the raw image data into a good representation and then transfers the feature representation into machine learning algorithms. This thesis seeks to develop data-adaptive feature learning algorithms, which can effectively and efficiently learn good, semantically meaningful features for a variety of image data.

Visual feature learning is a challenging problem for long because of the high dimensionality and high variability of images, which can take a variety of forms such as static images, video sequences, views of multiple cameras, etc.. Existing visual feature learning algorithms can be briefly divided into: holistic and local based methods. The holistic methods extract a single feature vector from the whole image region and the local methods extract a series of feature vectors (bag-of-features) from a number of local subregions. While holistic methods can preserve the global shape and overall structure of the image data, local methods have better performances in coping with local deformations. Recently, it is argued that global feature learning based on local feature analysis is preferable than holistic or local based methods. State-of-art algorithms adopt a local patch based multi-layer framework to learn hierarchical feature presentations, which are named as *"hierarchical feature learning"* [1] methods. High-level features are formulated by the composition of low-level features according to a deep architecture. Although existing methods have achieved many impressive results, they still have some unsolved issues, which motivates the work of this dissertation.

[1] IST, Graduate School of Informatics, Kyoto University, Kyoto 606–8501, Japan
[a)] zhang@vision.kuee.kyoto-u.ac.jp
[b)] xliang@i.kyoto-u.ac.jp
[c)] tm@i.kyoto-u.ac.jp

1.1 Challenging issues

There have been many important issues in developing visual feature learning algorithms. The most important issues include the learning architecture (features learned from the planar image surface or from hierarchical architectures), data availability (small training data or large data), image characteristics (static image or dynamic video sequence). Considering the two stages of a typical hierarchical learning pipeline: low-level feature extraction and high-level feature integration, one can conceive the central issues in training hierarchical learning architectures are:

- How to extract lower-level features in order to provide adequate input to higher-level features;
- How to adjust higher-level features to make good use of lower-level features.

Based on these two issues, the main challenges faced by most feature learning algorithms can be summarized as below:

Challenge 1: Robust local feature analysis: Existing algorithms usually employ image block based low-level feature extraction, where the image region is partitioned into blocks (patches) of the same size and low-level features are extract from each block with the same weight. However, the block based method is straightforward and not adaptive enough to fit the image statistics, espeically for natural images taken under complex situations.

Challenge 2: Sophisticated deep structure for hierarchical feature learning: Several deep architectures have been proposed in the literature. Early works utilize fully-connected neural networks, where the parameters of low-level features are learned

based on backpropgation [2]. Since the backpropgation is very likely to get stuck in local minima, these method do not achieve success finally. Recent works utilize local-connectivity based networks, which are defined by image spatial pyramid. However, the image pyramid is a complete tree structure, which is not flexible and not adaptive to investigate the image hierarchical structure.

Challenge 3: Small training data availability: Existing visual feature learning algorithms usually require a large number of training samples. However, this can not meet in many practical applications due to the difficulty in data collection. The small training data leads to the small sample size (SSS) problem arising from the small number of training samples compared to the high dimensionality of the feature space. The SSS problem challenges existing methods severely.

Challenge 4: Difficulty in video based feature learning: While existing visual feature learning algorithms can be applied to static images relatively easily, it is not so video sequences especially when the video is captured by a moving camera. In the moving camera videos, local object motions and camera motion and mixed and dependent with each other making the segmentation of local object motions difficult. The video based feature learning is a more challenging problem.

1.2 The core-aspect of this thesis

A major factor that prevents the better performance of state-of-the-art algorithms might have been the lack of image structure. This thesis aims at solving the above mentioned challenges by explicitly predicting the image structure using Quad-Trees and employ the learned image structures in feature learning to make the learning procedure data-adaptive and more powerful. Specifically, instead of arbitrarily defining block based local region partition and image spatial pyramid, we use Quad-Tree to explicitly learn data-adaptive local subregions for low-level feature extraction and data-adaptive tree structure for high-level feature integration. The benefits are three-fold: (1) the data-adaptive block based image region partition makes the low-level feature extraction be region specific and locally adapt to the image statistics; (2) the data-adaptive tree structure makes the deep architecture more flexible and data-adaptive for hierarchical feature learning; (3) the Quad-Tree structure can define the weights of local-subregions making the composition of low-level features to high-level features more effectively and accurately.

1.3 The main contributions of this thesis

The main contributions of this thesis can be summarized as follows: Towards the aforementioned four feature learning situations, we develop four Quad-Tree based image encoding methods for data-adaptive visual feature learning, including:

Algorithm 1: Feature learning from data-adaptive blocks decomposed by Quad-Tree (Section 2): For the local feature analysis, we develop a feature learning algorithm using data-adaptive blocks decomposed by Quad-Tree. Instead of arbitrarily dividing the image region into fixed-size image blocks (patches), Quad-Tree partitions the image region into local subreigons of variant sizes according to the feature distribution in an image dataset. Features extracted from data-adaptive blocks can be directly transferred to machine learning applications or act as low-level features for more sophisticated feature learning algorithms (e.g hierarchical learning). As it is difficult to find an appropriate threshold for Quad-Tree partition, we define a set of thresholds and get a set of Quad-Tree partitions accordingly. The Quad-Tree partitions on Uchimura database are illustrated in Fig.1.

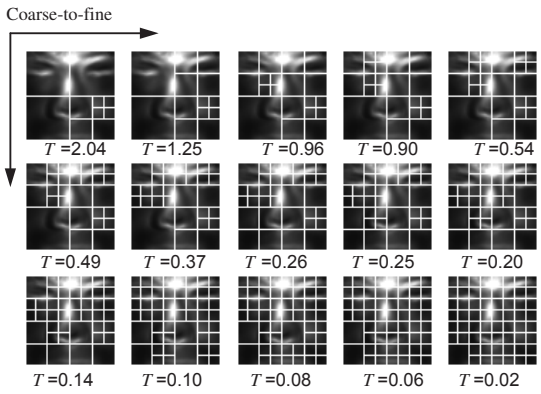


Fig. 1: An example of Quad-Tree partitions on a face image in Uchimura database. (Copyrighted by ICPR2012 [3])

Algorithm 2: Hierarchical feature learning using Quad-Tree structure of images (Section 3): After local feature analysis, the next step is on how to combine local features together. We develop a hierarchical feature learning algorithm using Quad-Tree structure of images. In contrast to image pyramid based hierarchical models, Quad-Tree defines a data-adaptive image tree structure for hierarchical feature learning. And the parameters of local subregions can be defined according to the tree structure, which makes hierarchical feature learning more efficient. The overview of Quad-Tree based hierarchical feature learning is shown in Fig.2.

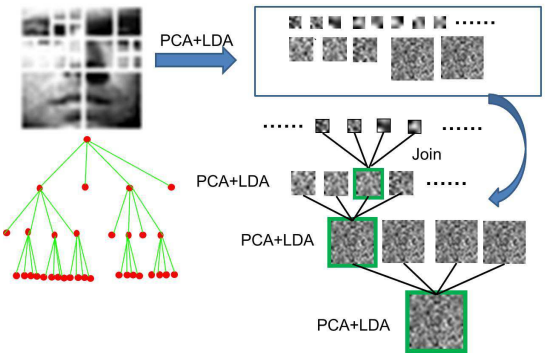


Fig. 2: Hierarchical feature learning using Quad-Tree structure of images. (Copyrighted by CRC Press / Balkema [4])

Algorithm 3: Feature learning from enlarged training data encoded by Quad-Tree for Small Sample Size problem (Section 4): To solve the SSS problem, we develop a training data extension method by generating new samples encoded by Quad-Trees and then perform base classifier ensemble to improve the final recognition accuracy. In contrast to existing ensemble methods which are performed using the original small training data, our method can generate more diverse base classifiers. Moreover, since existing ensemble methods suffer from the Diversity/Accuracy dilemma by integrating all base classifier together,

we develop a base classifier selection algorithm using a tailored 0–1 Knapsack solution, which alleviates the dilemma effectively. While the 0–1 Knapsack solution is developed for feature selection in Algorithm 1, this tailored 0–1 Knapsack solution is employed for classifier selection. The main idea of data extension is illustrated by Fig.3.

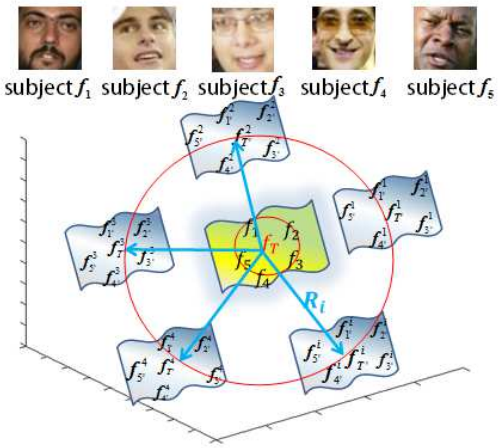


Fig. 3: Feature space expansion through generating new samples [5].

Algorithm 4: Feature learning in dynamic environments using Helmholtz-Hodge decomposition and Quad-Tree (Section 5): To make feature learning in video sequences more naturally, we develop a motion segmentation method for moving camera videos. This method can segment and recover object motions from the scene effectively. As Quad-Tree is sensitive to the dynamic background, it is rather difficult to apply Quad-Tree based image region partition directly in dynamic environments. To solve this problem, we introduce an amended Helmholtz-Hodge Decomposition first for camera motion compensation. After camera motion compensation, a data-driven Quad-Tree partition can be performed in the rested motion field for object motion segmentation. The framework of HHD and Qua-Tree based motion segmentation is shown in Fig.4.

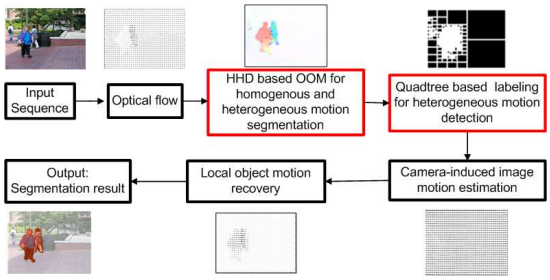


Fig. 4: Illustration of the feature learning algorithm in dynamic environments using HHD and Quad-Tree, which HHD is performed for camera motion compensation and Quad-Tree is performed for object motion segmentation.

In the next, we describe each algorithm with experimental results in Sections 2–5, respectively. The relationship among these algorithms is specified in Section 6 followed by the conclusion in Section 7.

2. Feature Learning from Data-Adaptive Blocks Decomposed by Quad-Tree

Local feature analysis is the first step of many feature learning algorithms. This section describes our feature learning algorithm from data-adaptive blocks decomposed by Quad-Tree.

2.1 Problem definition and related works

Principal Component Analysis (PCA) [6] and Linear Discriminant Analysis (LDA) [7] are two widely used appearance based face recognition algorithms. However, being based on a global description, local variations, such as motion deformation, illumination changing, data missing again form a major problem. Multi-subregion fusion methods ([8], [9]) were proposed as a solution for this problem. They divide the image into a set of disjoint subregions, perform recognition on each subregion and fuse the results.

