The Journal of Supercomputing (2020) 76:2597–2621
https://doi.org/10.1007/s11227-019-03041-y

Mining image frequent patterns based on a frequent pattern list in image databases

Ye-In Chang[1] · Jun-Hong Shen[2,3] · Chia-En Li[1] · Zih-Siang Chen[1] · Ming-Hsuan Tu[1]

Published online: 19 October 2019

Abstract

The goal of image mining is to find the useful information hidden in image databases. The 9DSPA-Miner approach uses the Apriori strategy to mine the image database, where each image is represented by the 9D-SPA representation. It presents a reasoning method to reason the unknown spatial relation that satisfies the spatial consistency. However, it may generate invalid candidates with the impossible relations that cannot be found in the 2D space or in the input database. Moreover, in this approach, counting the support of the pattern needs to intersect the associated image sets by searching the index structure, taking a long time. Therefore, in this paper, we propose an approach with a frequent pattern list, which generates all valid candidates of frequent patterns. Based on the frequent pattern list, the proposed approach presents two conditions in the candidate generation for finding frequent spatial patterns to avoid generating impossible candidates. Moreover, the proposed approach uses an additional verification step to further avoid generating impossible spatial relations. Therefore, the proposed approach generates fewer candidates than the 9DSPA-Miner approach, reducing the processing time. The experimental results have verified that the proposed approach outperforms the 9DSPA-Miner approach.

Keywords Frequent image patterns · Image databases · Image mining

✉ Jun-Hong Shen
shenjh@asia.edu.tw

[1] Department of Computer Science and Engineering, National Sun Yat-sen University, Kaohsiung 80424, Taiwan

[2] Department of Information Communication, Asia University, Taichung 41354, Taiwan

[3] Department of Medical Research, China Medical University Hospital, China Medical University, Taichung 40447, Taiwan

1 Introduction

Spatial data mining [1, 2] is the process of discovering interesting and previously unknown, but potentially useful patterns from large spatial databases, which must consider spatial data types and spatial relationships. The major accomplishments [3–8] only discover the frequent object classes occur together closely from spatial databases. Image mining [9–12] is an extension of the spatial data mining to image domain. Image mining deals with the extraction of implicit knowledge, image data relationship or other patterns not explicitly stored in the images [10]. Recently, there are several researches [13–17] which focus on mining spatial association rules based on the high-level spatial relations (such as the pairwise orientation or topology) among objects from the image databases. The features used in content-based image retrieval (CBIR) can be roughly divided into two categories, the low-level visual features [18–23] (such as color, texture and shape) and the high-level features [24–27] (such as pairwise spatial relations between objects).

In [16], Saritha and Santhosh proposed an intelligent medical image database system to mine the spatial patterns existing in medical images. Their work identified the spatial relationships existing between the anatomical structures in MRI axial scans of the brain, where a few anatomical structures like the caudate nucleus, thalamus, lateral ventricle and putamen are recognized as the prominent spatial objects. The spatial relations existing between the anatomical structures in normal and pathological situations are different so that the results of the spatial association mining in medical images can be used as diagnosis rules for doctors.

Finding frequent spatial patterns in an image database is fundamental to mining spatial association rules. The association rules can be easily derived from the frequent spatial patterns. Therefore, in this paper, we focus on mining frequent spatial patterns in the image database, where objects are located in the 2D space in images. Moreover, an image contains at least two objects so that spatial relations between any two objects exist. Figure 1 shows an example of the image database containing four images I_1–I_4. Setting the minimum support to 1/2, we can discover implicit spatial relations of objects from the image database. If a spatial pattern exists at least in two ($=1/2 * 4$) images, it is a frequent spatial pattern. For the concern of mining spatial relationships, two frequent spatial patterns considering both objects and their spatial relations are found out, as shown in Fig. 2.

Mining spatial association rules in image databases is first introduced by Hsu et al. [13] through the viewpoint mining algorithm. In their work, the distance and orientation radians among objects in an image are encoded as the representation for each image in the database, and the concept of the Apriori algorithm [28] is adopted. Since human is more sensitive with orientation than distance, to mine only spatial orientation, Wei and Shan propose the pattern growth algorithm which employs the 2D strings [9] to represent each image in the database. They use the concept of the PrefixSpan algorithm [17, 23] to efficiently mine the spatial co-orientation patterns from an image database.

In other way, Lee et al. [14] propose the 9DLT-Miner approach to mine the image database, where each image is represented by the 9DLT representation

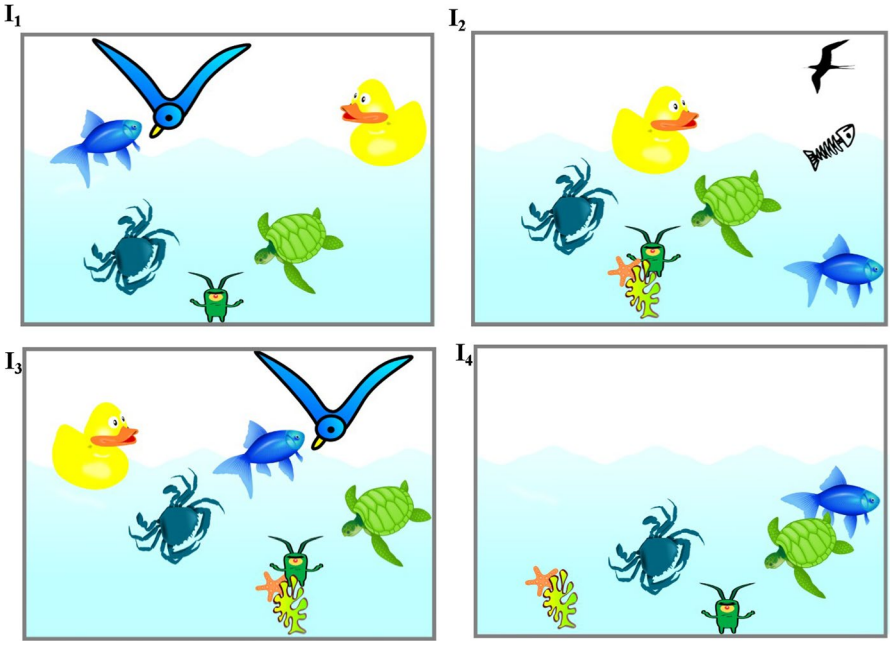

Fig. 1 An example of the image database

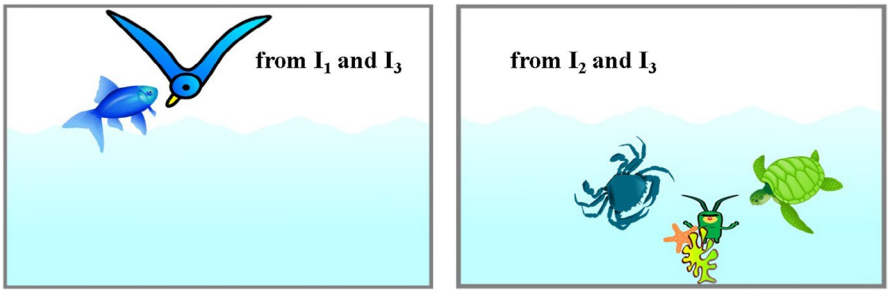

Fig. 2 The result of frequent patterns considering both objects and their spatial relations

[24]. In their work, an Apriori-like algorithm is used and the characteristic of 9DLT representation is exploited to prune most of invalid candidates. However, only nine directions can be handled and the mining results are still too detail to describe human visual perception. Lee et al. [15] propose the 9DSPA-Miner approach, which employs the 9D-SPA representation [27] to more correctly discriminate the images than the previous approaches. They adopt the Apriori strategy to discover the frequent patterns from the image database. Since there exists an unknown spatial relation in the generated candidate when performing the join step, they propose a reasoning method to reason the unknown relation. With the usage of the reasoning method, they can reason the unknown relation, which

satisfies the spatial consistency with known k relations ($k \geq 2$). Moreover, they scan the index structure to count the support of the discovered patterns. In this way, they will intersect the image sets of the spatial relations found in the index structure to count the support.

Although the 9DSPA-Miner can reason the unknown relation, it may also generate the impossible relations, which cannot be found in the 2D space or in the input database. That is, they may generate a large number of invalid candidates. Moreover, when counting the support of the pattern, they intersect the image sets of spatial relations in the pattern. In this way, they will recompute the intersections of image sets, taking too much time.

Therefore, to solve those problems and improve the performance of the 9DSPA-Miner, in this paper, we propose an approach based on the frequent pattern list to find all frequent spatial patterns in an image database. The proposed approach considers that spatial objects in a 2D image are well recognized via their corresponding minimum bounding rectangles (MBRs). The proposed approach includes two major conditions to avoid generating candidates that are impossible to be frequent whatever the unknown relation is. In the first condition, it is not necessary to generate a candidate of size $(k+1)$ under the case of missing any its frequent subset of size k. In the second condition, it is not necessary to generate a candidate of size $(k+1)$ under the case of the number of the intersection of instances of any two joinable subsets of size k less than the minimum support. If either one is satisfied, we then will not generate any candidate. Moreover, if both two conditions are not satisfied, we will use the frequent relations of size-2 frequent patterns to discover the unknown relation and use a verification step to avoid generating the impossible spatial relations. Therefore, in our approach, every generated candidate will be valid. Moreover, each discovered pattern has an associated image set. The image set stores the image identifiers where each image contains the discovered patterns. In this way, we do not need to scan the index structure to count the support of the patterns. Hence, we will not recompute the intersection of the image sets.

We summarize the contributions of this work as follows.

1. The proposed approach creates the frequent pattern list to quickly retrieve the frequent 2-patterns, which can prune most of impossible candidates while finding all frequent spatial patterns in an image database.
2. The proposed approach presents two conditions in the candidate generation for finding frequent spatial patterns to avoid generating impossible candidates.
3. The simulation results have verified that the proposed approach is more efficient than the 9DSPA-Miner.

