Nautilus: Recovering Regional Symmetry Transformations for Image Editing

MICHAL LUKÁČ, Adobe Research and Czech Technical University in Prague, Faculty of Electrical Engineering
DANIEL SÝKORA, Czech Technical University in Prague, Faculty of Electrical Engineering
KALYAN SUNKAVALLI, Adobe Research
ELI SHECHTMAN, Adobe Research
ONDŘEJ JAMRIŠKA, Czech Technical University in Prague, Faculty of Electrical Engineering
NATHAN CARR, Adobe Research
TOMÁŠ PAJDLA, Czech Technical University in Prague, Faculty of Electrical Engineering

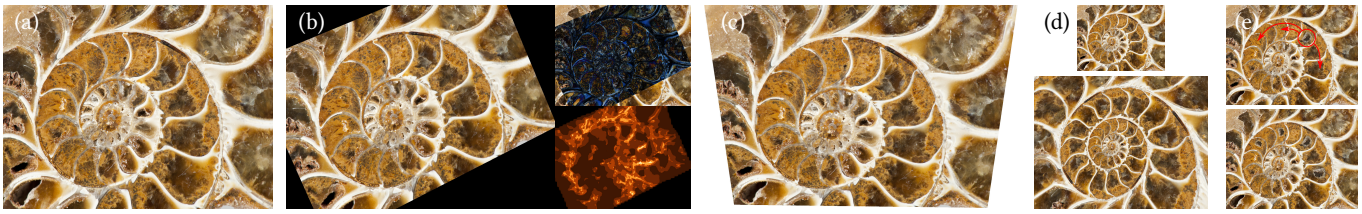


Fig. 1. From the input image (a), which exhibits a complex symmetry under perspective and with irregularities, our method automatically recovers the dominant symmetry transformation (b), and its spatial support map (lower right). Recovering this symmetry transformation facilitates a plethora of image editing applications. For example, we can analyse the transformation to compute its projective component and rectify the image (c). We can use it to facilitate symmetry-aware image completion, as we demonstrate by expanding the original image to a canvas twice the size while preserving the structure of the content (d). We can also use the transformation to propagate any edits we make along the symmetry (e). Source image: © Adobe Stock

Natural images often exhibit symmetries that should be taken into account when editing them. In this paper we present *Nautilus* — a method for automatically identifying symmetric regions in an image along with their corresponding symmetry transformations. We compute dense local similarity symmetry transformations using a novel variant of the Generalised PatchMatch algorithm that uses Metropolis-Hastings sampling. We combine and refine these local symmetries using an extended Lucas-Kanade algorithm to compute regional transformations and their spatial extents. Our approach produces dense estimates of complex symmetries that are combinations of translation, rotation, scale, and reflection under perspective distortion. This enables a number of automatic symmetry-aware image editing applications including inpainting, rectification, beautification, and segmentation, and we demonstrate state-of-the-art applications for each of them.

CCS Concepts: • **Computing methodologies → Image manipulation**; *Computational photography*;

ACM Reference format:
Michal Lukáč, Daniel Sýkora, Kalyan Sunkavalli, Eli Shechtman, Ondřej Jamriška, Nathan Carr, and Tomáš Pajdla. 2017. Nautilus: Recovering Regional Symmetry Transformations for Image Editing. *ACM Trans. Graph.* 36, 4, Article 108 (July 2017), 11 pages.
DOI: http://dx.doi.org/10.1145/3072959.3073661

DOI: http://dx.doi.org/10.1145/3072959.3073661

1 INTRODUCTION

Symmetries occur all around us; both natural organisms (like the ammonite shown in Fig. 1) and man-made objects exhibit symmetries in shape, texture, and form. These symmetries result in repetitive patterns that play a vital role in the human perception of natural images. Symmetries have been studied extensively (see Liu et al. [2009] for an overview), and it is important to properly account for them in image manipulation applications like image inpainting [He and Sun 2012; Huang et al. 2013], image resizing [Wu et al. 2010], image segmentation [Teo et al. 2015], perspective rectification [Pritts et al. 2014], and planarisation of textured surfaces [Liu et al. 2015]. These techniques discover symmetric repetitions of image patterns and use them as high-level constraints for the underlying manipulation algorithm. However, all these works detect only a small constrained set of symmetries like fronto-parallel translational regularity [He and Sun 2012] or translations under perspective [Huang et al. 2014; Liu et al. 2015; Wu et al. 2010].

The goal of our work is to enable a general class of symmetry-aware image editing operations. This requires us to address two important aspects: first, we need to handle a broad range of complex symmetries. Second, in order to edit an image, we need to isolate the region of the image that an estimated symmetry applies to. To this end, we present a method to estimate general symmetry transformations that are combinations of translation, rotation, scale, and reflection, even under perspective distortions. Our approach is more powerful than common symmetry detection schemes, since it is not limited to a single type of symmetry (e.g. a translational grid).

In addition, it is robust to partial occlusions, and automatically extracts both the pertinent symmetric transformation, and the region of the image that exhibits that symmetry. To our knowledge, our work is the first to address both these aspects. This allows us to estimate types of complex symmetries that are not handled in previous work, including the spiral symmetries shown in Fig. 1(a-b). Further, it allows us to use these estimated symmetries to enable a wide range of symmetry-aware image editing applications such as inpainting, perspective rectification, beautification, and segmentation (Fig. 1(c-e)).

Our technique starts by detecting dense local symmetries that are represented as similarity transformations. We do this by extending the Generalized PatchMatch algorithm [Barnes et al. 2010] to search for transformations that capture local image repetitions. This is a large search space that is slow to evaluate but has a sparse set of good solutions. We significantly improve the convergence of the search algorithm by using a sampling scheme based on Metropolis-Hastings [1953] that avoids transformations that are likely to be inaccurate. The similarity transformations found are accurate only in local regions because of perspective distortions that the similarity transformations cannot capture. In order to recover regional symmetries, we use each of the local similarity transformations as an initialization to a Lucas-Kanade-based iterative optimization that estimates a free-form homography that best explains the local similarity transformations. This optimization also estimates the corresponding image region where each regional symmetry transformation applies.

