Learning Salient Structures for the Analysis of Symmetric Patterns

Jaime Lomeli-R. and Mark S. Nixon

University of Southampton, UK
jlr2g12@soton.ac.uk

Abstract. Feature-based symmetry detection algorithms have become popular amongst researchers due to their dominance in performance, nevertheless, these approaches are computationally demanding. Also they are reliant on the presence of matched features, therefore they benefit from the abundance of detected keypoints; this implies that a trade-off between performance and computation time must be found. In this paper both issues are addressed, the detection of large sets of keypoints and the computation time for feature-based symmetry detection algorithms. We present an innovative process to learn rotation-invariant salient structures by clustering self-similarities. Keypoints are detected as local maxima in feature-maps computed using the learnt structures. Keypoints are described using BRISK. We consider an axis of symmetry to be a dense cloud of points in a parameter-space, a density-based clustering algorithm is used to find such clouds. Computing times are drastically shortened taking an average of 0.619 seconds to process an image. Detection results for single and multiple, straight and curved, reflection and glide-reflection symmetries are similar to the current state of the art.

Keywords: Feature detection, Moments, Matching, Symmetry.

1 Introduction

Symmetry is a pervasive cue in both natural and artificial structures. The automated detection of symmetry finds applications in areas such as interest-point detection [7, 21], image segmentation [8, 9], saliency models [19, 22] and medical image analysis [5, 6] amongst others.

In 2D there are four recognisable *primitive symmetries* namely, reflection, rotation, translation and glide-reflection [20]. Computer vision tackles the detection of these symmetries in a variety of ways including correlation-based [18], feature-based [16, 3] and machine-learning aided methods [4]. In [10], results of reflection, rotation and translation symmetry detection algorithms are shown. Reflection symmetry detection in the literature concentrates predominantly in straight and skewed reflections, few aim at the detection of curved axes; arguably the feature-based approach is the most popular due to its dominance in performance. Yet, the study of the glide-reflection group has not received much attention despite its abundant existence in real-world scenarios.

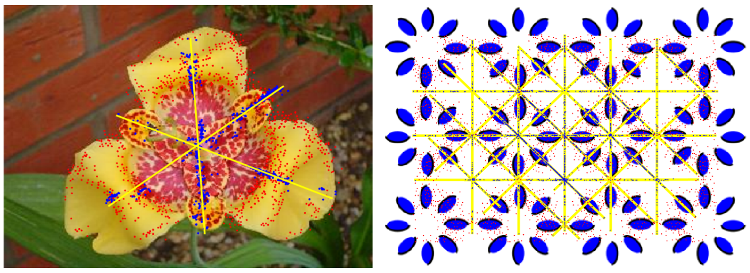
Fig. 1. Reflection and glide-reflection example results of the presented method. Yellow indicates the detected axes. Our method finds dense sets of keypoints even in low-textured images (image on the right).

Feature-based approaches begin by extracting keypoints in the image. These points are then described, mirrored and compared to find local similarities, each successful matching pair of keypoints votes for an axis of symmetry. This voting process benefits from large amounts of matches thus, large sets of detected keypoints are crucial. Lee and Liu highlight this problem, their method detects keypoints in several transformed versions of the image to generate a denser sampling [12]. Manual selection of image transformations requires the implementation of different operators and deep knowledge of various fields.

We address this problem with a novel keypoint detection method that clusters self-similarities. The central idea is to let the computer learn repetitive visual salient structures in a rotation-invariant space. The learning scheme can be depicted as a three step process; randomly extracting image patches from unlabelled data, normalisation, and the use of an unsupervised learning algorithm to learn *keypoint models*. Each keypoint model is used to create a feature-map in which non-maxima suppression can be used to detect keypoints. The detected sets of keypoints are plentiful even in low textured images (figure 1).

Extracted keypoints are described with a modified BRISK [13] pattern that can be efficiently mirrored with a look-up table. Successful keypoint matches are mapped into a parameter-space, where axes of reflection and glide-reflection symmetry are recognised as dense clouds of points that are rapidly detected using a density-based clustering algorithm.

Larger sets of keypoints require more computation time, therefore a trade-off between performance and time must be found. The set of techniques we use (figure 2), dramatically reduce the computing time in each step of the symmetry detection process. Our approach takes an average of 0.619 seconds to detect one or many axes of symmetry in an image (6% of the time reported in [12], calculated over a set of 64 images).