The rest of the paper is organized as follows. In Sect. 2, we describe the related researches for mining spatial association rules from the image databases. In Sect. 3, we present our proposed approach. In Sect. 4, we compare the performance of the proposed approach with that of the 9DSPA-Miner [15]. Finally, in Sect. 5, we summarize conclusions.

2 Related work

In this section, we describe two related researches for finding spatial association rules among objects from the image databases, including the 9DLT-Miner [14] and the 9DSPA-Miner [15].

2.1 The 9DLT miner approach

With an image representing as 2D strings [9], the pattern growth algorithm was proposed [17]. However, the 2D strings representation cannot indicate the directional relation between any two objects directly. Directional information is one of the most important types of information in an image database, and the 9DLT representation [24] is fundamental in this method. Lee et al. [14] proposed the 9DLT-Miner approach, which employs the 9DLT representation to mine an image database where each image is represented by the 9DLT representation. In the 9DLT representation, each object in an original image is represented as an icon whose center is used as the reference point. The directional codes of the 9DLT representation are shown in Fig. 3a, where R denotes the centroid of the referenced object. For a symbolic image as shown in Fig. 3b, the corresponding 9DLT matrix is shown in Fig. 3c. The lower-triangular codes indicate the spatial relations among the objects. For example, the spatial relation between A and B is 5. That is, B is to the south of A. The symbolic image can be expressed by a 9DLT string (A, B, C, D, 5, 6, 6, 7, 6, 6).

The 9DLT-Miner approach is similar to the Apriori algorithm [28]. Two frequent size-k patterns are joinable if the first $(k-1)$ objects and the corresponding relations between them are identical in both size-k patterns. The advantage of the 9DLT-Miner is that it uses the characteristics of the 9DLT representation to prune most of impossible candidates when generating a candidate size-$(k+1)$ pattern by joining two joinable frequent size-k patterns ($k \geq 2$). For example, when joining two size-2 frequent patterns (A, B, 3) and (A, C, 4) to generate a candidate (A, B, C, 3, 4, 2), there is an unknown relation 2 between B and C. Without exploiting the characteristic of the 9DLT representation, it has to examine all the nine directions. However, by considering the relations which exist in (A, B, 3) and (A, C, 4), the possible relations between B and C are only (A, B, C, 3, 4, $\underline{4}$), (A, B, C, 3, 4, $\underline{5}$) and (A, B, C, 3, 4, $\underline{6}$). Therefore, the 9DLT-Miner approach can reduce the large amount of candidates.

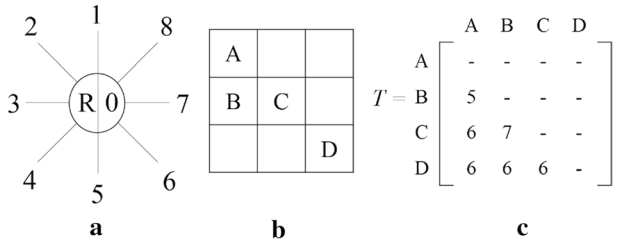

Fig. 3 The 9DLT representation and an example: **a** the directional codes of 9DLT, **b** a symbolic image and **c** a 9DLT matrix for the symbolic image in (**b**)

2.2 The 9D-SPA miner approach

In the 9DLT-Miner approach [14], the 9DLT representation [4] is still too detailed to describe the human visual perception. Moreover, only nine directions can be handled between any two objects. Huang et al. [27] proposed the 9D-SPA representation, which uses the directional and topological information to capture the spatial relations among objects in an image. Representing an image with the 9D-SPA representation can provide much better approximation for spatial relations between objects than that with 2D strings [9] or 9DLT representation [24]. Based on the 9D-SPA representation, Lee et al. [15] proposed the 9DSPA-Miner approach to mine the spatial association rules from an image database where each image is represented by the 9D-SPA representation. The 9DSPA-Miner approach exploits the characteristics of the 9D-SPA representation to prune most of impossible candidates. Moreover, it applies the anti-monotone strategy used in the Apriori approach [28] to prune the candidates whose sub-patterns are not frequent. Moreover, the approach modifies the index structure proposed by Huang and Lee [27] to more efficiently determine the support count of a candidate.

3 The proposed approach

Our proposed approach proceeds in two phases. First, we scan the image database to create the frequent pattern list and determine the frequent 2-patterns. Next, we join two frequent k-patterns to generate candidate $(k+1)$ patterns and determine the support of each candidate. The second phase is repeated until no more frequent patterns found. In the proposed approach, we define two conditions to avoid generating impossible candidates during the second phase. Moreover, we use the frequent relations to discover the unknown spatial relation in the generated candidate. We then exploit the result of the reasoning method [15] to prune the frequent relations that do not satisfy the spatial consistency. Furthermore, each discovered pattern is associated with an image set. With this usage, we do not need to scan the index structure to count the support and can efficiently count the support of the candidate.

In this section, we use an example image database *IDB*1 to explain our approach. Figure 4 is the image database *IDB*1 containing eight images, where there are five objects extracted and approximated by their MBRs. We consider only the image that contains at least two objects such that they could form a spatial relation.

3.1 The 9D-SPA representation

To facilitate the image retrieval, the proposed approach expresses each image in the image database as the 9D-SPA representation [27]. Suppose an image I contains n objects $(O_1, O_2, \ldots, O_n)$. Then, the 9D-SPA representation of I can be encoded as a set of 4-tuples as follows.

$$I = \{(O_{ij}, D_{ij}, D_{ji}, T_{ij}) \,|\forall O_i, O_j \in I,\ 1 \le i < j \le n\}.$$

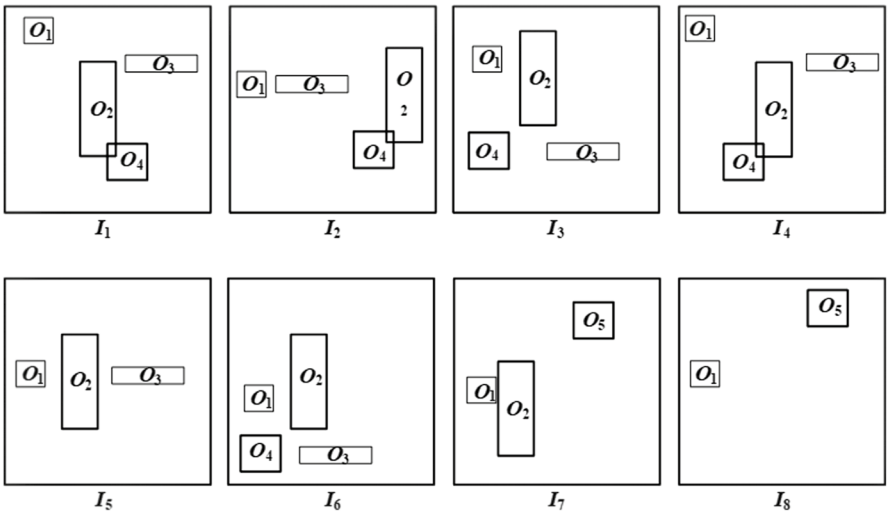

Fig. 4 An example database *IDB*1 containing eight images and five objects

O_{ij} is the code for the object-pair (O_i, O_j), where O_i is the ith object in the image database and i is assigned to object O_i as its object number [15]. Given two objects O_i and O_j, O_{ij} can then be easily computed by Eq. (1).

$$O_{ij} = \left[(j-1)\times(j-2)\right]/2 + i. \tag{1}$$

To obtain two numbers i and j from O_{ij} (or to decode O_{ij}), we use Eq. (2)

$$i = O_{ij} - (a\times(a+1)/2), \tag{2}$$

where a is the largest integer such that $a\times(a+1)/2 < O_{ij}$ and $j = a + 2$ [27].

D_{ij} is the code for the directional relation between objects O_i and O_j with O_j as the reference object [26]. Each area w_k cut by the four boundaries of the MBR of object O_j horizontally and vertically, $0 \le k \le 8$, is assigned with a binary code of 8 bits. For example, for area 6, the binary code is $(00100000)_2 = 32$; that is, the 6th bit from the rightmost is turned out. The value of D_{ij} is obtained by Eq. (3).

$$D_{ij} = \sum_{k=1}^{8} b_k w_k, \tag{3}$$

where $b_k = 1$ if object O_i overlaps area w_k; otherwise, $b_k = 0$. D_{ji} is the code for the directional relation between objects O_i and O_j with O_i as the reference object [26].

T_{ij} is the code for the topological relation between O_i and O_j. All the possible topological relations between any two objects are shown in Fig. 5. In Fig. 5e, f, although the topological code between "contain" or "inside" is same, the directional code between them is different. Figure 5c–f shows the cases for dealing with the overlapping spatial objects. With these topological codes, the

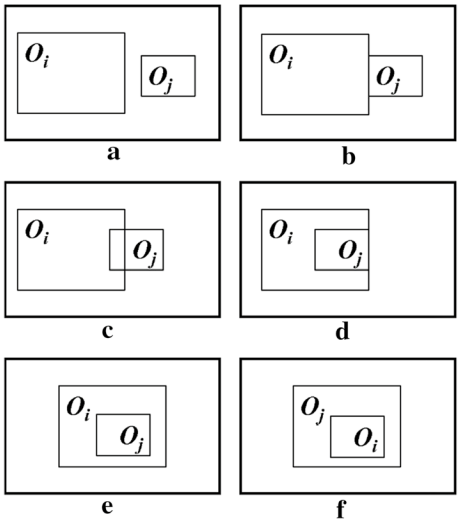

Fig. 5 All the topological relations between any two objects: **a** disjoint ($T_{ij}=0$); **b** meet ($T_{ij}=1$); **c** partly overlap ($T_{ij}=2$); **d** cover ($T_{ij}=3$); **e** contain ($T_{ij}=4$; $D_{ij}=255$); **f** inside ($T_{ij}=4$; $D_{ij}=0$)

overlapping data issue of spatial objects in images can be efficiently recognized and processed in the proposed approach.

