Detection of 2D Lattice Patterns of Repetitive Elements and their Use for Image Retrieval

Petr Doubek, Jiří Matas

doub@cmp.felk.cvut.cz, matas@cmp.felk.cvut.cz

CTU–CMP–2009–16

November 10, 2009


The authors were supported by U.S. government project 08-255.

Research Reports of CMP, Czech Technical University in Prague, No. 16, 2009

Published by Center for Machine Perception, Department of Cybernetics Faculty of Electrical Engineering, Czech Technical University
Technická 2, 166 27 Prague 6, Czech Republic
Detection of 2D Lattice Patterns of Repetitive Elements and their Use for Image Retrieval

Petr Doubek, Jiří Matas

November 10, 2009

Abstract

Repetitive structures in images can be an obstacle to matching and retrieval, if they are treated only as an unorganized set of elements. On the other hand, the repetitive structures can be turned into an advantage by identifying the pattern and matching or retrieving by the pattern. Our method detects planar repetitive structures (possibly multiple per image) under perspective distortion, such as windows on a facade. Each pattern is represented by a shift-invariant descriptor of the basic repeating element which can be matched to similar patterns in another views. The method is evaluated indirectly by an image retrieval experiment, where images of similar object are retrieved from the dataset based solely on the repetitive pattern description.

1 Introduction

Man-made environment contains many repeating elements, e.g. windows on a facade, tiles on the floor or poles of a railing. These repetitive patterns are distinctive for humans. However, they pose a problem even for state-of-the-art image matching and retrieval algorithms, because the repeating elements are treated independently and thus increase the number of tentative correspondences and possible mismatches, see Fig. 1.

Our goal is to detect the repetitive patterns and turn them into advantage by matching the entire repetitive structure instead of single repeating elements.

Different classes of repetitive patterns can be encountered in images: repetition of the basic building block – tile – on a 2D lattice, repetition along 1D line, scattered tiles (0D). In this paper, we focus on tiles repeating on a
regular 2D lattice, tolerating perspective distortion of the tile, slight variation in appearance of the tile and to some extent missing tiles in the pattern. As a by-product, we also detect 1D repetitions on a line. Figure 2 shows examples of repetitive patterns we aim to detect.

Of course, repetitive patterns were already studied by several researchers. One of the first works by Leung and Malik [1] grows the pattern from local seed window by SSD registration into a possibly deformed 2D lattice. Schaffalitzky and Zisserman [11] use very similar approach, investigating deeper the geometric structure of the pattern – they defined perspectively distorted line repetition as conjugate translation and lattice repetition as conjugate grid.

Tuytelaars et al. [13] took a global approach by clustering the possible basic elements of the repetitive pattern using cascaded Hough transform. They focus on detecting symmetries, detection of repetitive patterns is demonstrated only on one image of a tiled floor.

A computational model for periodic pattern was proposed by Liu et al. [2] using the theory of groups. The detection of patterns is performed on frontoparallel images of textures, often synthetic. The benefits of classification of patterns into crystallographic groups are unclear as it was rarely used in subsequent papers.

Park et al. [10] present impressive results on deformed lattice discovery,
however their method is evaluated only on images with a single dominant lattice structure. Their most recent work [9] was tested on images that contain multiple repetitive patterns, however authors do not show results with multiple detected patterns nor do they describe if and how they are detected. The evaluation metric is the percentage of tiles detected in a pattern. For matching or retrieval applications however, the detection of entire pattern is only of minor importance.

To our knowledge, only Schindler et al. [12] attempted matching or retrieval by repetitive patterns. However, the matching is not inter-image, but against manually prepared groundtruth database of facades. The database is very small, containing only nine patterns.

In contrast to the previous work, we aim to detect multiple repetitive patterns in image (if existent), find a shift-invariant representation of the basic element of the pattern and use it together with the geometric structure of the pattern to retrieve different images of the same object from a medium-sized image dataset. The detection task is formulated in Sec. 2, detection and representation of the repetitive pattern is described in Sec. 3, the match score calculation and retrieval is introduced in Sec. 4 and finally the method is evaluated by retrieval experiment in Sec. 5.