We show our results on a meaningful set of images widely used in the symmetry detection community. This set contains real-world and synthetic images with single and multiple, curved and straight and, reflection and glide-reflection symmetries. Our results are comparable with the state-of-the-art, while maintaining small computation times.

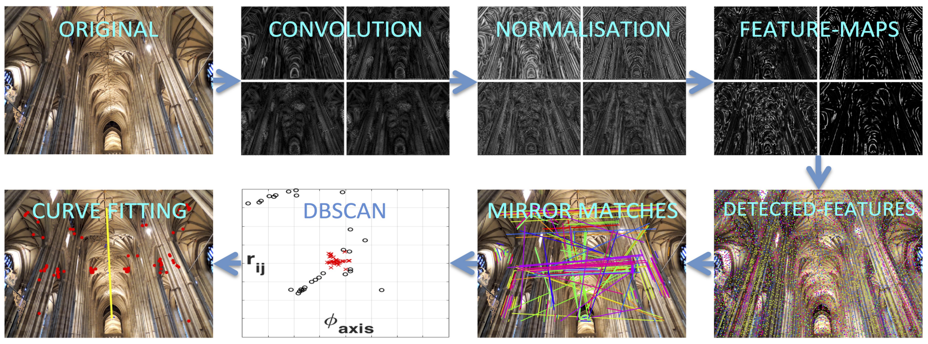

Fig. 2. The image is convolved with M Bessel-Fourier filters, structural-normalisation is computed on the complex modulus of the response of such convolution (as explained in Section 2). The probability of each pixel to belong to each of the pre-learnt keypoint models is calculated using equation 5, K sets of keypoints are extracted using non-maxima suppression in the resulting feature-maps. We compute and mirror the BRISK descriptors and compare all possible pairs of keypoints that belong to the same keypoint class (same k). Each successful match is transformed into a point in the parameter-space, where a density-based clustering algorithm is used to find dense point-clouds. Polynomial regression is used on each of the clusters found in the previous step to calculate the optimal axes of symmetry.

2 Learning Salient Structures

Keypoints are commonly recognised as local-maxima of a similarity measure on an image. For instance, convolving an image with a *Laplacian of Gaussian* kernel one can find blob-like structures; another example is the task of edge detection, in which a gradient operator is used. Let us analyse such filters in terms of the Fourier transform in cylindrical coordinates.

$$H_m(\rho) = \int_0^{2\pi} \int_0^{\infty} J_m(2\pi r\rho) e^{-im\theta} f(r,\theta) r \delta r \delta\theta, \tag{1}$$

$$F(\rho,\gamma) = \sum_{m=-\infty}^{\infty} (-i)^m H_m(\rho) e^{im\gamma}. \tag{2}$$

Equation 2 shows the Fourier transform of $f(r,\theta)$ in cylindrical coordinates. The function $H_m(\rho)$ is known as the *Hankel transform* of integer order m and, $J_m(2\pi r\rho)$ is the *Bessel function* of the first kind of integer order m (equation 1).

The family of functions $H_m(\rho)$ are the *Fourier coefficients* of the function $f(r,\theta)$ in the continuous domain of ρ. This family of functions is discretized by setting the variable $\rho = \lambda_{m,n}/2\pi$, where $\lambda_{m,n}$ are the roots of the Bessel function (corresponding to order m and scaling n). The integration range of r is reduced to obtain the set $B_{m,n}$ in equation 3.

$$B_{m,n} = \int_0^{2\pi} \int_0^{1} J_m(\lambda_{m,n} r) e^{-im\theta} f(r,\theta) r \delta r \delta\theta. \tag{3}$$

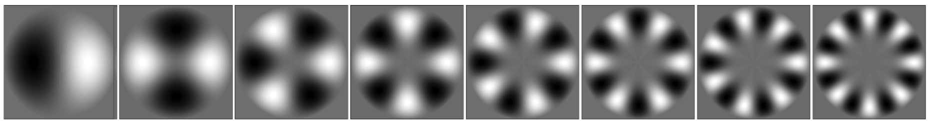

Fig. 3. The real part of the BF moment-generating functions for $n = 0$ and $m = 1, 2, \ldots, 8$.