We use the image containing O_1 and O_2 shown in Fig. 6, retrieved from I_1 in Fig. 4, to illustrate how to organize it with the 9D-SPA representation. First, we have $O_{12} = [(2-1)\times(2-2)]/2+1 = 1$. The nine rectangular areas based on the MBR of O_2 are shown in Fig. 6b, and O_1 overlaps area 4. Thus, D_{12} can be calculated as $(2^{4-1})=8$. Next, we cut the whole image into nine rectangular areas based on the MBR of O_1, as shown in Fig. 6c, and O_2 overlaps area 8. Thus, we have $D_{21}=(2^{8-1})=128$. Finally, since the topological relation of O_1 and O_2 is disjoint, its T_{12} is 0. As a result, the 9D-SPA representation of the image in Fig. 6a is {(1, 8, 128, 0)}. Table 1 lists the corresponding 9D-SPA representations for the images in database *IDB*1 shown in Fig. 4.

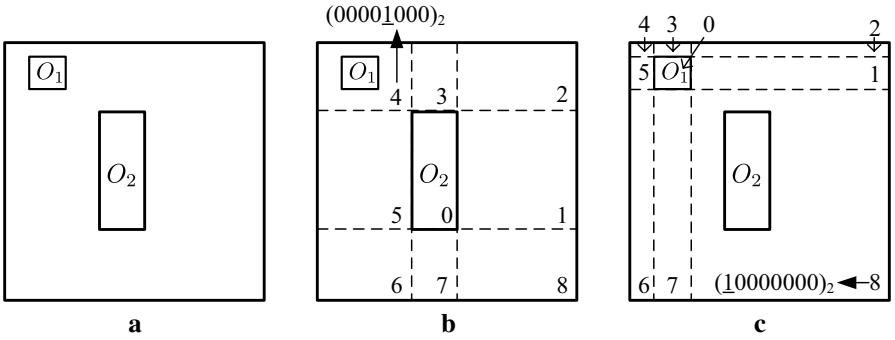

Fig. 6 An image represented with the 9D-SPA representation: **a** O_1 and O_2 in I_1; **b** O_2 as the reference object; **c** O_1 as the reference object

Table 1 The 9D-SPA representation of database *IDB*1

Image	The 9D-SPA representation
I_1	{(1, 8, 128, 0), (2, 8, 128, 0), (3, 48, 3, 0), (4, 8, 128, 0), (5, 28, 193, 2), (6, 6, 96, 0)}
I_2	{(1, 16, 131, 0), (2, 56, 1, 0), (3, 131, 16, 0), (4, 8, 128, 0), (5, 7, 112, 2), (6, 8, 128, 0)}
I_3	{(1, 16, 131, 0), (2, 8, 128, 0), (3, 12, 192, 0), (4, 4, 224, 0), (5, 2, 32, 0), (6, 1, 56, 0)}
I_4	{(1, 8, 128, 0), (2, 8, 128, 0), (3, 48, 3, 0), (4, 8, 128, 0), (5, 7, 112, 2), (6, 2, 32, 0)}
I_5	{(1, 16, 131, 0), (2, 56, 1, 0), (3, 56, 1, 0)}
I_6	{(1, 16, 131, 0), (2, 8, 128, 0), (3, 12, 192, 0), (4, 4, 224, 0), (5, 2, 32, 0), (6, 1, 56, 0)}
I_7	{(1, 16, 131, 1), (7, 32, 2, 0), (8, 32, 2, 0)}
I_8	{(7, 32, 2, 0)}

3.2 Problem definitions

Mining spatial association rules for an image database contains two phases, fining all frequent spatial patterns and generating spatial association rules from these frequent patterns. After the first phase is processed, the corresponding spatial association rules can be easily derived from the frequent patterns [15]. Therefore, finding the spatial frequent patterns dominates the overall performance efficiency. Therefore, the goal of this paper is to efficiently find all the frequent spatial patterns.

In an image, a spatial relation (*s*-relation) r_{ij} between two objects O_i and O_j in the 2D is expressed as a 4-tuple of the 9D-SPA representation (O_{ij}, D_{ij}, D_{ji}, T_{ij}). A spatial *k*-pattern X is denoted by $X=\{O_{p(1)}, O_{p(2)},\ldots, O_{p(k)}, r_{p(1)p(2)}, r_{p(1)p(3)},\ldots, r_{p(k-1)p(k)}, X.imgs\}$, where k is the number of objects in pattern X, $k\geq 2$, and $p(a)$ is the object number for object O, $1\leq a\leq k$. The spatial *k*-pattern X records all *s*-relations, $r_{p(b)\,p(c)}$, of the combinations of any two objects $O_{p(b)}$ and $O_{p(c)}$, $p(b)<p(c)$, $1\leq b\leq(k-1)$, $1\leq c\leq k$. $X.imgs$ is an image set containing the images that have all the corresponding *s*-relations.

The support of an *s*-relation r_{ij}, $sup(r_{ij})$, is the fraction of images in the image database containing r_{ij}. If $sup(r_{ij})$ is not less than the user-specified minimum support, *min_sup*, this s-relation is a frequent *s*-relation. The support of a spatial *k*-pattern X, $sup(X)$, is the fraction of images in the image database containing this pattern [15]. If $sup(X)$ is not less than *min_sup*, this pattern is a frequent spatial pattern.

3.3 Data structures

In the proposed approach, two data structures are used, the two-level index structure [15] and the frequent pattern (FP) list. The two-level index structure [15] is used to find the frequent 2-patterns. We add every *s*-relation (i.e., the 2-pattern) found in each image to build the index structure. Figure 7 shows an index structure built from *s*-relations in Table 1. The first level index is an array of size $N * (N-1)/2$, where N is the number of distinct objects in the image database [15]. The array index can be hashed by the object-pair O_{ij} in the *s*-relation. Each entry in this array contains

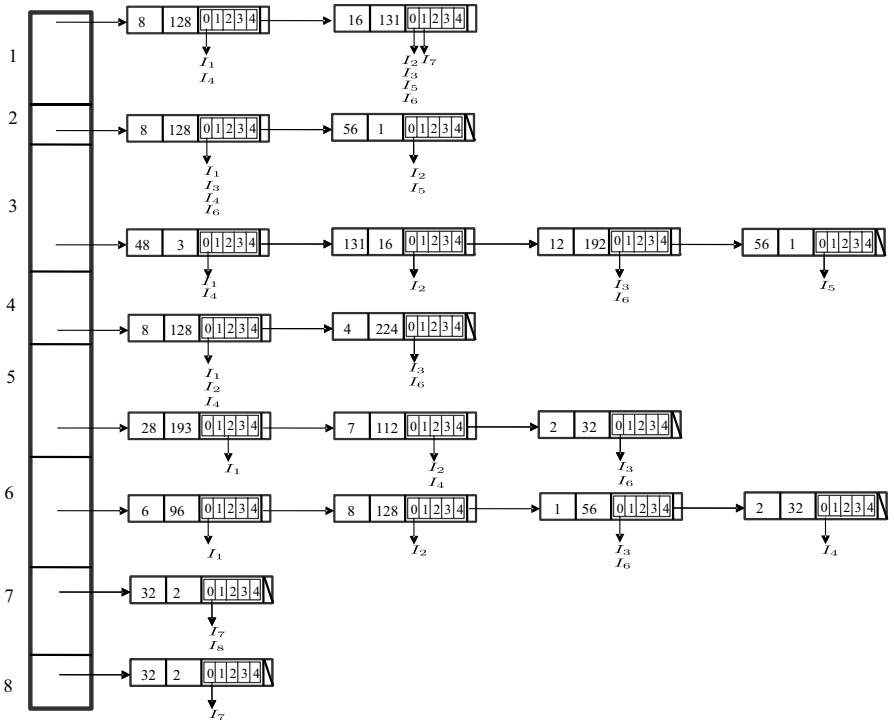

Fig. 7 The two-level index structure constructed from database *IDB*1

a pointer to a list of nodes. Each node in the list contains four elements, (1) the directional relation-code D_{ij}, (2) the directional relation-code D_{ji}, (3) the topological array which is an array of five pointers for five possible topological codes where each code points to an image set having directional relation-codes D_{ij} and D_{ji}, and (4) a pointer to the next node in the current 9D-SPA code list.

By setting the minimum support, *min_sup*, to 1/4, we find the frequent 2-patterns when building the index structure. Whenever adding an *s*-relation to the index structure, the size of one of the referenced image set is incremented, and we then check whether the size is not less than (*min_sup* * |*D*|), where |*D*| is the number of images in our database. In our running example, this value is 2 (= 1/4 * 8). If it is true, we then consider that the *s*-relation is a frequent 2-pattern. Without rescanning the index structure, the frequent 2-patterns can be found whenever the construction of the index structure is finished. All the frequent 2-patterns for the images shown in Fig. 4 are shown in Fig. 8.

