Visual Pattern Discovery for Architecture Image Classification and Product Image Search

Wei-Ta Chu and Ming-Hung Tsai
National Chung Cheng University, Chiayi, Taiwan
wtchu@cs.ccu.edu.tw, tmh96m@cs.ccu.edu.tw

ABSTRACT
Many objects have repetitive elements, and finding repetitive patterns facilitates object recognition and numerous applications. We devise a representation to describe configurations of repetitive elements. By modeling spatial configurations, visual patterns are more discriminative than local features, and are able to tackle with object scaling, rotation, and deformation. We transfer the pattern discovery problem into finding frequent subgraphs from a graph, and exploit a graph mining algorithm to solve this problem. Visual patterns are then exploited in architecture image classification and product image retrieval, based on the idea that visual pattern can describe elements conveying architecture styles and emblematic motifs of brands. Experimental results show that our pattern discovery approach has promising performance and is superior to the conventional bag-of-words approach.

Categories and Subject Descriptors
H.2.9 [**Database Management**]: Database Applications – *data mining, image databases*. I.2.10 [**Artificial Intelligence**]: Vision and Scene Understanding – *modeling and recovery of physical attributes, texture*.

General Terms
Algorithms, Performance, Experimentation.

Keywords
Pattern discovery, local feature, part-based model, visual pattern.

1. INTRODUCTION
Repetitive elements, or patterns, are ubiquitously presented in man-made objects and natural environments, such as buildings, decorations, leaves, and animal fur. Many objects have characteristic repetitive elements, and these elements provide clues to object identification. For example, an object with sash windows implies that it is a building, whereas an object with petals reveals that it is a flower. Therefore, finding repetitive patterns in images is important for image understanding, object recognition, and other applications.

The objective of this work is to automatically discover repetitive patterns in images and elaborately describe patterns for practical applications. We focus on man-made objects with clear and visually similar substructures. A found pattern is called as a "visual pattern", which is a contrast to the widely-used term "visual word" [1]. With visual patterns, we are able to describe elements conveying architecture styles. Furthermore, for product image retrieval or the so-called product search, visual patterns depict emblematic motifs of each brand.

To discover visual patterns, we need to extract image features, and then find patterns of features that appear frequently. In real-world images, a pattern may have different visual appearances under different lighting, viewpoints, scales, and occlusions. The part-based visual word representation [1] is ideal to address these issues, which first extracts local features from images, and then quantizes feature descriptors into visual words. However, visual words are limited to describe appearance in a local region, which fail to describe objects that cover large areas. Due to visual diversity of local image patches, visual words barely carry explicit semantics to represent image content. One possible solution is to group individual features into a more discriminative configuration, which is a common approach in object recognition. However, instances of the same pattern may occur at arbitrary positions with overlaps or gaps, which increase the difficulty of pattern discovery.

To tackle with these problems, we propose a higher-level feature representation that takes into account spatial relationships between local features. We treat visual patterns as subgraphs embedded in a root graph, which is induced by local features extracted from an image. Pattern discovery is therefore transformed into the problem of finding frequent subgraphs.

The primary contributions of this work are summarized as follows:

- A higher-level feature representation is constructed. It is more discriminative than visual word by modeling spatial relationships between local features, and is flexible to tackle with object rotation, scaling, and viewpoint changes.
- We exploit a graph mining algorithm to automatically detect and localize repeated elements and discover their common feature configurations.
- We demonstrate practicality of visual patterns by conducting architecture image classification and product search.

The rest of this paper is organized as follows. Related works are reviewed in Section 2. The visual pattern discovery framework is introduced in Section 3. In Section 4, we present two applications based on visual patterns. Section 5 provides evaluation results of pattern discovery and image classification/retrieval. Conclusion and future works are given in Section 6.

ICMR'12, June 5-8, Hong Kong, China

2. RELATED WORKS

2.1 Part-based Models

In order to apply information retrieval techniques to image retrieval, high-dimensional feature vectors of local image patches are mapped into discrete visual words [1]. Each visual word represents a visual concept, which is analogue to a unit word in text documents. Based on visual words, the bag-of-words model (aka bag-of-features or bag-of-keypoints) which has been successfully adopted in text processing fields can be applied to computer vision problems [12]. This approach is one extreme of part-based models [8][9], where an image is characterized by its statistical distribution of visual words, but the spatial relationships between image patches are totally ignored. The low discrimination and ambiguity issue of visual words is still an open problem [10].

Comparing with the bag-of-words approach, another extreme is the constellation model, which models spatial locations and appearances of parts as a joint Gaussian distribution, but results in significant computation cost. Between these two extremes, several models have been developed. Following the discussion in [8], there are the star models [14], the k-fan models [14], the tree models [16], the hierarchical structures [17], and the sparse flexible model [8]. Most of these models describe appearances and spatial relationships by a generative approach, which requires parameter estimation and large amount of training data.

2.2 Pattern Discovery

Data mining techniques such as frequent itemset mining algorithms have been adopted to identify frequently co-occurred visual words. Visual words presented in a spatial neighborhood are viewed as a database transaction, and frequent visual word patterns are then found within the sampled neighborhoods. Each transaction or a found pattern is treated as orderless bag of visual words [22][24] or visual words with loosely defined spatial relationships [21][23]. Drawbacks of itemset-based mining approaches are the difficulty in defining transactions on an image, e.g. the positions and scales of neighborhood, and the lack of precise description of spatial relationships between local features.

Other than itemset-based mining approaches, Nowozin et al. [26] used a multiple-graph mining algorithm to capture frequent structures of visual words across images. Graph-based representation can better describe spatial relationships between local features. But in their implementation, the number of local feature in an input image is very limited, and the mined structures are assumed to appear in an image only once. Gao et al. [28] used a single-graph mining algorithm to extract structures of visual words that frequently occur in images.