Existing well-known local subregion methods can be mainly divided into two types: (1) block (patch) based algorithms, which divides an image region into regular blocks or utilizes a sliding window to locate subregions for feature extraction. These methods are very straightforward and not adaptive to image statistics. (2) hand-designed local subregion based methods. These methods provide several hand-designed spatial regions for feature extraction. For example, in [10] a set of 30 regions were designed for face recognition, see Fig. 5. They showed better performance than block based methods. However, these hand-designed regions are highly depending on the prior-knowledge of human beings. The process is time consuming, incomplete and usually relying on data registration.

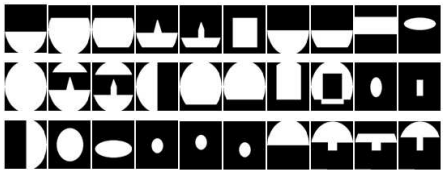


Fig. 5: The hand-designed 30 regions [10].

Apart from the image region partition, another problem with existing local subregion based methods is on how to select subregions to build up the best combination of them. Some local subregions may not appropriate for classification. Integrating classification results of inappropriate subregions may reduce the final classification accuracy. To address these two problems, this sections develops a data-adaptive image region partition method using Quad-Trees and introduce a local subregion selection method based on 0–1 Knapsack solution.

2.2 Algorithm architecture

The algorithm can be described as follows: first, Quad-Tree decomposes an image region into multiple local sub regions recursively according to an image homogeneity criterion function. In order to make the Quad-Tree partition adapts to variant databases, we perform Quad-Tree partition on a template image instead of original images. Then, a local subregion selection algorithm is proposed for the fusion of the results.

2.2.1 Modelling the feature distribution using a template

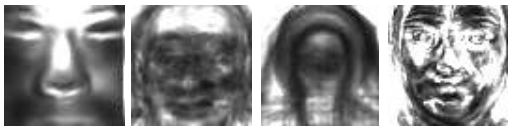
The generation of template image is motivated by the idea of LDA. which encodes the discriminative feature distribution by maximizing the between-class scatter matrix S_b and minimizing the within-class scatter matrix S_w (See Eq. 1). The template face, which is defined by Eq. 2, represents the distribution of discriminative features across the face region for all the images in a database. That is, the total variance of entire database equal to the variance of the template (see Eq. 3). Example template faces for four face databases are shown in Fig. 6.

$$\begin{cases} S_b = \sum_{i=1}^{c} N_i(\mu_i - \mu)(\mu_i - \mu)^T, \\ S_w = \sum_{i=1}^{c} \sum_{x_k \in X_i} (x_k - \mu_i)(x_k - \mu_i)^T, \end{cases} \tag{1}$$

$$template = diag(\frac{S_b}{S_w}), \tag{2}$$

$$totalVar = variance(diag(\frac{S_b}{S_w})), \tag{3}$$

where μ denotes the mean face of all face classes, μ_i is the mean face of class X_i, N_i denotes the sample number of class X_i, and x_k represents the k-th sample of class X_i.



(a) (b) (c) (d)

Fig. 6: Templates for four databases: (a) Uchimura 3D, (b) ATT (renamed as ORL), (b) IFD, (c) JFFE database. (Copyrighted by ICPR2012 [3])

2.2.2 Quad-Tree based image region partition

Quad-Tree decomposition is performed based on a criterion function defined in Eq.4. If the variance of a region R (block) or one of its four sub-blocks (subR) is higher than a threshold variance ($T * totalVar$) , then R is split into smaller blocks. The variance here actually relates to the density of discriminant features in a region.

$$\begin{cases} doSplit(R) = totalVar(R) > T * totalVar, \\ doSplit(R) = doSplit(R) | doSplit(subR). \end{cases} \tag{4}$$

The difficulty in Quad-Tree decomposition is how to define an appropriate threshold T. Local variances usually vary on different databases, so one universal threshold is not a good idea in our method. Even in one database, it is rather difficult to find the most appropriate threshold to get the best partition. In this dissertation, we define a set of thresholds instead of a single one. The thresholds are defined between a maximum value and a minimum value in a descending order. The maximum value is the start point where the template image begins to partition. The minimum value is the end point, where the template is partitioned into smallest blocks completely. The maximum value and the minimum value are also depending on databases. The template is split into less and bigger blocks for a large threshold, but into more and smaller blocks for a small threshold. Given a set of thresholds, we can get a set of Quad-Tree partitions with different resolutions (the partitions are from coarse to dense). Figure 1 illustrates the Quad-tree partitions on the template image of Uchimura 3D database [11].

2.2.3 Data-adaptive block based feature extraction

For each Quad-Tree partition, we re-organize the face images in the original image dataset according to the partition result. As explained above, larger partitions (blocks) mean that the density of discriminative features in them is low. Thus, there is no need to keep its original size. We will do downsampling on large blocks (the size is larger than the smallest size) by resizing them to a smaller size ($(d/2) \times (d/2)$), where d is the width of large blocks (in pixel). For smallest blocks mean they the density of their discriminant features are high. Thus, we will not do down-sampling on them. Finally, block resizing result in a set of new images whose sizes are smaller than original face images. For each Quad-Tree partition, we will get a new set of images, from each appearance based feature extraction method is applied to generate feature subspace. In this dissertation, we use PCA+LDA for the extraction of features, where PCA is utilized for dimension reduction and LDA is employed to generate discriminant features in the PCA feature subspace.

2.3 Selection of local subregions

The subregions obtained from Quad-Tree partitions represent portions of the face region in different locations. Their performances (recognition accuracy) are different from each other since they contain variant discriminative features. A single subregion is usually unlikely to achieve the best performance since the discriminative features contained in each single suregion are usually limited. To solve this problem, integration of the classification results of multiple subregions may work better than a single subregion. However, it is difficult to find the best combination of all subregions, some subregions may not be appropriate for classification. For example, in our method, some thresholds may not be good for Quad-Tree partition. Thus, the classification results of such local subregions are not accurate. In our work, motivated from the idea of 0-1 Knapsack problem, we covert the problem of subregion selection and fusion into an optimization problem, which is explained in the following section.

2.3.1 0–1 Knapsack algorithm

The conventional 0–1 knapsack problem is depicted like this: given a set of items and a knapsack, each item has a mass and a value while the knapsack has its own capacity. We would like to select some items and put them into the knapsack so that the total value of selected items can be maximized while the total weight should not beyond the capacity of K. This can be interpreted as:

$$\begin{aligned} &max(value(K)) \\ &subject\ to \quad mass(K) \le t_c \end{aligned} \tag{5}$$

where $value(K)$ and $mass(K)$ represent the total value and total mass of selected items in K, and t_c denotes the capacity of K. This problem is very similar to our problem. Given a set of subregions $S = S_1, S_2, \cdots, S_n$, each subregion S_i has two parameters: the size W_i and the discriminant power V_i. We aim at choosing a sub-

set O of S such that the total size of selected subregions does not exceed the capacity of O and the total value is maximized. The size (in pixel) and discriminative power are similar as the weight and the value in the conventional 0–1 Knapsack problem. As mentioned before, the discriminative power of a subregion can be represented by the density of feature distribution. Thus, we use the trace of $S_b^i./S_w^i$ in a subregion to represent its discriminant power (see Eq.6).

$$V_i = trace(\frac{S_b^i}{S_w^i}) \tag{6}$$

The capacity of the subset O is defined by the total size of the 30 regions illustrated in Fig. 5 since we want to investigate whether our method can work better than or equal to the performance of the state-of-the-art [10] under the same computational cost. Like the conventional 0–1 Knapsack problem, we can also use dynamic programming to solve this optimization problem.

2.4 Experimental rvaluation

2.4.1 Datasets

Our method was evaluated on seven databases: a 3D database (Uchimura 3D database [3]) and six widely used 2D databases: IFD [12], JFFE database [13], ORL (pre-name: ATT database [14]), Extended Yale (Yale2) [15], AR [16], and FERET [17]. The face data in these databases are under varied conditions including a variety of head poses (Uchimura, ORL, IFD, AR), illumination changing(Uchimura, ORL, IFD, Yale2, AR, FERET), partial data missing (Uchimura), facial expressions (ORL, IFD, JFFE, Yale2, AR, FERET), and facial details (e.g. with glasses or not: ORL, AR, FERET).

Fig. 7: Sample images of seven databases: (a)Uchimura 3D database, (b) IFD, (c) JFFE, (d) ORL, (e) Extended Yale (Yale2), (f) AR, (g) FERET.

- **Uchimura 3D database**: A 3D point cloud data of 640×480 was created along with color images. This dataset consists of 38 subjects. Each subject has 10 samples, of various head poses and illumination conditions. From the 3D data, we first generate a range image as the visual feature image according to the method in [11]. For recognition, 5 samples of each subject are randomly selected as training data while the rest as test data.
- **Indian Face Database (IFD)**: this database contains 40 subjects. Each subject has 11 samples varied in head poses and facial expressions. Five samples per subject are randomly selected as training data and the rest as test data.
- **The Japanese Female Facial Expression Database (JFFE)**: Ten subjects are contained in the databases. Each subject has 7 samples according to 7 basic facial expressions. For recognition, 4 samples are randomly selected as training data while the rest as test data.
- **ORL database (previous ATT database)**: There are 40 subjects in this databases. Each subjects has 10 samples with variances in facial expressions, open or closed eyes, with glasses or no glasses, scale changes (up to about 10 percent), head poses. Five samples per subject are randomly selected as training data while the left ones as test data.
- **Extended Yale database (Yale2 database)**: more than 20,000 single light source images of 38 subjects with 576 viewing conditions (9 poses in 64 illumination conditions) are contained in the database. To evaluate the robustness of our method on the illumination changes, 5 samples of the 64 illumination conditions are randomly selected as training data, the left 59 images as test data.
- **AR database**: 4000 color face images of 126 people (70 men and 56 women) are contained in this database. They are frontal view faces with different facial expressions, illumination conditions, and occlusions (sun glasses and scarf). Each subject has 20 samples, which are divided into two sections taken in different time (separated by two weeks). As in [18], the fist session images of 50 male subjects and 50 female subjects was selected as a subset for performance evaluation. For each subject, we randomly choose 5 samples for training and the left 9 images for test.
- **FERET database**: consists of 13539 images corresponding to 1565 subjects. Images differ in facial expressions, head position, lighting conditions, ethnicity, gender and age. To evaluate the robustness of our method to facial expressions, we worked with subset of front faces labeled as *Fa, Fb*, where *Fa* with regular facial expressions, and *Fb* with alternative facial expressions. There are 1201 subjects in the dataset. *Fa* samples of each subject are selected as training data, while *Fb* as test data.

The performances of the first three databases (Uchimura 3D, ifd, and JAFEE) are reported in the published work [3]. However, these databases are very small and out-of-date. To make the evaluation more convinced, we add four more databases (ORL, Yale2, AR, FERET) in the experiments, which are considered as benchmark databases to evaluate face recognition algorithms. Face images in the ORL, Yale2, and AR databases are aligned to 32×32 using the method in [19]. FERET database is normalized using the *CSU Face Identification Evaluation System 5.1* [20]. The face images are cropped to the same size 32×32. Sample images of all databases are shown in Fig. 7.