The frequent pattern (FP) list is simultaneously built when constructing the two-level index structure. The main advantage of the FP list is to quickly retrieve the frequent 2-patterns. The FP list has two levels. The first level is an array which size is equal to the size of the first level of the two-level index structure. The array index can be hashed by the object-pair O_{ij} in the *s*-relation. Each entry in the array contains a pointer to a list of nodes. Each node contains five elements,

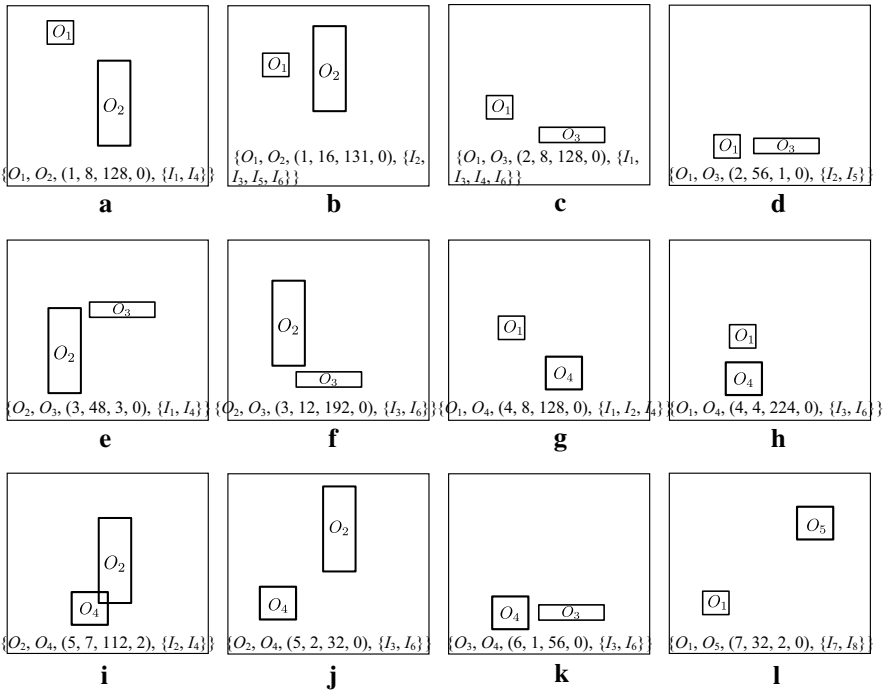

Fig. 8 The frequent size-2 patterns found in database *IDB*1

(1) the directional relation-code D_{ij}, (2) the directional relation-code D_{ji}, (3) the topological code T_{ij}, (4) a pointer to the image set containing the *s*-relation, and (5) a pointer to the next node in the current list. Whenever adding an *s*-relation to the index structure, the size of one of the referenced image set is incremented, and we then check whether the size is not less than (*min_sup* * |*D*|); in the running example, this value is 2. If it is true, we will add the *s*-relation and the reference of the image set to the FP list. Without rescanning the index structure, the FP list can be constructed whenever the construction of the index structure is finished. Figure 9 shows a completed FP list, which is built from the two-level index structure in Fig. 7. The number of 2-patterns in the FP list is always less than the two-level index structure, since the FP list stores only the frequent 2-patterns.

Given the object-pair code O_{ij} between two objects O_i and O_j, we can retrieve the frequent *s*-relations between O_i and O_j from the FP list. For example, if we want to retrieve the frequent relations between O_1 and O_2, since its object-pair code O_{12} is 1, we then can retrieve two frequent *s*-relations {(1, 8, 128, 0), {I_1, I_4}} and {(1, 16, 131, 0), {I_2, I_3, I_5, I_6}} from the FP list shown in Fig. 9. During generating frequent $(k+1)$-patterns from joining two joinable *k*-patterns, we can prune the most impossible candidates by retrieving the frequent *s*-relations from the FP list. The pruning strategy will be discussed later.

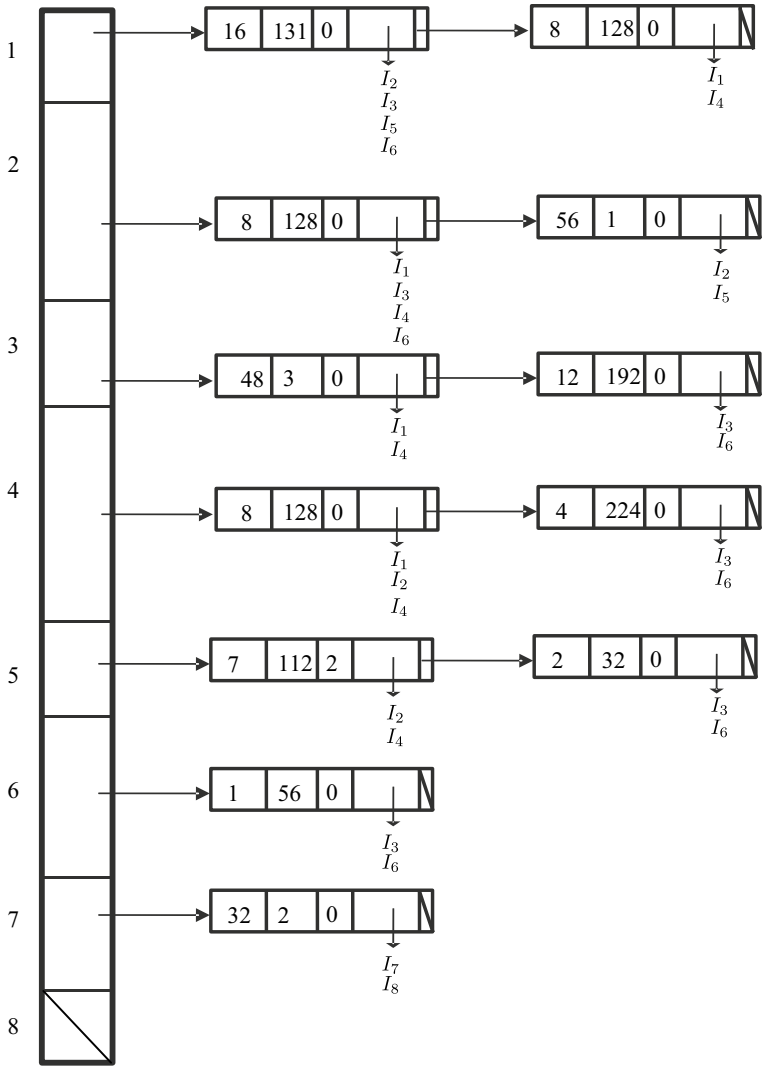

Fig. 9 An FP list built from the two-level index structure

3.4 Frequent pattern generation with the pruning strategy

When joining two joinable k-patterns to generate $(k+1)$-candidate patterns, there exists an unknown s-relation in the generated candidates. We propose two conditions to avoid generating invalid candidates during frequent pattern generation. When satisfying one of the two conditions, we will not generate candidates. Moreover, we describe the pruning strategy used in our approach. The pruning strategy involves two stages, candidate generation and verification. In the candidate generation stage, we will find the unknown s-relation to generate the candidates when performing joining process. Since some

generated candidates are impossible to exist in the 2D space, we will prune them in the verification stage.

3.4.1 The FPL and IntImage conditions

The proposed approach provides two conditions to avoid generating invalid candidates of frequent patterns, the FPL (frequent pattern list) and IntImageSet (intersected image set) conditions. If either one is met, no candidate will be generated. According to the Apriori property [28], each spatial relation in a frequent pattern is frequent. This indicates that if the candidate pattern is frequent, its unknown *s*-relation r_{ij} should be frequent. Therefore, if the unknown *s*-relation r_{ij} cannot be found in the FP list, no candidate will be generated. Then, we have the FPL condition: if $|\text{FP}[O_{ij}]|=0$, no candidate will be generated. Note that $|\text{FP}[O_{ij}]|$ returns the number of *s*-relation nodes attached to the object-pair O_{ij}.

For example, from Fig. 8a, l, these two joinable frequent 2-patterns can be possibly joined to generate candidate 3-patterns, $\{O_1, O_2, (1, 8, 128, 0), \{I_1, I_4\}\}$ and $\{O_1, O_5, (7, 32, 2, 0), \{I_7, I_8\}\}$. In this case, there is an unknown *s*-relation r_{25}. We can then compute that the object-pair O_{25} is 8 and retrieve frequent *s*-relation r_{25} from the FP list shown in Fig. 9. However, since $|\text{FP}[O_{25}]|$ is 0, no frequent *s*-relation r_{25} can be found in the FP list. Therefore, in this case, no candidate is generated.

On the other hand, the IntImageSet condition is that two joinable *k*-patterns will not generate any frequent $(k+1)$-pattern if the number of the intersection of their associated image sets is less than (*min_sup* * |*D*|). The potential frequent $(k+1)$-pattern exists at most in all the images that are in the intersected image set of its associated joinable *k*-patterns. If the number of the intersected image set is not less than (*min_sup* * |*D*|), this $(k+1)$-pattern will not be frequent.

For example, from Fig. 8a, d, these two joinable frequent 2-patterns can be possibly joined to generate candidate 3-patterns, $\{O_1, O_2, (1, 8, 128, 0), \{I_1, I_4\}\}$ and $\{O_1, O_3, (2, 56, 1, 0), \{I_2, I_5\}\}$. In this case, there is an unknown *s*-relation r_{23}. Then, we have the object-pair O_{23} is 3, and find out that $|\text{FP}[3]|=2>0$, which satisfies the FPL condition. However, the intersection of the image sets of these two *s*-relations is empty; this implies that there exists no such image containing the generated candidate 3-pattern in the database. Therefore, no candidate will be generated.