Contributions. Our work makes the following contributions:

1. Our method is the first approach that can estimate complex symmetries that are combinations of repeated rotation, translation, scaling and reflection under perspective distortion.
2. We do this in a robust but efficient manner by estimating dense local symmetries using a novel variant of the Generalised PatchMatch algorithm, and aggregating them with an extended Lucas-Kanade refinement.
3. We estimate multiple symmetries in an image and isolate the region where each symmetry applies using a per-pixel support measure that can detect imperfections and occlusions.
4. We present a new manually-annotated benchmark dataset for evaluating complex symmetries that we demonstrate the quality of our results on.
5. We present several symmetry-aware image editing applications that improve on task-specific state-of-the-art techniques.

2 RELATED WORK

Symmetry Analysis. Liu et al. [2009] provide a thorough overview of the work on symmetry detection in natural images. Kiryati et al. [1998] estimate reflection in undistorted images. Liu e al. [2004] discover translational symmetries by using auto-correlation to vote for parameters in transformation space. This idea was extended to detect rotations under affine skew [Lee and Liu 2010] and reflections along curved paths [Lee and Liu 2012].

Later work has focused on using affine-invariant feature points to detect regular repetitions under perspective [Tuytelaars et al. 2003], reflections under perspective [Cornelius et al. 2007], and reflections and rotations in undistorted images [Loy and Eklundh 2006]. Hays et al. [2006], Park et al. [2009], and Liu et al. [2015] recover near-regular translational lattices [Liu et al. 2004] from sparse features and dense correspondences respectively.

These previous approaches focus on detecting specific classes of symmetries. In contrast, our framework can detect general symmetry transformations – that can be arbitrary combinations of translation, rotation, scale, and reflection – under perspective or affine distortion. Unlike previous techniques, which often detect sparse feature points and then fit symmetries to these points, we search for dense per-pixel symmetries and then group them into global estimates. This allows us to search for a broader class of transforms in a robust manner. While this approach can be computationally expensive, we show that it can be solved efficiently by building on the Generalised PatchMatch algorithm [Barnes et al. 2010]. In addition, while some of the above approaches [Lee and Liu 2010; Loy and Eklundh 2006] compute a region of support, this is typically in the form of coarse estimates like the center and perimeter for rotational symmetries and a lattice for translational symmetries. In contrast, our dense approach recovers regional symmetry support on a per-pixel basis, and thus is robust to occlusions and imperfections in the symmetry.

Symmetry detection has also been addressed in the context of 3D data (see Mitra et al. [2012] for a comprehensive overview), and techniques have been proposed to recover reflections and rigid transformations [Mitra et al. 2006] and combinations of rotation, translation, and scale [Pauly et al. 2008]. Like them, we estimate local symmetries which we then group into global symmetries with their associated regions of impact. However, we do this under perspective distortion which makes the global symmetry analysis challenging because it "smears" the local transforms and precludes the use of direct voting in the transformation space.

Applications of Symmetry Analysis. Symmetry detection has been used to drive a number of analysis and editing tasks. Lobay and Forsyth [2006] recover 3D shape from the distortions of repetitive textures. Translational symmetries can be used to replace textures in images [Liu et al. 2015, 2004] and remove regular occluders like fences [Liu et al. 2008]. He et al. [2012] exploit fronto-parallel translational symmetries to improve inpainting. Huang et al. [2014] extend this to out-of-plane symmetries by using vanishing lines to rectify planes as a pre-processing step. Huang et al. [2013] show that more general symmetries can improve inpainting results, but rely on the user to specify them manually. Repeated patterns can also be utilized for image rectification. Zhang et al. [2012] rely on a sparsity model of texture to estimate translational symmetry and rectify an image. However, their method is sensitive to background clutter and requires the user to select representative parts in the image. Aiger et al. [2012] detect congruent line segments while Pritts et al. [2014] measure the change in scale of matching features. A key drawback of these local measurements is that they are content dependent and can be affected by noise. In our method the projective component is removed just by analysing the homography estimated from local symmetries over large image regions. Wu et al. [2010] detect translational lattices and perform content-aware resizing of images that does not distort the lattice. Kim et al. [2012] propose a texture

Method	L	T	G	R	A	H	F	D
[Kiryati and Gofman 1998]	•							
[Tuytelaars et al. 2003]	•	•			•	•		
[Loy and Eklundh 2006]	•		•	•				
[Hays et al. 2006]		•			•	•	•	
[Cornelius et al. 2007]	•				•	•		
[Lee and Liu 2010]				•	•			
[Lee and Liu 2012]			•				•	
[Zhang et al. 2012]	•	•			•	•		
[Pritts et al. 2014]	•	•	•	•	•	•		
[Liu et al. 2015]		•			•	•	•	
Our approach	•	•	•	•	•	•		•

Table 1. Capabililties of previous symmetry detection methods: reflection (**L**), translation (**T**), glide reflection (**G**), rotation (**R**), under affine skew (**A**), under perspective (**H**), under free-form deformation (**F**), dense symmetry support mask (**D**).

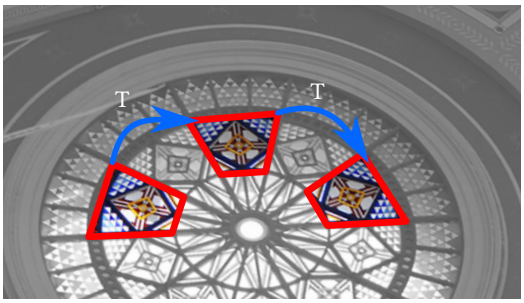


Fig. 2. Symmetry as a repeatable transformation. A homography transformation T maps a part of the image to a similar part under perspective, as does T^2. Source image: © Isabella Josie

perturbation technique that optimizes an auto-correlation-based symmetry representation and use it for symmetry transfer, filtering, and deformation.

While these techniques develop application-specific symmetry analysis methods, we show that our method can estimate general symmetries that support a number of image editing tasks including rectification, hole-filling, beautification, segmentation, etc.

3 TASK FORMULATION

Classical symmetry group theory [Liu et al. 2009] defines a symmetry as a global isomorphism. This definition is too restrictive to apply to natural images which exhibit occlusions, geometric and photometric inconsistencies, and other forms of noise. More recent work has resolved this by robustly matching a global appearance template [Hays et al. 2006; Liu et al. 2015; Park et al. 2009] or fitting symmetries to sparse feature points [Pritts et al. 2014]. However, image editing applications require dense symmetry estimates. This in turn requires a local, point-wise notion of symmetry that can adapt to local image content and still be computed robustly.