2 Repetitive Pattern Detection Task

Our task is to detect a set $C$ of all observable repetitive patterns $C_k$ in given image $I$. The objects in the pattern $C_k$ can be arranged on a regular lattice or periodically on a line. Frontoparallel image of one lattice cell is called tile. Figure 3 shows the structure of a $K \times L$ lattice formed by rectangular tiles of size $\Delta x \times \Delta y$ starting at $(x_0, y_0)$.

The ideal case, where the object is present at each tile and entire tile repeats itself without any variation between the tiles, is rare. Therefore, we allow incomplete lattice and incomplete repeating tile in our model:

$$\forall (k, l) \in S \subseteq \{0 \ldots K - 1\} \times \{0 \ldots L - 1\} \text{ (for subset of lattice tiles)}$$

$$\forall (\alpha, \beta) \in T \subseteq (0, 1)^2 \text{ (for repeating part of tile)}$$

$$P(x_0 + \alpha \Delta x, y_0 + \beta \Delta y) = P(x_0 + (k + \alpha) \Delta x, y_0 + (l + \beta) \Delta y), \quad (1)$$

where $P$ is the appearance of the pattern at a given point. The pattern $P$ is perspectively projected into an image $I$ by homography $H$. 

3
Figure 3: Structure of a $K \times L$ lattice. Appearance of the corresponding points $(\alpha, \beta)$ on each tile is the same.

## 3 Global Detection of Lattice Patterns

We assume that the tile of the repetitive pattern contains some distinguished feature, otherwise it would not be recognized even by humans. Detector of distinguished regions [6, 4] provides features $L_i$ in entire image, which are then grouped into clusters $C_k$ by shape and appearance represented as SIFT descriptor [3]). Each cluster is tested for presence of a lattice or line pattern. If such pattern is found, its perspective distortion is estimated, a square pattern tile extracted and the pattern is grown further by cross-correlation.

### 3.1 Clustering Interest Regions

The interest regions $L_i$ are detected by MSER detector [4] and blob detector [6]. The shape of each region is described by a local affine frames (LAF) [5] and its appearance is represented as a SIFT descriptor [3].

**Clustering by appearance.** For each region $L_i$, we find up to 512 of its nearest neighbors $N_i$ according to SIFT appearance, using kd-trees [7]. Only neighbors that are under SIFT distance threshold $t_{SIFT} = 250$ for MSER or $t_{SIFT} = 300$ for blobs or are accepted. The region with most neighbors creates first cluster $C_1$, taking its neighbors into the cluster and removing them from list. Iteratively, additional clusters $C_k$ are created using the next region with most clusters in the list as a center. If a cluster $C_k$ is to be created, but majority of the neighbors of the new cluster center are already
in another cluster $C_l$, the remaining regions of $C_k$ are also added into $C_l$.

Finding connections between tiles. Next, we want to find the connections between neighboring tiles in each cluster, as they are needed to uncover the geometric structure. The center of regions $L$ is considered as the center of the tile. Inside a cluster $C$, up to 12 near neighbors $N_{i,j}$ are found for each region $L_i \in C$, where $j \in \{1 \ldots 12\}$. The metric is image distance. The search for near neighbors is separated into four $90^\circ$ sectors, see Fig. 4, in each up to 3 near neighbors are found. The reason to use sectors is that in strongly deformed or elongated lattice we might get all nearest neighbors only in one direction, either horizontal or vertical. To correctly detect the lattice, we need connections in both directions. In each sector, the search is limited to 1.75 multiple of the distance of closest neighbor found in the given sector, to prevent finding connections that are just integer multiples of the base lattice vector. The neighbors are also tested for affine frame consistency – due to possible perspective distortion, the frame cannot be used for global clustering, but locally the frames of neighboring tiles can be matched.