3.4.2 Candidate pattern generation

Candidate $(k+1)$-patterns are generated by joining two joinable *k*-patterns, $k \geq 2$, where the first $(k-1)$ items and the corresponding *s*-relations between them must be identical [15]. Given two joinable *k*-patterns *A* and *B*,

$$A = \{O_{p(1)}, O_{p(2)}, \ldots, O_{p(k-1)}, \boldsymbol{O_i}, r_{p(1)p(2)}, r_{p(1)p(3)}, \ldots, r_{p(k-1)i}, A.imgs\}$$
$$B = \{O_{p(1)}, O_{p(2)}, \ldots, O_{p(k-1)}, \boldsymbol{O_j}, r_{p(1)p(2)}, r_{p(1)p(3)}, \ldots, r_{p(k-1)i}, B.imgs\},$$

where $O_i \neq O_j$, the candidate $(k+1)$-pattern *P* is generated as follows.

$$P = \{O_{p(1)}, O_{p(2)}, \ldots, O_{p(k-1)}, \boldsymbol{O_i}, \boldsymbol{O_j}, r_{p(1)p(2)}, r_{p(1)p(3)}, \ldots, \boldsymbol{r_{ij}}, P.imgs\},$$

where there exists an unknown s-relation r_{ij}. $p(a)$ is the object number of the object in the joinable k-pattern, $p(a) < p(a+1)$, $1 \le a \le (k-1)$. $P.imgs$ is the image set of pattern P containing the images that have all the corresponding s-relations. The image set containing the unknown s-relation r_{ij} is $r_{ij}.imgs$. Then, we have $P.imgs = (A.imgs \cap B.imgs) \cap r_{ij}.imgs$.

Given the unknown s-relation r_{ij}, if the frequent s-relation r_{ij} exists in the FP list and the number of the intersection of two image sets of two joinable patterns is not less than (min_sup * $|D|$), we can then find the unknown s-relation to generate the candidate. For example, from Fig. 8b, d, we can join

$$\{O_1, \boldsymbol{O_2}, (1, 16, 131, 0), \{I_2, I_3, I_5, I_6\}\}$$
$$\{O_1, \boldsymbol{O_3}, (2, 56, 1, 0), \{I_2, I_5\}\},$$

where there is an unknown s-relation r_{23}. We can then compute that O_{23} is 3. Since $|\mathrm{FP}[O_{23}]|$ $(=2) > 0$ and the number of the intersection of both image sets, $\{I_2, I_5\}$, is not less than 2 $(= 1/4 * 8)$, we can continue to generate the candidate. We can use the Apriori property that "any relation in a frequent pattern must be frequent" [26] to find the possible values of r_{23}. Since O_{23} is 3, we can then use the FP list shown in Fig. 9 to find it out in an efficient way. From searching in the FP list in Fig. 9, the possible values of r_{23} are (3, 48, 3, 0) and (3, 12, 192, 0). Therefore, we will generate two candidates

$$\{O_1, O_2, O_3, (1, 16, 131, 0), (2, 56, 1, 0), \underline{(3, 48, 3, 0)}, A.imgs\}$$
$$\{O_1, O_2, O_3, (1, 16, 131, 0), (2, 56, 1, 0), \underline{(3, 12, 192, 0)}, B.imgs\}.$$

However, the generated candidates may not be valid in the 2D space. In other words, some candidates may not be found in the 2D space. For example, one of the generated candidates contains the s-relations (1, 16, 131, 0), (2, 56, 1, 0) and (3, 48, 3, 0). We have to ensure that these three s-relations can coexist within in a pattern or in an image. When retrieving the possible values of r_{23} from the FP list, we have to consider the other known s-relations in the candidate. That is, we have to ensure that the s-relation from the FP list can coexist with other known s-relations, or the s-relations in a pattern are consistent among each other. When the s-relations within a pattern cannot coexist, we say that the s-relation from the FP list does not satisfy the *spatial consistency*. That is, no such image containing all these s-relations at the same time can be found in the image database. We define this problem as the *inconsistent problem*. When a pattern has the inconsistent problem, it is an impossible pattern that cannot be found in the 2D space.

In the current join example, to make the s-relation from the FP list to satisfy the spatial consistency, we have to check whether the frequent s-relation r_{23} can coexist with the known s-relations (1, 16, 131, 0) (r_{12}, the s-relation between O_1 and O_2) and (2, 56, 1, 0) (r_{13}, the s-relation between O_1 and O_3). Figure 10a shows these two known s-relations (1, 16, 131, 0) and (2, 56, 1, 0). Figure 10b, c shows the two frequent s-relations (3, 48, 3, 0) and (3, 12, 192, 0) from the FP list shown in Fig. 9, respectively. For the frequent s-relation (3, 48, 3, 0) in Fig. 10b,

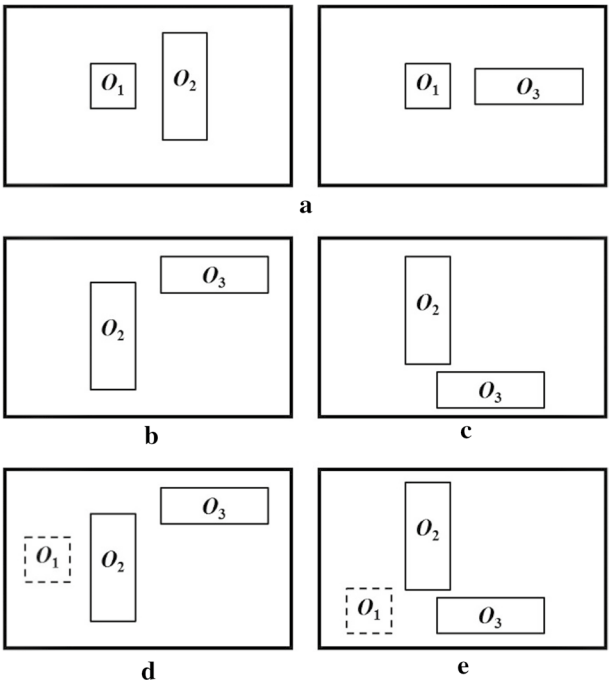

Fig. 10 The example to illustrate the consistent property: **a** the known *s*-relations in the candidate; **b** one of the frequent *s*-relations r_{23}; **c** one of the frequent *s*-relations r_{23}; **d** the possible location of O_1 when considering the *s*-relation r_{23} shown in (**b**); **e** the possible location of O_1 when considering the *s*-relation r_{23} shown in (**c**)

in order to coexist with both known *s*-relations shown in Fig. 10a, we have to locate O_1 to form the candidate containing O_1, O_2 and O_3. However, we cannot generate both known *s*-relations (1, 16, 131, 0) and (2, 56, 1, 0) whatever the position located with O_1. That is, we cannot generate the candidate containing (1, 16, 131, 0), (2, 56, 1, 0) and (3, 48, 3, 0). For example, Fig. 10d shows one of the locations of O_1 in Fig. 10b. In Fig. 10d, we can find that the *s*-relation between O_1 and O_3 is (2, 32, 2, 0). However, the known relation between O_1 and O_3 is (2, 56, 1, 0). Therefore, if r_{23} is (3, 48, 3, 0), it cannot satisfy the spatial consistency. Similarly, Fig. 10c shows the other *s*-relation (3, 12, 192, 0) from the FP list. One of possible locations of O_1 is shown in Fig. 10e. In Fig. 10e, the relation between O_1 and O_2 is (1, 48, 3, 0), but the known relation between O_1 and O_2 is (1, 16, 131, 0). Whatever the location of O_1 is, we can deduce that (3, 12, 192, 0) cannot coexist with (1, 16, 131, 0) and (2, 56, 1, 0), so (3, 12, 192, 0) does not satisfy the spatial consistency.

Therefore, in the verification stage, we need to verify the frequent *s*-relations from the FP list. After the verification, we can prune the frequent *s*-relations that do not satisfy the spatial consistency. Hence, the impossible candidates can be pruned.

3.4.3 Verification

In our proposed approach, we use a reasoning method [15] to reason the *s*-relations that satisfy the spatial consistency. In the 9DSPA-Miner [15], the reasoning method may also discover a lot of the impossible relations when reasoning the *unknown* relation, generating invalid candidates. Therefore, based on the result of the reasoning method, we can verify whether that the frequent *s*-relations from the FP list satisfy the spatial consistency. That is, we can prune a large number of candidates that are impossible to be found in the 2D space.

When finding the possible values of the unknown *s*-relation r_{ij}, we need to retrieve them from the FP list. However, the values from the FP list may not satisfy the spatial consistency. Therefore, there may exist some candidates that are impossible to be found in the 2D space. To further prune these candidates, we need to find the values of r_{ij} satisfying the spatial consistency. Assume that O_f, O_i and O_j are within in an image, where $f<i<j$ and the *s*-relation r_{ij} is an unknown relation. Let the sets of reasoned valid values of D_{ij}, D_{ji} and T_{ij} in r_{ij} are RD_{ij}, RD_{ji} and RT_{ij}, respectively. Based on the reasoning method in [15], we can use D_{if} and D_{jf} to reason these three valid sets. If there is more than one O_f, all the reasoned valid sets for each O_f should be obtained, and each final valid set is the intersection of the corresponding reasoned valid sets for all of O_f's. The detail of the reasoning method can be found in [15]. The combinations of three valid sets contain the *s*-relations that satisfy the spatial consistency. However, in [15], they reason the unknown relations of all combinations of these three valid sets, which may generate impossible relations that do not exist in the image database. Therefore, in our proposed approach, given a frequent *s*-relation $r_{ij}=(O_{ij}, D_{ij}, D_{ji}, T_{ij})$ from the FP list, if $D_{ij} \in RD_{ij}$, $D_{ji} \in RD_{ji}$ and $T_{ij} \in RT_{ij}$, r_{ij} is a consistent *s*-relation in the image. Since we use the frequent relation stored in the FP list to discover the unknown relation, we can avoid generating the impossible relations, which exist in the 9DSPA-Miner [15].