$B_{m,n}$ is known as the (complex) *Bessel-Fourier* (BF) moment-generating function [14] (orthogonal in the range $0 \leq r \leq 1$). Blob detection operators are filters that have non-zero magnitude in $B_{0,n}$ and, commonly used gradient operators respond to the set $B_{1,n}$. The variable m represents the rotational frequency, and n represents the radially *expanding* waves around the origin (or a given point if the BF functions are used as filters).

We empirically set $n = 0$ and $m = 1, 2, \ldots, M$ where $M = 8$ (figure 3). An image $I(x, y)$ is convolved with such M BF-filters at a single scale with a kernel size $kernSize = 0.04 \times max(I.height, I.width)$, then we obtain the magnitude of the complex responses. There are M functions $I_m(x, y)$ thus, each pixel is represented by an M-dimensional vector. This vector is known as the *rotational power spectrum*, we treat it as function of the image space *i.e.* $RPS(x, y)$. The RPS of each pixel is normalised so the sum of all M dimensions sums up to one (Equation 4, $|\boldsymbol{x}|_1$ denotes the L^1-norm). We call this process *local structural normalisation* as it is independent from the location of the pixel.

$$RPS_{norm}(x_i, y_j) = \frac{RPS(x_i, y_j)}{|RPS(x_i, y_j)|_1}. \tag{4}$$

In practice, all elements of $RPS_{norm}(x_i, y_j)$ are set to 0 when a threshold is not met *i.e.* $\{|RPS(x_i, y_j)|_1 < RPS_{thresh}\}$, this is to avoid detecting features in regions of low energy or prevent division by 0 in its case. Low values of RPS_{thresh} highlight weaker responses, while large values discard detections with low contrast. Any visual cue can be recognised as a keypoint. After the local structural normalisation process, each pixel is a point $RPS_{norm}(x_i, y_j)$ in an M-dimensional simplex and pixels that are visually similar are close to each other in this space.

The probability density function (PDF) of such space describes repeatability, clustering algorithms find local-maxima on the PDF of a discrete random variable. To find repeatable structures, we randomly build set of 10^5 RPS_{norm}'s from various images and run K-Means clustering to find K centres (we set $K = 10$ for the results shown here). During the training process we replace the sample $RPS_{norm}(x_i, y_j)$ for a different one if the condition $\{|RPS(x_i, y_j)|_1 < RPS_{thresh}\}$ is met. The clustering process can be performed off-line, the location of the pixels and the image they came from is not accounted for. The found centres are known as *keypoint models* (denoted as C_k). Centres lie in regions of high-density, this means that visual self-similarities get clustered together. The training process is rapid, usually converging after only eight iterations.

We calculate $RPS_{norm}(x, y)$, we then use equation 5 to calculate the probability for each pixel to belong to each of the pre-learnt keypoint models. This creates a set of K rotation-invariant feature-maps $0 \leq M_k(x, y) \leq 1$ in which non-maxima suppression can be performed to detect keypoints (figure 4).

$$M_k(x_i, y_j) = \frac{|RPS_{norm}(x_i, y_j) - C_k|_2^{-4}}{\sum_{k'=1}^{K} |RPS_{norm}(x_i, y_j) - C_{k'}|_2^{-4}}. \tag{5}$$

The operator $|\boldsymbol{x}|_2$ denotes the L^2-norm. Keypoints will be denoted henceforth as P_i^k where i is the index of the keypoint and k is the feature-map in which it was found. Keypoints found on the same feature-map are said to be of the same class.

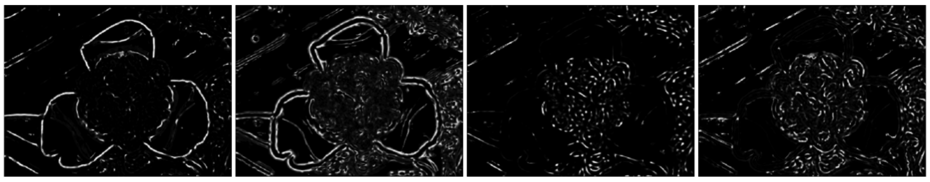

Fig. 4. Six (of the ten) feature-maps created from the image on the left in figure 1. Learnt keypoint models represent different structures, for instance the leftmost feature-map has a good response to step-edges and the two rightmost feature-maps respond to highly textured regions on the image.

3 Matching Mirrored Keypoints

We use a BRISK descriptor [13] to compare keypoints across the image and find local similarities, this descriptor is much faster to calculate than SIFT or SURF [1]. BRISK samples the image around a keypoint with a circular pattern. Samples are compared to each other, if a sample a is brighter than sample b the comparison result is a logic 1 or 0 otherwise. The orientation of the keypoint is calculated using the shorter comparisons, the longer ones form the descriptor.

