First Symmetry Detection Competition: Summary and Results

Ingmar Rauschert, Kyle Brocklehurst , Somesh Kashyap**, Jingchen Liu and Yanxi Liu
Department of Computer Science and Engineering
The Pennsylvania State University
University Park, PA, 16802
** Blue Belt Technologies
Contact Person: yanxi@cse.psu.edu

CSE Dept Technical Report No. CSE11-012, October, 2011

Abstract

Symmetry is a pervasive phenomenon presenting itself in all forms and scales in natural and manmade environments and its detection plays an essential role at all levels of human as well as machine perception. The recent resurging interest in computational symmetry for computer vision and computer graphics applications has motivated us to conduct an US NSF funded symmetry detection algorithm competition as a workshop during 2011 Computer Vision and Pattern Recognition Conference. This competition sets the first benchmark in computer vision history for symmetry detection algorithms. In this report we explain the evaluation metric and the automatic execution of such evaluation workflow; we present and analyze the algorithms submitted, and show their results on three sets of real world images depicting reflection, rotation and translation symmetries respectively. This competition establishes a performance baseline for future work on symmetry detection. .

Keywords

Reflection, Rotation, Translation Symmetry, Competition, Performance Evaluation, Frieze Pattern, Wallpaper pattern, Lattice Detection, Benchmark, Groundtruth

1 Introduction

In the arts and sciences, as well as in our daily lives, symmetry has made a profound and lasting impact. Likewise, a computational treatment of symmetry and group theory (the ultimate mathematical formalization of symmetry) has the potential to play an important role in computational sciences. Although seeking symmetry from digital data has been attempted for over four decades, a fully automated symmetry-savvy recognition system still remains a challenge for real world applications. However, the recent resurging interests in computational symmetry for computer vision and computer graphics applications have shown promising results. Recognizing the fundamental relevance and potential power that computational symmetry affords, we conducted a survey of the current state of the arts symmetry detection algorithms and performed a first quantitative benchmark on a diverse set of real world images [1,6,7], the first of a proposed set of three benchmarks.

In this report we present the results of the first symmetry detection competition in computer vision, which we shared at a dedicated workshop at the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2011 in Colorado Springs, Colorado. The competition was divided into three parts, each focusing on one of the three types of symmetries: reflection, rotation and translation respectively. For each symmetry category, we collected images depicting objects with representative symmetry features. In order to minimize bias towards specific symmetries, we also obtained a large variety of symmetry images from professional and amateur photographers who signed up and submitted images to our Flickr photo sharing website [2]. We collected a total of four training and six test sets, totaling 124 images (see Figure 1).

Figure 1: Four of the six image data sets are shown. (Top Left) Reflection Symmetry, (Top Right) Rotation Symmetry, (Bottom Right) Translation Symmetry (Urban Buildings), (Bottom Left) Translation Symmetry

Each image was annotated, yielding a total of 167 reflection and rotation symmetries and over two thousand wallpaper tiles for translation symmetry. During the annotation phase we identified and specified several ambiguities that can arise during the labeling process. In all cases of ambiguities a tradeoff between local and global context had to be made before it could be resolved. Because this trade of is in almost all cases subjective in nature, we have marked such symmetries as ambiguous and discounted any detections by the tested algorithms.

We received eight submissions for symmetry detection, three for reflection, two for rotation and three for translation symmetry. Adding one baseline algorithm to each symmetry group for comparison, we evaluated a total of eleven algorithms. The evaluation process was completely automated, counting the number of correct detections (true positives) and the number of incorrect detections (false positives). Detection performance is then reported in terms of precision and recall rates., for which the later only considers the number of true positives versus the total number of symmetries present, while the former also considers and penalizes for the number of false positives.

The quite remarkable outcome of all evaluations is that overall, none of the submitted algorithms performed better than their respective baseline algorithm (including execution speed of the algorithms). In comparison, tested algorithms performed best on reflection symmetry, second best on rotation symmetry and last translation symmetry. However, looking at sub categories of reflection symmetries, we found that one algorithm performed superior to all other tested algorithms on real images with multiple reflection axis present (judged on recall rate). A different algorithm outperforms all other algorithms on real and synthetic images depicting only a single reflection symmetry, when the number false positives has been taken into account (judged on precision rate).

We established a testbed for the evaluation of symmetry detection algorithms, devised evaluation metrics and automated the evaluation process. We tested our process on eleven algorithms and established a performance baseline that can be used as reference for future work on symmetry detection. .

2 Data Sets and Annotation

In this section we provide information on the collected dataset, the employed method for annotation and how it will be used in the evaluation process.

2.1 Data Collection

The competition was divided into three parts, each focusing on one of three types of symmetries: reflection, rotation and translation respectively. For each symmetry category, we collected images depicting objects with representative symmetry features from personal photos sets of our students. In order to minimize bias towards specific symmetries, we also obtained a large variety of symmetry images from professional and amateur photographers who signed up and submitted images to our Flickr photosharing website [5]. We collected a total of four training and six test sets, totaling 124 images. Figure 1 illustrates four of the six test image sets.

2.2 Categorization of Image Data Sets

For each symmetry type we split the obtained datasets into a number of relevant sub categories. For example, we split al 30 images of the reflection symmetry test set into four groups: images depicting synthetic versus real images and images depicting a single vs. multiple reflection symmetries. The purpose of this categorization is to evaluate and identify strength or weaknesses of the tested algorithms on specific image categories. The individual number of images and symmetries in each category can be extracted from Table 1.

For rotation symmetry we replaced the categories *real* and *synthetic* with *discrete*, *continuous* and *deformed* symmetry elements. The purpose here is test for performance differences amongst images with discrete symmetry elements such as the pedals of a flower or the spikes of a car wheel, versus images with no discernable symmetry elements, such as a smooth ring. We also test images with deformed symmetry objects, in which the object is partially occluded or affected by some affine or perspective distortion. We expect algorithms to perform best on images with discrete symmetries and worst on images with deformed symmetry objects. Examples and the number of images and symmetries in each category are given in Table 2.