In [19] and [20], boost classifiers were used to select the most discriminative visual word combinations. In [15], the authors used a Page-Rank like algorithm to achieve the same goal. Zhang and Chen [27] identify co-occurred visual words by training transformation matrices, but they didn't handle object scaling or rotation issues. In [25], an iterative learning procedure was proposed to automatically learn structured appearance models corresponding to a given annotation word.

2.3 Lattice Detection

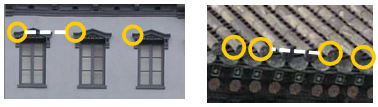
Works about lattice detection are related to pattern discovery. Liu et al. [3] developed a set of algorithms to find the underlying lattice of a given periodic texture and identify its symmetry group. More recent works in [4] and [5] were proposed to detect lattices of near-regular textures in real images. In [6], wallpaper patterns were extracted and used to match with building images in a 3D database, so that geo-tagging can be automatically achieved in urban environments. With consideration of perceptual grouping, Park et al. [7] advance related works to detect multiple, semantically relevant lattices in a scene simultaneously.

Lattice detection differs from our work from the following aspects. Firstly, it mainly focuses on detecting periodic structure in images, with little attempt to describe repetitive patterns for recognition purpose. Schindler et al. [6] did describe patterns and conduct pattern-based matching. However, from the second aspect, not all repetitive objects are placed in as a lattice. Objects of varied sizes would be located as a radial type or an irregular type. Thirdly, lack of mechanisms for eliminating noisy patterns or coping with sparsity of repetitive patterns prevents current lattice detection from retrieval and recognition applications.

Most of the literature mentioned above strives to discover how to describe spatial context between feautres. A comparative study of spatial context used for image anlaysis can be found in [29].

3. VISUAL PATTERN DISCOVERY

We define a visual pattern as a set of visual words with a specific spatial configuration that frequently occurs in an image. A visual pattern is represented by a graph $G = \langle V, E \rangle$. Each vertex $v_i \in V$ carries appearance features encoded as a particular visual word, and each edge $e_{ij} \in E$ encodes the pair-wise spatial relationship between v_i and v_j. An edge is established between v_i and v_j if they have consistent spatial relationship across its occurrences. Furthermore, a vertex in the graph should be spatially related to at least one other vertex, i.e. G should be a connected graph. To enhance discriminability of visual pattern, a pattern cannot have two vertices encoded as the same visual word, e.g. the corners of windows in Figure 1(a) correspond to the same visual word. Although this visual word appears frequently, it strides across two instances of the same element (across two windows) and thus cannot stand for a particular element. A similar situation can be seen on the roof tiles in Figure 1(b). In this article, we use the terms "pattern" and "visual pattern" interchangeably. An occurrence of a pattern is called an *instance*.



(a) (b)

Figure 1. An example of low discriminative feature configuration.

3.1 Feature Extraction and Categorization

An image I is represented by a set of SIFT [13] feature descriptors $I = \{p_i\}$. Each feature p_i is represented by a four-tuple: $p_i = [\boldsymbol{x}_i, s_i, \theta, \boldsymbol{f}_i]$, where the 2-D vector $\boldsymbol{x}_i$ is the xy-coordinate of p_i, s_i is the scale, $\theta_i \in [-\pi, +\pi]$ is the orientation, and $\boldsymbol{f}_i$ is the 128-D SIFT descriptor that encodes the appearance feature surrounding $\boldsymbol{x}_i$.

A pre-trained visual vocabulary, denoted by W, is used to identify each feature's corresponding visual word. To construct this visual vocabulary, local features extracted from training images are clustered using the K-means algorithm. Centroids of clusters form

a visual vocabulary $W = \{w_1, w_2, ..., w_{|W|}\}$, where $|W|$ denotes size of this visual vocabulary and it may depend on different applications. Each local feature p_i is quantized into its nearest visual word w_{a_i}, where a_i is the visual word index corresponding to p_i. That is,

$$a_i = \arg\min_{j=1,...,|W|} dist(\boldsymbol{f}_i, w_j). \quad (1)$$

The function $dist(.)$ calculates the Euclidean distance between the SIFT vector $\boldsymbol{f}_i$ and the visual word w_j. After this step, an image is translated into a bag-of-visual words representation $\{a_i\}$.

3.2 Visual Pattern Description

The spatial relationship between two local features p_i and p_j is characterized by a 4-D vector $r_{ij} = [D_{ij}, S_{ij}, H_{ij}, H_{ji}]$. We adopt the representation originally designed by [28], with slight modification on the information of relative scale. The value D_{ij} is the spatial distance between p_i and p_j, which is normalized by the corresponding scales to resist image scaling. The value S_{ij} is the relative scale. The value H_{ij} is the relative heading from p_i to p_j, i.e. the angle from $\boldsymbol{x}_j$ to $\boldsymbol{x}_i$ relative to θ_i, which makes it invariant to image rotation. Similarly, the value H_{ji} is the relative heading from p_j to p_i. An example of relative headings is illustrated in Figure 2. The symbol $\|.\|_2$ in Equation (2) denotes the Euclidean distance, and the function $\Delta(.)$ in Equations (4) and (5) denotes the principle value, which is in the range $[-\pi, \pi]$. This representation is invariant to translation, scale and rotation, and is robust to small distortion.

$$D_{ij} = \frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2}{\sqrt{s_i^2 + s_j^2}}, \quad (2)$$

$$S_{ij} = \frac{\min(s_i, s_j)}{\max(s_i, s_j)}, \quad (3)$$

$$H_{ij} = \Delta_\theta(\arctan(\boldsymbol{x}_i - \boldsymbol{x}_j) - \theta_i), \quad (4)$$

$$H_{ji} = \Delta_\theta(\arctan(\boldsymbol{x}_j - \boldsymbol{x}_i) - \theta_j). \quad (5)$$

Among all the characteristics in r_{ij}, we found that the relative headings H_{ij} and H_{ji} are the most distinctive ones. Therefore, we compare two relationships r_{ij} and $r_{i'j'}$ by their quantized heading values. Given a spatial relationship r_{ij}, its two heading values are quantized into a pair of indices by using the quantization function

$$QH(r_{ij}) = \left[\left\lfloor \frac{conv(H_{ij})}{2\pi/NUMBINS} \right\rfloor, \left\lfloor \frac{conv(H_{ji})}{2\pi/NUMBINS} \right\rfloor \right], \quad (6)$$

where the function $conv(.)$ converts a principle value ranged $[-\pi, +\pi]$ to $[0, 2\pi]$, and the constant $NUMBINS$ denotes the number of bins to quantize the interval $[0, 2\pi]$. The resulting index is from 0 to $NUMBINS - 1$. After heading values quantization, two relationships r_{ij} and $r_{i'j'}$ are considered consistent if $QH(r_{ij}) = QH(r_{i'j'})$.