![Figure 4: Sector search for near neighbors of a single region (a). Blue lines mark the sectors, blue arcs are the 1.75 limit of each sector. Red arrows point to the near neighbors found. Nearest neighbor connections between regions of one cluster are plotted in yellow (b).](image)

Finally, each cluster is treated as a graph with regions as vertices and near neighbor connection as edges and the graph is tested for presence of disjoint components. If multiple disjoint components exist in a single cluster, the cluster is divided.
3.2 Finding Geometric Structure of Cluster

In a lattice pattern, the tile connections in the direction of two lattice base vectors should prevail. The obstacle to finding the base vectors and thus the lattice is perspective distortion. Under perspective distortion, the tile connecting lines belonging to the same base vector do not have the same azimuth, only the same vanishing point.

**Test for 1D line pattern.** Before looking for vanishing points, we perform a test for a single line pattern in cluster $C$. The first cheap test is low variation in $L_iN_{i,j}$ connections azimuth, $L_i \in C$. Then RANSAC is used to find a line that fits at least 80% of cluster regions $L_i$. If such line is found, cluster $C$ is labeled as 1D line pattern.

**Finding vanishing points by Hough transform.** For each cluster $C$, we search for the vanishing points belonging to the lattice base vectors by Hough transform on the tile connections $L_iN_{i,j}$ themselves. For increased accuracy, we add chains of connections which lie on the same line – they are likely to form a lattice row or column and due to larger distance between endpoints are less prone to noise than connections between neighbor tiles. We use the parameterization of Hough space presented in [14] to decrease the resolution in large coordinate values, as vanishing points are often far from the image.

![Figure 5: Selected pair of vanishing points and supporting tile connections.](image)

**Selecting two vanishing points.** The Hough transform finds not only two vanishing points corresponding to the two base lattice vectors, but also vanishing points corresponding to the diagonal vectors. Note that due to the
perspective distortion the diagonal connection between tiles in image can be shorter than the base connection. Among the vanishing point candidates, we look for a pair that explains most of the connections $L_i, N_{ij}$. Then, the position of vanishing points is re-estimated by a least squares fit, using only inlier connections for each vanishing point candidate. Figure 5 shows an example of found pair of vanishing points and the tile connections corresponding to the pair.

**Lattice rectification.** Knowing the pair of vanishing points corresponding to the two main lattice directions of cluster $C$, the rectifying homography $H$ is estimated, which maps the lattice into frontoparallel projection with square tile, see Fig. 6. The positions of interest regions $L_i \in C$ are mapped by homography to $L'_i$.

![Pattern extension on rectified image](image)

Figure 6: Rectified image with the mean tile in the top-left corner. Blue crosses mark the tile centers, green squares are positions of distinguished regions $L_i$ at given tiles.
3.3 Extending and Merging the Patterns

We have to assume that some interest regions belonging to the pattern were not detected or that they were not assigned to the correct cluster. It would result in missing the tile, unless we find it by cross-correlation against representative tile of the pattern. We obtain the representative tile as a mean tile in the rectified image.

Calculating the mean tile. Calculating the mean tile directly from patches at positions $L'_i$ of the rectified image $I'$ would result in significantly blurred edges due to the noise in tile positions. Therefore, we refine the position of each tile by local cross-correlation against template tile, see Fig. 7, and only the tiles with cross-correlation above 0.7 are used to calculate the mean tile. The template tile is obtained as the best-correlated tile against all other tiles at initial positions $L'_i$.

Figure 7: Mean tile calculation. The top row shows all tiles of the cluster at positions $L'_i$, the bottom row shows refined tiles against the best correlated tile (left). The resulting mean tile is on right.

Extending into empty tiles. The empty tiles (i.e. without interest regions belonging to the cluster) inside the lattice and on its borders are tested for presence of the repetitive element by cross-correlation against the mean tile and added to the pattern if cross-correlation is above 0.7.