We use a descriptor of 512 comparisons $\{d_p \mid p = 0, 1, \ldots, 511\}$ which can be stored in char arrays of size $512/8 = 64$ bytes. The traditional BRISK pattern is slightly modified, we make the pattern symmetric across the orientation of the keypoint and index it in such a way that d_p is the mirror pairing across the keypoint orientation of d_{511-p}. In this manner, we can bit-swap each byte with a look-up table and then swap the order of the array.

The Hamming distance between two binary descriptors can be calculated with an XOR operation and a bit-count. We calculate the distance between all possible pairs of features of the same class, and use only the closest match. A successful matching pair $P_{ij}^k = (P_i^k, P_j^k)$ exists if the Hamming distance between the descriptors of P_i^k and P_j^k is less than a threshold, we empirically choose $BRISK_{thresh} = 120$.

4 Parameter-Space Analysis

A parameter-space is created from successful matches. We use a two-dimensional parameter-space with coordinates (ϕ_{axis}, r_{ij}) calculated with equations 6, 7 and 8 (figure 5). In this 2D parameter-space, pure reflection symmetry axes are single points whilst curved reflection axes are smooth contours. As we use a single scale, a keypoint is represented by a location and an orientation *i.e.* $P_i^k = (x_i, y_i, \phi_i)$.

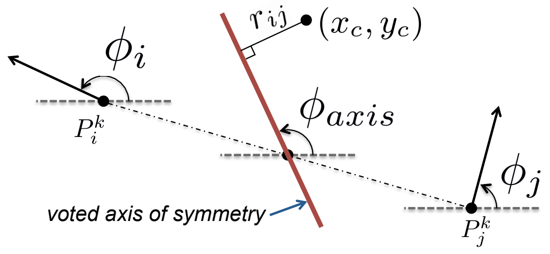

Fig. 5. Every matched pair of features $P_{ij}^k = (P_i^k, P_j^k)$, is mapped to a point in the parameter-space with coordinates (ϕ_{axis}, r_{ij}).

The angle in equation 6 is defined in the range $\{-\pi \leq \phi'_{axis} < \pi\}$. As we only care for the angle of the axis and disregard the direction in which it points, we use equation 7 to re-bound this parameter in $\{0 \leq \phi_{axis} < \pi\}$. The second coordinate of the parameter-space is the perpendicular distance between the axis and the centre of the image with coordinates (x_c, y_c).

$$\phi'_{axis} = \frac{\phi_i + \phi_j}{2}. \tag{6}$$

$$\phi_{axis} = \begin{cases} \phi'_{axis} & \text{if } 0 \leq \phi'_{axis} \\ \phi'_{axis} + \pi & \text{if } \phi'_{axis} < 0 \end{cases}. \tag{7}$$

$$r_{ij} = \left(\frac{x_i + x_j}{2} - x_c\right) sin(\phi_{axis}) - \left(\frac{y_i + y_j}{2} - y_c\right) cos(\phi_{axis}). \tag{8}$$

In other feature-based symmetry detection approaches, each sample casts a vote in an accumulator matrix. This accumulator matrix is a discrete approximation of a density map of the parameter-space where points of local-maxima represent axes of symmetry. To be able to detect such points, the accumulator matrix has to be smoothed and then a non-maxima suppression algorithm must be used. The set of successful matches that generated a point of high-density represent an axis of symmetry, this procedure is appropriate for straight axes as these are single points in the parameter-space. On the other hand, curved axes of symmetry are contours in the parameter-space; therefore the single-point detection approach is no longer suitable. We consider an axis of symmetry to be a dense cloud of points in the parameter-space.

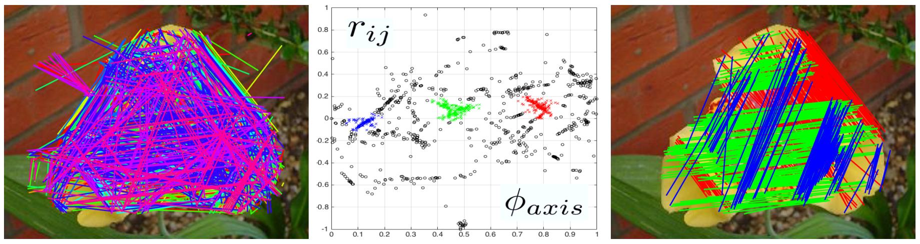

Fig. 6. On the left, all successful keypoint matches; each colour represents a different keypoint class. After DBSCAN (middle), matches voting for axes of symmetry are easily identified (right).