To sum up, a visual pattern is represented as a graph $G = \langle V, E \rangle$. Each vertex $v_i \in V$ carries appearance feature encoded by a particular visual word, and is represented by a two-tuple $v_i = [i, a_i]$, where a_i is the visual word index encoding appearance of v_i. Each edge $e_{ij} \in E$ encodes the spatial relationship between v_i and v_j, and is represented by a three-tuple $e_{ij} = [i, j, QH(r_{ij})]$, where $QH(r_{ij})$ describes quantized spatial relationship between v_i and v_j.

Now we are able to detect instances of a pattern and compare two sets of local features. Given a set of local features $\{p_m\}$ in an image, $\{p_m\}$ is said to be an instance of G if there exists an bijective mapping (i.e. one-to-one and onto) from i to m, such that $\forall v_i \in V$, $a_i = a_m$ and $\forall e_{ij} \in E$, $QH(r_{ij}) = QH(r_{mn})$. Similarly, we can use this approach to compare two sets of local features. Given two sets of local features $\{p_i\}$ and $\{p_{i'}\}$ of the same size, which can be obtained from the same or different images, $\{p_i\}$ and $\{p_{i'}\}$ are said to be two instances of the same pattern if there exists an bijective mapping from i to i', such that $\forall p_i$, $a_i = a_{i'}$ and $\exists p_j : QH(r_{ij}) = QH(r_{i'j'})$. This is the foundation to unsupervisedly find patterns from instances.

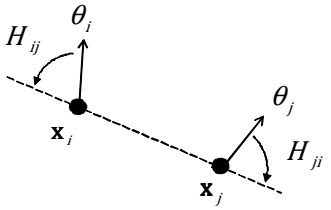


Figure 2. An illustrative example of relative headings. The value H_{ij} is counter-clockwise, and H_{ji} is clockwise.

3.3 Visual Pattern Discovery

Motivated by [26] and [28], a visual pattern can be treated as a connected subgraph embedded in a root graph. We first show how to construct a root graph for an image, and then show how to apply graph mining techniques to find visual patterns. Different from previous sections, where we use the term *instances* to denote the occurrences of a visual pattern in images, here we use the term *embeddings* to denote the occurrences of a subgraph in the root graph. Size of a graph is defined as the number of edges in it.

3.3.1 Graph Construction

Given the set of image features $I = \{p_m\}$, we build an undirected graph $\mathcal{G} = \langle V_r, E_r \rangle$ to represent these features and their spatial relationships, called the root graph of this image. The vertex $v_m \in V_r$ corresponds to the m th local feature p_m, and is represented by a 2-tuple $v_m = [m, a_m]$, where a_m is the vertex label equal to the visual word index for p_m. Each edge represents the spatial relationship between two features, and is represented by a three-tuple $e_{mn} = [m, n, b_{mn}]$, where b_{mn} is the edge label that uniquely identifies a possible value of $QH(.)$. If two edges e_{mn} and $e_{m'n'}$ have the same edge label, it means that the two relationships r_{mn} and $r_{m'n'}$ are consistent, i.e. $QH(r_{mn}) = QH(r_{m'n'}) \leftrightarrow b_{mn} = b_{m'n'}$.

The root graph is not necessary to be a connected graph, whereas a visual pattern is a connected graph. Given a root graph $\mathcal{G}$, any connected subgraph would potentially be a pattern. In other words, if an edge e_{mn} exists, then the endpoint features p_m and p_n would potentially belong to a pattern instance. Clearly, not all pairs of features would be a pattern instance. Therefore, it is not necessary to create edges between all pairs of features. To decrease complexity of the mining process, we use some criteria to create appropriate edges. First, any two vertices with the same vertex label cannot form an edge (see Figure 1). Second, we assume that the spatial scatter of a pattern would be in an appropriate range. Two features in a pattern should not be highly overlapped because they represent the same portion of an image. Third, we should consider a pattern's repeatability across different images. A pattern with features sampled in far apart scales has low repeatability across images. It's better that the features of a pattern

are sampled in nearby scales. Overall, we construct edges between any two vertices with different vertex labels, and the spatial relationship of its endpoints should fulfill the following equation.

$$E_r = \{e_{mn} | a_m \neq a_n, D_{mn} \in [T_{D_{min}}, T_{D_{max}}], S_{mn} > T_{S_{min}}\}. \quad (7)$$

The values $T_{D_{min}}$ and $T_{D_{max}}$ are the thresholds for D_{mn} (Equation (2)), and the value $T_{S_{min}}$ is the threshold for S_{mn} (Equation (3)).

3.3.2 Visual Pattern Discover Using Graph Mining

We adopt the VSIGRAM (Vertical Single-Graph Mining) algorithm [2] to find frequent subgraphs because of efficiency and and less memory requirement. There are mainly three components: subgraph frequency counting, graph isomorphism checking, and subgraph lattice exploration.

- Subgraph frequency counting

To determine frequency of a subgraph, we can count the maximum number of edge-disjoint (vertex-disjoint) embeddings [2]. Two embeddings are edge-disjoint (vertex-disjoint) if they do not share edges (vertices). In our work, we adopt the setting of vertex-disjoint embedding, i.e. the frequency of a subgraph is the maximum number of its vertex-disjoint embeddings in a graph.