At the crux of our method is the definition of a symmetry as a *repeatable correspondence*. While a regular correspondence is a transformation T that when applied to a patch p_x results in a dissimilarity $d(p_x, T * p_x)$ that is low, a repeatable correspondence is one where the dissimilarity $d(p_x, T * T * p_x)$ is *also* low, i.e., repeated applications of the symmetry produces patches that are all similar (see Figure 2).

We use the above definition to formulate and quantify the cost of a symmetry at a point as follows: using the *Sum of Squared Differences* (SSD) as a measure of patch-to-patch dissimilarity, we measure the cost of a candidate transform T at patch p_x, centered at pixel x, as:

$$e(T, p_x) = \max\Big(\text{SSD}\left(p_x, T(p_x)\right), \text{SSD}\left(T(p_x), T^2(p_x)\right)\Big). \quad (1)$$

This definition allows us to compute dense symmetries using patches centered at every pixel in an image. In practice, these patches are small and not sufficient to recover perspective effects. Therefore, we compute our symmetries in two stages. First, we compute dense *local symmetries* that are restricted to similarity transformations and are evaluated at a patch level. Next, we fuse these local symmetries into *regional symmetries* that are modeled as homographies to account for perspective effects. These symmetries map parts of an image to themselves in such a way that the transformed regions are similar to the original ones. This approach allows us to compute globally applicable symmetries along with their dense spatial support maps.

4 ALGORITHM

As illustrated in Figure 3, the core principle of our algorithm is to discover patch-wise dense local symmetries in the image plane, and then deduce regional out-of-plane symmetries from these. To make the task of finding dense correspondences computationally tractable, we build upon the well-known Generalised PatchMatch (GPM) [Barnes et al. 2010] algorithm, enriching it with a Metropolis-Hastings sampling scheme to improve performance. We then cluster these results and use them as initializations for hierarchical Lucas-Kanade registration [Baker and Matthews 2004] combined with a novel fit detection scheme to find homographic transformations representing regional symmetries. The details of these steps are described below.

4.1 Local Symmetry Search

In the first step of our algorithm, we search for dense repeatable similarity correspondences that represent local symmetries. We limit the search to only similarity transformations because any usable patch size would be too small to infer perspective effects. This effectively restricts the search to image-plane symmetries, and we later extend these into non-fronto-parallel planes.

Our algorithm is built upon the k-NN GPM for generalised transformations which we modify to minimize Equation 1. However our search space is much sparser than usual, as there are far fewer symmetries than just correspondences. Using the random sampling step in GPM to find them would thus produce unreasonable runtimes. To make this search practical, we extend the GPM algorithm in two principal ways. First, we parameterize the transformation in such a way that the parameters can be interpreted as sets of points in image space. Second, we take advantage of this parametrization to alter the random search step to a guided variant that focuses on

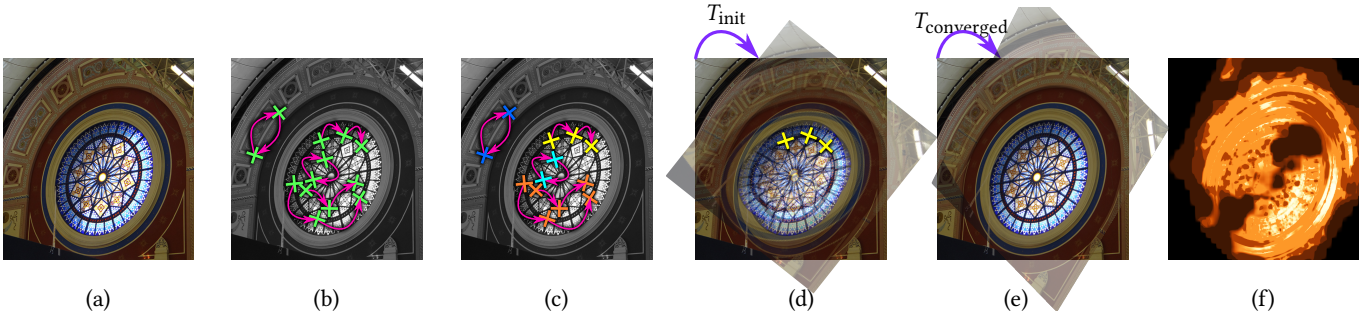


Fig. 3. High-level overview of our method. Given a natural image (a), we find dense repeatable similarity correspondences (b). We cluster these correspondences into groups (c), and for each of the transformations represented by one of these clusters (d) (in this example, focusing on the outer rim of the window), we refine the similarity transformation to compute a locally optimal homography symmetry (e). During the fitting step, we also extract a support map (f). Source image: © Isabella Josie

exploring the more promising regions of the search space, resulting in a much faster convergence.

4.1.1 Transformation Representation. A similarity transformation in 2D homogeneous coordinates may be expressed using four parameters (a, b, c, d):

$$T = \begin{pmatrix} a & -b & c \\ b & a & d \\ 0 & 0 & 1 \end{pmatrix}. \tag{2}$$

A transformation of this form is fully determined by two correspondences; that is to say, the system of equations

$$\begin{aligned} Tx &= y \\ Ty &= z, \end{aligned} \tag{3}$$

uniquely determines the similarity transformation T for any three points x, y, z such that $x \neq y \wedge y \neq z$. Thus at any point x, given a pair of points y, z there is a unique transformation T so that the above equation holds, and conversely, we can easily compute unique y, z given a transformation T.

We use these three points to measure the cost of the transformation and interpret each transformation as a triplet of points in image space. We take advantage of this equivalence to quickly check whether the transformation is valid or not (i.e., all the projected points are within the bounds of the image), detect how different two transformations are at a given point by measuring the distance between the corresponding projected points (as in $d_x(T_1, T_2) = ||T_1x - T_2x||^2 + ||T_1^2x - T_2^2x||^2$), as well as re-interpret perturbations of the transformation in the random sampling step in image space.

We can easily extend the space of transformations to include reflections by flipping the signs of the values in the first column of the symbolic matrix given. This alters the structure of Equation (3) when determining the parameters (a, b, c, d). To handle this we simply track whether a given transformation is reflective or not.

Our implementation uses 7×7 patches in the search, and maintains a queue of 10 best transformations for each pixel. The queue only keeps transformations which are both sufficiently distinct from each other as well as from the identity transformation. We consider two transformations to be distinct if they either differ in their reflective component (the sign of the determinant of T), or if their mutual distance, as defined above, is larger than 3 pixels.