However, this extension is possible only when there is a sufficient variation in the intensity of the mean tile, i.e. when the mean tile is not homogenous.

3.4 Merging and Sorting the Patterns

Among the set of detected patterns, we can expect divided parts of a single pattern or weaker diagonal lattice patterns overlapping the base lattice pattern. To clean the output set of patterns, we merge the divided pattern parts and suppress weaker patterns occluded by stronger.
Merging patterns. In the clustering phase, some patterns could be split due to missing local connections. Now that the lattices are known, we check each pair of lattices for 1) overlap of their bounding boxes, 2) matching vanishing points and 3) lattice vectors of comparable size. If all three conditions are fulfilled, the patterns are merged.

Sorting patterns. The remaining patterns are sorted by the following criteria: 1) number of tiles $L_i$ and number of tiles added by cross-correlation; 2) completeness of the lattice connections $L_iN_{i,j}$ corresponding to the selected pair of vanishing points; 3) overlap with another larger pattern (taken as negative value).

3.5 Shift-Invariant Tile Representation

For matching and retrieval, we have to able to compare mean tiles $M$ between patterns. If the two compared patterns are images of the same real-world pattern, their mean tiles should be similar, except for possible 90, 180 or 270 degrees rotation, different scale and translation. The rotation ambiguity arises due to different ordering or sign of the two lattice vectors. The translation ambiguity can be caused by different choice of tile centers. A tile can contain multiple different interest regions and the tile center is set to the center of interest region which generates the largest cluster. In different images of the same pattern, the choice of the same interest region is not guaranteed, see Fig. 8.

![Figure 8: Tile shift-ambiguity and zero-phased shift-invariant mean tile. The same pattern can result in shifted lattice detections in different views (a,b) and thus shifted mean tiles (c). Our zero-phased mean tile representation is shift-invariant (d).](image)

**Fourier transform magnitude descriptor.** For the shift and scale invariant representation we can use a normalized vector $d$ of the first $10 \times 10$
real coefficients of discrete 2D Fourier transform of the grayscale tile \( F = \mathcal{F}(M) \), omitting the first coefficient (zero frequency).

**Fourier transform with zero phase of the first harmonic.** In an idea similar to [8], we adjusted the Fourier transform phase coefficients \( \phi[F] \) to reach zero for the phase of the first harmonic coefficients in both horizontal and vertical direction (\( \phi[F(1,0)] \) and \( \phi[F(0,1)] \))

\[
F^* = \mathcal{F}(M(m,n))e^{-j\phi[F(0,1)]}e^{-j\phi[F(1,0)]k}.
\]

The “zero-phased” tiles \( M^* = \mathcal{F}^{-1}(F^*) \) corresponding to the zero-phased descriptor \( d \) are shown in right column of Fig. 8.

The comparison of the zero-phased Fourier transform descriptor with the simpler Fourier transform magnitude descriptor ended with a tie as each prevailed on one dataset, see Fig. 14 for details. We believe that in some cases the Fourier transform magnitude descriptor might prove useful because of its higher robustness, albeit it is less descriptive. We also tested the description by SIFT descriptor of the zero-phase tile \( M^* \), however it showed the least successful, probably due to an increased sensitivity to inaccuracies in the tile size.

**Peaks in color RGB histogram.** The above mentioned descriptors take into account only image intensity. To take advantage of the color information, we also calculate and store two largest peaks in the color RGB histogram of the mean tile. However, it should be noted that our main test objects, facades, contain reflective surfaces and therefore color is only a minor cue.

## 4 Image Retrieval by Repetitive Patterns

We formulated the task of image retrieval as follows: for a given image query \( I_q \) (and set of its repetitive patterns \( C_q \)), retrieve from the dataset \( \mathcal{I} \) set of images \( \mathcal{R}_q \) containing objects most similar to some object in the query image. Figure 9 shows an example of successful retrieval.