We use a density-based clustering algorithm called *DBSCAN* [15] to detect dense clouds of points. The parameter-space is normalised in the range $\{0 \leq \phi_{axis} \leq 1, -1 \leq r_{ij} \leq 1\}$ as the clustering relies on the Euclidean distance of the points. DBSCAN takes two parameters, ϵ indicates how densely clustered are the points and $MinPts$ indicates the minimum number of density-reachable points for a cluster to exist. We set these parameters empirically as $\epsilon = 0.2^2$ and $MinPts = 50$, although these values may change depending on the application.

DBSCAN does not need to know the number of clusters to be found, this property is useful as the number of axes is usually unknown. It also discards outliers, allowing larger values of $BRISK_{thresh}$ to increase the number of matches; this improves the sampling of the parameter space even further. The middle image in figure 6, shows clustering results for the multiple axes detection task on the original image in figure 1. The other two images in figure 6, show the matched features before and after clustering.

5 Optimal Axes

Each of the L detected clusters is a subset AX_l of the set of all successful matches. As in [12], to find each optimal axis of symmetry we perform twenty polynomial regressions over the middle points of the matching pairs $P^k_{ij} \in AX_l$ (calculated with equation 9). Each polynomial regression is performed with degree $d = 1, 2, 3, 4, 5$ and rotation $r = 0°, 45°, 90°, 135°$. To calculate the regression in different angles we rotate the middle points using the rotation matrix in equation 10. Out of the twenty possible axes, we consider as optimal the one with the smallest root-mean-square error.

$$\begin{bmatrix} P^k_{ij,x} \\ P^k_{ij,y} \end{bmatrix} = \begin{bmatrix} \frac{x_i+x_j}{2} \\ \frac{y_i+y_j}{2} \end{bmatrix} . \tag{9}$$

$$\begin{bmatrix} P^k_{ij,x,r} \\ P^k_{ij,y,r} \end{bmatrix} = \begin{bmatrix} cos(r) & -sin(r) \\ sin(r) & cos(r) \end{bmatrix} \begin{bmatrix} P^k_{ij,x} \\ P^k_{ij,y} \end{bmatrix} . \tag{10}$$

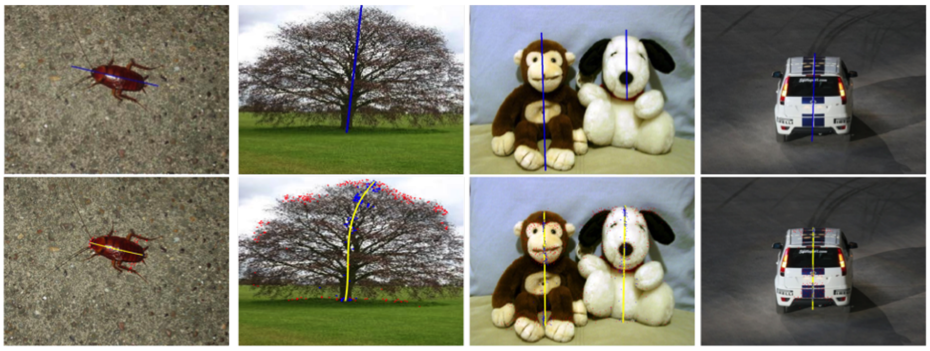

Fig. 7. Results of our approach (bottom), compared against ground-truth marked images (top).

6 Results

The percentages in table 1 are the true positive rates compared against human annotated ground truth on the 64 images of the PAMI2012 dataset [12]. Results using both SIFT and our method for keypoint detection are shown. Our approach compares to the state-of-the-art using only 6% of the computation time. We provide result in images commonly used in the symmetry detection literature in figure 8, this is to give the reader a means to visually compare our results with those obtained by other authors. Figure 7 shows our results alongside ground truth annotations from [10]. Our code is available at https://github.com/jimmylomro/symmetry.

Table 1. Results comparing our approach to those showed by Lee and Liu in [12].