4.1.2 Guided Random Sampling. The PatchMatch family of algorithms computes a nearest neighbour field by alternating *random search* and *propagation* steps. In the random search step, each image pixel samples correspondences in an exponentially shrinking window around the current best match, approximating an exponential distribution. The best matches found are transferred between neighboring image pixels in the propogation step. This scheme works well in translational PatchMatch, where the size of the search space is roughly equal to the number of pixels in the image, and thus the aggregate probability of finding a good match for a region is high. Generalised PatchMatch increases the search space combinatorially by adding more dimensions to it, while the number of valid solutions remains roughly the same. This results in the need for more search iterations to achieve a comparable result quality. The distance measure we use is even more discriminative than in the GPM scenario (there are fewer repeatable correspondences than regular correspondences in an image), and we would require even more computation to achieve comparable results.

To overcome this, we propose a *guided sampling* approach. In this, the new guesses are still selected randomly, but the sampling is biased towards more promising parts of the search space through the use of an *oracle* – a lightweight estimator that is roughly correlated to the final cost.

The guided sampling algorithm works just like GPM, except the transformation we are using is interpreted as the point triplet as explained above. A random perturbation of the transformation can be implemented as a random perturbation of two of the points. We do this by keeping the point at which we are running the sampling, x, fixed, and perturbing the other two points in a spatial window around their current estimates. However, instead of drawing a uniform random sample from this window, we use the Metropolis-Hastings algorithm [Metropolis et al. 1953] to draw a sample from a probability distribution, p_x, that we design to better represent good matches for the point, x, as:

$$p_x(y) \propto o_x(y), \tag{4}$$

where $o_x(\cdot)$ is the oracle function at x. In our scenario the final distance function is the patch-to-patch SSD. We therefore base the oracle function on colour. For a candidate point y being sampled by pixel x we define it as:

$$o_x(y) = \exp\left(-||G(y) - G(x)||^2\right), \tag{5}$$

where $G(x)$ is the colour at pixel x convolved with a Gaussian filter, with $\sigma = 3$ pixels (half patch width). This gives a rough estimate of the patch distance around these two pixels which is rotation-invariant, reflection-invariant and to a degree also scale-invariant. This convolution is pre-computed so evaluating this function only requires sampling an image twice, and is thus very fast.

We constrain $p_x(\cdot)$ so that it is non-zero only within an exponentially shrinking spatial window around the current best match (as is done in GPM). Thus we spend more time sampling the parts of the search space we consider to be more promising, while staying spatially close to known good matches from GPM.

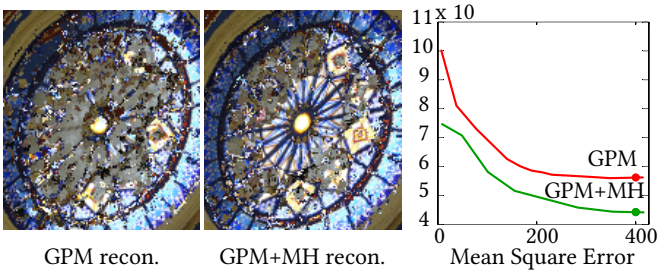


Fig. 4. Comparison between standard GPM and our Metropolis-Hastings sampling scheme (GPM+MH). The nearest neigbour fields produced by our method leads to better reconstructions (left) and lower image reconstruction errors (right) after the same processing time. Source image: © Isabella Josie

In practice, this guided-sampling variant of GPM produces similar results in terms of average patch-wise SSD, but substantially outperforms standard GPM in terms of reconstruction error (i.e. the difference between the image reconstructed using the retrieved nearest-neighbour field vs. the original image, as shown in Figure 4). The reason for this is that the guided search finds more meaningful symmetries which are propagated over larger areas, while the unguided variant diverges into pixel-wise local minima. This also means that guided search performs better on more structured images which have a sparser solution space, while in unstructured images where the error landscape is much more forgiving, performance is comparable.

Finally, flat, non-textured patches are much more tolerant to their correspondences, and as a result extensively searching for their best matches does not contribute to finding meaningful symmetries in the structured parts of the image. Therefor, we perform the random sampling step only in patches that are *salient*, as determined by the measure used by Liu et al. [2015]. Non-salient patches still contribute by propagating correspondences, but we focus the symmetry exploration effort to the structured parts of the image.

4.1.3 Pre-clustering. While the local symmetries computed by our method capture local repetitions well, we are interested in symmetries that are supported over large areas of the image. The obvious way to do this is to cluster the detected point-based symmetries over larger regions. Since we are already using a variant of GPM to find these point symmetries, we take advantage of some properties of this algorithm to directly generate a preliminary clustering.

GPM creates new transformations only in the random sampling step. During the propagation step, existing transformations are propagated from one pixel to another without any changes. We take advantage of this property by assigning a unique label to each new generated transformation, which is then propagated along with the transformation itself. At the conclusion of the local symmetry search step, we then have a set of pre-clustered symmetry transformation labels, which we use in further steps of our algorithm.

This pre-clustering allows us to coalesce regions with identical transformations and consider them in aggregate. We are also able to discard transformations with low salient spatial support (less than 10 pixels in our implementation), further saving computational effort.

4.2 Global Homography Search

The local symmetry search step estimates a set of similarity transformations that capture the symmetries in small local regions of the image. We limit the search to similarity transformations because it is challenging to recover perspective effects from small image patches. In the next step, we refine these similarity transformations to recover regional symmetries – encoded as homographies for handling perspective distortions – that capture symmetries over large image regions. For this, we first define the notion of a symmetry that "explains" a pixel, and then build a gradient-descent algorithm on top of it.

4.2.1 Spatial Support Estimation. Given a candidate symmetry transformation T, each patch p may be isomorphically mapped onto a counterpart $p' = T * p$ within the image. We could thus measure the dissimilarity between p and p' to evaluate how well T aligns the two patches. However this measure highly depends on the patch content, e.g, it can be very high when a highly structured patch is even slightly misaligned, but low when a non-structured patch is poorly aligned. Thus, we found it to be an inappropriate measure of the quality of a transformation.

We follow the intuition that if two same-frequency image-domain signals are correlated, the energy of their difference will be lower than the energy of either of them. While this intuition breaks down when the frequency content of the two signals differs, we can isolate this variable by decomposing the images into a Laplacian Pyramid [Burt and Adelson 1983]. Each level of the pyramid then represents a separate frequency band on which we can consider the signals to be comparable.