2.5 Experimental results

The performance of our method is compared with: (1) the block based method; (2) the 30 region method [10]. In [10], since they donot know how to select the best combination of 30 regions, we use the following four combinations: the total 30 regions, 28 regions of highest accuracy, 20 regions of highest accuracy, and the maximum single region (the single region of the highest performance). The results are shown in Table 1.

Table 1: Recognition rate of our Quad-tree based method compared to the block and 30-region based method [10] on seven face databases.

Method \ Database		3D	IFD	JFFE	ORL	Yale2	AR	FERET
Block		**99.9**	88.1	95.0	92.7	76.3	83.2	59.6
30-region+ PCA-LDA	30 regions	98.9	87.3	95.0	93.5	91.6	88.6	75.0
	28 regions	98.9	86.0	95.0	94.0	91.2	88.6	73.3
	20 regions	98.9	87.7	95.0	93.5	91.9	88.6	**75.3**
	max single	99.3	87.7	**99.9**	94.5	89.9	86.6	67.3
Data-Adaptive Block		**99.9**	**91.4**	**99.9**	**96.0**	**91.9**	**89.6**	70.3

From the experimental results, we find that for the above mentioned first 6 databases our method outperforms the conventional PCA-LDA method and the state-of-the-art 30-region method obviously. Our method benefits from the data-driven image partition and it can be widely applied to diverse databases, especially for those with a large number of variations. Moreover, the 0-1 knapsack solution guarantees our method to select the best combination of multiple subregions automatically. We need to acknowledge that our method performs worse than 30-region on FERET database, which only contains frontal face images and has been every well registered using the *CSU Face Identification Evaluation System 5.1* [20]. We argue that the performance of 30-region method relies much on human beings and data registration.

3. Hierarchical Feature Learning using Quad-Tree Structure of Images

In Section 2, we have investigated local feature analysis using data-adaptive blocks decomposed by Quad-Tree. In this section, we will explore how to combine local features to form a global feature representation.

3.1 Problem definition and related works

Early works utilize simple liner combination rules such as weighted sum rule to combine local features. However, the linear combination is not sophisticated enough to investigate the global image structure. New trends of feature learning utilize a multi-layer framework to learn global feature from local features, where high-level features can be formulated by the composition of low-level features.

Recent developed hierarchical feature learning methods utilize local connectivity based networks such as Convolutional Neural Networks (CNNs) [21] and Deep Belief Networks (DBNs) [22], [23]. CNNs utilize local connected feedforward networks to investigate the spatial correlationship between neuros of adjacent layers. A typical pipeline of CNNs comprises a convolution process and subsampling process. These two processes are alternated until the size of the final feature representation is as small as what we desired [24]. DBNs are multi-layer graphical models working based on greedy layer-wise unsupervised learning of Restricted Boltzmann Machines (RBMs). RBMs are utilized for parameter initialization of the whole architecture. Although CNNs and DBNs have achieved many success, they still have some problems. The first problem relates to the hierarchical architecture, which are defined based on image pyramid. Image pyramid is a complete tree structure. We will have the same structure for all the images when the size of the image pyramid is fixed. Thus, these methods are not data-adaptive and not flexible enough to investigate the image hierarchical structure. The second problem relates to the parameter learning. Existing methods (e.g. DBNs) utilize a layer-wise unsupervised pre-training for parameter learning. However, the unsupervised pre-training is not well-suited for supervised learning applications.

To solve these problems, this section develops an alternative hierarchical feature learning algorithm using Quad-Tree structure of images. Quad-Tree explicitly learns the image hierarchical structure, which can guide in hierarchical feature learning. The benefits are two-fold: (1) The Quad-Tree structure is a data-adaptive tree structure and more flexible for hierarchical feature learning; (2) the parameters (weights) of local subregions can be defined according to the tree structure other than using unsupervised pre-learning.

3.2 Algorithm architecture

The proposed algorithm comprises a top-down Quad-Tree decomposition and a bottom-up deep feature integration [4]. An overview of the hierarchical feature learning algorithm is illustrated in Fig.2. Unlike evenly partitioning image, Quad-Tree decomposes an image into uneven subregions according to the density of discriminant information across the image region. Low-level features can be extracted from each subregion and higher-level features can be formed by low-level features according to the tree structure.

3.2.1 Top-down image structure prediction

Instead of dividing the face region into uniform blocks, Quad-Tree partitions the face region into data-adaptive blocks of variant sizes by means of local discriminative variance. Larger partition implies that the block has lower density of discriminant features, and vice versa. To make the Quad-Tree partition more robust to local noises, we consider to use the variance of all face images in the entire image dataset. As stated in section 2, we define a template image T and then perform Quad-Tree decomposition on T in the same way as stated in Section 2. Since it is rather difficult to find the best partition using a universal threshold. We also define a set of thresholds in a descending order as in section 2 and generate a series of Quad-Trees.

3.2.2 Bottom-up hierarchical feature learning

For each Quad-Tree partition, we first extract low-level features from each subregion using PCA+LDA and then higher-level features can be formulated according to split-and-joint path encoded in the tree structure.

Specifically, as Fig.2 shows that a face image is partitioned into many blocks with varied sizes. Blocks without green rect-

angle denote the leaf nodes, which are used as the input for local feature extraction based on PCA+LDA. Thee select the top k_i vectors as a feature basis, where k_i is less than the corresponding block size, i denotes the level index in the tree hierarchy. The smaller i is, then bigger k_i becomes. Each block is projected into a feature subspace encoded by PCA+LDA, and the four reduced-dimensionality neighbor blocks are then joined together back to their father node in their original split order. For the new joined-nodes, we apply PCA+LDA again to generate more abstract features. And these features will act as new input to create up the higher layer. This process is repeated until reaching the root node. Since we apply PCA+LDA on each layer, we name this method as deep PCA [24]. That is, our hierarchical feature learning is performed based on split-and-joined process according to the Quad-Tree structure and using a deep PCA as dimension reduction and feature extraction.

3.2.3 Weight assignment according to Quad-Tree structure

The difficulty in high-level feature formulation is how to define the dimension of each local subregion. In Algorithm 1, we assume that each local subregion has the same discriminant power for classification. Thus, we assign them the same weight and extract the same dimension of features. However, from experimental results, we find that the actual discriminant power of each local subregion is different from each other. Thus, in this algorithm, we explore to assign different weights to local suregions (including leaf nodes and non-leaf nodes). The weight corresponds to the dimension of the PCA subspace. High weight implies that we need to extract high dimensional features, and vice versa.

Specifically, we use the number of leaf nodes under each local subregion to define its weight. Specifically, the weight of each node $subR_k$ is defined by the number of leaf nodes in the subtree rooted at $subR_k$ against that number rooted by its father node $subR_j$. An example of the Quad-Tree structure with weight assignment is shown in Fig. 8. When each group of four neighbour nodes are joined to generate a higher level node (their father node) $subR_j$, the dimension of the feature subspace of $subR_k$ is defined by the $w_k * d_j$, where w_k is the weight of $subR_k$ and d_j is the dimension of the feature subspace of $subR_j$.

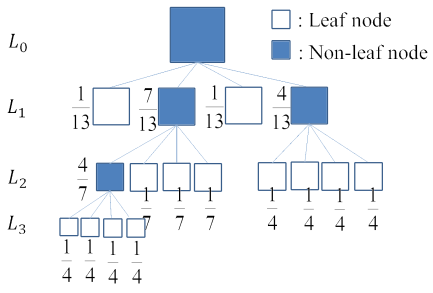


Fig. 8: An example of Quad-Tree partition with weight assignment.

3.3 Experimental evaluation

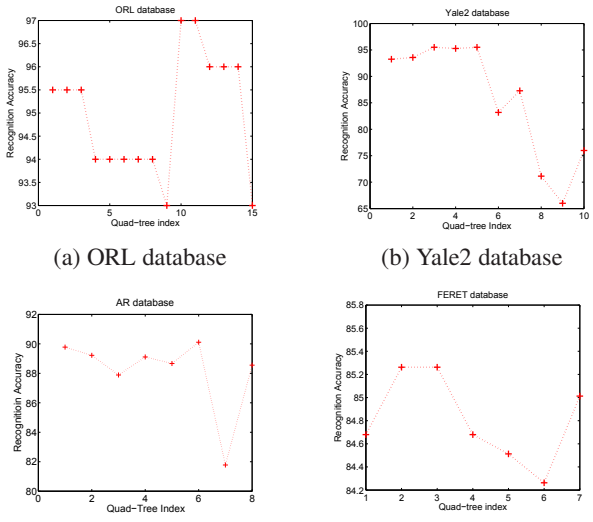
In order to compare with Algorithm 1, we use the same four databases in the experiments: ORL [25], Extended Yale (Yale2) [15], AR [16], and FERET database [17]. Please refer to Section 2.4.1 for the description of each database.

We compared this method with several existing algorithms using global feature, local features, and canonical deep feature, respectively. Beyond hierarchical learning, there are also some other global feature learning algorithms such as the multi-scale patch-based collaborative representation (MPCRC)[18]. Thus, the comparison methods are: (1) the conventional PCA+LDA method; (2) MPCRC [18]; (3) the 30-region method [10], mentioned in Section 2; (4) the Data-Adaptive Block based method developed in Section 2; (5) the Deep PCA [24]. Table 2 illustrates the comparison results.

Table 2: Recognition accuracy of our Quad-Tree based hierarchical learning method compared to other state-of-the-arts.

Method \ Database	ORL	Yale2	AR	FERET
PCA+LDA	92.70	76.30	83.22	59.62
MPCRC	91.50	92.11	88.60	73.64
30Regions	93.50	91.64	88.6	75.02
Data-Adaptive Block	96.00	91.86	89.56	70.28
deep_PCA	95.50	90.33	89.78	84.44
QT-Hierarchical	**97.00**	**92.94**	**90.44**	**85.43**

From Table 2, we find that our method outperforms others in most experiments. To explore how varied deep structures influence the performance of feature hierarchy, we introduced a set of thresholds for Quad-Tree partition. Fig. 9 plots the recognition accuracy of variant Quad-Tree partitions (the indices correspond to different thresholds in ascending order). Please note that the leftmost point indicates the accuracy of the original Deep PCA, which is performed based on a complete tree structure. From this figure, we can find that hierarchical feature learning with different Quad-Tree partitions performs quite different. And the best performance usually is achieved at a certain tree index instead of the first one depending on databases. Our method benefit from the data-driven Quad-Tree partition, can generate the most appropriate partition adapt to different databases automatically.



(a) ORL database (b) Yale2 database

(c) AR database (d) FERET database

Fig. 9: Influence of Quad-Tree partitions to the deep_PCA based face recognition on four databases. (Copyrighted by CRC Press / Balkema (Taylor & Francis Group) [4])

3.4 Comparison with data-adaptive block based algorithm

The relationship between Algorithm 1 and Algorithm 2 is that: Algorithm 1 is about feature learning from the planar image surface while Algorithm 2 is hierarchical feature learning using tree structure. In Algorithm 1, we treat each block equally and assign the same weight to them for combination. In Algorithm 2, we assign different weights to different blocks according to the tree structure.