For translation symmetry we spilt the dataset into three categories: easy, medium and hard, each of which holds various challenges to the algorithm. The easy category holds images with clearly visible wallpaper structure and only mild affine distortions. The medium category holds images with strong wallpaper structure but more severe affine and perspective distortions, as well as irregular structures (e.g. wrinkles on clothing). The hard category is most challenging due to the presence of strong and distracting background clutter, which in some cases is unstructured and in some cases is itself

Table 1: The dataset for reflection symmetry is divided into four categories, synthetic vs. real images, and images with single vs. multiple reflection axis. We use a total of 30 images with a combined count of 66 reflection axis.

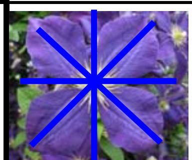
Reflection Symmetry Image Categories & Number of Symmetries								
Category	Single			Multiple			Total	
		#Imgs	#Syms		#Imgs	#Syms	#Imgs	#Syms
Synthetic		8	8		7	30	15	38
Real		6	6		9	22	15	28
Total		14	14		16	52	30	66

Table 2: The dataset for rotation symmetry is divided into six categories, images with discrete vs. continuous symmetry elements, deformed symmetry objects, and images with single vs. multiple reflection axis. We use a total of 40 images with a combined count of 81 rotation symmetries.

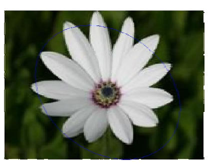
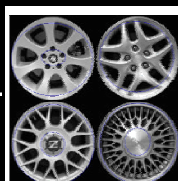
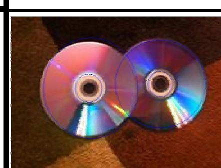
Rotation Symmetry Image Categories & Number of Symmetries								
Category	Single			Multiple			Total	
		#Imgs	#Syms		#Imgs	#Syms	#Imgs	#Syms
Discrete		11	11		3	16	14	27
Continuous		10	10		5	25	15	35
Deformed		7	7		4	12	11	19
Total		28	28		12	53	40	81

Table 3: The dataset for translation symmetry is divided into three categories, easy, medium and hard. We use a total of 31 images with a combined count over 2000 wallpaper tiles.

Translation Symmetry Image Categories & Number of Symmetries							
Easy		Medium		Hard		Total	
#Images	#Texels	#Images	# Texels	#Images	# Texels	#Images	#Texels
11		10		10		31	
							

structured in form of translational symmetry. Examples and the number of images and wallpaper texels in each category are given in Table 3.

2.3 Groundtruth Annotation

Employing the manual power of 30 students from our course on “Symmetry for Image Processing” at Penn State, each image was annotated using annotation programs, developed specifically for this purpose, and yielding a total of 167 reflection and rotation symmetries and over two thousand wallpaper tiles for translation symmetry. An example of annotation labels for each of the three symmetry groups is shown in Figure 2.

Reflection symmetry axis are marked as a line with a start point p1=(x1,y1) and an end point p2=(x2,y2). The length of the line covers the respective support region of the annotated symmetry. For rotation symmetry an ellipse is defined that covers the maximal support region, with center point c=(cx,cy), major and minor axis length L=(a,b) and the orientation θ of major axis with respect to the image x-axis. For translation symmetry, a lattice is defined with a start point P=(x,y) and a vector field T (two vectors for each tile), and each tile represents the texel in a wallpaper pattern.

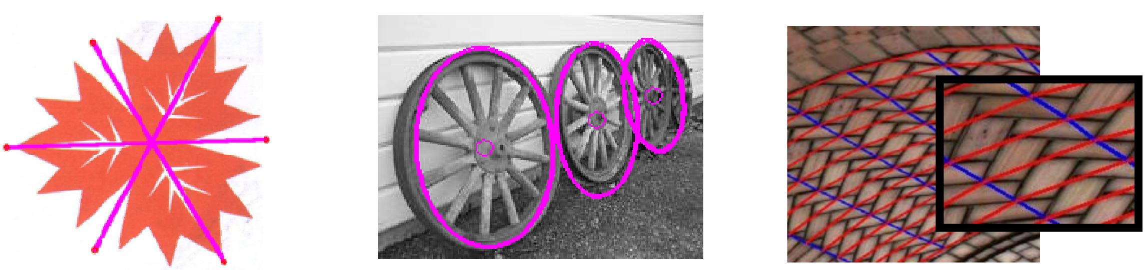


Figure 2: Images with annotation labels for (Left) reflection, (Mid) rotation and (Right) translation symmetry.

2.4 Annotation Ambiguities

During the annotation phase we identified a number of ambiguities that can arise during the labeling process. In all cases of ambiguities a tradeoff between local and global context seems to play a major role in deciding how the ambiguity can be resolved. Here, we give two examples from reflection symmetry that highlight ambiguities caused by scale of context and object deformations:

- Hierarchical Ambiguity
- Shape Ambiguity

Looking at hierarchical ambiguity, we refer to Figure 3. Symmetry is defined as a transformation g of a set of points S such that g(S) = S. Traditionally S represents the entire set of points, or in the case of a 2D space, the entire image. Given such a global definition of symmetry, only few true symmetries can be defined. However, to the human eye many more symmetries appear when viewed on a local rather than global scale.