Given input image I_o and the transformed image I_T, we construct their respective Laplacian pyramids, P_o and P_T, and measure alignment confidence, $c(x, l)$, at pyramid level l at patch p_x as:

$$c(x,l) = 1 - \min\left(1, \frac{||P_o^l(p_x) - P_T^l(p_x)||}{\max(||P_o^l(p_x)||, ||P_T^l(p_x)||) + \epsilon}\right), \tag{6}$$

where ϵ is a small constant (10^{-7}) to prevent division by zero. This yields a similarity measure in the range $[0, 1]$ for each pixel at every scale. To consider a patch well-aligned, we require that it be

well-aligned at all scales. To ensure this, we aggregate alignment confidence across scales as follows:

$$c(x) = \max_{l \in \{0..l_{max}\}} \left(\min \left(s_l, \min_{j \in \{l..l_{max}\}} \left(c(p_x, j) \right) \right) \right), \quad \text{where} \quad s_l = 2^{-l}. \tag{7}$$

Here $l = 0$ is the finest scale and $l = l_{max}$ the coarsest. s_l thus corresponds to the scale factor of the pyramid level, and the aggregated alignment confidence approximates the reciprocal of the smallest scale value up to which the given pixel aligns well. This aggregated aligned confidence captures the regions of the images that support a particular symmetry transformation, and we refer to it as a *spatial support map*.

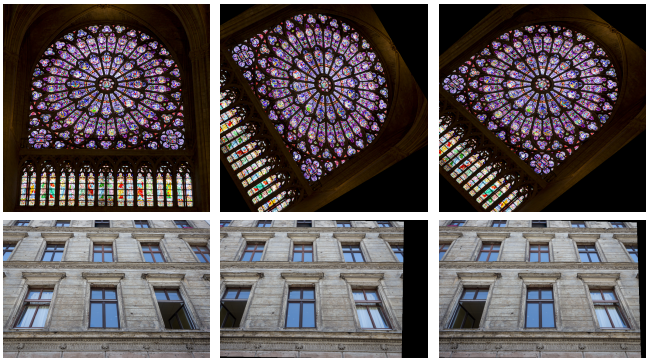
As a side effect, the support map may be thresholded at s_l to gain a binary support mask of a fit up to a certain scale level. This allows effective handling of irregular images, because the user can just linearly scale the weight map based on the desired tolerance.

4.2.2 Fitting homographies. Each similarity transformation produced by our local symmetry search step (Sec. 4.1) is now associated with a spatial support map that indicates the image regions that it applies to. We then use the Lucas-Kanade registration algorithm [Baker and Matthews 2004] to refine each local similarity transformation into a more general homography transformation. This algorithm uses gradient descent to find a homography that is close to the initial similarity, while minimising the difference between the original and transformed images. The locality of this search in the transformation space is a desired behaviour in our context, as a globally optimal solution is always trivially an identity transform.

In order to make the descent robust to image noise, we use the Levenberg-Marquardt Inverse Compositional Lucas-Kanade algorithm (Sec. 4.5 in [Baker and Matthews 2004]). Also, to make the descent robust to occlusions and distractors, we take advantage of the previously defined spatial support map $c(x)$ to focus on the regions that we believe the current symmetry applies to. We do this by weighting the per-pixel Hessians and gradients (Eqns. 92 and 93 in [Baker and Matthews 2004]), and the image error by the spatial support map. Each time the transformation is refined, we recompute the spatial support map using Equation. 7.

4.2.3 Learning to filter symmetries. The above algorithm produces the best global homography estimates for all detected local symmetries, some of which are correct and some are not. In order to determine the homographies that represent meaningful symmetries, we use a Support Vector Machine classifier [Cortes and Vapnik 1995] that we train using manually annotated reasonable and unreasonable symmetries on a dataset of 32 images (for a total of 21744 labeled transformations).

We first compute a goodness weight for every homography that evaluates how well it explains the image by summing the spatial support map across the entire image, i.e., $w(x) = \sum_x c(x)$. The input features to the classifier are: 1) the goodness weight of the candidate normalized by size of the image, 2) percentile of said weight within the candidate set (i.e., compared to all the other homographies estimated for this image), 3) relative density of the weight value within the candidate set as measured by kernel density estimation (i.e., how many of the homographies for this image have a similar weight value), 4) size of the spatial support region (estimated by thresholding the support map using Otsu's method [Otsu 1979]), and 5) the difference between the initial weight of the candidate (from the local search step) and its converged weight (after fitting the homography). We use RBF kernels for the SVM and train using hard negative mining to improve performance. The SVM was evaluated using a training/test split of 80%/20% and tuned to achieve a testing performance of 91.17% precision and 41.33% recall. We opted for higher precision and lower recall to ensure that the output symmetries are more likely to be meaningful.



(a) Input (b) Local similarity (c) Final homography

Fig. 5. A before-and-after comparison of the homography fitting. Given an input image (a), our local similarity search computes a coarse symmetry transformation (b), that is refined by the homography fitting (c) to produce transformed images that are better aligned with the input. Source images: © Adobe Stock

5 RESULTS

For a first-approximation of verification of correctness, we ran our algorithm on a synthetically generated dataset, manually comparing the output to the known ground truth. This allowed us to test, in isolation, the robustness of our algorithm to high-frequency noise, low-frequency random distortion, as well as perspective distortion. We have found it to be surprisingly robust to both noise and distortion, and satisfactorily resilient to perspective for all of the examined symmetry types. These results are included in the supplemental material.

We further sought to evaluate the quality of the returned results on natural images, such as one might use in editing applications. Evaluating quality of symmetry detection is a very challenging problem; indeed, it seems to be at least as challenging as the detection itself. The state-of-art work in this area is the 2013 symmetry detection competition [Liu et al. 2013], but we found its methodology ill-suited for evaluation of complex symmetries. The evaluation there is special-cased for individual symmetry types, and ground-truth labels are established by perceptual consensus rather than pixel-wise fit, which is insufficient for our image editing applications. Likewise, the reference algorithms there do not provide output that can be easily compared with a homography transformation matrix; for example, for reflections, the evaluation is by design unconcerned with the extent and form of the symmetry perpendicular

to the axis, and for rotations no information on the angle thereof is considered.