Method	**Symmetry** True Positive Rate (# False Positives)				**Timings**
	Straight	Straight Glide	Curved	Curved Glide	mean(std) secs
proposed (SIFT)	81%(6)	40%(7)	62%(5)	40%(6)	0.332(0.06)
proposed	91%(4)	80%(5)	78%(4)	60%(6)	0.619(0.18)
Lee [12]	86%(3)	80%(3)	83%(3)	60%(4)	9.7(10.3)
Loy [11]	91%(9)	7%(46)	0%(38)	0%(26)	6.1(9.4)

7 Conclusion

We have presented a new procedure for detecting symmetric structures in images. The detection of symmetry benefits from the plentiful sets of keypoints provided by our method, as the parameter-space is more densely sampled. We consider axes of symmetry as dense clouds of points in the parameter space, allowing more flexibility when detecting curved axes.

It was also shown that it is possible to drastically shorten computing times while maintaining similar detection results. The computation times can be further improved. For example, equation 5 is an evaluation of the PDF of the keypoint model C_k generated by all other keypoint models and its time complexity is $O(K^2)$; it is possible to use a different non-linear distance function that can be calculated in $O(K)$. The time complexity of DBSCAN is $O(N_m log(N_m))$ where N_m is the number of matching pairs. Compared to the constant complexity of the Hough transform approach, DBSCAN is better when there is little symmetry in the image *i.e.* few matching pairs.

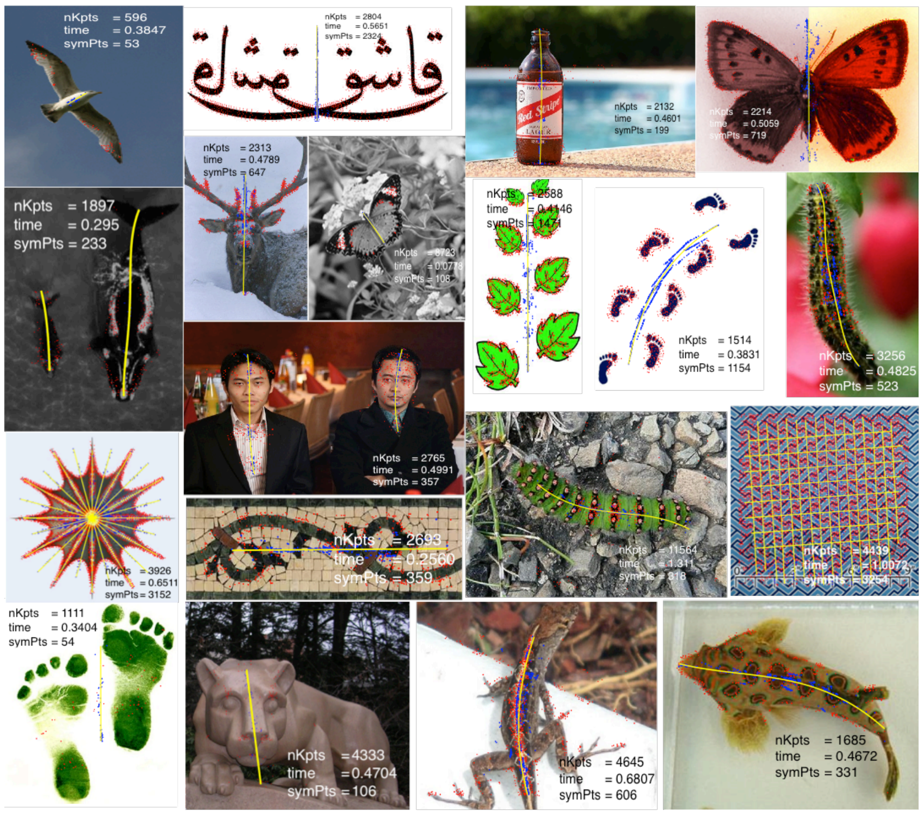
nKpts = 596 time = 0.3847 symPts = 53
nKpts = 2804 time = 0.5651 symPts = 2324
nKpts = 2132 time = 0.4601 symPts = 199
nKpts = 2214 time = 0.5059 symPts = 719
nKpts = 2313 time = 0.4789 symPts = 647
nKpts = 1897 time = 0.295 symPts = 233
nKpts = 6723 time = 0.0778 symPts = 108
nKpts = 2588 time = 0.4146 symPts = 1474
nKpts = 1514 time = 0.3831 symPts = 1154
nKpts = 3256 time = 0.4825 symPts = 523
nKpts = 2765 time = 0.4991 symPts = 357
nKpts = 3926 time = 0.6511 symPts = 3152
nKpts = 2693 time = 0.2560 symPts = 359
nKpts = 11564 time = 1.311 symPts = 318
nKpts = 4439 time = 1.0072 symPts = 3254
nKpts = 1111 time = 0.3404 symPts = 54
nKpts = 4333 time = 0.4704 symPts = 106
nKpts = 4645 time = 0.6807 symPts = 606
nKpts = 1685 time = 0.4672 symPts = 331