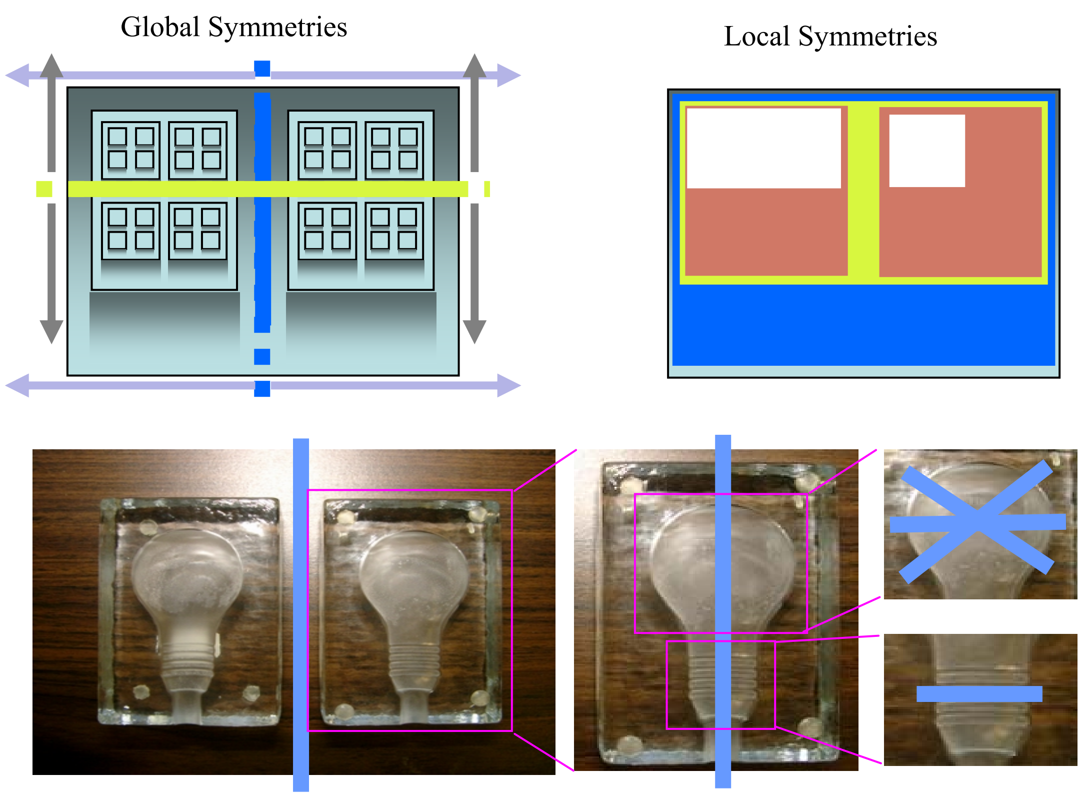


Figure 3: Hierarchical Ambiguity of reflection symmetry. (Top Left) Without local context, symmetry is defined over the entire image. (Top Right) When using a subset of the 2D plane (local context) many different reflection symmetries can be defined. (Bottom) An example of scale dependent annotation of reflection symmetry (blue lines).

When looking at Shape Ambiguity we are confronted with the problem that the definition of symmetry g(S)=S seldom holds true in practice. In real images, symmetric sub-parts rarely are exact copies of each other. Instead, slight deformation of shape and subtle differences in texture, color or lighting are commonplace, yet to the human eye such differences are often of little significance when judging symmetry (see Figure 4). Similar scenarios of ambiguity can be constructed for rotation and translation symmetry as well.

Because a tradeoff in weighting local versus global symmetry elements is required in almost all cases and is extremely subjective in nature, we mark such symmetries as ambiguous and discount any detection by the tested algorithms so that neither the number of true positives nor the number of false positives is affected.

Eventually, what is required is to define a symmetry transformation that is invariant to small and local disturbances of object shape and appearance. A more formal definition of such symmetry ambiguities is required and we see this as a very interesting and essential future work.

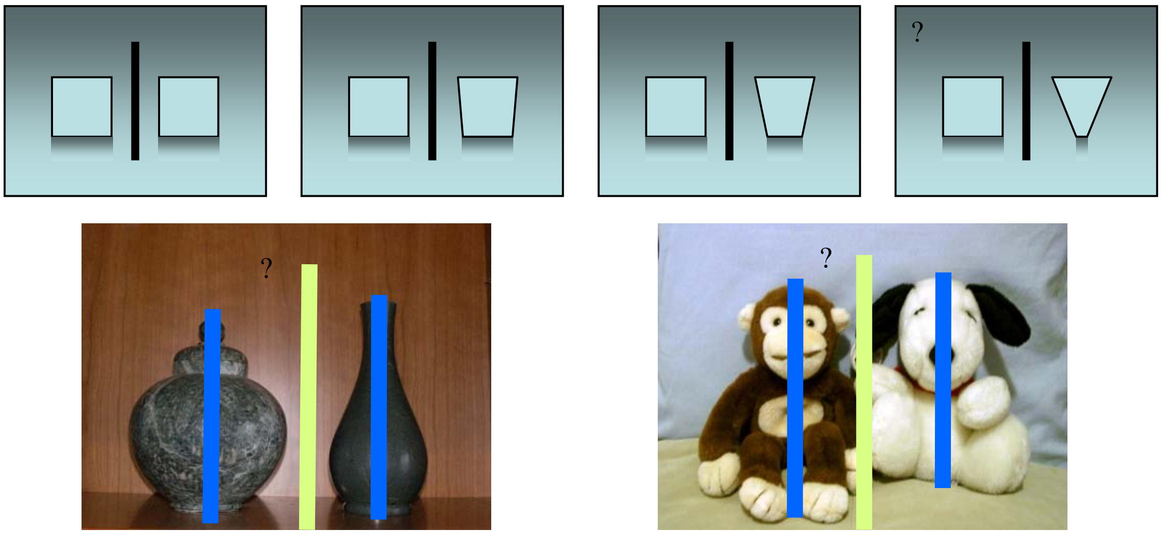


Figure 4: Shape Ambiguity in reflection symmetry. (Top Row) Two squares form a perfect reflection symmetry along their mirror axis. However, as one of the squares changes into a triangle, the boundary between valid and invalid symmetry fades. (Bottom row) Real world examples of shape ambiguity. While reflection symmetry within an object seems legitimate, symmetry between objects seems to be more subjective and application dependent.

3 Contestants and Algorithm Execution

In this section we outline how the execution and evaluation of submitted algorithms has been carried out. We received eight submissions for symmetry detection, three for reflection, two for rotation and three for translation symmetry. Adding one baseline algorithm to each symmetry group for comparison, we evaluated a total of eleven algorithms. Table 4 shows all contestants and baseline algorithms used in this competition.

Prior to the submission deadline, we provided potential contestants with a set of training images for each symmetry group. Contestants used these images to adapt and potentially fine-tune their free parameters in order to achieve best possible results on these training sets.

However, we nonetheless had various issues in executing the submitted algorithms on our image test set. While some algorithms came with a nice Graphical User Interface (GUI), it forced us to manually run the detection algorithm for each test image separately. Other algorithms came in form of Matlab code and a single point of entry (SPE), which made automatic batch processing of all images easy. We also received algorithms that required a number of manual steps for pre-processing, and for which documentation was rather scarce. After communicating with the authors most of the issues could be resolved. Yet, several algorithms still had abnormal terminations on a subset of images.

Table 4: List of research groups who have submitted symmetry detection algorithms for benchmark evaluation in three symmetry categories.