To obtain the vertex-disjoint embeddings of a subgraph G_s, we first create *complement overlap graph* $\mathbb{G} = \langle \mathbb{V}, \mathbb{E} \rangle$, using all its non-identical embeddings. Each vertex of $\mathbb{G}$ corresponds to an embedding of G_s, and each edge of $\mathbb{G}$ corresponds to a pair of vertex-disjoint embeddings, i.e. an edge is established in $\mathbb{G}$ if its endpoint embeddings are vertex-disjoint. After that, the maximum clique in $\mathbb{G}$ is found by using the maximum clique algorithm. Number of vertices in the maximum clique means the frequency of the subgraph G_s.

- Graph isomorphism checking

To check isomorphism, we encode a subgraph as a string code, called the *canonical label*, which is a unique identifier invariant to the ordering of vertices. Given a subgraph, the canonical label is obtained by concatenating all its vertex labels and the upper-triangular entries of its adjacency matrix. In order to make this string invariant to vertex ordering, a naïve way is to try all possible permutations of vertices, produce a set of strings from all such permutations and its corresponding adjacency matrix, and then choose the lexicographically largest one as the canonical label for this subgraph [11].

Figure 3 shows an example of canonical labeling. Figure 3(c) is one adjacency matrix of Figure 3(a). The matrix header shows the vertex labels (in numerical), and the non-empty matrix entries are filled by edge labels (in alphabet). The string obtained from Figure 3(c) is “1 1 2 Ø b c”, where the substring ”1 1 2” is the concatenated vertex labels, and the substring “Ø b c” is obtained by concatenating the upper-triangular entries (except for the ones in the main diagonal) of the adjacency matrix, from top to down and left to right. The ‘Ø’ symbol corresponds to the empty entry indicating relationship between v_0 and v_1 , and it is lexicographically smaller than any possible edge label. Figure 3(b) shows another graph, which is isomorphic to Figure 3(a) but with different vertex ordering. To know whether Figure 3(a) and Figure 3(b) are isomorphic, we list all possible vertex ordering of these two graphs and the corresponding canonical labels, as shown in Figure 3(c)~(h) and Figure 3(i)~(n), respectively. The adjacency matrices that produce the lexicographically largest strings are shown in Figure 3(h) and Figure 3(l). Both produce “2 1 1 c b Ø”, and thus these two graphs are isomorphic.

This approach has time complexity $O(|V|!)$ for a graph containing $|V|$ vertices. To reduce the count of permutation, we adopt the partition-based canonical labeling approach [11]. Vertices are first partitioned into disjoint groups by their degrees and vertex labels, and vertex ordering is only permuted within each partition. Details of partition-based canonical labeling please refer to [11].

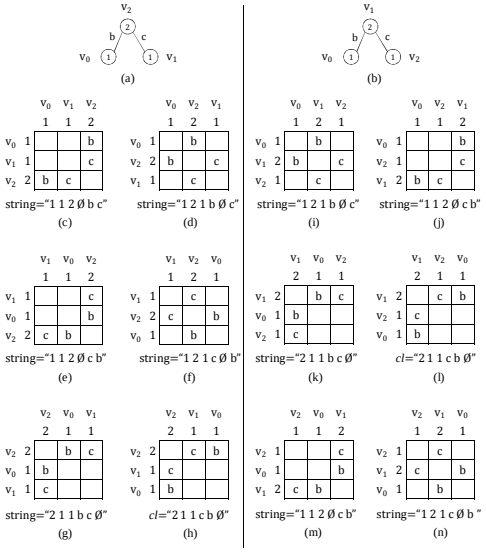


Figure 3. An example of naïve canonical labeling.

- Subgraph lattice exploration

The VISGRAM algorithm finds frequent subgraphs in a depth-first fashion. It starts from a size-1 (number of edge = 1) frequent subgraph (denoted by F^1), finds all the size-2 frequent subgraphs that are one-edge extension of the currently visited subgraph F^1, explores one of the size-2 frequent subgraphs, and goes deeper and deeper. The process backtracks when there is no frequent size-($k+1$) subgraph being produced from the currently visited size-k subgraph, or the limitation of size of subgraph is reached.

- The VSIGRAM algorithm

Given a root graph $\mathcal{G}$, the VSIGRAM algorithm [2] finds subgraphs with occurrence frequency larger than T_{freq}. We further set two constraints for the mining process: vertex-disjoint embeddings constraint and subgraph size constraint. To count the instance of a visual pattern more accurately, we constrain that two instances of a pattern cannot share the same local feature. In other words, two embeddings of a subgraph cannot share the same vertex. To enhance the discriminability of visual pattern, we constrain that the size (i.e. the number of edges) of a subgraph should be at least two. The maximal subgraph size can be determined by users. A larger-size visual pattern is more discriminative but less repeatable across images.

For graph construction, the complexity is $O(N^2)$ if N feature

points are extracted. However, with the visual label constraint and spatial constraint, the constructed graph is much smaller than an N-vertex complete graph. For subgraph frequency counting, finding the maximum clique is NP-hard. Fortunately, the constructed graph is sparse and small, and thus counting subgraphs is fast. For checking subgraph isomorphism, it is not known to be either in P or in NP-complete, but the method we adopt [2] provides an algorithm with heuristics to efficiently handle this problem. For subgraph lattice exploration, if there are M different vertex labels in a graph, the complexity of generating the subgraph lattice in the worst case is $O(M!)$, followed by the linear-time depth-first-search algorithm.

4. APPLICATIONS

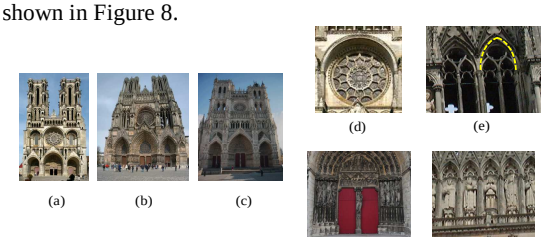
4.1 Architecture Image Classification

4.1.1 Architectural Style and Visual Patterns

An architectural style is a description for architecture of a specific geographical region, time period, or techniques. In this paper, we focus on four types of architecture styles: Gothic, Korean, Georgian, and Islamic architecture. Brief introduction of their characteristic features are listed below, in which some descriptions are from Wikipedia[1] and Buffalo Architecture[2].