In light of these problems, we decided to design our own benchmark along with an evaluation algorithm, which is included and described in the supplemental material. We selected a set of 30 photographs which together exhibit a broad mixture of various symmetries. For the purposes of the evaluation, we interpret a symmetry as an isomorphic mapping between the feature points of the image. We have manually annotated the selected images with feature points and defined the ground-truth *primitive* symmetries, out of which composite symmetries such as translational grids are composed. We then assign each returned result a fit value between 0 and 1 based on how closely it matches the most similar ground-truth transformation.

Based on this evaluation, we found that our method returns an average of 4.23 symmetries per image ($\sigma^2 = 2.34$) with an average quality of 0.632 ($\sigma^2 = 0.39$). Full breakdown by image is shown in the supplemental material. The examples that have proven challenging to our method can be found in Section 7 where we discuss our limitations. We further present a sampling of other interesting results in Figure 7.

In Figure 6, we compare our estimation of reflection and rotation symmetries against the method of Loy and Eklundh [2006] which is the baseline algorithm for these symmetries in [Liu et al. 2013]. As can be seen here, our technique is able to match the performance of a state-of-the-art technique developed specifically for rotations and reflections. In addition, we are able to densely predict the spatial extent of the symmetries; previous techniques, including Loy and Eklundh, which rely on sparse feature points, cannot do so.

In practical terms, the applications shown in this paper also demonstrate that the detected symmetries are accurate enough to support a variety of image editing methods. In fact, the results obtained by our method work as well as manually annotated symmetries would, and provide comparable or better results than either competing techniques, or symmetry-agnostic ones.

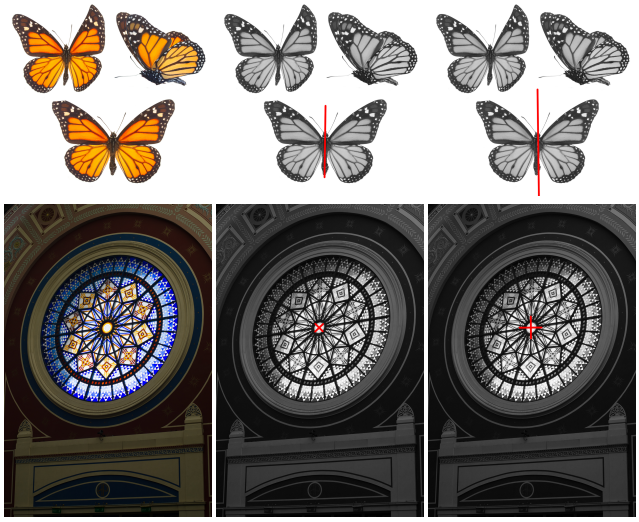
Our unoptimized prototype has been implemented in a combination of C++ and MATLAB. Runtimes currently span tens of minutes per a megapixel image, with the majority of this being taken up by the Lucas-Kanade iteration. While processing an individual symmetry is fast and takes a minute at most, a great number of symmetries has to be processed in total. We hope to remove this bottleneck in the future by on-line clustering of candidate local symmetries during the course of the global symmetry estimation.

6 APPLICATIONS

To demonstrate the practical utility and versatility of our method, we have implemented several image editing applications where the knowledge of the symmetric transformation and its spatial support simplifies the task or removes the need for manual input. We compare our results on each of these applications to their current state-of-the-art methods, and demonstrate that our general framework is able to replicate and improve upon their quality.

6.1 Image Rectification

Previous techniques for removing perspective distortion and shear from a photograph require the manual specification or detection of



(a) Input (b) Loy [2006] (c) Our result

Fig. 6. We compare against the reflection and rotation detection method of Loy and Eklundh [2006] (reflection axis and center of rotation marked in red). As can be seen here, we closely match their performance while retaining the general nature of our method. In addition, we can predict which regions of the image satisfy this symmetry, which their method does not. Source credit: © Adobe Stock (top), CC-BY Cat Burston @ Flickr (bottom)

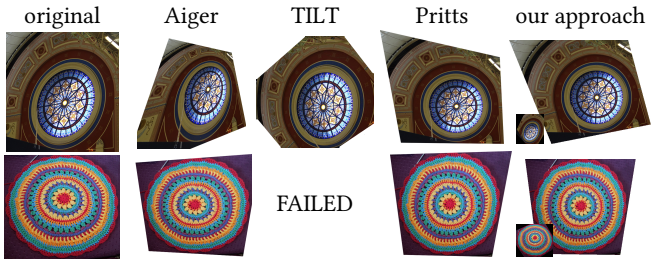


Fig. 8. Rectification of images captured under perspective (a). We compare Aiger et al. [2012] (b), Zhang et al. [2012] (c), and Pritts et al. [2014] (d) against our approach (e, used symmetry inset). Source images: © Isabella Josie.

corresponding points, right angles, vanishing lines [Liebowitz and Zisserman 1998], congruent line segments [Aiger et al. 2012], global measurements such as the rank of the image matrix [Zhang et al. 2012] or change of scale [Pritts et al. 2014].

In our solution we rectify the image using one of our detected global symmetry homographies, $\mathbf{H}$. The key is to assume that $\mathbf{H}$ has the following structure:

$$\mathbf{H} = \mathbf{PSP}^{-1}, \tag{8}$$

where

$$\mathbf{P} = \begin{bmatrix} 1 & x_1 & 0 \\ 0 & x_2 & 0 \\ x_3 & x_4 & 1 \end{bmatrix}, \quad \mathbf{S} = \begin{bmatrix} x_5 & x_6 & x_7 \\ x_8 & x_9 & x_{10} \\ 0 & 0 & 1 \end{bmatrix}. \tag{9}$$

$\mathbf{P}$ is the pure perspective part of $\mathbf{H}$ and $\mathbf{S}$ is an in-plane symmetry which we assume to be a general similarity transform, i.e., uniform