Symmetry Group	Contestant(s)	Institution(s)	Algorithm/Code
Reflection	Mo and Draper	Colorado State, USA	Matlab, SPE
	Eckhardt	Frauenhofer ISOB, Germany	Matlab, Pipelines
	Kondra and Petrosino	Uniparthenope, Napoli, Italy	Matlab, Mex-Files, SPE
Baseline	Loy and Eklundh		Windows Executable, (publicly available [3])
Rotation	Kim, Cho and Lee	Seoul National University, South Korea	Matlab
	Kondra and Petrosino	Uniparthenope, Napoli, Italy	Matlab
Baseline	Loy and Eklundh		Windows Executable, (publicly available [1])
Translation	Cai	Polytechnic, Hongkong, China	Matlab, Mex-Files, GUI
	Wu	University of North Carolina, USA	Windows Executable
	Eckhardt	Frauenhofer ISOB, Germany	Matlab, Pipelines
Baseline	Park, Brocklehurst, Collins and Liu	Pennsylvania State University, USA	Windows Executable (publicly available [4])

3.1 Algorithm Evaluation

For all three symmetry groups the algorithm performance is measured in terms of precision and recall rates (see Figure 5), which are two popular metrics for evaluating the correctness of a pattern recognition algorithm. Both can be seen as extended versions of *sensitivity*, a simple metric that computes the fraction of instances for which the correct result is returned.

When using precision and recall, the set of possible labels for a given detection is divided into two subsets, one of which is considered "true" or “correct” for the purposes of the metric, the other “false” or “incorrect”. Recall is then computed as the fraction of correct detections (true positives) among all instances that *actually* belong to the correct subset (number of groundtruth symmetries), while precision is the fraction of correct detections among those that the algorithm *believes* to belong to the relevant subset, which includes all those detections that are “true” (true positives) and “wrong” (false positives).

Precision = TP / (TP + FP)

Recall = TP / (TP + FN)

TP = True Positives	or Correct Detection
FN = False Negatives	or Missed Detection
FP = False Positives	or Incorrect Detection

Figure 5: Precision and recall rates are used to evaluate the correctness of a symmetry detection algorithm.

Precision can be seen as a measure of exactness or fidelity, whereas recall is a measure of completeness. In practice, a high **recall** means most symmetries have been correctly detected, without considering a possibly high number of incorrect detections (which would imply low **precision**). High **precision** means that most detections are a correct detections, although not all symmetries might have been found (which would imply low **recall**).

While for reflection and rotation symmetry evaluation, recall and precision rates are computed for the total number of true positives and false positives over all images, the recall and precision scores for translation symmetry is calculated for each image separately and then combined and averaged over the number of all images. Because the number of texels on which we count the number of correct detections varies vastly between images (e.g. some images have 200+ texels while others might have only twelve), this averaging is necessary to avoid any bias towards images with a high count of texels.

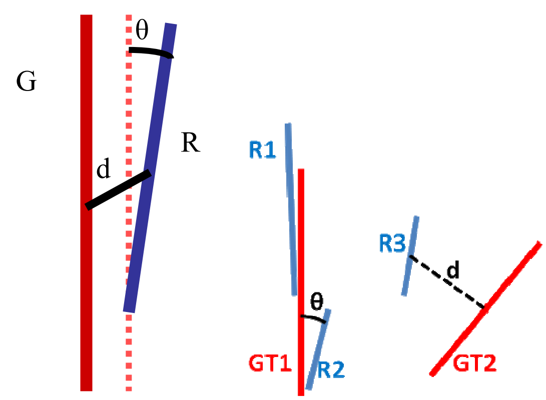
We now explain the process of determining the number of correct and incorrect detections for each symmetry group, which has been completely automated for this study.

Reflection Symmetry

For each detection result R, we measure the angle between the detected symmetry axis (R) and the ground-truth axis (GT). We also measure the distance (d) between the centers of both lines. A correct detection (true positive) is achieved, if the orientation between the two axis is less then some threshold t1, and the distance between the two axis is less then some threshold t2. For an illustration please see Figure 6.

For t1 and t2 we choose t1=10 degrees and t2 = 20% of the ground-truth axis length.

Multiple valid detections (R1, R2) can be clustered if they associate with the same ground-truth axis. One detection result, however, can only be associated with one ground truth axis. The size of the symmetry support region is not further considered.



a) **b)**

Figure 6: a) A detected reflection axis R (blue) is compared to the groundtruth G (red) by measuring the distance d and the relative angle *θ* between the two axes. b) shows two examples.

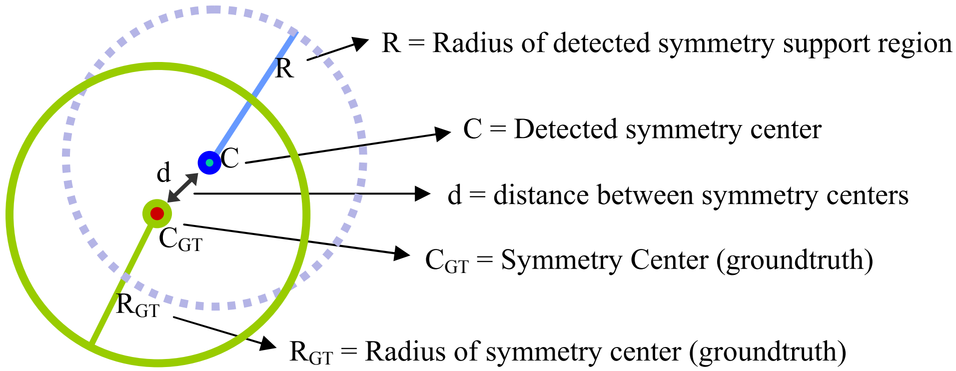


Figure 7: A detected rotation symmetry with center C and circular support region with radius R is compared against a groundtruth symmetry with center C_{GT} and region radius R_{GT} . If the distance d between the two centers is below some threshold and the symmetries have a similar support region, then the detected symmetry is considered valid.

3.2 Rotation Symmetry

For each detection result we measure the distance d between detected (C) and ground-truth symmetry center (C_{GT}). We also record the radius R of the detected symmetry region (and use the length of the major axis, if the detection algorithm reports an ellipse instead of a circle as support region). A correct detection (true positive) is achieved, if the distance d between detected and some ground-truth symmetry center is below some threshold t1, and if the radius R of the detected support region is within some bounds [b1, b2]. For t1 we choose t1=max(5, 0.02*R_{GT}) and for [b1, b2] we choose b1=0.8* R_{GT} and b2=1.2* R_{GT} . For an illustration please see Figure 7.