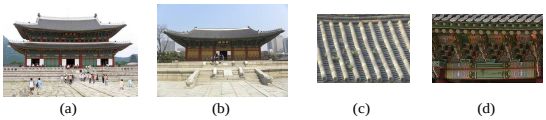
- Gothic architecture: Gothic architecture often has a decorated big round window in the centre of the facade, called the rose window (c.f. Figure 4).
- Korean architecture: Buildings of the Joseon dynasty are regarded as the representation of Korean architecture. Examples of the Korean architectural elements include roof tiles (c.f. Figure 5).
- Georgian architecture: The hung sash windows in the facade are the most distinguishable feature of Georgian architecture (c.f. Figure 6).
- Islamic architecture: We consider Islamic buildings, particularly mosques, with decorative patterns on the walls. These patterns are formally named Arabesque, which is an important element in Islamic art.

To show that visual patterns would correspond to distinctive architectural elements, some of the patterns found by our system are shown in Figure 8.



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

Figure 4. Examples of Gothic architectures: cathedrals and common architectural elements.



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

Figure 5. Examples of Korean architectures and architectural elements.

[1] http://en.wikipedia.org/
[2] http://buffaloah.com/

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

Figure 6. Examples of Georgian architectures.

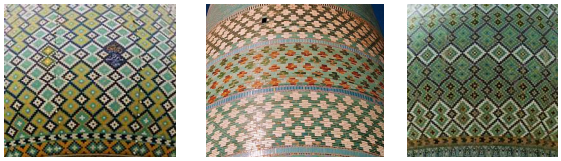


Figure 7. Examples of Islamic architectures.

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

Figure 8. Examples of visual patterns that correspond to architectural elements. They are the rose window (a), the roof tiles of Korean architecture (b), the hung sash windows of Georgian architecture (c), and the Arabesque of Islamic architecture (d).

4.1.2 Architecture Image Classification

We construct a classifier to conduct architecture image classification. This classifier is trained based on the co-occurrence statistics of visual patterns. For generality, we will use the term "class" or "image class" instead of "architectural style" in the following description. By using the pattern comparison method described in Section 3.2, we obtain the union of all pattern sets extracted from training images. In test images, a pattern is used for classification if it occurs in some training images, i.e. it is one of the training patterns.

To classify a test image into a class, we define f_j to be the random event that a training pattern occurs in this image, and $C = \{c_1, c_2, ..., c_K\}$ to be the discrete random variable of classes. $C = \{\text{"Gothic"}, \text{"Korean"}, \text{"Georgian"}, \text{"Islamic"}\}$ in our work. The occurrences of training patterns in this image is denoted by $\{f_j\}$. Assume that different pattern occurrences are conditionally independent given class c, we can use a Bayesian classifier to infer the probability that an image with pattern occurrences $\{f_j\}$ belongs to an image class c:

$$\psi(c) = \frac{p(c|\{f_j\})}{p(\bar{c}|\{f_j\})} = \frac{p(c)}{p(\bar{c})} \prod_{\{f_j\}} \frac{p(f_j|c)}{p(f_j|\bar{c})}. \quad (8)$$

The term $\frac{p(c)}{p(\bar{c})}$ is the prior probability ratio of image class presence c versus absence $\bar{c}$, which controls classification bias toward different classes. The term $\frac{p(f_j|c)}{p(f_j|\bar{c})}$ is the likelihood ratio of pattern occurrence f_j under class presence c versus absence $\bar{c}$, which reflects the distinctiveness of this pattern in class c. We assume that the prior probability ratios are the same for all classes. The likelihood ratios are estimated from training images, based on the co-occurrence statistics of pattern occurrences f_j with the class c. The conditional probability $p(f_j|c_k)$ is estimated by

$$p(f_j|c_k) = \frac{p(f_j, c_k)}{p(c_k)}. \quad (9)$$

To cope with data sparsity, we use a Dirichlet regularization parameter d to populate event counts:

$$p(f_j|c_k) \propto \frac{freq(f_j, c_k)}{freq(c_k)} + d. \quad (10)$$

The value $freq(f_j, c_k)$ is the number of class c_k training images having pattern f_j, and the value $freq(c_k)$ is the number of class c_k images in the training set. The value d is set as 0.01.

The classifier is constructed as the prior ratios and the likelihood ratios are estimated. With this classifier, given a test image with training pattern occurrences $\{f_j\}$, we can infer the probability of this image belonging to each class. The class c^* maximizing $\psi(c)$ is chosen as the most probable class for a test image.

$$c^* = \arg\max_c \psi(c). \quad (11)$$

4.2 Product Image Retrieval

Visual patterns can also be used to retrieve images containing objects with specific texture appearance. Given a query image with texture-like contents, our goal is to retrieve images that contain objects with texture appearance similar to the query image.

Some fashion houses have their own representative motifs featured on their products, which serve as the emblematic codes of its brand. Examples are the Monogram Canvas of Louis Vuitton, the double-G logo Gucci, the big-C logo of Coach, the double-F tab of Fendi, and the Anagram motif of Loewe (c.f. Figure 9). Searching products of a particular brand is equivalent to finding images containing the emblematic motifs of this brand.

To search products of a particular brand, we first discover visual patterns from a classical patch like one of the figures in the bottom row of Figure 9. Suppose taht $\{f_1^c, f_2^c, ..., f_N^c\}$ denotes the set of patterns extracted for the brand c, the likelihood of an image with visual patterns $\{g_1, g_2, ..., g_M\}$ having products of brand c is computed by

$$\frac{1}{M}\sum_{i=1}^{N}\sum_{j=1}^{M}\delta(f_i^c, g_j), \quad (12)$$

where $\delta(f_i^c, g_j) = 1$ if $f_i^c = g_j$, and $\delta(f_i^c, g_j) = 0$ otherwise. An image with the likelihood larger than a predefined threshold is claimed to have the corresponding products. The threshold is set loosely, because corresponding products would be occluded or have significant affine transformation in real cases. It is noted that an image may have several products of different brands and can be detected by this approach.