The list of repetitive patterns \( C_q \) of the query image is compared to repetitive patterns \( C_i \), where \( i \in \{1 \ldots |\mathcal{I}|\} \), of each image in the dataset and match score \( S_{q,i} \) is calculated, see Sec. 4.1. The retrieved images are chosen as the ones with the highest match score.

We are aware that this naive each-to-each matching with complexity of a single query linear in the dataset size is acceptable only for a limited size of the dataset.
4.1 Match Score of an Image Pair

The match score $S_{ij}$ of an image pair $(I_i, I_j)$ is calculated solely from their repetitive patterns $(C_i, C_j)$, the images themselves are not used at this stage. For each pattern $C^i_k \in C_i$, we find the best matching pattern $C^j_l \in C_j$ with match score $s_k$, see Sec. 4.2 for calculation of the match score between two patterns. Figure 10 shows an example of matching sets of patterns from two images.

The matching score $s_k$ between patterns $C^i_k$ and $C^j_l$ is adjusted by ranking
of the patterns inside their respective lists $C_i, C_j$

$$s'_k = \left(1 - \frac{1}{2} \frac{k + l}{|C_i| + |C_j|}\right) s_k .$$  

(3)

This adjustment reflects the intention that large and strong patterns should have greater influence on the match score between images than small local patterns. The ranking inside lists can be used as a measure of the pattern strength because the pattern list in each image is sorted as described in Sec. 3.4.

A linear combination of the highest three pattern matches $s'_1, s'_2, s'_3 \in \{s'_k\}$ form the match score of an image pair $(I_i, I_j)$

$$S_{i,j} = 0.65s'_1 + 0.3s'_2 + 0.05s'_3$$  

(4)

where the coefficients were obtained as weights of linear classifier on feature vector $(s'_1, s'_2, s'_3)$ learned by SVM.

### 4.2 Match Score of two Repetitive Patterns

The match score $s_{k,l} = s_T s_C$ of two repetitive patterns $C_i^k$ and $C_j^l$ is a product of two measurements: similarity $s_T \in (0, 1)$ of grayscale mean tiles and similarity $s_C \in (0, 1)$ of color histograms.

**Tile similarity $s_T$.** The similarity between normalized vector descriptors of grayscale tiles is calculated as $s_T = 1 - |d_i^k - d_j^l|$, where $d_{i,k} = (r, g, b) \in \{(0, 1)^4, n$ being the relative count of pixels in the peak bin. The similarity formula has to take into account the possibility that the peaks are switched.

$$s_C = 1 - \frac{\min \left( |(p_{i,k,1}^j, p_{i,k,2}^j) - (p_{i,l,1}^j, p_{i,l,2}^j)|, |(p_{i,k,1}^j, p_{i,k,2}^j) - (p_{i,l,1}^j, p_{i,l,2}^j)| \right)}{\max \left( |(p_{i,k,1}^j, p_{i,k,2}^j) - (p_{i,l,1}^j, p_{i,l,2}^j)|, |(p_{i,k,1}^j, p_{i,k,2}^j) - (p_{i,l,1}^j, p_{i,l,2}^j)| \right)} .$$  

(5)

**Lattice size similarity $s_L$.** We also tested a third factor, lattice size similarity. Each lattice has its width and height expressed in a number of tiles as $l = (w, h)$. The similarity is computed as

$$s_L = 1 - \frac{1}{2} \left( \frac{|w_k - w_l|}{\max(w_k, w_l)} + \frac{|h_k - h_l|}{\max(h_k, h_l)} \right) .$$  

(6)

We neglect the possibility that the width and height are switched, because in the prevailing type of images in our test data – building facades – the orientation remains almost always constant (sky up).
In our experiments, we evaluated the influence of color similarity and lattice size similarity on retrieval success, see Fig. 11. The color similarity proved useful on one dataset while not hindering the performance on the second dataset. However, the lattice size similarity showed as misleading.

Figure 11: Influence of color and lattice similarity on the results on both tested datasets. ROC curve for match score calculated using both measures is red, the green curve shows result without the color similarity and the blue curve without the lattice similarity.