We compare the result of these two algorithms in Table 2. We observe that Algorithm 2 outperforms Algorithm 1 for all databases, especially on FERET database, which is largest databases containing a large number of classes. These experimental results demonstrate the strength and effectiveness of utilizing hierarchical learning to generate global feature representation.

4. Feature Learning from Enlarged Training Data Encoded by Quad-Tree for Small Sample Size Problem

In Sections 2–3, we have investigated feature learning with large training data. In order to evaluate the reliability of our method, this section investigates feature learning with small training data, which suffers from the SSS problem. An extreme case of SSS is single sample per person (SSPP), where only one sample is available for each subject. Existing visual feature learning algorithms suffer from overfitting problem under SSS severely. Even worse, SSPP makes some feature learning methods fail, e.g. LDA since we can not generate the within-class scatter matrix under SSPP. Thus, it is very necessary to investigate feature learning with small training data.

4.1 Problem description and related works

There have been many attempts in the literature towards the SSS problem, such as the virtual sample generation based methods, multi-subregion combination methods [26]. However, these methods still have limited generalization ability when the training samples are insufficient [18]. Recently, a machine-learning technique known as ensemble learning has been recognized as an effective way in alleviating the SSS problem. The basic idea of ensemble learning is that a pool of weak classifiers can have a better classification ability than a strong classifiers under SSS. It generates multiple base classifiers, which offer complementary information for classification. Benefit from base classifier collaboration, the overall performance can be higher than a single classifier.

Existing ensemble learning algorithms for face recognition can be mainly classified into three groups: (1) global feature selection based on random subspace; (2) multi-subregion based local feature extraction; (3) global and local feature integration.

The first kind of methods introduce *random subspace* for ensemble, which is based on the literature finding that strong and stable base classifiers defined by subspace methods (e.g. PCA, LDA) are not suitable to ensemble rules [28]. The second kinds of methods utilize local feature extraction based on multi-subregions[18]. An early attempt [29] partitioned each face image into six elliptical sub-regions. Topcu et al. [30] developed an alternative way by partitioning face regions into small patches of the same size. Base classifiers can be trained from each patch separately and the final recognition results were obtained based on the decision fusion rule of base classifiers. Considering that both global and local features can provide complementary information, the third category of methods integrate both global and local features together for classification. Well known method include a hierarchical face recognition algorithm [31], MPCRC [18], the 30region method [10].

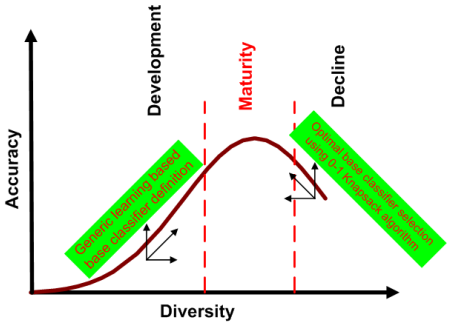


Fig. 10: The relationship between diversity and accuracy, which is summarized into three stages: development, maturity, and decline [5]. (This figure is plotted based on the experimental results shown in Fig. **??**)

Although existing ensemble methods can alleviate the SSS problem in some sense, they still have limited ability since they can not generate diverse base classifiers from insufficient training data. This section develops a training data augmentation method by generating new samples encoded by Quad-Tree and then performing ensemble from the expanded training data more effectively. The novel idea of this work is illustrated in Fig. 3. The original small training sample set locates at the center of the face space in the figure. To explore new possibilities of face space, we introduce a set of random matrices $R_i, i = 1, 2, ..., L$, the elements of which are generated based on uniform distribution. Random matrices are not added to the original face images directly. Instead, we first model the feature distribution of the a small sample set by a template image f_T and then R_i is added to f_T to transform it to a new place $f^i_{T'}$. After that, Quad-Tree decomposition [3] is performed on each $f^i_{T'}$ to generate an image encoding pattern. Original face images are re-organized according to each Quad-Tree partition to generate a new training sample set. For each random matrix, we can have a Quad-Tree partition, and generate a new training sample set. Given N random matrices, we can have N Quad-Tree partitions, and N new training sample set accordingly. Base classifier are learned from each new training set for ensemble.

Even having diverse base classifiers, we still need to consider another problem. Ensemble learning algorithms aim to achieve higher accuracy and higher diversity meanwhile. It is difficult to achieve these two objectives at the same time. Figure 10 illustrates the relationship between accuracy and diversity. We can see that they are not linearly related. The increase of diversity can improve the accuracy in a certainty degree, but not always so. This observation is known as the Diversity/Accuracy dilemma [32]. To solve this problem, we introduce an amended 0–1 Knapsack solution.

4.2 Algorithm architecture

The developed Quad-Tree based ensemble framework QT-E is illustrated in Fig. 11, which involves the following three steps: (1) Base classifier definition after data expansion; (2) Base classifier selection using an optimal solution; (3) Base classifier integration through majority voting.

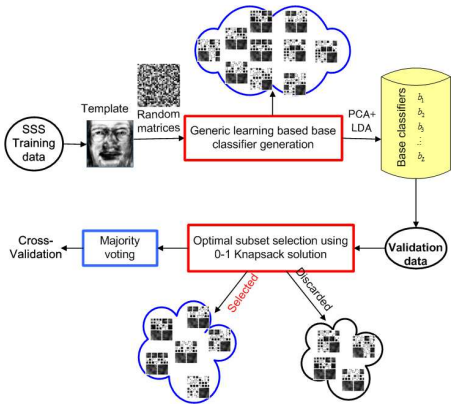


Fig. 11: Illustration of Quad-Tree based ensemble framework (QT-E) [5].

4.2.1 Quad-Tree based image data expansion

The procedure of base classifier definition is illustrated in Fig. 12, which consists of three operations: (1) template image generation and random matrix introduction; (2) new sample generation based on Quad-Tree decomposition, and (3) the definition of accuracy and diversity of base classifiers.

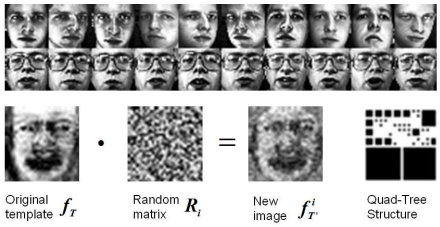


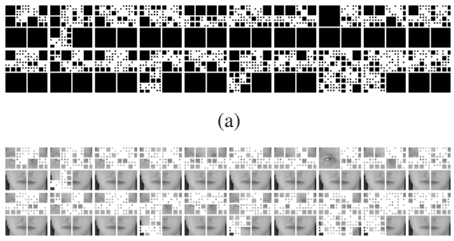
Fig. 12: Example of template face generation and random matrix introduction on ORL database [5].

The generation of the original template image is the same as that in section 2. Since the template image f_T is generated from the small training data, it is of weak ability to represent the whole face space. To solve this problem, we introduce a set of random matrices $R = \{R_1, R_2, ..., R_i, ..., R_L\}$ and add them to f_T to generate several new template images $f_{T'}^i = \{f_{T'}^1, f_{T'}^2, ..., f_{T'}^i, ...f_{T'}^L\}$. Here, each random matrix R_i, $i = 1, 2, ..., L$ if of the same size as f_T and its elements are randomly generated based on a uniform distribution in [0, 1]. The new image $f_{T'}^i$ is generated by the dot product of f_T and R_i as:

$$f_{T'}^i = f_T \cdot R_i. \tag{7}$$

After obtaining several template images, Quad-Tree decomposition is performed on each $f_{T'}^i$ in the same way as stated in Sections 2. The main difference is on the threshold definition. In this section, the random matrix is the main variations introduced to face images, which influence the feature distribution over the image region. We had better keep t_v as an invariant parameter in order for better investigation the influence of the random matrix to the face region partition. In this dissertation, we define $t_v = 0.5 * var(wholeR)$ without loss of generality, where $wholeR$ denotes the whole region of $f_{T'}^i$.

Each Quad-Tree partition refers to a face encoding pattern (depicted by blocks of variant sizes) as illustrated in Fig. 13 (a). Given L random matrices, we can have L encoding patterns. For each Quad-Tree partition, we re-organize the face images in the original image dataset according to the partition result. The image re-organization is performed in the same way as that in section 2.2.3. For L Quad-Tree partitions, we can have L new training sample set. Then, we train a base classifier b_i from each new training sample set using PCA+LDA.



(a)

(b)

Fig. 13: An example of 20 Quad-Trees (a) Quad-Trees (b) Quad-Tree partitions on a face image. (Copyrighted by IET [27])

4.2.2 Base classifier definition and calculation

Each base classifier has two important parameters: (1) the accuracy ($acc(E)$); and (2) the diversity ($div(E)$). The accuracy of each base classifier b_i is defined by the ratio of correctly classified samples against the total number of samples in Eq. (8).

$$acc(b_i) = \frac{num(correctSamples(b_i))}{num(totalSamples)}, \tag{8}$$

The measurement of diversity ($div(E)$) is investigated in many literature works. In this section, we use the *disagreement measurement* to calculate $div(E)$ since it was originally designed for ensemble problem [33] and its intuition was coincide with ours, which suggest that two diverse base classifiers should perform differently on the same training data.

4.2.3 Base classifier selection and ensemble

To solve the diversity/accuracy dilemma, we develop a base classifier selection algorithm using a tailored 0–1 Knapsack solution. The selection algorithm can be considered as on optimal combinatorial task. The conventional 0–1 Knapsack problem has been depicted in Section 2.3.1. This is very similar to our problem. We have a set of base classifiers. Each base classifier are with two parameters: (1) recognition accuracy, and (2) the diversity with other base classifiers in making decisions. Our goal is to find an optimal subset of base classifiers E which can maximize the final accuracy of E and the total diversity should be still higher than or equal to a threshold diversity t_d. Our problem can be interpreted in a mathematical way as:

$$\begin{aligned} &max(acc(E)), \\ &subject\ to \quad D(E) \geq t_d, \end{aligned} \tag{9}$$

where t_d is the diversity threshold and $D(E) = div(E)$ denotes the

diversity of E.

We define the diversity threshold t_d as such a diversity, at which the final recognition accuracy of an ensemble achieves at its highest value. Such t_d has different values at different databases. In this section, in order to make the definition of t_d adapts to databases, we calculate t_d as the average value of such t_ds on four databases in the experiment.

After base classifier selection, we need to integrate the classification results of selected base classifiers $B' = \{b_1, b_2, ..., b_{L'}\}$ together. Here, we use majority voting as the integration scheme. For each test sample, we assign the class label that receives the largest vote to it.

4.3 Experimental evaluation

To evaluate the developed system, we also use the four standard face recognition datasets, namely: ORL [25], Extended Yale (Yale2) [15], AR [16], and FERET [17], which have been introduced in Section 2.4.1. In this section, we evaluated our method by comparing with a large number of face recognition algorithms on both SSS problem and SSPP problem. And then, we tested the performance of the base classifier selection algorithm. For the better comparison of our method with existing methods, we conduct the SSS problem with different number of training samples per subject. For the training and test data partition, we randomly select p samples for each subject as training data and the rest samples as test data. For each database, we perform 10 splits in this way. For each split, we use k-fold cross validation ($k = 5$) for the evaluation of perform with and without base classifier selection, respectively. Our method with and without base classifier selection (denoted by *'Our-Sel.'* and *'Our-Org.'*), respectively. The final recognition accuracy of our method is reported by averaging over $10 * k$ trials(10 random splits by k folds-cross validation). In our method, we use PCA+LDA for feature extraction and use the nearest neighbor classification with L_2-norm for matching. For each databases, we define the feature dimension of PCA+LDA subspace as $subjectNumber - 1$.