As with reflection symmetry, one detection result can match to only one ground-truth symmetry, but multiple different detections can be matched to one ground-truth center. Not considered in this evaluation are more detailed support region descriptors (such as an ellipse instead of a circle), the number of symmetry folds and the type of symmetry (e.g. discrete or continuous).

3.3 Translation Symmetry

We count the number of correctly detected tiles in a lattice structure that encodes the wallpaper pattern present in an image. A quadrilateral lattice tile is correct if all its four corners match up to corners in the ground-truth lattice. Because many different lattice structures can correctly define a wallpaper pattern, a global matching between detected and recoded ground-truth lattice has to be devised first.

We have created an automated method of lattice evaluation [4] that establishes a mapping between a detected lattice T and the ground truth lattice G by minimizing a distance cost-function between paired lattice points using a globally unique affine transformation to all detected lattice points.

This method works reliably in most cases. In rare cases, it can generate false positives due to odd boundary conditions caused by occluding objects or the edge of the image. For this reason, a visual inspection has been carried out to ensure that only texels occluded half or less are counted as valid.

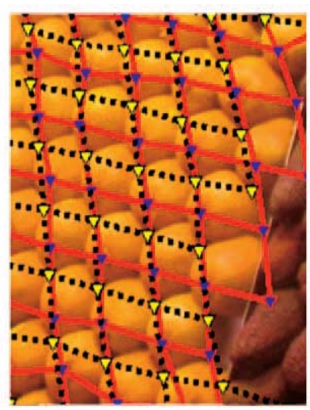


Figure 8: A global offset between ground truth (red) and detected lattice (dotted black) is found by minimizing the distance between all lattice points under some optimal global affine transformation, applied to all tiles simultaneously.

4 Results

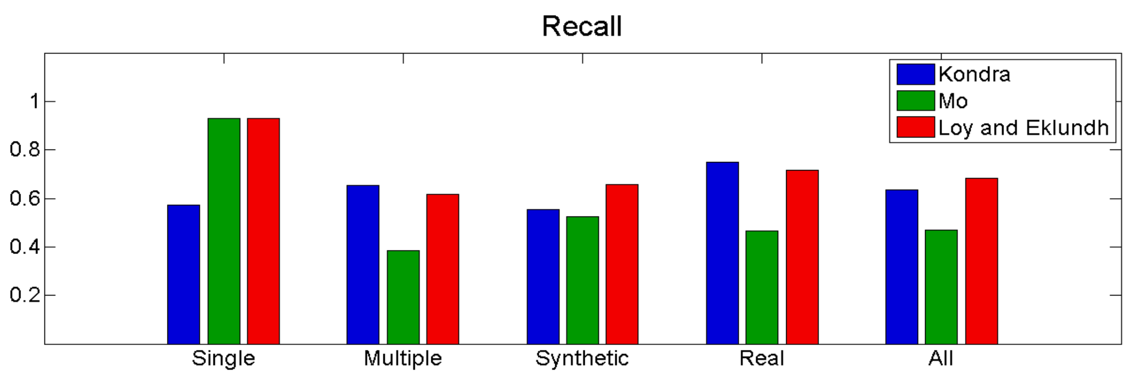
We now present results on the symmetry detection competition. For each symmetry group we show precision and recall rates overall and for each contestant and for each image category. Sample detection outputs and side-by-side comparisons can be found in the figures at the end of this document.

4.1 Reflection Symmetry

Overall, the algorithm by Loy and Eklundh (our baseline) outperforms all contestants with a recall rate of 0.68, compared to the algorithm by Kondra with a recall rate of 0.64 and that of Mo with a recall rate of 0.47. This performance advantage is due to a single image category, namely synthetic images, in which Loy and Eklundh outperform all others with a clearer margin (0.68 versus 0.55 and 0.53). In all other categories the algorithms by our contestants perform equally well or better than our baseline algorithm.

Among the contestants, Kondra's algorithm performs better than Mo's algorithm (0.64 versus 0.47). In fact Kondra's algorithm performs better than Mo's in three out of four image categories. Multiple symmetry Axis: 0.65 versus 0.38, Synthetic Images: 0.55 versus 0.53, and Real Images: 0.75 versus 0.46. Only in the single symmetry axis category does Mo's algorithm perform better than Kondra's algorithm (0.93 versus 0.57).

When we look at precision rates, the picture changes. Kondra's algorithm exhibits a relatively high number of false positives, when compared to Mo's or Loy and Eklundh's algorithm and therefore offers consistently less precision. Again, as with recall, Mo's algorithm achieves best precision on images with a single reflection axis. In all other categories our baseline algorithm performs best.



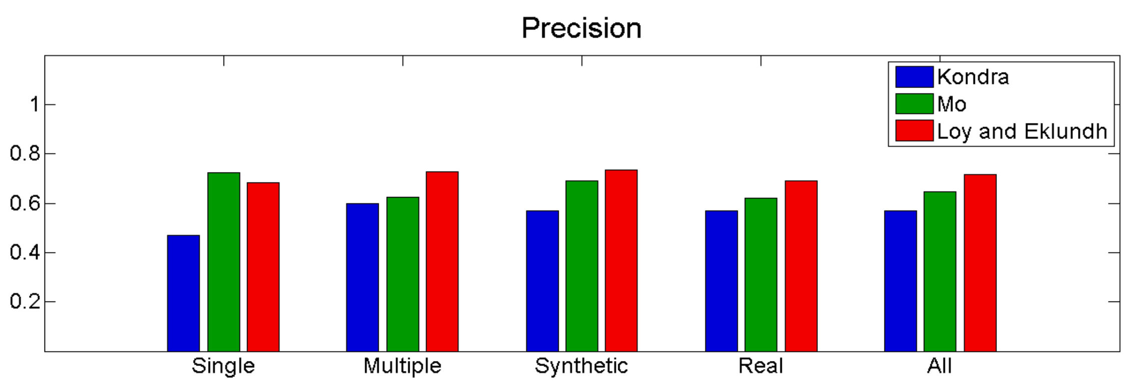


Figure 9: Recall and precision rates for reflection symmetry detection

We also compare execution times of all algorithms (see Figure 10). The baseline algorithm again performs best with an average execution time per image of 0.97 seconds. Mo's algorithm compares favorably with 3.8 seconds, while Kondra's algorithm takes on average 15.6 seconds per image. All computations were performed on a Windows Vista 64bit machine with an i7, 2.67G cpu (8 core), 6GB ram and used Matlab R2008b.