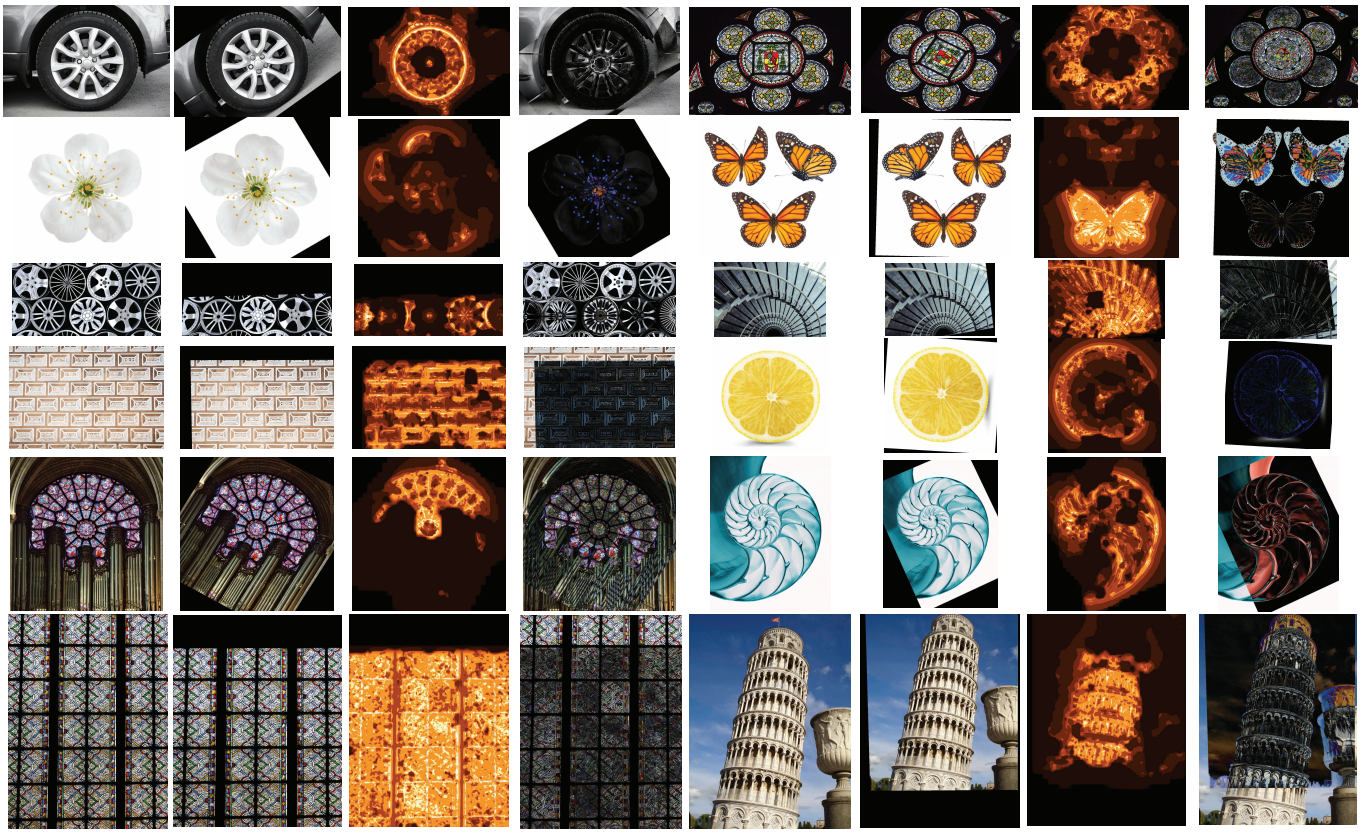


Fig. 7. A selection of symmetries detected by our method; for each example from left to right: the input image, the input image transformed according the detected symmetry, spatial support map of the detected symmetry, and difference image produced by subtracting the original input image from its transformed counterpart. In this selection, our method is able to estimate rotations (e.g., row 1, row 2 right, row 5 right), translations (row 3, row 6), and combinations of scale and rotations like the Nautilus (row 3 right, row 5 right). Many of these are captured under perspective that our technique is able to account for. Source images: © Adobe Stock

scale, rotation, and translation. In addition, we assume that this similarity transformation has a non-zero rotational component, failing which the decomposition can become under-constrained.

To obtain the rectified image we need to estimate $\mathbf{P}$ and then apply its inverse. To do that we formulate the task as a non-linear optimization problem, with the following objective function:

$$E(\mathbf{x}) = \lambda \left\| \begin{pmatrix} x_5 \\ x_8 \end{pmatrix} - \begin{pmatrix} -x_9 \\ x_6 \end{pmatrix} \right\|^2 + \sum_{i=1}^{4} ||\mathbf{H}\mathbf{v}_i - \mathbf{P}\mathbf{S}\mathbf{P}^{-1}\mathbf{v}_i||^2 \qquad (10)$$

The first term of (10) enforces $\mathbf{S}$ to be as close to a similarity as possible, and the second term ensures that the decomposition is close to the input homography $\mathbf{H}$ (measured using the re-projection error of the four image corners $\mathbf{v}_{1\ldots4}$).

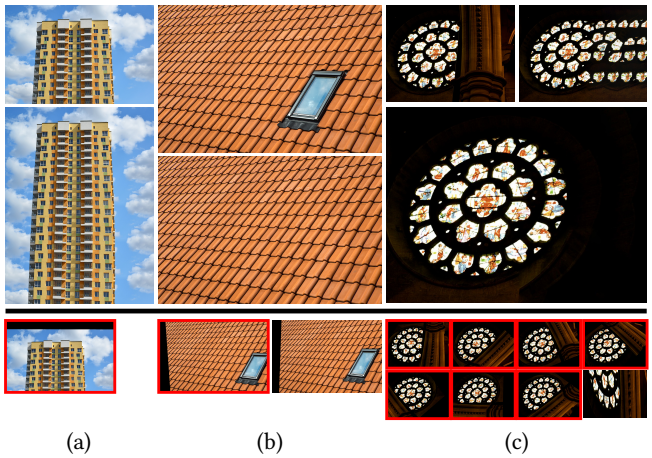
Minimizing $E(\mathbf{x})$ is feasible only when the rotation angle of the underlying similarity is away from 0° and 180°. This can be verified by finding the closest similarity $\mathbf{S}'$ to the input homography $\mathbf{H}$ and computing $\alpha = \arctan(x_8/x_5)$.When $\alpha \in (10^\circ, 170^\circ)$ we initialize $\mathbf{P}$ with the identity matrix, $\mathbf{S} = \mathbf{S}'$ and then run a non-linear optimization using the L-BFGS algorithm [Liu and Nocedal 1989; Nocedal 1980] where exact partial derivatives of $E(\mathbf{x})$ are computed using dual numbers [Piponi 2004]. To avoid getting stuck in an inappropriate local minima, we allow for greater freedom at the beginning of the optimization. We start with $\lambda = 1$ to let the algorithm explore a fruitful direction, and gradually increase it until it reaches $\lambda = 10^9$ which allows us to strictly enforce $\mathbf{S}$ to be a similarity transform. To ensure that the resulting decomposition is meaningful we use the value of $E(\mathbf{x})$. In practice, we observed $E(\mathbf{x}) < 1$ indicates a good rectification (see Fig. 8).