Fig. 8. Single and multiple axes detected on images containing one or more of the reflection, curved reflection and glide reflection symmetry groups. Each image shows the total number of keypoints detected (*nKpts*), the runtime of the algorithm (*time*) and the number of matches supporting the axes of symmetry (*symPts*). For wallpaper images DBSCAN uses $minPts = 15$, the default parameters are used in all other cases. Images were obtained from [12, 10, 11, 17, 18].

References

1. Heinly, J., Dunn, E., and Frahm, JM.: Comparative evaluation of binary features. In ECCV 2012.
2. Lee, S., Collins, RT., and Liu, Y.: Rotation symmetry group detection via frequency analysis of frieze-expansions. In CVPR 2008.
3. Wang, Z., Tang, Z., and Zhang, X.: Reflection symmetry detection using locally affine invariant edge correspondence. In IEEE Transactions on Image Processing (2015).
4. Brachmann, A., and Redies, C.: Using Convolutional Neural Network Filters to Measure Left-Right Mirror Symmetry in Images. Symmetry (2016).
5. Yu, CP., Ruppert, GCS., Nguyen, DTD., Falcao, AX. and Liu, Y.: Statistical Asymmetry-based Brain Tumor Segmentation from 3D MR Images. In Biosignals (2012)
6. Mancas, M., Gosselin, B., Macq, B., *et al.* : Fast and automatic tumoral area localisation using symmetry. In ICASSP (2005)
7. Salti, S., Lanza, A. and Di Stefano, L.,: Keypoints from symmetries by wave propagation. In CVPR (2013)
8. Levinshtein, A., Dickinson, SJ., and Sminchisescu, C.: Multiscale symmetric part detection and grouping. In ICCV (2009)
9. Teo, CL., Fermuller, C., and Aloimonos, Y.: Detection and segmentation of 2D curved reflection symmetric structures. In ICCV (2015)
10. Liu, J., Slota, G., Zheng, G., Wu, Z., Park, M., Lee, S., Rauschert, I., and Liu, Y.,: Symmetry detection from real-world images competition 2013: Summary and results. In CVPR Workshops (2013)
11. Loy, G., and Eklundh, JO.: Detecting symmetry and symmetric constellations of features. In ECCV (2006)
12. Lee, S., and Liu, Y.: Curved glide-reflection symmetry detection. In TPAMI (2012)
13. Leutenegger, S., Chli, M., and Siegwart, RY.: BRISK: Binary robust invariant scalable keypoints. In ICCV (2011)
14. Xiao, B., Ma, JF., and Wang, X.: Image analysis by Bessel–Fourier moments. In Pattern Recognition (2010)
15. Ester, M., Kriegel, HP., Sander, J., Xu, X., *et al.* : A density-based algorithm for discovering clusters in large spatial databases with noise. In KDD (1996)
16. Patraucean, V., Grompone von Gioi, R., and Ovsjanikov, M.: Detection of mirror-symmetric image patches. In CVPR Workshops (2013)
17. Kovesi, P., *et al.* : Symmetry and asymmetry from local phase. In Tenth Australian joint conference on artificial intelligence (1997)
18. Lee, S., Collins, RT., and Liu, Y.: Rotation symmetry group detection via frequency analysis of frieze-expansions. In CVPR (2008)
19. Kootstra, G., Nederveen, A., and De Boer, B.: Paying attention to symmetry. In BMVC (2008)
20. Liu, Y., Hel-Or, H., and Kaplan, CS.: Computational symmetry in computer vision and computer graphics. Now Publishers Inc. (2010)
21. Hauagge, DC., and Snavely, N.: Image matching using local symmetry features. In CVPR (2012)
22. Kootstra, G., and Schomaker, LRB.: Prediction of human eye fixations using symmetry. In CogSci (2009)