For example, we take the two joinable 2-patterns shown in Fig. 8b, d to illustrate our pruning strategy. Figure 8b shows the frequent 2-pattern containing O_1 and O_2 with the *s*-relation $r_{12}=(1, 16, 131, 0)$, and Fig. 8d the one containing O_1 and O_3 with the *s*-relation $r_{13}=(2, 56, 1, 0)$. Whenever joining them, we have to find the frequent *s*-relations of r_{23} from the FP list in Fig. 9, and the results are (3, 48, 3, 0) and (3, 12, 192, 0), which are the candidates in the candidate generation stage. Among these *s*-relations for r_{23}, some may generate the impossible candidates, since they are not consistent with the other *s*-relations (i.e., (1, 16, 131, 0) and (2, 56, 1, 0)) within an image. Therefore, we can use D_{21} and D_{31} to reason three valid sets RD_{23}, RD_{32} and RT_{23} to discover the *s*-relation r_{23} that satisfies the spatial consistency. After that, we will prune the *s*-relation r_{23} that does not satisfy the spatial consistency.

In this case, based on the reasoning method in [15], three valid sets for r_{23} can be reasoned, $RD_{23}=\{56, 68, 124, 131, 199, 255\}$, $RD_{32}=\{0, 1, 16, 17\}$ and $RT_{23}=\{0, 1, 2, 3, 4\}$. Moreover, all these possible *s*-relations for r_{23} are shown in Fig. 11. Let us verify the frequent *s*-relation $r_{23}=(3, \underline{48}, 3, 0)$. Since D_{23} is 48 that is not in the valid set RD_{23}, it does not satisfy the spatial consistency. In the same way, the other *s*-relation $r_{23}=(3, \underline{12}, 192, 0)$ does not satisfy the spatial consistency, since D_{23} is 12 that is not in the valid set RD_{23}. In other words, when joining these two joinable

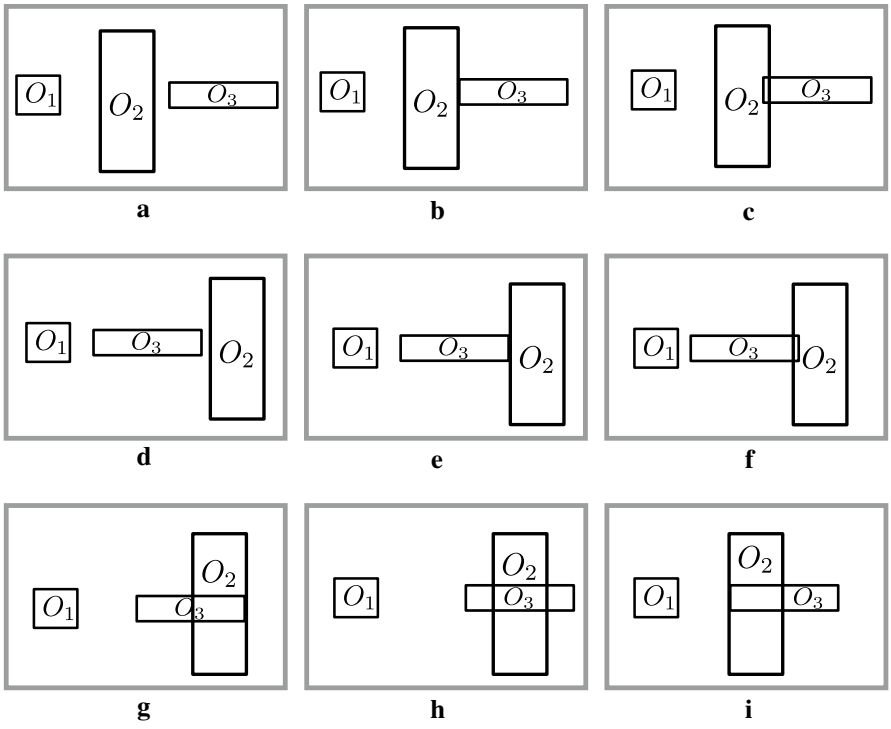

Fig. 11 The valid candidates: **a** $r_{23}=(3, 56, 1, 0)$; **b** $r_{23}=(3, 56, 1, 1)$; **c** $r_{23}=(3, 56, 1, 2)$; **d** $r_{23}=(3, 131, 16, 0)$; **e** $r_{23}=(3, 131, 16, 1)$; **f** $r_{23}=(3, 199, 16, 2)$; **g** $r_{23}=(3, 68, 16, 2)$; **h** $r_{23}=(3, 68, 17, 2)$; **i** $r_{23}=(3, 68, 1, 2)$

2-patterns from Fig. 8b, d, all candidates generated in the early stage (i.e., the candidate generation stage) are pruned, since all the frequent *s*-relations for r_{23} do not satisfy the spatial consistency. Note that, for the same case, the 9DSPA-Miner [15] generates 120 candidates, which is equal to $|RD_{23}| * |RD_{32}| * |RT_{23}|$.

3.4.4 The frequent 3-pattern

From Fig. 8b, c, we can join the following two joinable frequent 2-patterns.

$$\{O_1, \boldsymbol{O_2}, (1, 16, 131, 0), \{I_2, I_3, I_5, I_6\}\}$$
$$\{O_1, \boldsymbol{O_3}, (2, 8, 128, 0), \{I_1, I_3, I_4, I_6\}\}.$$

From the FP list shown in Fig. 9, the frequent *s*-relations of r_{23} are (3, 48, 3, 0) and (3, 12, 192, 0), where $r_{23}.imgs$ for (3, 48, 3, 0) is $\{I_1, I_4\}$ and that for (3, 12, 192, 0) is $\{I_3, I_6\}$. According to the reasoning method in [13], we can use D_{21} and D_{31} to reason the following three valid sets, $RD_{23}=\{2, 3, 4, 6, 7, 8, \underline{12}, 14, 24, 28, 31, 56, 68, 124, 131, 199, 255\}$, $RD_{32}=\{0, 1, 16, 17, 32, 48, 64, 96, 112, 128, 129, \underline{192}, 193, 224, 241\}$ and $RT_{23}=\{\underline{0}, 1, 2, 3, 4\}$. After the verification, only the *s*-relation

(3, 12, 192, 0) satisfies the spatial consistency and the number of intersection of the corresponding image sets is not less than (*min_sup* * |*D*| = 2). Therefore, the following candidate of the frequent 3-pattern is generated, {O_1, $\boldsymbol{O_2}$, $\boldsymbol{O_3}$, (1, 16, 131, 0), (2, 8, 128, 0), (3, 12, 192, 0), {I_3, I_6}}, as shown in Fig. 12a. In addition, any sub *k*-patterns in a frequent (*k* + 1)-pattern must be frequent as mentioned in [26]. In this case, any sub 2-patterns in this generated 3-pattern shown in Fig. 12a are all frequent, as shown in Fig. 8, so that this 3-pattern is definitely a frequent 3-pattern. In the same processing, all the other frequent 3-patterns for the database *IDB*1 shown in Fig. 4 are shown in Fig. 12b–e.

3.4.5 The frequent 4-pattern

Any two joinable 3-patterns shown in Fig. 12 are joined to generate the candidate 4-patterns if the number of the intersection of their corresponding image sets is not less than (*min_sup* * |*D*|). In Fig. 12, only the 3-patterns from Fig. 12a, c are processed, since the number of the intersections of the image sets of other joinable 3-patterns is less than (*min_sup* * |*D*|) = 2.

$$\{O_1, O_2, \boldsymbol{O_3}, (1, 16, 131, 0), (2, 8, 128, 0), (3, 12, 192, 0), \{I_3, I_6\}\}$$
$$\{O_1, O_2, \boldsymbol{O_4}, (1, 16, 131, 0), (4, 4, 224, 0), (5, 2, 32, 0), \{I_3, I_6\}\}.$$

For this joining, there is an unknown *s*-relation r_{34}. From searching in the FP list shown in Fig. 9, the *s*-relation r_{34} is (6, 1, 56, 0), and $r_{34}.imgs = \{I_3, I_6\}$. According to the reasoning method in [15], for object O_1, we can use D_{31} and D_{41} to reason the following three valid sets, $RD'_{34} = \{0, 1, 2, 3, 4, 6, 7, 8, 12, 14, 16, 17, 24, 28, 31, 32, 48, 56, 64, 68, 96, 112, 124, 128, 129, 131, 192, 193, 199, 224, 241, 255\}$, $RD'_{43} = \{0, 1, 2, 3, 4, 6, 7, 8, 12, 14, 16, 17, 24, 28, 31, 32, 48, 56, 64, 68, 96, 112,$

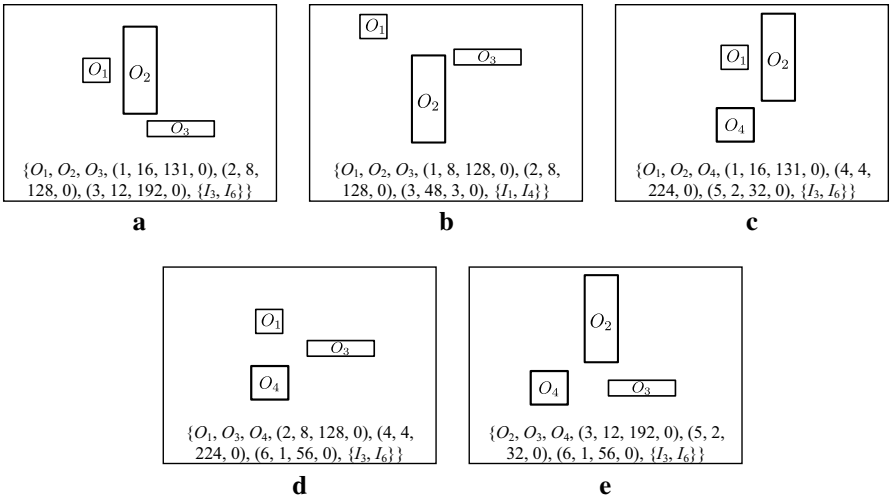

Fig. 12 All frequent 3-patterns for the database *IDB*1 shown in Fig. 5

124, 128, 129, 131, 192, 193, 199, 224, 241, 255} and $RT"_{34}=\{0, 1, 2, 3, 4\}$. In addition, according to the reasoning method in [15], for object O_2, we also need to use D_{32} and D_{42} to reason the following three valid sets, $RD"_{34}=\{1, 128, 129, 192, 193\}$, $RD"_{43}=\{8, 12, 24, 28, 56, 124\}$ and $RT"_{34}=\{0, 1, 2\}$. Then, the three valid sets of D_{34}, D_{43} and T_{34} are the intersections of their corresponding valid sets in aspects of objects O_1 and O_2, respectively.