Figure 9. Products of five different brands and their emblematic motifs: (a) the Monogram Canvas of Louis Vuitton, (b) the double-G logo Gucci, (c) the big-C logo of Coach, (d) the double-F tab of Fendi, and (e) the Anagram motif of Loewe.

5. EXPERIMENTS

5.1 Performance of Pattern Discovery

We collect different types of images that contain repetitive objects, and evaluate quality of the discovered patterns by human judgement. An extracted pattern is considered a good pattern if all its instances correspond to the same type of object, e.g. instances on the butterflies in Figure 10. To extract visual patterns, the parameter $NUMBINS$ stated in Equation (6) is set as 8, and the thresholds $T_{D_{min}}$, $T_{D_{max}}$, and $T_{S_{min}}$ in Equation (7) are set as 2, 10, and 0.6, respectively. In pattern discovery, the minimal frequency threshold of a visual pattern is 4, and the size of a pattern (number of edges) is 2 or 3.

Figure 10 shows sample results extracted based on our approach and [28]. Overall, our approach is capable to find patterns under scaling, rotation, illumination changes, and partial occlusion, and we can easily identify meaningful parts of images. On the contrary, most of the patterns found by the approach in [28] have visually inconsistent instances, even for the computer-generated graphics. They assume edges are sortable, and expect that spatially consistent edges would be put together after edge sorting. However, their edge sorting criterion actually causes many inconsistent edges being put together. Gao et al. [28] claimed that the found associations are just "candidate associations", and should be merged to produce the "true associations". Unfortunately, they didn't clearly state how to merge these associations in [28]. Figure 11 shows sample results for an architecture image.

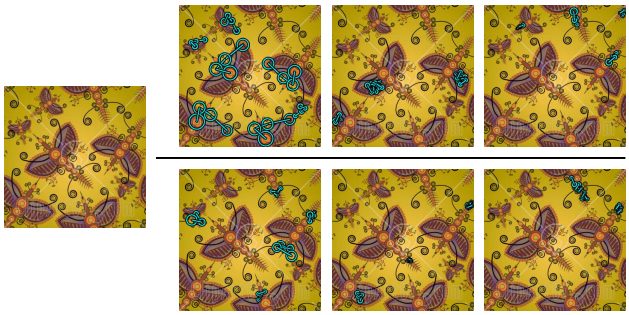


Figure 10. Sample results for computer generated graphics. Left: input images. Top row shows our results, and the bottom row shows results by [28].

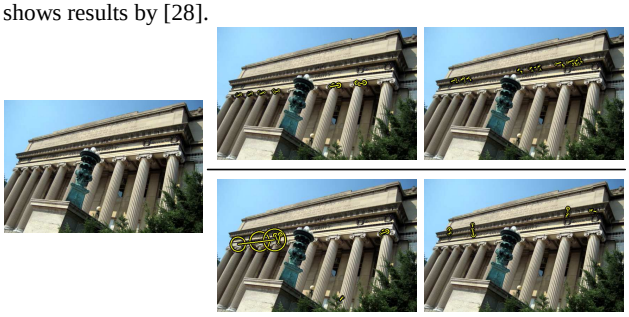


Figure 11. Sample results for an architecture image.

5.2 Performance of Architecture Image Classification

Because there is no appropriate benchmark for architecture image, we collect the evaluation dataset from the web. There are 111 Gothic images, 156 Korean images, 75 Georgian images, and 81 Islamic images in the evaluation dataset. Some of the Gothic images are from the Paris dataset[3]. The datasets for evaluation are available on our website[4]. The 10-fold cross validation scheme is used to evaluate the performance. For each fold, 30 images are randomly selected from each class as the training images, and the

[3] http://www.robots.ox.ac.uk/~vgg/data/parisbuildings/
[4] http://www.cs.ccu.edu.tw/~wtchu/projects/VP/index.html

remaining is for testing. We use a size-20 visual vocabulary to do feature categorization. Although the size of visual vocabulary seems small, we have to note that a visual pattern consisting of three vertices actually encode $20^3 = 8000$ possible visual word combinations. The parameters for pattern discovery are the same as that in Section 5.1.

To evaluate whether visual patterns are beneficial to classification, the standard bag-of-words (BoW) representation is used for comparison. Two classifiers based on k-nearest neighbor (kNN) and support vector machine (SVM) are used to classify the resulting vectors, respectively. For the kNN classifier, the Euclidean distance between a test vector and a training vector is calculated. For each test set, the classification accuracy is the average over all $(|c| * n + 1)$-nearest neighbor results, where $|c| = 4$ is the number of image class, and $n = \{0, 1, 2, 3, 4\}$. For the SVM classifier, we use the package provided by [18] for parameter setting and constructing a multi-class SVM classifier.

Figure 12 exhibits the classification result. Our method outperforms the BoW approach in three classes. We obtain worse performance for Gothic architectures because sometimes spatially consistent features cannot be found. Some of the Gothic images have nearly duplicate content, which makes the BoW approach work fine in classifying Gothic architectures. The SVM classifier works worse than the kNN classifier in two classes, which may be due to insufficiency of training data. The most prominent repetitive element in Korean architecture is the roof, and features on roofs vary largely in images captured in bottom-up angles. This may be the reason that performance for Korean architecture is generally the worst. However, by further considering spatial configurations of feature points, our method more accurately captures the characteristics of Korean architecture, and takes the largest performance lead over other three classes. The average classification accuracy for our approach, the BoW approach with the kNN classifier, and the BoW approach with the SVM classifier are 0.81, 0.74, and 0.73, respectively.