Our explanation of the failure of the lattice similarity measure is that in different views of the same pattern different parts of the lattice are often occluded or out of the image frame, which affects the lattice size and it is therefore not a reliable similarity measure.

In future, this match score calculated ad-hoc as a product of two measurements could be replaced by better justified score learned on a feature vector of multiple measurements, as we have done for the match score of an image pair. However, that would require pattern correspondence groundtruth, so far we have labels only on the image level.

5 Experiment

The method was tested on two dataset: 1) our dataset Pankrac+Marseilles with 106 images of app. 30 buildings appearing in more than one image, number of appearances ranges from 2 to 6; and 2) the building subset of near-regular texture dataset PSU-NRT [9], containing 117 images with over 20 buildings appearing in more than one image, number of appearances ranges from 2 to 26. Figure 12 shows all images from both datasets.

1dataset can be downloaded from http://cmp.felk.cvut.cz/~doub/SterIm
Figure 12: Images from Pankrac+Marseilles and PSU-NRT datasets.
For each image $I_i \in \mathcal{I}$, the groundtruth is labeled as a set of images $\mathcal{G}_i \subseteq \mathcal{I}$ that contain some object from $I$, i.e. $\mathcal{G}_i$ is the groundtruth response to the query by image $I_i$. The method is evaluated by comparing the response $\mathcal{R}_i$ against groundtruth $\mathcal{G}_i$. Detection rate on entire dataset equals to ratio of true positives to the size of the groundtruth response and false positive rate equals to ratio of false positives to number of all non-matching pairs

$$\text{detection rate} = \frac{\sum_i |\mathcal{R}_i \cap \mathcal{G}_i|}{\sum_i |\mathcal{G}_i|}, \quad \text{false positive rate} = \frac{\sum_i |\mathcal{R}_i \cap \bar{\mathcal{G}}_i|}{\sum_i |\bar{\mathcal{G}}_i|},$$

where $\bar{\mathcal{G}}_i = \mathcal{I} \setminus (\mathcal{G}_i \cup \{i\})$ is the complement of groundtruth response.

The response $\mathcal{R}_i$ is either set of $n$ images with the highest matching score $S_{i,j}$ or thresholded set $\mathcal{R}_i = \{I_j : S_{i,j} \geq \theta\}$, where $\theta$ is the threshold on the image match score. Figure 13 shows example responses $\mathcal{R}_i$ with three best matching images.

The trade-off between detection rate and false positive rate can be adjusted by $n$ or by $\theta$. Each value of these parameters gives us one point on the ROC curve in Fig. 14.

As expected, the retrieval by repetitive patterns performed better on the Pankrac+Marseilles dataset, where the average size of the tile image is larger and more details are observable. It corresponds to the purpose of these two datasets, authors of the PSU-NRT dataset used it solely for detection of repetitive patterns in single images, whereas our Pankrac+Marseilles dataset was created to test retrieval.

The average detection time by our Matlab implementation on 1000 $\times$ 700 image is 50 seconds, the time to run a single query on 106 images dataset is 1 second, increasing linearly with the size of the dataset.

6 Conclusion

We presented a method for detection of repetitive patterns with lattice and line structure aimed at using the detected patterns for image retrieval. The contribution of the paper lies in 1) representing the pattern by a shift-invariant tile that can be matched to tiles of the same pattern detected in different views and 2) demonstrating that this repetitive pattern representation can be used to retrieve images from a dataset. Our dataset used for testing is publicly available together with the groundtruth.

Although the retrieval results of our method alone would not be sufficient especially on larger datasets, the repetitive pattern matching can be used to boost performance of standard matching methods based on single image features.
Figure 13: Retrieval experiment – examples of successful queries. The query image in left column, followed by three best matches for each query.
The drawbacks of the detection method are its primary focus on lattice patterns and the top-down direction of search, unsuitable for high resolution images with many small patterns (such as aerial photographs).

References