$$RD_{34} = RD'_{34} \cap RD''_{34} = \{\underline{1}, 128, 129, 192, 193\}$$
$$RD_{43} = RD'_{43} \cap RD''_{43} = \{8, 12, 24, 28, \underline{56}, 124\}$$
$$RT_{34} = RD'_{34} \cap RD''_{43} = \{\underline{0}, 1, 2\}.$$

After the verification, the frequent s-relation (6, $\underline{1}$, $\underline{56}$, $\underline{0}$) satisfies the spatial consistency, and the number of intersection of the corresponding image sets is not less than ($min\_sup * |D|$) = 2. Therefore, this candidate 4-pattern can be generated. Note that for this join example, the 9DSPA-Miner [15] will generate 90 candidates, which is equal to $|RD_{34}| * |RD_{43}| * |RT_{34}|$. The generated candidate 4-pattern is $\{O_1, O_2, O_3, O_4, (1, 16, 131, 0), (2, 8, 128, 0), (4, 4, 224, 0), (3, 12, 192, 0), (5, 2, 32, 0), (\underline{6, 1, 56, 0}), \{I_3, I_6\}\}$, as shown in Fig. 13. Any sub 3-patterns in this frequent 4-pattern are all frequent shown in Fig. 12 so that this generated 4-pattern is frequent. Since there is only one frequent 4-pattern, it is impossible to generate a size-5 candidate. Therefore, the mining process can be terminated.

4 Performance evaluation

In this section, we make a comparison of the performance between our proposed approach and the 9DSPA-Miner [15]. We evaluate the performance efficiency on the processing time for the following experimental results.

4.1 Simulation model

The synthetic data generation algorithm in our performance evaluation is similar to that in the 9DSPA-Miner [15]. We generate N objects in the synthetic database,

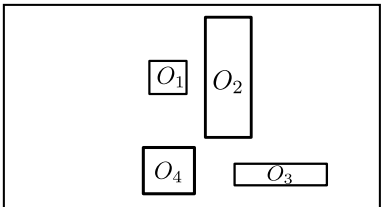

Fig. 13 The frequent 4-pattern for the database *IDB*1 shown in Fig. 4

where each object has its length and width, which are selected from a uniform distribution. Since there exist 9D-SPA spatial relations [27] within an image, every object, which is randomly selected from N objects, is uniformly distributed in the 2D space. Therefore, the spatial relations between any two objects can be determined. We generate rectangular objects that are uniformly distributed in an image of size 100 * 100. Every image generated is expressed as the 9D-SPA representation.

N is the number of objects, ranging from 30 to 70. T is the average size of images, measured in the number of objects in an image, and its range is from 6 to 14. D is the number of images in thousands, ranging from 10 to 100. I is the average size of potential frequent patterns, which is set to 4. L is the number of potential frequent patterns, which is set to 2000. *min_sup* is the minimum support threshold, ranging from 0.1 to 0.5%. The number of objects for each generated image is determined by the Poisson distribution with a mean equal to T. Each image consists of a series of potentially large patterns, where those patterns are chosen from a set of L large potentially patterns. Each pattern in L has an associated weight generated from the Zipf distribution that determines the probability that this pattern will be chosen. The size of each potentially large patterns in L is determined from a Poisson distribution with a mean equal to I.

4.2 Computational complexity

For constructing a two-level index structure and an FP list that stores frequent 2-patterns, the proposed approach needs to scan the whole input image database once. Consider that there are D images having an average T objects in the input database. Each image has $\binom{T}{2} = \frac{T\times(T-1)}{2}$ s-relations. As a result, the computational cost of processing this part requires $O\left(D \times \frac{T\times(T-1)}{2}\right)$.

To generate candidate frequent k-patterns, $k \geq 3$, the proposed approach joins pairs of joinable frequent $(k-1)$-patterns, where the first $(k-2)$ objects between them must be identical. Therefore, each joining operation has at most (-2) equality comparisons [29] and examines at most l unknown s-relations from the FP list. In the worst case, for each iteration, the proposed approach needs to join every pair of joinable frequent $(k-1)$-patterns, F_{k-1} [29]. As a result, the computational cost of processing this part requires $O\left(\sum_{k=3}^{T}(k-2+l)\left|F_{k-1}\right|^2\right)$.

4.3 Experimental results

In the first experiment, we compare the performance of our proposed approach and the 9DSPA-Miner [15] with the synthetic data D50T10L2000I4N36, where the number of images $D=50$ K, the average size of images $T=10$, the number of potential frequent patterns $L=2000$, the average size of the potential frequent patterns $I=4$, and the number of objects $N=36$. Figure 14 shows the comparison of the processing time under the different values of *min_sup* from 0.1 to 0.5%. When the value of *min_sup* decreases, the processing time of the 9DSPA-Miner increases dramatically, but that of our proposed approach increases slowly. In this figure, we can

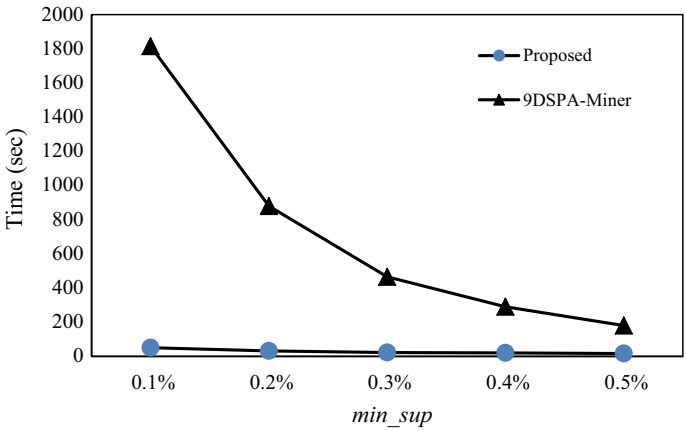

Fig. 14 A comparison of the processing time under the different minimum support

observe that the proposed approach has a shorter processing time than the 9DSPA-Miner in all cases. When the value of *min_sup* is set to 0.1%, the processing time of the proposed approach is about 36 times faster than that of the 9DSPA-Miner.

In the second experiment, we evaluate the effect of the different average size of images. In this experiment, we generate the dataset D10L2000I4N36sup0.5%, where *min_sup* is set to 0.5% and the average size of images varies from 6 to 14. Figure 15 shows the comparison of the processing time with the different average size of images. When the average size of images increases, the processing time of both algorithms increases. The reason for the increase of the processing time of both algorithms is that the number of frequent patterns and candidates increases when the average size of images increases. Therefore, both algorithms need long time in checking whether the candidates are frequent. However, the processing

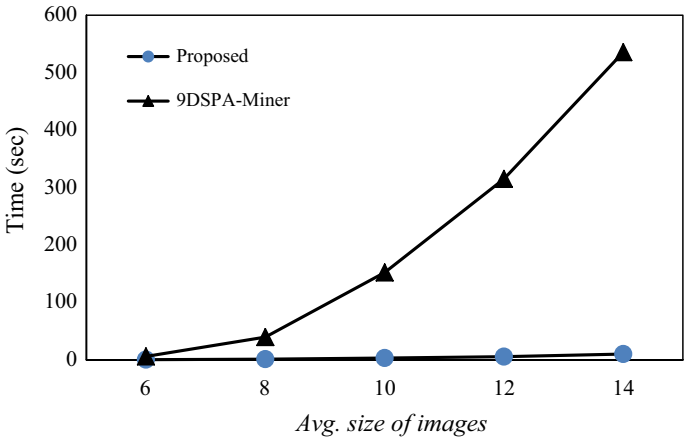

Fig. 15 A comparison of the processing time under the different average size of images

time of our approach increases slowly as compared to that of the 9DSPA-Miner. Our approach is more scalable than the 9DSPA-Miner, since we can prune a large number of invalid candidates.

In the third experiment, we evaluate the effect of the different number of objects. In the experiment, we generate the dataset D10T10L2000I4sup0.5%, where the number of objects varies from 30 to 70 and *min_sup* is set to 0.5%. Figure 16 shows the comparison of the processing time with the different number of objects. Because the average support for the patterns decreases as the number of objects increases, the processing time of both algorithms decreases when the number of objects increases. Our proposed algorithm is faster than the 9DSPA-Miner, especially, when the number of objects decreases.

In the final experiment, we evaluate the effect of the different number of images. In the experiment, we generate the dataset T10L2000I4N36sup0.5%, where the number of images varies from 10 to 100 K and *min_sup* is set to 0.5%. Figure 17 shows the comparison of the processing time with the different number of images. When the number of images increases, the number of candidates increases. Moreover, both approaches need long time in building the index structure as the number of images increases. Furthermore, since both approaches count the support of a candidate by the intersection of image sets, they need long time in intersecting when the average support of patterns increases. In the 9DSPA-Miner, they recompute the intersections of image sets to count the support of the discovered patterns. However, since every discovered pattern is associated with an image set in our approach, we do not need to scan the index structure to count the support and do not recompute the intersections of image sets. Therefore, the proposed approach takes less time in counting the support than the 9DSPA-Miner, as can be seen in Fig. 17.