4.3.1 Experimental results

We compared our method with several existing face recognition algorithms on SSS problem, including: conventional face recognition algorithms without ensemble and several existing ensemble methods [10], [18], [26], [34], [35], [36], [37]. The results on ORL, Yale2, and FERET-1 databases are shown in Tables 3–5, respectively. In these tables, the fist column are the comparison methods while the rest columns report the rank-one recognition accuracy of comparison methods using p samples per subject as training data.

Comparison with conventional face recognition algorithms without ensemble The comparison methods without ensemble include: PCA family [36], LDA family [19], LPP family (Locality Preserving Projections) [19], CCA family (canonical correlation analysis) [35], and several other representative methods such as SVM, neural network based methods (MLP (multilayer perceptron) and RBF (radial basis function network)) [36].

From Tables 3–5, we can see that our method obtains higher performance in all experiments compared to all conventional methods without ensemble. This thanks to two main reasons:

Table 3: Evaluation on the ORL database

Method		p=2	p=3	p=4	p=5
PCA		66.9	76.6	82.1	86.3
LDA family	LDA	72.5	84.0	89.4	92.8
	PCA+LDA	77.7	86.1	90	92.7
	R-LDA	79.1	89.0	93.7	96.4
	S-LDA	82.9	91.9	95.9	97.7
CCA family	PCA+CCA	81.3	87.5	89.2	91.8
	CCA+Perturbation	81.3	87.8	89.5	92.7
	KPCA+CCA	81.5	88.8	92.8	93.5
	2DCCA	85.0	89.5	93.3	95.0
LPP family	LPP	78.0	86.2	90.3	93.2
	R-LPP	79.1	89.1	93.6	96.4
	S-LPP	82.9	91.9	95.9	97.7
	OLPP	79.5	89.2	93.6	96.2
Ensemble	RS	76.8	83.6	87.8	93.3
	PCRC	70.6	81.8	87.9	89.5
	MPCRC	78.4	84.3	87.1	91.5
	30Region	80.6	87.8	90.7	94.8
	Data-Adaptive Block	79.1	87.5	93.3	96.0
Our	Org.	86.3	92.2	95.4	97.5
	Sel.	**87.1**	**93.1**	**96.2**	**98.1**

Table 4: Evaluation on the Yale2 database

Method		p=5	p=10
PCA		36.4	53.6
LDA family	LDA	75.5	87.5
	PCA+LDA	76.3	87.0
	R-LDA	77.2	89.6
CCA family	PCA+CCA	73.0	86.0
	CCA+Perturbation	73.0	87.0
	KPCA+CCA	75.0	88.0
	2DCCA	87.5	91.5
LPP family	LPP	67.9	81.5
	Tensor-LPP	71.7	82.9
	OLPP	71.6	83.7
Ensemble	RS	84.3	95.8
	PCRC	91.0	98.8
	MPCRC	92.8	**99.1**
	30Region	90.3	97.8
	Data-Adaptive Block	91.9	98.7
Our	Org.	95.0	98.8
	Sel.	**95.5**	98.9

Table 5: Evaluation on the FERET-1 database

Method		p=2	p=3	p=4
PCA family	PCA	82.4	86.6	89.4
	2DPCA	81.9	86.4	89.2
	KPCA	82.3	87.8	91.6
SVM family	SVM	68.8	91.7	95.1
	PCA+SVM	91.5	95.8	97.2
Neural Networks	MLP	72.9	83.4	85.9
	RBF	85.3	93.2	96.8
Ensemble	RS	75.7	81.5	85.7
	LBP-5 × 5	89.4	92.1	94.2
	LBP-7 × 7	91.5	94.4	96.0
	LBP-7 × 7w	**92.9**	95.1	96.6
	30Region	73.0	92.3	82.1
	Data-Adaptive Block	84.3	93.9	94.5
Our	Org.	85.1	96.9	96.4
	Sel.	91.9	**96.9**	**98.2**

(1) the original training sample set can be enlarged effectively by generating new samples from QT-E such that the face space can be represented more accurately by our method; (2) bases classifiers learned from enlarged training data are more diverse for collaboration such that the overall discriminative power is much greater than a single classifier. Our method performs better than conventional methods in dealing with the SSS problem thanking

Table 6: Evaluation on the AR and FERET-2 databases for SSPP face recognition

Method	AR							FERET	Year
	B	C	D	E	F	G	H		
PCA	97	87	60	77	76	67	38	84.0	1991
$(PC)^2A$	97	87	62	77	74	67	40	84.5	2002
$E(PC)^2A$	97	87	63	77	75	68	41	85.5	2004
2DPCA	97	87	60	76	76	67	37	84.5	2004
$(2D)^2PCA$	98	89	60	71	76	66	41	85.0	2005
SOM	98	88	64	73	77	70	42	91.0	2005
LPP	94	87	36	86	74	78	20	84.0	2005
SVD-LDA	73	75	29	75	56	58	19	85.5	2005
Block PCA	97	87	60	77	76	67	38	84.5	2004
Block LDA	85	79	29	73	59	59	18	86.5	2004
UP	98	88	59	77	74	66	41	90.0	2010
30region	91	94	37	91	66	81	22	86.0	2012
MPCRC	87	95	25	**96**	80	**88**	9	79.0	2012
ADA	N/A	N/A	N/A	N/A	N/A	N/A	N/A	92.6	2013
DMMA	**99**	93	**69**	88	**85**	79	45	93.0	2013
Our-Sel.	98	**96**	55	90	83	80	**48**	**94.5**	2014

to both new sample generation and ensemble learning.

Comparison with existing ensemble methods: We compared our method with three kinds of representative ensemble methods: (1) global feature selection based on random subspace [34]; (2) patch based local feature extraction such as PCRC [18] and a patch based method using LBP (Local Binary Pattern) [36]; (3) global and local feature integration methods such as the 30 region method [10] which defines 30 regions with large overlaps based on experimental experience and the multi-scale patch based method MPCRC [18] which integrates the collaboration of multi-scale patches to construct the global feature representation. The performance of LBP methods are reported in [36]. From Tables 3–5, we find that our method outperforms the existing ensemble methods in almost all experiments thanking to three benefits: (a) Our method expands the training sample set by generating new samples. Then, we can generate more diverse and accurate base classifiers to estimate the face space; (b) In contrast to existing methods, which divide face regions into separated patches, QT-E divides face region into blocks of variant sizes according to a tree-structure, which preserves the geometric correlationship between local blocks; (c) Other than existing methods, which involve all base classifiers for ensemble, our method develops a base classifier selection algorithm, which just selects appropriate base classifiers for ensemble. From Tables 3–5, we can see that Our-Sel. has better performance than Our-Org. for almost all splits of databases, which verifies the effectiveness of base classifier selection.

Comparison with State-of-the-arts on SSPP problem: We compared our method with 15 state-of-the-arts on SSPP problem, including PCA, (PC)2A [38], E(PC)2A [39], 2DPCA [40], (2D)2PCA [41], SOM [42], LPP [43], SVD-LDA [44], Block PCA [45], Block LDA [46], UP [47], 30region [10], MPCRC [18], ADA [37], and DMMA [48]. Table 6 shows the rank-one recognition accuracy of these methods on the AR and FERET-2 databases. We find that our method outperforms most of the comparison methods and obtains comparable performances with the recently developed DMMA algorithm on AR database and outperforms DMMA with an accuracy gain of 1.0 percent on the FERET-2 database. The reason why our algorithm is comparable to these state-of-the-arts thanking to the strategy of new sample generation which does expand the face space in the training stage. Benefit from introduction of random matrices, our new generated samples are quite diverse and different from the original training data compared with existing virtual sample generation methods. In addition, our method not only encodes discriminant features but also geometric information, which is also an important cue for recognition.

4.4 Comparison with data-adaptive block based algorithm

We compared this algorithm with Algorithm 1, which performs feature learning with large training data. The main difference between these two algorithms is that Algorithm 1 utilizes the original small data while this algorithm generate base classifiers after data argumentation. From Tables 3–5, we observe that when the training data is very small for example (P=2,3 for ORL, p=5 for Yale 2, and p=2 for FERET-1), Algorithm 1 has a very low-performance. However, along with the increasing of the training data (e.g. p=5 for ORL, p=10 for Yale 2, and p=3, 4 for FERET-1) the difference between these two method becomes obscure. This suggest that Algorithm 1 can work well when the training data is large enough. However, it suffers from performance degradation when the training data is small, which verifies the effectiveness of this method in solving the SSS problem by generating new samples.

5. Feature Learning in Dynamic Environments using Helmholtz-Hodge Decomposition and Quad-Tree

In the last three sections (section 2–4), we have investigated feature learning on static images (image dataset). In order to further evaluate the reliability of our method, in this section, we investigates feature learning in moving camera videos.

5.1 Problem review and related works

Video based feature learning is a hot research topic in the recent years thanking to the increasing development of mobile devices such as smart watch, smart phone, wearable devices (e.g. Google glasses, Baidu Eyes), and so on. In the moving camera videos, camera motion and local object motions are mixed and dependent with other. Local object motions can offer motion features to many video based applications such as action recognition, tracking, content based surveillance, etc.. Since the camera mo-

tion and local object motions can influence with each other, if we use the original mixed motion field for these applications, object motion based applications will be very challenging and less accurate. Thus, we need to segment object motions from the scene and recover them to their true values.

As local object motions can be very complicated, it is difficult to model and segment them directly. An effective approach is to compensate the camera-induce image motion first. And the residue motions must belong to moving objects. Dense motion based methods are important camera motion compensation methods, which perform pixel-level segmentation on image plane motion (optical flow). They usually assume the camera-induced image motion by a parametric transformation ranging from translation to perspective transformation using different parameters [49]. Pixels that are consistent with the estimated model are supposed to be inlier, while others are supposed to outliers. Since the inlier estimation is often affected by outliers, several outlier removal methods were proposed, such as a regression scheme using gradient descent (GD) [49] or least squares (LS) [50], outlier rejection filter [51], RANSAC [52], joint inlier estimation and motion segmentation method [53], [54], state-of-art algorithm [55]. In contrast to feature based methods, dense based methods do not rely on the strong key features tracked throughout the scene. However, they suffer from two big problems: (1) the parametric models used for inlier estimation are just approximations of camera motions which only hold for the restricted cases of camera motion.(2) The image plane motion depends on the distance of 3D points from the camera.