Figure 13 presents sample classification results. Failure classification may be caused by scale of object (no local feature can be extracted from small-scale building like Figure 13(e)), or pattern statistics in training set (in Figure 13(f), some of the found patterns have high likelihood values in the Islamic class). Image contents may also cause failure classification. There is a building with root tiles shown in bottom-left of Figure 13(g), and this image is erroneously classified as Korean architecture. On the other hand, deterministic patterns like root tiles cannot be accurately extracted from the extreme viewpoint in Figure 13(h).

We also evaluate performance under different number of training images. The average number of patterns found in the training set of 10, 20, 30, and 40 images are 53035, 99161, 139737, and 179240, and the average classification accuracy under these four settings are 0.75, 0.81, 0.83, and 0.82, respectively. Based on sufficient number of patterns, our classification approach performs well.

5.3 Performance of Product Image Retrieval

To evaluate product image retrieval, we collect 343 images from the web. The evaluation dataset includes 86 positive images, in which 37 images contain products of Louis Vuitton (LV), 26 images contain products of Gucci, and 34 images contain products of Coach. The other 257 images are junk images that do not have any product of these three brands. All of the positive images are products presented in very cluttered scenes, and some of them are collected from the Flickr group "What's in your bag"[5]. Three classical motifs of these three brands are used as query images.

A visual word dictionary of 50 visual words is used to discover visual patterns, and the parameter settings are identical to the previous section. We show retrieval performance under various pattern sizes and minimal frequency thresholds. Given a set of visual patterns extracted from a test image, only the patterns with occurrence frequency larger than or equal to T_{freq} are used to perform the retrieval task. Table 1 presents retrieval result based on size-2 patterns. It is shown that when increasing the minimal frequency threshold, we can eliminate more noise patterns, but the true patterns corresponding to brand motifs may be filtered out, i.e. precision increases and recall decreases. It is hard to extract patterns from the Coach's big-C logo. For Coach, we cannot find visual patterns with occurrence frequency greater than 6. Table 2 presents retrieval performance on size-3 patterns. It is hard to find size-3 patterns with occurrence frequency greater than 3, and therefore we only show the result of $T_{freq} = 3$. The precision on size-3 patterns is much higher than that on size-2 patterns, because size-3 patterns provide more discriminative descriptions.

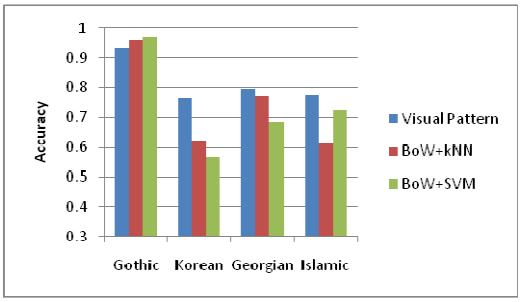
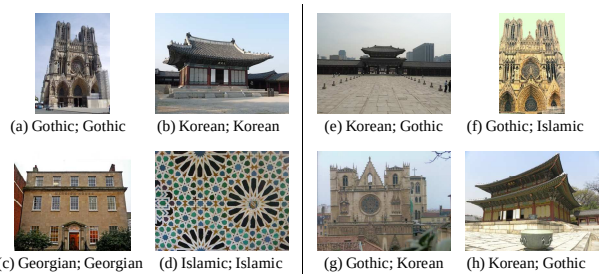


Figure 12. Performance comparison of architecture image classification.



(a) Gothic; Gothic (b) Korean; Korean (e) Korean; Gothic (f) Gothic; Islamic

(c) Georgian; Georgian (d) Islamic; Islamic (g) Gothic; Korean (h) Korean; Gothic

Figure 13. Sample classification results. The caption "Korean; Gothic" means that it is truly a Korean architecture, and is misclassified as the Gothic.

Table 1. Performance of image retrieval with size-2 patterns.

	T_{freq}	3	4	5	6	7	8
Louis Vuitton	Prec.	0.40	0.53	0.60	0.67	0.75	0.71
	Recall	0.32	0.22	0.16	0.16	0.16	0.14
Gucci	Prec.	0.25	0.37	0.41	0.5	0.62	0.71
	Recall	0.69	0.53	0.46	0.42	0.38	0.38
Coach	Prec.	0.34	0.55	1	1	N/A	N/A
	Recall	0.32	0.18	0.06	0.03	N/A	N/A

[5] http://www.flickr.com/groups/whats_in_your_bag/

Table 2. Performance of image retrieval with size-3 patterns.

	Louis Vuitton	Gucci	Coach
Precision	1.00	1.00	1.00
Recall	0.17	0.31	0.06

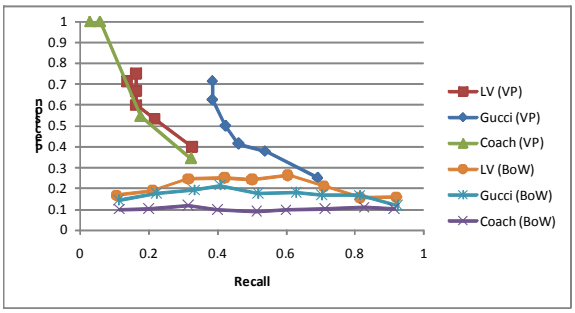


Figure 14. Performance comparison of image retrieval based on the BoW approach and based on the size-2 visual patterns.

Figure 14 shows the performance based on the BoW approach and size-2 visual patterns (VP). The BoW approach characterizes global statistics of visual words, and cannot resist background clutter or occlusion. In contrast, our method well distinguishes texture elements from background clutter and achieves high precision and reasonable good recall values.

6. CONCLUSION

We have presented an approach to automatically detect and localize frequent spatial feature configurations, which can successfully describe characteristic features of repetitive objects. Relationships between local features are transformed into a root graph, and visual patterns as subgraphs embedded in the root graph are then found through the graph mining process. Evaluation results of two applications show that our approach is capable to find patterns under object scaling, rotation, illumination changes, and partial occlusion.