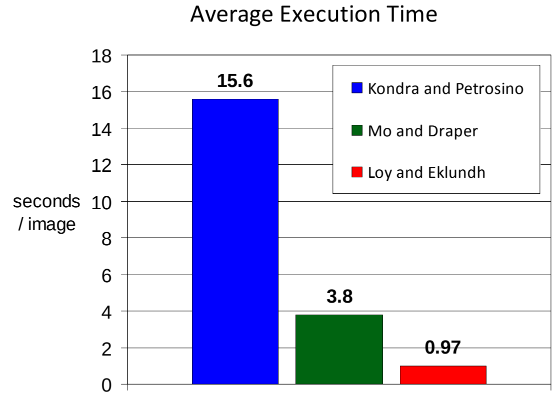


Figure 10: Average algorithm execution time in seconds for one reflection symmetry image.

4.2 Rotation Symmetry

Overall, our baseline algorithm by Loy and Eklundh clearly outperforms all other algorithms by a significant margin: Recall=0.51 versus 0.21 (Kondra) and 0.16 (Kim). It also performs best in each image category (see Figure 11).

Amongst the contestants, Kondra's algorithm performs overall better than Kim's algorithm (Recall=0.21 versus 0.16). Kondra's algorithm is especially better than Kim's algorithm on images with single (0.31 versus 0.14) and deformed rotation symmetries (0.21 versus 0.12). On images with continuous symmetries, we observe that Kim's algorithm has not produced a single correct detection. However, Kim's algorithm performs better then Kondra's algorithm on images with discrete (0.41 versus 0.626) and multiple rotation symmetries (0.17 versus 0.15).

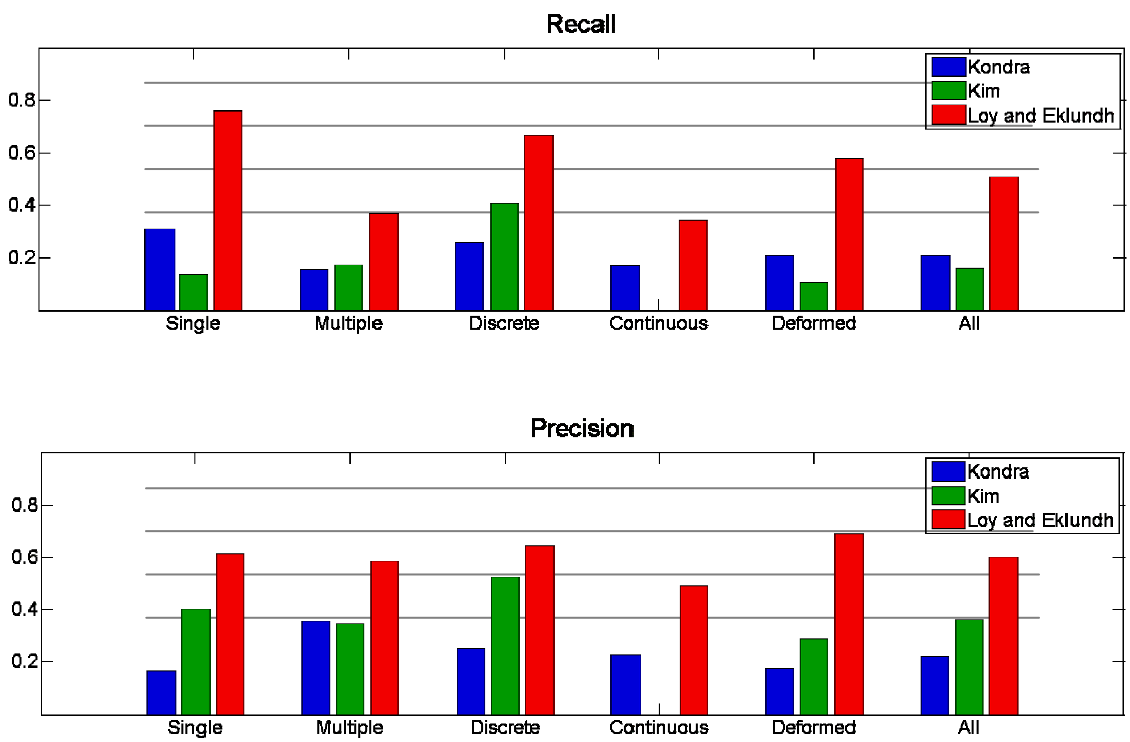


Figure 11: Precision and Recall rates for rotation symmetry detection.

Again similar to reflection symmetry, when we look at precision rather than recall rates, our baseline algorithm performs best overall, while Kondra's relatively high number of false positives changes the performance picture in favor of Kim's algorithm in all image categories.

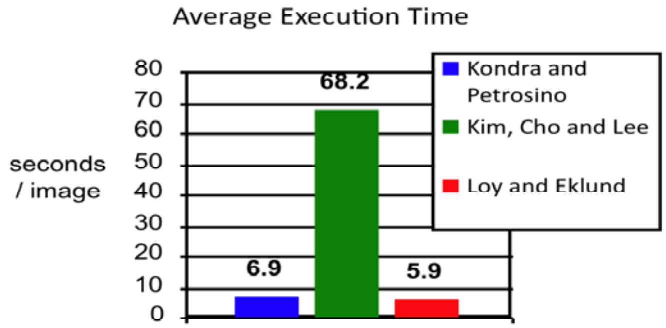


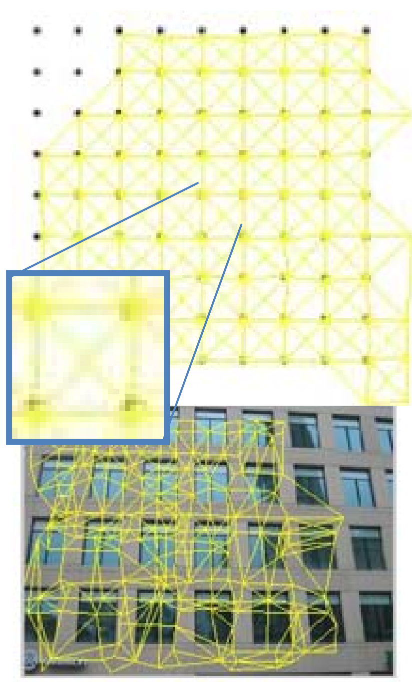
Figure 12: Average execution time in seconds for one rotation symmetry image.