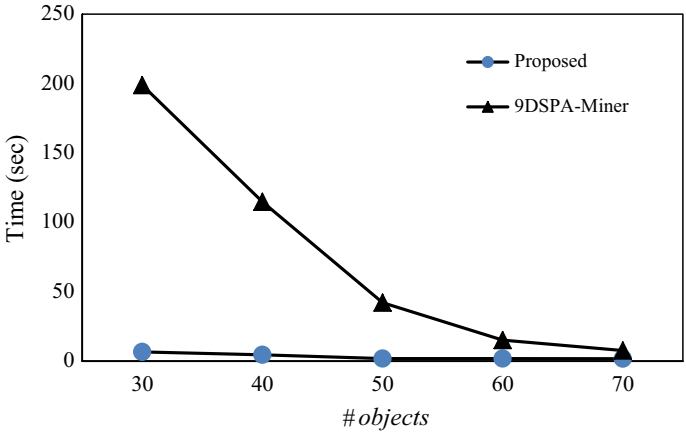

Fig. 16 A comparison of processing time under the different number of objects

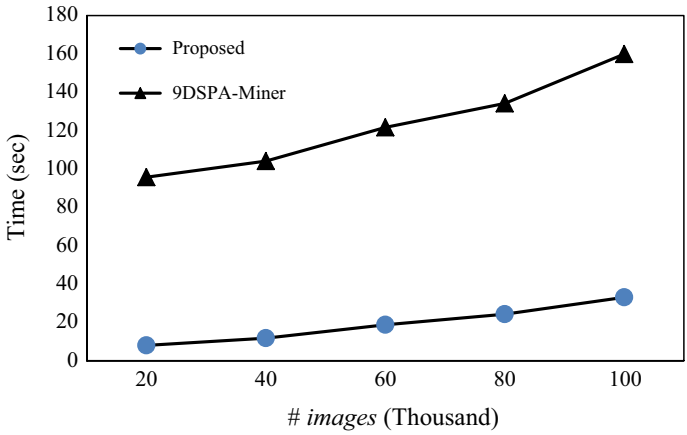

Fig. 17 A comparison of processing time under the different number of images

5 Conclusions

In this paper, based on the frequent pattern list, we have proposed an approach that can avoid generating the invalid relations which could not be found in the 2D space or in the current input database. After creating size-k candidates that satisfy the spatial consistency from size-$(k-1)$ frequent patterns by following the reasoning method [15], we will do the verification stage to prune invalid candidates. In addition, when performing the join step, the FPL and IntImageSet conditions are used. If either one condition is satisfied, we then will not generate any candidate that will not be frequent whatever the value of the unknown relation is. Furthermore, when counting the support of the candidate, we will not scan the index structure. Since each discovered pattern is associated with an image set, we will not recompute the intersections of image sets. From our simulation results, we have shown that our proposed approach is more efficient than the 9DSPA-Miner. How to handle uncertain or missing data [30, 31] for image data mining is a possible future research direction. How to handle data insertion/deletion for incremental mining [32] in image databases is another possible future research direction.

Acknowledgements This research was supported by Grant MOST 107-2221-E-110-064 from the Ministry of Science and Technology, Taiwan.

References

1. Shekhar S, Zhang P, Huang Y, Vatsavai RR (2004) Trends in spatial data mining. In: Joshi A, Kargupta H (eds) Data mining: next generation challenges and future directions. AAAI/MIT Press, Cambridge, pp 357–380

2. Wu X, Zhang X (2019) An efficient pixel clustering-based method for mining spatial sequential patterns from serial remote sensing images. Comput Geosci 124:128–139. https://doi.org/10.1016/j.cageo.2019.01.005
3. Han J, Koperski K, Stefanovic N (1997) GeoMiner: a system prototype for spatial data mining. ACM SIGMOD Rec 26:553–556. https://doi.org/10.1145/253262.253404
4. Huang Y, Shekhar S, Xiong H (2014) Discovering colocation patterns from spatial data sets: a general approach. IEEE Trans Knowl Data Eng 16:1472–1485. https://doi.org/10.1109/TKDE.2004.90
5. Hudelot C, Atif J, Bloch I (2008) Fuzzy spatial relation ontology for image interpretation. Fuzzy Sets Syst 159:1929–1951. https://doi.org/10.1016/j.fss.2008.02.011
6. Koperski K, Han J (1995) Discovery of spatial association rules in geographic information databases. In: Egenhofer MJ, Herring JR (eds) Lecture notes in computer science. Springer, Berlin, Heidelberg, pp 47–66
7. Morimoto Y (2001) Mining frequent neighboring class sets in spatial databases. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM Press, New York, pp 353–358
8. Shekhar S, Huang Y (2001) Discovering spatial co-location patterns: a summary of results. In: Jensen CS, Schneider M, Seeger B, Tsotras VJ (eds) Lecture notes in computer science. Springer, Berlin, Heidelberg, pp 236–256
9. Chang S-K, Shi Q-Y, Yan C-W (1987) Iconic indexing by 2-D strings. IEEE Trans Pattern Anal Mach Intell PAMI-9:413–428. https://doi.org/10.1109/tpami.1987.4767923
10. Hsu W, Lee ML, Zhang J (2002) Image mining: trends and developments. J Intell Inf Syst 19:7–23. https://doi.org/10.1023/A:1015508302797
11. Koli R, Pal R, Chaube N, Joshi K, Maithani A (2019) Agile data mining approach for medical image mining. In: 2019 International Conference on Automation, Computational and Technology Management. IEEE, pp 246–251
12. Voravuthikunchai W, Crémilleux B, Jurie F (2014) Image re-ranking based on statistics of frequent patterns. In: Proceedings of International Conference on Multimedia Retrieval. ACM Press, New York, pp 129–136
13. Hsu W, Dai J, Lee ML (2003) Mining viewpoint patterns in image databases. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM Press, New York, pp 553–558
14. Lee AJT, Hong R-W, Ko W-M, Tsao W-K, Lin H-H (2007) Mining spatial association rules in image databases. Inf Sci (NY) 177:1593–1608. https://doi.org/10.1016/j.ins.2006.09.018
15. Lee AJT, Liu Y-H, Tsai H-M, Lin H-H, Wu H-W (2009) Mining frequent patterns in image databases with 9D-SPA representation. J Syst Softw 82:603–618. https://doi.org/10.1016/j.jss.2008.08.028
16. Saritha S, Santhosh KG (2011) Interestingness analysis of semantic association mining in medical images. In: Venugopal KR, Patnaik LM (eds) Communications in computer and information science. Springer, Berlin, Heidelberg, pp 1–10
17. Wei L-Y, Shan M-K (2006) Efficient mining of spatial co-orientation patterns from image databases. In: 2006 IEEE International Conference on Systems, Man and Cybernetics. IEEE, pp 2982–2987
18. Agarwal S, Verma AK, Singh P (2013) Content based image retrieval using discrete wavelet transform and edge histogram descriptor. In: 2013 International Conference on Information Systems and Computer Networks. IEEE, pp 19–23
19. Flickner M, Sawhney H, Niblack W, Ashley J, Huang Q, Dom B, Gorkani M, Hafner J, Lee D, Petkovic D, Steele D, Yanker P (1995) Query by image and video content: the QBIC system. Computer 28:23–32. https://doi.org/10.1109/2.410146
20. Liu G-H, Yang J-Y (2013) Content-based image retrieval using color difference histogram. Pattern Recognit 46:188–198. https://doi.org/10.1016/j.patcog.2012.06.001
21. Murala S, Wu QMJ (2014) Expert content-based image retrieval system using robust local patterns. J Vis Commun Image Represent 25:1324–1334. https://doi.org/10.1016/j.jvcir.2014.05.008
22. Singha M, Hemachandran K (2012) Content based image retrieval using color and texture. Signal Image Process Int J 3:39–57. https://doi.org/10.5121/sipij.2012.3104
23. Wang X-Y, Zhang B-B, Yang H-Y (2014) Content-based image retrieval by integrating color and texture features. Multimed Tools Appl 68:545–569. https://doi.org/10.1007/s11042-012-1055-7
24. Chang C-C (1991) Spatial match retrieval of symbolic pictures. J Inf Sci Eng 7:405–422
25. Chang SK, Yan CW, Dimitroff DC, Arndt T (1988) An intelligent image database system. IEEE Trans Softw Eng 14:681–688. https://doi.org/10.1109/32.6147

26. Huang P-W, Hsu L, Su Y-W, Lin P-L (2008) Spatial inference and similarity retrieval of an intelligent image database system based on object's spanning representation. J Vis Lang Comput 19:637–651. https://doi.org/10.1016/j.jvlc.2007.09.001
27. Huang P-W, Lee C-H (2004) Image database design based on 9D-SPA representation for spatial relations. IEEE Trans Knowl Data Eng 16:1486–1496. https://doi.org/10.1109/TKDE.2004.92
28. Agrawal R, Srikant R (1994) Fast algorithms for mining association rules. In: Proceedings of 20th International Conference on Very Large Data Bases, pp 487–499
29. Tan P-N, Steinbach M, Karpatne A, Kumar V (2019) Introduction to data mining, 2nd edn. Pearson, London
30. Ovi JA, Ahmed CF, Leung CK, Pazdor AGM (2019) Mining weighted frequent patterns from uncertain data streams. In: Kacprzyk J (ed) Advances in intelligent systems and computing. Springer, New York, pp 917–936
31. Rahman MM, Ahmed CF, Leung CK-S (2019) Mining weighted frequent sequences in uncertain databases. Inf Sci (NY) 479:76–100. https://doi.org/10.1016/j.ins.2018.11.026
32. Sumalatha S, Subramanyam RBV (2019) A MapReduce solution for incremental mining of sequential patterns from big data. Expert Syst Appl 133:109–125. https://doi.org/10.1016/j.eswa.2019.05.013