Acknowledgement: The work was partially supported by the National Science Council of Taiwan, Republic of China under research contract NSC 100-2221-E-194-061.

7. REFERENCES

[1] J. Sivic and A. Zisserman, "Video Google: A Text Retrieval Approach to Object Matching in Videos," Proc. of ICCV, pp. 1470-1477, 2003.

[2] M. Kuramochi and G. Karypis, "Finding Frequent Patterns in a Large Sparse Graph," Data Mining and Knowledge Discovery, vol. 11, no. 3, pp. 243-271, 2005.

[3] Y. Liu, R. T. Collins, and Y. Tsin, "A Computational Model for Periodic Pattern Perception Based on Frieze and Wallpaper Groups," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 26, no. 3, pp. 354-371, 2004.

[4] J. Hays, M. Leordeanu, A. A. Efros, and Y. Liu, "Discovering Texture Regularity as a Higher-Order Correspondence Problem," Proc. of ECCV, Part II, LNCS 3952, pp. 522-535, 2006.

[5] M. Park, K. Brocklehurst, R. T. Collins, and Y. Liu, "Deformed Lattice Detection in Real-World Images Using Mean-Shift Belief Propagation," IEEE Trans. on PAMI, vol. 31, no. 10, pp. 1804-1816, 2009.

[6] G. Schindler, P. Krishnamurthy, R. Lublinerman, Y. Liu, and F. Dellaert, "Detecting and Matching Repeated Patterns for Automatic Geo-tagging in Urban Environments," Proc. of CVPR, 2008.

[7] M. Park, K. Brocklehurst, R.T. Collins, and Y. Liu, "Translation-Symmetry-Based Perceptual Grouping with Applications to Urban Scenes," Proc. of ACCV, 2010.

[8] G. Carneiro and D. Lowe, "Sparse Flexible Models of Local Features," Proc. of ECCV 2006, Part III, LNCS 3953, pp. 29-43, 2006.

[9] D.-Q. Zhang, "Statistical Part-Based Models: Theory and Applications in Image Similarity, Object Detection and Region Labeling," PhD Thesis, Columbia University, 2005.

[10] J.C. van Gemert, C.J. Veenman, A.W.M. Smeulders, and J.-M. Geusebroek, "Visual Word Ambiguity," IEEE Trans. on PAMI, vol. 32, no. 7, 2010, pp. 1271-1283.

[11] M. Kuramochi and G. Karypis, "An Efficient Algorithm for Discovering Frequent Subgraphs," Technical Report, Department of Computer Science, University of Minnesota, 2002.

[12] J. Zhang, M. Marszałek, S. Lazebnik, and C. Schmid, "Local Features and Kernels for Classification of Texture and Object Categories: A Comprehensive Study," IJCV, vol. 73, no. 2, pp. 213-238, 2007.

[13] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," IJCV, vol. 60, no. 2, pp. 91-110, 2004.

[14] D. Crandall, P. Felzenszwalb, and D. Huttenlocher, "Spatial Priors for Part-based Recognition using Statistical Models," Proc. of CVPR, pp. 10-17, 2005.

[15] S. Zhang, Q. Tian, G. Hua, Q. Huang, and S. Li, "Descriptive Visual Words and Visual Phrases for Image Applications," Proc. of ACM MM, pp. 75-84, 2009.

[16] P. Felzenszwalb and D. Huttenlocher, "Pictorial structures for object recognition," IJCV, vol. 61, no. 1, pp. 55-79, 2005.

[17] G. Bouchard and B. Triggs, "Hierarchical Part-Based Visual Object Categorization," Proc. of CVPR, pp. 710-715, 2005.

[18] C.-C. Chang and C.-J. Lin, "LIBSVM : a library for support vector machines," 2001. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[19] D. Liu, G. Hua, P. Viola, and T. Chen, "Integrated Feature Selection and Higher-order Spatial Feature Extraction for Object Categorization," Proc. of CVPR, 2008.

[20] X. Liu, Z. Shi, Z. Li, and Z. Shi, "CoBoost Learning of Visual Categories with 1st and 2nd Order Features from Google Images," Proc. of ACM MM, pp. 533-536, 2009.

[21] T. Quack, V. Ferrari, and L. V. Gool, "Video Mining with Frequent Itemset Configurations," Proc. of ACM CIVR, LNCS 4071, pp. 360-369, 2006.

[22] J. Yuan, Y. Wu, and M. Yang, "Discovery of Collocation Patterns: from Visual Words to Visual Phrases," Proc. of CVPR, 2007.

[23] T. Quack, V. Ferrari, B. Leibe, and L. V. Gool, "Efficient Mining of Frequent and Distinctive Feature Configurations," Proc. of ICCV, 2007.

[24] Y.-T. Zheng, M. Zhao, S.-Y. Neo, T.-S. Chua, and Q. Tian, "Visual Synset: Towards a Higher-level Visual Representation," Proc. of CVPR, 2008.

[25] M. Jamieson, A. Fazly, S. Stevenson, S. Dickinson, and S. Wachsmuth, "Using Language to Learn Structured Appearance Models for Image Annotation," IEEE Trans. on PAMI, vol. 32, no. 1, pp. 148-164, 2010.

[26] S. Nowozin, K. Tsuda, T. Uno, T. Kudo, and G. BakIr, "Weighted Substructure Mining for Image Analysis," Proc. of CVPR, 2007.

[27] Y. Zhang and T. Chen, "Efficient Kernels for Identifying Unbounded-Order Spatial Features," Proc. of CVPR, pp. 1762-1769, 2009.

[28] J. Gao, Y. Hu, J. Liu, and R. Yang, "Unsupervised Learning of High-order Structural Semantics from Images," Proc. of ICCV, pp. 2122-2129, 2009.

[29] G.Th. Papadopoulos, C. Saathoff, H.J. Escalante, V. Mezaris, I. Kompatsiaris, M.G. Strintzis, "A comparative study of object-level spatial context techniques for semantic image analysis," CVIU, vol. 115, no. 9, pp. 1288-1307, 2011.