We also compare execution times of all algorithms (see Figure 12). Our baseline algorithm again performs best with an average execution time per image of 0.59 seconds. Kondra's algorithm now compares favorably with 6.9 seconds per image, while Kim's algorithm takes on average 68.2 seconds per image.

4.3 Translation Symmetry

Evaluation of translation symmetry detection was unfortunately not as straightforward as it has been for reflection and rotation symmetry detection. The three submissions we have received did not end up conforming to the evaluation standard (see Figure 13) as laid out and published by us on our website [5].

Cai's output is not a valid lattice structure



Wu's output often only shows vanishing lines

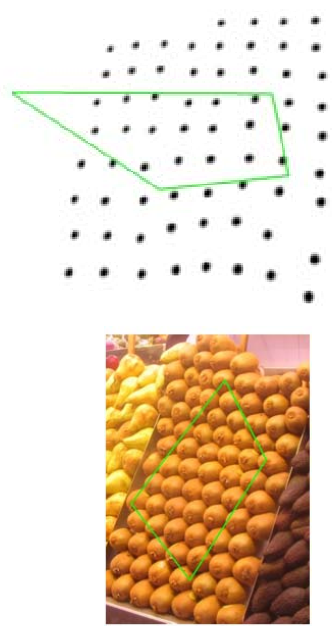


Figure 13: Issues with algorithm outputs for translation symmetry detection.

4.3.1 Issues with submitted algorithms

The algorithm by Y. Cai requires user input to provide an initial guess for the lattice structure, making a fair comparison to other algorithms, in particular our baseline, impossible. Further complicating the evaluation is that Cai's algorithm output format does not comply with our standard lattice format. We were unable to find a suitable technique to convert their output format to ours. The algorithm by C. Wu is primarily designed for 1D frieze pattern detection on building facades, and requires strong horizontal features for vanishing point detection. It nonetheless worked on some of our test images but failed on many others.

The algorithm by M. Eckhardt was initially designed only for reflection symmetry detection but an attempt has been made by its author to modify it for translation symmetry detection as well. However, after a long series of back and forth communication, and the realization that the modification did not bring about the desired results; the algorithm was eventually withdrawn from the translation symmetry detection competition.

4.3.2 Evaluation 1

We attempted quantitative evaluation by transforming the output lattice by Cai's algorithm into a valid lattice format when ever possible. We dismissed all results for which this transformation was not possible. The total number of test images for which all contestants reported valid results where then only four images (two from easy set, and one from each of medium and hard sets).

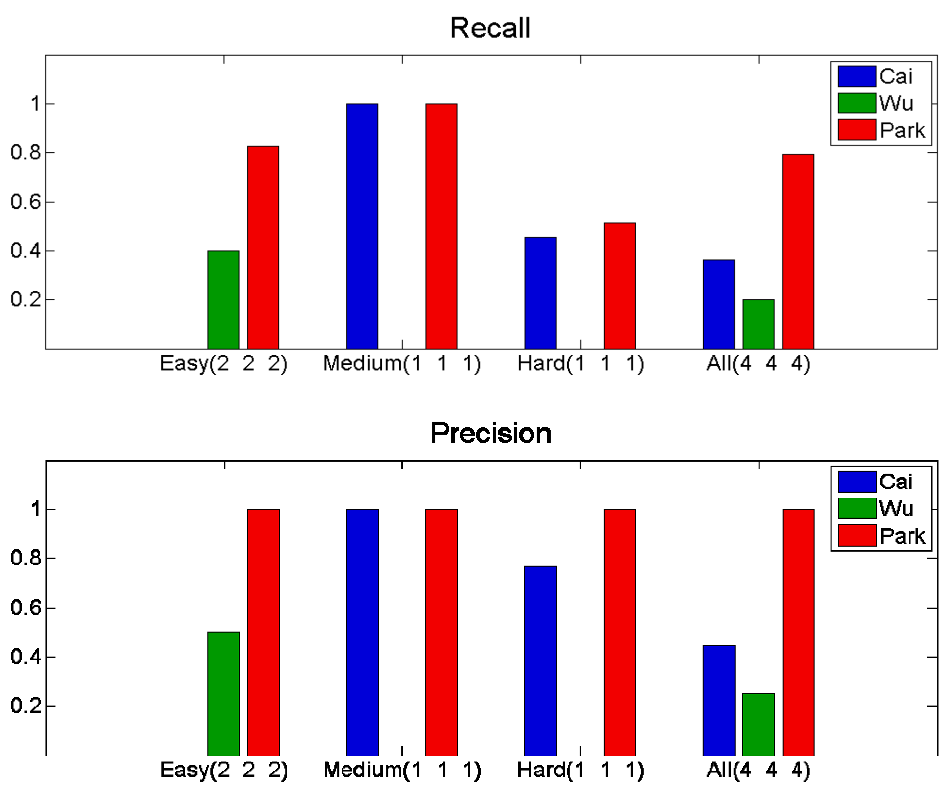


Figure 14: Precision and Recall rates for translation symmetry evaluation No. 1. Number triplet (a,b,c) means number of images used for each contestant. Here we used the only four images for which all contestants got valid results.

We observe that overall our baseline algorithm performs best with a recall rate of 76% compared to 36% (Cai) and 19% (Kim). Due to the small number of test images used for this evaluation, any further discussion of performance characteristics seems insensible.

4.3.3 Evaluation 2

We evaluated each algorithm separately on its own set of valid output images. However, by doing so we need to be cautious about comparing the obtained performance results, because each algorithm is evaluated on a different set of test images.