5.2 Algorithm architecture

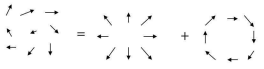
To solve the above mentioned two problems, in this section, we develop a dependent motion segmentation algorithm by introducing an amended Helmholtz-Hodge decomposition (HHD) and a data-driven Quad-Tree partition. HHD is used for camera motion compensation while Quad-Tree is for object motion segmentation. HHD can interpret the three kinds of camera-induced image motions by its two components: curl-free component and divergence-free component. And the Quad-Tree based object motion segmentation can deal with depth discontinuities in 3D scenes. The outline of our algorithm is illustrated in Fig. 4.

5.2.1 HHD based camera motion modelling in 3D scenes

Helmholtz-Hodge Decomposition

Helmholtz-Hodge decomposition is one of the fundamental theorems in fluid dynamics. The principle of HHD [56] is explained as follows: for an arbitrary image motion field $\vec{\xi}$, it decomposes it into two components: a curl-free component ∇E and a divergence-free component $\nabla \times \vec{W}$:

$$\vec{\xi} = \nabla E + \nabla \times \vec{W}. \tag{10}$$



Here, E and W are 3D potential surfaces defined as: 1. *Scalar potential surface*, whose gradient is the curl-free component ∇E; 2. *Vector potential surface*, whose curl operation denotes the divergence-free component $\nabla \times \vec{W}$ [56].

The three kinds of camera induced image motions can be interpreted by the two components of HHD in this way:

- Radial motion is irrational, and can be represented by the curl-free component;
- Rotation is incompressible, and can be represented by the divergence-free component.
- Translation is irrational and incompressible, and can be represented by both curl-free and divergence-free components without any change.

For the implementation, we add two assumptions: (1) the original motion field should be piece-wise smooth; (2) HHD is implemented based on global minimization. The implementation is as follows: since ∇E and $\nabla \times \vec{W}$ are the projection of the original motion field $\vec{\xi}$ to the space of the curl-free field and the divergence-free field, respectively, the distance between $\vec{\xi}$ and two projected components should be minimal. Therefore, we apply energy minimization to calculate the two components.

$$\begin{aligned} &min(D(E)) = min(\int_\Omega \parallel \nabla E - \vec{\xi} \parallel^2 d\Omega), \\ &min(G(\vec{W})) = min(\int_\Omega \parallel \nabla \times \vec{W} - \vec{\xi} \parallel^2 d\Omega). \end{aligned} \tag{11}$$

where Ω denotes the image domain. Please refer to [57] for implementation details.

Inlier/Outlier representation Inlier is the abbreviation of inlier optical flow, which refers to the camera-induced image motion. Outlier is the abbreviation of outlier optical flow, which refers to motion discontinuities due to the relative motion between moving objects and camera, depth discontinuities, and noises. For the inlier, most part of it is caused by camera motion, which is smooth and continuous. Here we use $\vec{V}_{inlier}^{homogeneous}$ to represent this part, where the "homogeneous" refers to the continuity of the motion field. However, motion discontinuities occur at the boundaries of scene objects due to depth discontinuities. Thus, we use $\vec{V}_{inlier}^{heterogeneous}$ to represent this part. In contrast to inlier, outliers belonging to the motion field of moving objects. It suffers from motion discontinuities because of the relative motion between camera and moving objects. Most part of outliers are heterogeneous and just very small part of them are homogeneous. Thus, we use $\vec{V}_{outlier}^{heterogeneous}$ and $\vec{V}_{ioutlier}^{heterogeneous}$ to represent the heterogeneous and homogeneous part, respectively. Both the inlier and outlier can be represented by the homogeneous part and heterogeneous part as below:

$$\begin{aligned} &\vec{V}_{inlier} = a_1 \vec{V}_{inlier}^{homogeneous} + b_1 \vec{V}_{inlier}^{heterogeneous}, a_1 >> b_1, \\ &\vec{V}_{outlier} = a_2 \vec{V}_{outlier}^{homogeneous} + b_2 \vec{V}_{outlier}^{heterogeneous}, a_2 << b_2. \end{aligned} \tag{12}$$

where a_1, b_1, a_2, b_2 are quantity coefficients. For the inlier, most part of it is homogeneous, thus $a_1 >> b_1$. It is opposite for the outliers, thus $a_2 << b_2$. The homogeneous part should be represented by a low-order polynomial function while the heterogeneous part should be represented by a high-order polynomial function.

Camera motion representation by HHD

Based on the above discussion, the optical flow field can be represented by a low-order polynomial function for $\vec{V}_{inlier}^{homogeneous}$, $\vec{V}_{outlier}^{homogeneous}$, and by a high-order polynomial function for $\vec{V}_{inlier}^{heterogeneous}$, $\vec{V}_{outlier}^{heterogeneous}$. In our algorithm, the polynomial function actually corresponds to the potential surface of HHD. As we

added two assumptions to the implementation of HHD, the potential surfaces of the amended HHD are piece-wise smooth. They approximate the basic shape of motion field and thus corresponds to a low-order polynomial function. Thus, the homogeneous motion field, which should be represented by a low-order polynomial function can be interpreted by the two components of HHD as follows:

$$\{\vec{V}_{inlier}^{Homogeneous}, \vec{V}_{outlier}^{Homogeneous}\} = k_1(\nabla E) + k_2(\nabla \times \vec{W}). \quad (13)$$

where k_1 and k_2 are two regularization parameters. They will be determined in the next section.

The heterogeneous motion, which corresponds to a high-order polynomial function, cannot be represented by HHD and will be reserved in a reminder. In the next section, we will construct an object-motion oriented map (OOM) from this remainder to detect heterogeneous motions.

5.2.2 Object motion modelling in 3D scenes

To construct the object motion oriented map, we fist need to represent the homogeneous motion according to Eq. (13). Then OOM can be calculated by subtracting the homogeneous motion from the original motion field $\vec{\xi}$ as follows:

$$OOM = \vec{\xi} - k_1(\nabla E) - k_2(\nabla \times \vec{W}). \quad (14)$$

The key point is the determination of the two parameters k_1 and k_2, which should be discussed according to the type of the camera motion involved in the scene. In this paper, we define two distance functions in Eq.(15) for this purpose.

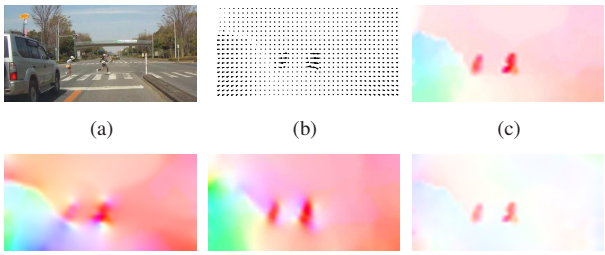
$$k_1 = \frac{\|\xi - \nabla E\|}{\|\xi\|}, \quad k_2 = \frac{\|\xi - \nabla \times \vec{W}\|}{\|\xi\|}, \quad (15)$$

where k_1 denotes the distance between $\vec{\xi}$ and the curl-free component, and k_2 denotes the distance between $\vec{\xi}$ and the divergence-free component.

We use an example 3D scene in Fig. 14 to illustrate the procedure of OOM construction. In the image sequence (a), two people are walking from left to right, a car is stopping at the left side. A camera is moving behind the car. From the original optical flow field in (b) and its color image in (c), we can see that the car shows larger motion than other pure flat scene area in distance. After HHD decomposition, the OOM in (f) demonstrates that most part of homogeneous motion field of static scene objects has been removed from OOM. It only contains heterogeneous motion caused by depth discontinuities and moving objects.

5.2.3 Quad-Tree based motion segmentation

To detect the heterogeneous motion, we need to label them from OOM. As depth discontinuities and moving objects vary at different locations, it is rather difficult to label them based on global thresholding. To this end, we introduce a data-driven Quad-Tree scheme, which casts the heterogeneous motion labeling to local subregions. If a region is determined as heterogeneous according to a criterion function in Eq.(16), it will be divided into four subregions. The criterion function considers the following two conditions:



(a) (b) (c) (d) (e) (f)

Fig. 14: An example 3D scene: (a) one frame; (b) the input optical flow; (c) visualization of the input optical flow by color coding [58]; (d) curl-free component; (e) divergence-free component; (f) OOM. (Copyrighted by Springer LNCS [57])

(a) (b)

Fig. 15: Quad-Tree partition on the example 3D scene: (a) OOM; (b) Quad-Tree partition. (Copyrighted by Springer LNCS [57])

- The variance of pixel values in a region R is higher than or equal to a threshold variance T_{var};
- The mean pixel value in R is higher than or equal to a threshold mean value T_{mean};

Mathematically it can be formulated as:

$$doS\,plit(R) = true, while \begin{cases} var(R) \geq T_{var} & or, \\ mean(R) \geq T_{mean}. \end{cases} \quad (16)$$

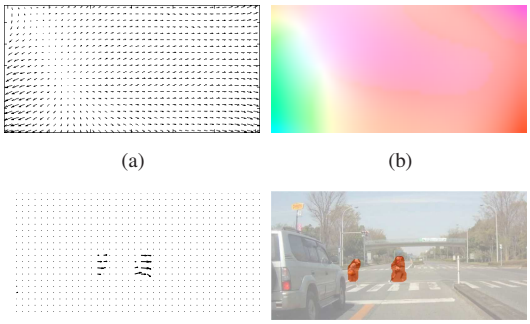
We apply the first condition to detect the heterogeneous motion caused by depth discontinuities ($\vec{V}_{inlier}^{heterogeneous}$), where the values changes violently in local regions and second condition to detect the heterogeneous motion caused by moving objects ($\vec{V}_{outlier}^{heterogeneous}$), where higher than the threshold implies the local object motions occupy larger part of the region. Partition performs till no more regions can be split. Regions of smallest size *smallR* are labeled as foreground regions containing heterogeneous motions and will be excluded from the inlier estimation in the next procedure. The rest larger regions *largeR* = *wholeR* − *smallR* represent the homogeneous region and will be evolved in inlier estimation, where *wholeR* represents the whole region of OOM.

5.2.4 Camera motion estimation using surface fitting

After Quad-Tree based heterogeneous motion labeling on OOM, we will use the rested regions, which correspond to homogeneous motion field for inlier estimation. We first illustrate the procedure on scalar potential surface E. The procedure on $\vec{W}$ is analogous.

As aforementioned, the potential surface of HHD should be smooth and represented by a low-order polynomial function. Thus, we formulate the problem of inlier estimation from E as construction of a new smooth surface E', which approximates the smooth basic shape of E. We employ a surface fitting solution using a low-order polynomial function as follows:

$$z = a_{d0}x^d + a_{0d}y^d + \cdots + a_{ij}x^i y^j + \cdots + a_{10}x + a_{01}y + a_{00}, \quad (17)$$



(a) (b) (c) (d)

Fig. 16: Inlier estimation and outlier recovery on the example 3D scene: (a) the estimated inlier optical flow; (b) color visualization of (a); (c) the recovered outlier optical flow; (d) segmentation result. (Copyrighted by Springer LNCS [57])

In our method, a polynomial of $d = 5$ is employed so as to produce a smooth and accurate surface E'. Similarly, we get a new smooth potential surface $\vec{W}'$ which approximate the base of $\vec{W}'$. The inlier of curl-free component is calculated by $G_1 = \nabla E'$. The inlier of divergence-free component is computed by $G_2 = \nabla \times \vec{W}'$. The final inlier optical flow is estimated by linear combination of G_1 and G_2 using Eg.(18). Figure 16 (a), (b) shows the estimated inlier of the example 3D scene. We can see that our method estimates the inlier accurately.

$$G = k_1 G_1 + k_2 G_2, \tag{18}$$

5.2.5 Object motion recovery

After inlier estimation, outliers can be recovered by subtracting the inlier from the original motion field subsequently. Figure 16 shows the recovered outlier motion field in (c) and the final segmentation map in (d).