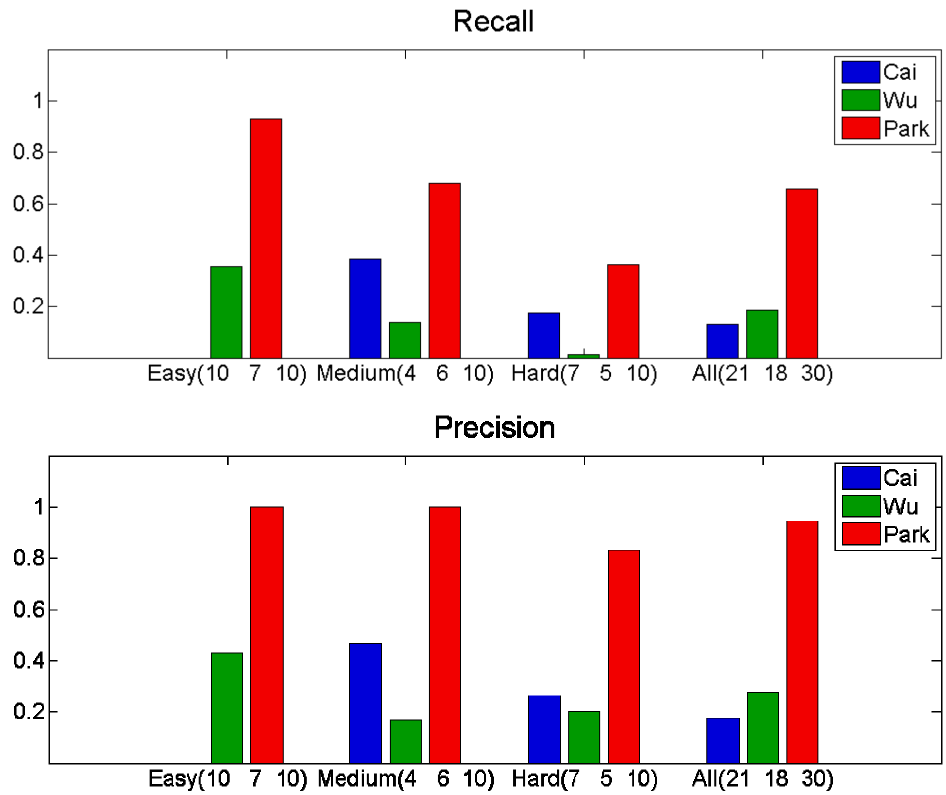


Figure 15: Precision and Recall rates for translation symmetry evaluation No. 2. Number triplet (a,b,c) means number of valid images considered for each contestant. Here each algorithm was evaluated independently on all its valid output results

We observe that out of a total of 31 test images, Cai's algorithm produced 21 valid output results, Wu's algorithm produced 18 and our baseline algorithm produced 30 valid output results. In terms of both recall and precision the baseline algorithm performs significantly better than the two contestants (Figure 15).

5 Conclusion

As the organizers of the first symmetry detection algorithm competition we have learned a lot through this process. First of all, from a research point of view, we learned how difficult it is to determine, unambiguously, groundtruth for real world symmetries. The difficulty resides in both local (hierarchical) symmetries and the degree of symmetry-ness. Further research by establishing a more adaptive real world symmetry model and using crowd sourcing will be carried out. Second, we realize that the algorithm evaluation using single point (single parameter setting given by the contestant) precision/recall rates is insufficient and partial for comparison. For future evaluation of each algorithm, we need to determine a 'tunable' knob inside the algorithm (provided by the contestant) to generate a receiver operating characteristic (ROC) curve, then compare the areas under the ROC curves. Third, standardizing the evaluation process becomes more and more important with the size increase of the test image sets. We have, for this competition, used such automated evaluation methods for the first time (reflection, rotation in particular). However, the output format of the submitted algorithms need to be enforced for future performance evaluations, especially for translation symmetry detection.

Overall we are pleased and excited that the first benchmark for symmetry detection algorithms has been set up for the computer vision community. Future effort can be built upon what we have established to further expand and solidify the test image set, annotations, quantitative evaluation standards and automatic evaluation tools. We have made our training and testing image sets with associated ground truth publicly available (http://vision.cse.psu.edu/research/symmComp/index.shtml). We welcome any comments, feedback, and additions to our existing image data sets and evaluation tools.

6 Acknowledgement

We gratefully acknowledge the support of NSF grant IIS-1040711 that makes this competition possible. We also thank all the participants of the competition, the contributors of real world photos on flickr (http://www.flickr.com/groups/1555886@N20/), the panel speakers at our CVPR 2011 workshop (http://vision.cse.psu.edu/research/symmComp/workshop/index.shtml) as well as our 11-member advisory committee:

Jacob Feldman (Rutgers)
Richard Hartley (ANU)
Takeo Kanade (CMU)
Jitendra Malik (U.C. Berkeley)
Doris Schattschneider (Moravian College)
Marjorie Senechal (Smith College)
Christopher Tyler (SKBIC)
Luc Van Gool (ETH Zurich & University of Leuven)
Laurent Younes (Johns Hopkins University)
Alan Yuille (UCLA)
Andrew Zisserman (Oxford)

7 References

[1] Yanxi Liu, Hagit Hel-Or, Craig S. Kaplan and Luc Van Gool (2010). **Computational Symmetry in Computer Vision and Computer Graphics**, *Foundations and Trends® in Computer Graphics and Vision*: Vol. 5: No 1-2, pp 1-195.

[2] http://www.flickr.com/groups/symmetrycompetition

[3] Loy,G. and Eklundh,J. (2006), **Detecting symmetry and symmetric constellations of features**, ECCV 2006.

[4] M. Park, K. Brocklehurst, R. T. Collins, and Yanxi Liu (2009), **Deformed Lattice Detection in Real-World Images using Mean-Shift Belief Propagation**, IEEE Transaction on Pattern Analysis and Machine Intelligence (TPAMI). Vol. 31, No. 10

[5] http://vision.cse.psu.edu/research/symmComp

[6] *M. Park, S. Lee, P.-C. Chen, S. Kashyap, A. A. Butt and Y. Liu* (2008) **Performance Evaluation of State-of-the-Art Discrete Symmetry Detection Algorithms** (CVPR '08)

[7] *P. Chen, J.H. Hays, S. Lee, M. Park, and Y. Liu* (2007) **A Quantitative Evaluation of Symmetry Detection Algorithms,** Tech. report CMU-RI-TR-07-36, Robotics Institute, Carnegie Mellon University , September, 2007. Tech Report PSU-CSE-07011