5.3 Experimental evaluation

5.3.1 Datasets and experiments design

The system's performance is evaluated on four benchmark datasets: Hopkins [59], Berkeley Motion Segmentation [60], Complex Background [55], and SegTrack [61]. The Hopkins dataset contains three category video sequences: checkerboard, car, and people. It provides ground truth segmentation on selected features tracked throughout the sequence. The Berkeley dataset is derived from the Hopkins dataset, which consists of 26 moving camera videos: car, people, and Marple sequences. This dataset has full pixel-level annotations on multiple objects for a few frames sampled throughout the video. We use the other two datasets: Complex Background and SegTrack, which contain extremely challenging scenes to highlight the strength of our method. They provide full pixel-level annotations on multiple objects at each frame within each video.

5.3.2 Experimental results

In this part, we compare our method with six recent developed dense motion segmentation methods including: (1) joint inlier estimation and segmentation (GME-SEG) [53], (2-3) iterative estimation based on least-square (LS) [50], and gradient decent (GD) [49], (4) outlier rejection filter (Filter) [51], (5) RANSAC [52], and (6) the latest developed FOF [55], which is considered as the state-of-art algorithm in motion segmentation. In [55], they present two versions: (1) FOF, which uses optical flow only, and (2) FOF+color+prior, which combines optical flow, color appearance and a prior model together. The first five methods were implemented using the source code in [53]. We reported the performances of FOF and FOF+color+prior [55] directly.

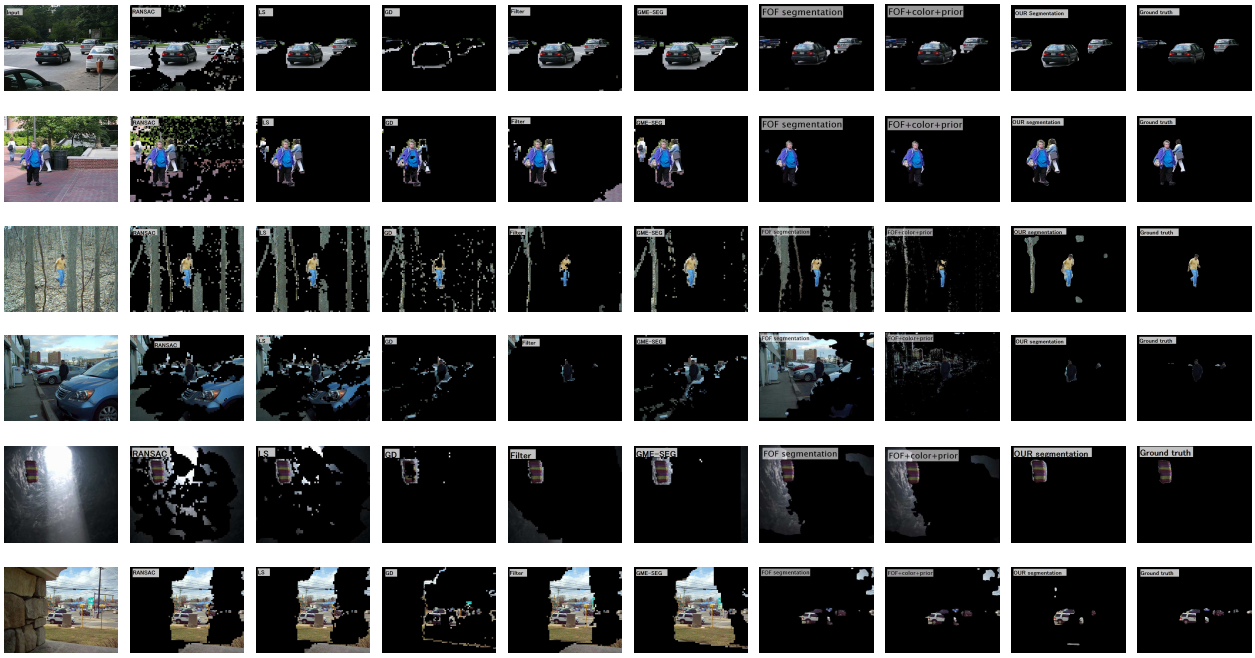
We used the F-measure [55] to evaluate the performance of each algorithm, which is an important performance analysis parameter. Table 7 reports the F-measure of dense based methods on three benchmark datasets: Berkeley, Complex Background, and SegTrack. From Table 7, we observe that our method achieves at the highest performance for almost all videos: it raises the F-measure by 10% – 30% on *Cars* 2, 3, 4, 7, and *People* 1 sequence in the Berkely dataset; around 10% on the *drive, parking*, and *store* sequences in the Complex Background dataset; and more than 20% on the *parachutte* and *monkeydog* sequences in the SegTrack dataset. This result is quite appealing even when videos contain extremely challenging scenes, such as the ones with complex camera motions, complex backgrounds, occlusions, etc.. The good quantitative results are confirmed by the good visual quality of the segmentation results. Some examples are shown in Fig. 17, where the last column shows the ground truth segmentation. In most cases, our segmentation agrees with the true object regions more than existing methods.

6. Discussion: relationship between four algorithms

We need to explicitly mention that the first three algorithms are highly related with each other while the fourth algorithm is different from them in some sense. The first three algorithms share the same research background (face recognition). Their problem definition and algorithm description are also similar. However, the research background of the last algorithm is on motion segmentation. It investigates feature extraction from video sequences rather than image dataset. Thus, the problem definition and algorithm description are also different from the first three algorithms. However, these four algorithms share the same Quad-Tree based image encoding, which is the core-aspect of this dissertation.

7. Conclusion

This dissertation develops four Quad-Tree based image encoding methods towards four visual feature situations: (1) Feature learning from data-adaptive blocks decomposed by Quad-Tree;(2) Hierarchical feature learning using Quad-Tree structure of images;(3) Feature learning from enlarged training data encoded by Quad-Tree for Small Sample Size problem ;(4) Feature learning in dynamic environments using Helmholtz-Hodge decomposition and Quad-Tree. They are highly related with each other and occupy a wide range of feature learning issues. Features can be extracted from the planar image surface or using hierarchical models, with large training data or small data, from static images or video sequences. The first three algorithms are developed for feature learning from static images (image dataset) while the last algorithm is for object motion segmentation in video sequences. We use face recognition as research background for the first three algorithms and motion segmentation for the last



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Fig. 17: Segmentation results of six existing dense based methods and ours on challenging scenarios: (a) input sequences, from top to bottom: cars2, people2, forest, store, parachute, traffic, segmentation by (b) RANSAC [52], (c) LS [50], (d) GD [49], (e) Filter [51], (f) GME-SEG [53], (g) FOF [55], (h) FOF+color+prior [55], (i) our segmentation, (j) ground-truth segmentation.

Table 7: F-measure of existing dense based methods and ours.

Sequences	GME-SEG	LS	GD	Filter	RANSAC	FOF	FOR+color	Our
Cars1	78.33	**86.18**	18.01	82.87	60.42	47.81	50.84	76.38
Cars2	55.90	65.97	15.70	78.28	34.21	46.37	56.60	**83.13**
Cars3	65.21	79.43	22.29	74.56	35.80	67.18	73.57	**87.37**
Cars4	45.69	49.78	22.96	77.22	22.81	38.51	47.96	**84.42**
Cars5	54.67	61.93	33.08	81.17	25.24	64.85	70.94	**84.82**
Cars6	33.01	51.23	28.77	57.53	13.40	78.09	84.34	**85.71**
Cars7	37.89	36.36	36.92	60.47	13.79	37.63	42.92	**86.10**
Cars8	62.20	81.24	8.57	78.44	37.02	87.13	87.61	**90.68**
Cars9	72.99	**80.99**	17.97	68.19	54.69	68.99	66.38	77.42
Cars10	60.01	66.04	14.34	**90.95**	81.78	53.98	50.84	54.83
People1	34.11	38.32	40.76	71.84	12.06	56.76	69.53	**80.03**
People2	78.16	84.45	69.30	81.70	37.53	85.35	88.40	**89.81**
drive	8.14	6.30	41.18	32.40	5.76	30.13	61.80	**83.03**
forest	15.42	11.01	15.41	19.87	10.67	19.48	31.44	**35.71**
parking	33.20	21.57	43.84	62.29	17.47	43.47	73.19	**83.47**
store	14.39	10.94	32.10	29.32	9.68	28.46	70.74	**80.10**
traffic	14.77	15.80	34.55	15.04	15.49	66.08	71.24	**71.67**
birdfall2	9.39	3.84	0.99	64.00	3.25	68.68	75.69	**76.23**
girl	22.51	20.26	15.36	18.21	12.33	75.73	**81.95**	78.06
parachute	23.01	18.97	12.88	16.30	44.03	51.49	54.36	**86.72**
cheetah	21.33	14.93	43.59	12.05	9.85	12.68	22.31	**55.67**
penguin	10.34	18.84	15.34	5.53	18.66	14.74	20.71	**21.61**
monkeydog	22.29	20.74	16.46	18.93	12.31	10.79	18.62	**45.44**

algorithm. Experimental results on a large number of benchmark datasets demonstrate the good performance of our methods.

Apart from face recognition and motion segmentation, our methods can be widely applied to many other computer vision applications thanking to the data-adaptive property. For the future direction, we are planning to extend our methods to other image classification and recognition problems, to explore more effective features and classifiers, to employ our Quad-Tree based methods in other hierarchical models such as Convolutional Neural Networks (CNNs), Dynamic Belief Networks (DBNs).

Acknowledgement

This work is supported by: the Japan Society for the Promotion of Science, Scientific Research KAKENHIfor the Grant-in-Aid for Young Scientists (ID: 25730113).

References

[1] Dai, J., Hong, Y., Hu, W. and Wu, Y. N.: Unsupervised Learning of Dictionaries of Hierarchical Compositional Models, *CVPR* (2014).

[2] Rumelhart, D. E., G. E., H. and Williams, R. J.: Learning representations by backpropagating errors, *Nature*, Vol. 333, pp. 533–536 (1986).

[3] Zhang, C., Liang, X. and Matsuyama, T.: Multi-subregion face recognition using coarse-to-fine Quad-tree decomposition, *ICPR*, pp. 1004–1007 (2012).

[4] Zhang, C., Liang, X. and Matsuyama, T.: Multi-Depth Deep Feature Learning for Face Recognition, *INIC* (2014).

[5] Zhang, C., Liang, X. and Matsuyama, T.: Generic Learning-Based Ensemble Framework for Small Sample Size Face Recognition in Multi-Camera Networks, *Sensors*, Vol. 14, No. 12, pp. 23509–23538 (2014).

[6] Turk, M. and Pentland, A.: Eigenfaces for recognition, *J. Cognitive Neuroscience*, Vol. 3, No. 1 (1991).

[7] Belhumeur, P. N., Hespanha, J. P. and Kriegman, D. J.: Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection, *TPAMI*, Vol. 20, No. 7, pp. 711–720 (1997).

[8] Queirolo, C. C., Silva, L., Bellon, O. R. P. and Segundo, M. P.: 3D face recognition using simulated annealing and the surface interpenetration measure, *TPAMI*, Vol. 32, pp. 206–219 (2010).

[9] Alyüz, N., Gökberk, B. and Akarun, L.: Regional registration and curvature descriptors for expression resistant 3D face recognition, *SIU*, pp. 544–547 (2009).

[10] Spreeuwers, L.: Fast and accurate 3D face recognition using registration to an intrinsic coordinate system and fusion of multiple region classifiers, *IJCV*, Vol. 93, No. 3, pp. 389–414 (2011).

[11] Zhang, C., Uchimura, K., Koutaki, G. and Zhang, C. M.: 3D face recognition using multi-level multi-feature fusion, *PSIVT* (2010).

[12] Jain, V. and Mukherjee, A.: The Indian Face Database, *http://vis-w.cs.umass.edu/ vidit/IndianFaceDatabase/*.

[13] Dailey, M. N., Joyce, C., Lyons, M. J., Kamachi, M., Ishi, H. and Gyoba, J.: Evidence and a computational explanation of cultural differences in facial expression recognition, *Emotion*, Vol. 10, No. 6, pp. 874–893 (2010).

[14] Nefian, A. V. and Hayes, M. H.: Hidden Markov models for face recognition, *Proc. Int. Conf. Acoustics, Speech, Signal Processing*, pp. 2721–2724 (1998).

[15] Georghiades, A. S., Belhumeur, P. N. and Kriegman, D. J.: From Few to Many: Illumination Cone Models for Face Recognition under Variable Lighting and Pose, *TPAMI*, Vol. 23, No. 6, pp. 643–660 (2001).

[16] Martinez, A. M. and Benavente, R.: The AR face database, *CVC Technical Report* (1998).

[17] Phillips, P. J., Moon, H., Rizvi, S. A. and Rauss, P. J.: The FERET evaluation methodology for face recognition algorithms, *TPAMI*, Vol. 22, No. 10, pp. 1090–1104 (2000).

[18] Zhu, P., Zhang, L., Hu, Q. and Shiu, S.: Multi-scale Patch based Collaborative Representation for Face Recognition with Margin Distribution Optimization, *ECCV* (2012).

[19] Cai, D., He, X., Hu, Y., Han, J. and Huang, T.: Learning a Spatially Smooth Subspace for Face Recognition, *CVPR* (2007).

[20] CSU: CSU Face Identification Evaluation System (2003).

[21] LeCun, Y., Bottou, L., Bengio, Y. and Haffner., P.: Gradient-based learning applied to document recognition, *Proceedings of the IEEE*, pp. 2278–2234 (1998).

[22] Hinton, G. E. and Osindero, S.: A fast learning algorithm for deep belief nets, *Neural Computation*, Vol. 18, pp. 1527–1554 (2006).

[23] Hinton, G. E., Osindero, S. and Teh, Y. W.: A fast learning algorithm for deep belief nets, *Neural Computation*, Vol. 18, pp. 1527–1554 (2006).

[24] Mitchell, B. and Sheppard, J.: Deep structure learning: beyond connectionist approaches, *ICMLA'12* (2012).

[25] Samaria, F. S. and Harter, A. C.: Parameterisation of a Stochastic Model for Human Face Identification, *Proceedings of 2nd IEEE Workshop on Applications of Computer Vision*, pp. 138–142 (1994).

[26] Lu, J., Tan, Y.-P. and Wang, G.: Discriminative Multimanifold Analysis for Face Recognition from a Single Training Sample per Person, *TPAMI*, Vol. 35, No. 1, pp. 39–51 (2013).

[27] Zhang, C., Liang, X. and Matsuyama, T.: Small Sample Size F ace Recognition using Random Quad-Tree based Ensemble Algorithm, *ICDP* (2013).

[28] Skurichina, M. and Duin, R. P. W.: Bagging, boosting and the random subspace method for linear classifiers, *Pattern Analysis and Applications*, Vol. 5, No. 2, pp. 121–135 (2002).

[29] Martinez, A. M. and Kak, A. C.: PCA versus LDA, *TPAMI*, Vol. 23, No. 2, pp. 228–233 (2001).

[30] Topcu, B. and Erdogan, H.: Decision Fusion for Patch-Based Face Recognition, *ICPR*, pp. 1348–1351 (2010).

[31] Su, Y., Shan, S., Chen, X. and Gao, W.: Hierarchical ensemble of global and local classifiers for face recognition, *IEEE Transactions on Image Processing*, Vol. 18, No. 8, pp. 1885–1896 (2009).

[32] Tang, E. K., Suganthan, P. N. and Yao, X.: An analysis of diversity measures, *Machine Learning*, Vol. 65, No. 1, pp. 247–271 (2006).

[33] Arodz, T.: Margin-based Diversity Measures for Ensemble Classifiers, *Advances in Soft Computing Volume*, Vol. 30, pp. 71–78 (2005).

[34] Wang, X. and Tang, X.: Random sampling for subspace face recognition, *IJCV*, Vol. 70, No. 1, pp. 91–104 (2006).

[35] Sun, N., Ji, Z., Zou, C. and Zhao, L.: Two-dimensional canonical correlation analysis and its application in small sample size face recognition, *Neural Computing and Applications*, Vol. 19, pp. 377–382 (2010).

[36] Oravec, M., Pavlovicova, J., Mazanec, J., Omelina, L., Feder, M. and Ban, J.: Efficiency of Recognition Methods for Single Sample per Person Based Face Recognition, *Refinements and New Ideas in Face Recognition, Dr. Peter Corcoran (Ed.), ISBN: 978-953-307-368-2, InTech*, pp. 1885–1896 (2011).

[37] Kan, M., Shan, S., Su, Y., Xu, D. and Chen, X.: Adaptive discriminant learning for face recognition, *Pattern Recognition*, Vol. 46, No. 9, pp. 2497–2509 (2013).

[38] Wu, J. and Zhou, Z.-H.: Face recognition with one training image per person, *Pattern Recognition Letters*, Vol. 23, No. 14, pp. 1711–1719 (2002).

[39] Chen, S., Zhang, D. and Zhou, Z.-H.: Enhanced (PC)2A for face recognition with one training image per person, *Pattern Recognition Letters*, Vol. 25, No. 10, pp. 1173–1181 (2004).

[40] Yang, J., Zhang, D., Frangi, A. and Yang, J.: Two-dimensional PCA: A new approach to appearance-based face representation and recognition, *TPAMI*, Vol. 26, No. 1, pp. 131–137 (2004).

[41] Di, W., Zhang, L., Zhang, D. and Pan, Q.: Studies on Hyperspectral Face Recognition in Visible Spectrum With Feature Band Selection, *TPAMI*, Vol. 40, No. 6, pp. 1354–1361 (2010).

[42] Tan, X., Chen, S., Zhou, Z.-H. and Zhang, F.: Recognizing partially occluded, expression variant faces from single training image per person with SOM and soft k-NN ensemble, *IEEE Transactions on Neural Networks*, Vol. 16, No. 4, pp. 875–886 (2005).

[43] He, X., Yan, S., Hu, Y., Niyogi, P. and Zhang, H. J.: Face recognition using Laplacianfaces, *TPAMI*, Vol. 27, No. 3, pp. 328–340 (2005).

[44] Zhang, D., Chen, S. and Zhou, Z.-H.: A new face recognition method based on SVD perturbation for single example image per person, *Applied Mathematics and Computation*, Vol. 163, No. 2, pp. 895–907 (2005).

[45] Gottumukkal, R. and Asari, V. K.: An improved face recognition technique based on modular PCA approach, *Pattern Recognition Letters*, Vol. 25, No. 4, pp. 429–436 (2004).

[46] Chen, S., Liu, J. and Zhou, Z.-H.: Making FLDA applicable to face recognition with one sample per person, *Pattern Recognition*, Vol. 37, No. 7, pp. 1553–1555 (2004).

[47] Deng, W., Hu, J., Guo, J., Cai, W. and Feng, D.: Robust, accurate and efficient face recognition from a single training image: A uniform pursuit approach, *Pattern Recognition*, Vol. 43, No. 5, pp. 1748–1762 (2010).

[48] Lu, J., Tan, Y.-P. and Wang, G.: Discriminative Multi-Manifold Analysis for Face Recognition from a Single Training Sample per Person.

[49] Su, Y., Sun, M.-T. and Hsu, V.: Global motion estimation from coarsely sampled motion vector field and the applications, *IEEE Transactions on Circuits Systtem and Video Technology.*, Vol. 15, No. 2, pp. 232–242 (2005).

[50] Smolic, A., Hoeynck, M. and Ohm, J.-R.: Low-complexity global motion estimation from P-frame motion vectors for MPEG-7 application, *ICIP*, pp. 271–274 (2000).

[51] Chen, Y. M. and Bajic, I. V.: Motion vector outlier rejection cascade for global motion estimation, *IEEE Signal Processing Letters*, Vol. 17, No. 2, pp. 197–200 (2010).

[52] Fischler, M. and Bolles, R.: RANSAC random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography, *Communications of the ACM*, Vol. 26, pp. 381–395 (1981).

[53] Chen, Y. M. and Bajic, I. V.: A joint approach to global motion estimation and motion segmentation from a coarsely sampled motion vector field, *IEEE Transactions on Circuits Systtem and Video Technology*, Vol. 21, No. 9, pp. 1316–1328 (2011).

[54] Qian, C. and Bajic, I. V.: Global motion estimation under translation-zoom ambiguity, *Proc. IEEE PacRim*, pp. 46–51 (2013).

[55] Narayana, M., Hanson, A. and Learned-Miller, E.: Coherent Motion Segmentation in Moving Camera Videos using Optical Flow Orientations, *ICCV* (2013).

[56] Bhatia, H., Norgard, G., Pascucci, V. and Bremer, P.-T.: The Helmholtz-Hodge Decomposition - A Survey, *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, Vol. 19, No. 8, pp. 1386–1404 (2013).

[57] Liang, X., Zhang, C. and Matsuyama, T.: Inlier Estimation for Moving Camera Motion Segmentation, *ACCV* (2014).

[58] Baker, S., Scharstein, D., Lewis, J. P., Roth, S., Black, M. J. and Szeliskiv, R.: A Database and Evaluation Methodology for Optical Flow, *Internations Journal of Computer Vision*, Vol. 92, pp. 1–31 (2011).

[59] Tron, R. and Vidal, R.: A benchmark for the comparison of 3D motion segmentation algorithms, *CVPR* (2007).

[60] Brox, T. and Malik, J.: Object segmentation by long term analysis of point trajectories, *ECCV* (2010).

[61] Tsai, D., Flagg, M. and M.Rehg, J.: Motion Coherent Tracking with Multi-label MRF optimization, *BMVC* (2010).