6.2 Image Beautification

Recently, Dekel et al. [2015] presented an algorithm to detect and enhance/suppresses non-local irregularities in input photographs. While this approach produces impressive results on many images, it exhibits bias when beautifying radially symmetric patterns, warping them into octagonal rather than circular forms. This bias is due to the translational model used to detect and represent geometric variations. We can extend their method using our more general high-level symmetries. We do this by replacing the single image input into Dekel et al.'s algorithm with an extended *image set* that contains images transformed according to the dominant symmetries

(a) Input (b) Dekel et al. [2015] (c) Our result

Fig. 9. Image beautification with detected symmetries. Input image (a), result using Dekel et al. [2015] (b, note the distortion of the shape and unpleasing contraction of thin features), and beautification obtained using Dekel et al.'s approach when the original image is extended by a set of our detected rotational symmetries (c). Source images: © Adobe Stock (top), © Isabella Josie (bottom).



(a) (b) (c)

Fig. 10. Image completion using detected symmetries under perspective. We manipulate the input images (top) to create results (middle) where the edits include retargeting/extending images (left), removing objects under perspective (middle) and even difficult tasks like removing occluders (right, symmetry-unaware result from Photoshop Content-Aware Fill in the top right inset). We also show the symmetries detected by our algorithm (bottom), with the highlighted ones being used for the completion. Source images: (a), (b) © Adobe Stock, (c) CC-BY Simon Cocks @ Flickr.

we found. Their algorithm can then sample transformed patches from symmetric locations in the nearest neighbor step. As demonstrated in Figure 9, this leads to results that preserve the global shape as well as thin structures of the input image. As an additional extension, the input images can be pre-rectified as described in Section 6.1 perform beautification under perspective distortion.

6.3 Image Completion and Editing

Patch-based image completion techniques typically cannot preserve higher-order symmetry relations and instead rely on the user to specify guiding constraints [Barnes et al. 2009; Huang et al. 2013]. Generalized PatchMatch was used in [Darabi et al. 2012] to adapt structural changes locally, however, this approach can be applied only when the region that needs to be filled is relatively small. He and Sun [2012] improved image completion using fronto-parallel translational symmetries, and Huang et al. [2014] detected translated planar patterns under perspective and used it to guide synthesis. Nevertheless, this approach is limited only to translational symmetry.

Thanks to our approach's ability to detect complex symmetries under perspective, we have a powerful tool to impose symmetry constraints in these tasks. We use a PhotoMontage-based method [Agarwala et al. 2004] to demonstrate image completion using our detected symmetries. In particular, we transform an input image with each detected symmetry transformation to create a *transformed image stack*. We also transform each symmetry transformation's spatial support map to create a corresponding *spatial support stack*. At each pixel of the edited image, we can pick samples from one of the transformed images and the corresponding transformed spatial support map indicates whether that sample is valid. We use graph cuts [Boykov et al. 2001] to compute a label map that indicates which transformed image each edited image pixel pulls its value from. The unary term is set according to our transformed spatial support maps, and the binary term is identical to that used in PhotoMontage. We combine the images using the computed label map and use Poisson blending [Pérez et al. 2003] to improve the merging. As demonstrated in Figure 10, this framework can be used to retarget images as well as to edit out some regions. The resulting image preserves the global structure of complex symmetries even in the presence of perspective effects. Most importantly, this technique is almost completely automatic, requiring the user to only specify which symmetry to use for the task.

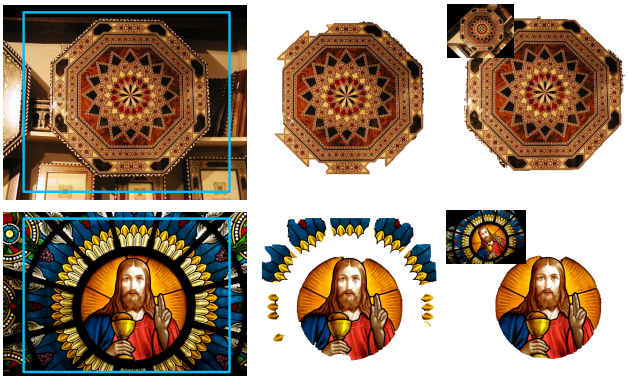
For less regular scenarios where running synthesis is necessary, the transformation matrix and fit map may be used to deduce input constraints for an approach like Tranfsormation-Guided Image Completion [Huang et al. 2013].

Finally, we can use the knowledge of symmetries to propagate edits made by the user along these symmetries, as seen in Figure 11. This is trivially achieved by allowing the user to apply one of the detected transformations, or multiplies thereof, to create transformed copies of the user's input which are then transparently fed into any editing process of choice.

6.4 Symmetry Segmentation

Symmetry detection can provide valuable prior information about the relationship between regions in an image and used to guide a segmentation algorithm. Previous approaches to image segmentation based on symmetries were either limited to a certain type of symmetry such as reflection [Sun and Bhanu 2012] or required manual annotation of specific symmetry locations [Teo et al. 2015]. Using our approach we can detect symmetries with corresponding support maps (see Fig. 7). This information can be used as a prior to constrain the global optimization process to automatically segment symmetric objects within the image. As a demonstration of this capability we implemented a simple graph cut based approach which uses the energy minimization framework developed by Boykov and Funka-Lea [2006] to perform image segmentation. We replaced the regional data term (originally based on appearance) with the spatial support maps of our detected symmetries and then run their graph cut optimization. The algorithm can either segment a symmetric object or the non-symmetric part of the input image based on the

Fig. 11. Edit propagation with detected mirror symmetry. The user draws scribbles (blue circles with white outline on the left) which are used to change the color of selected segments on the left wing. Detected mirror symmetry is applied to propagate the user scribbles to the right wing (blue circles with yellow outline), and finally the color from individual scribbles is propagated to fill the segments (right). Source image: © Adobe Stock



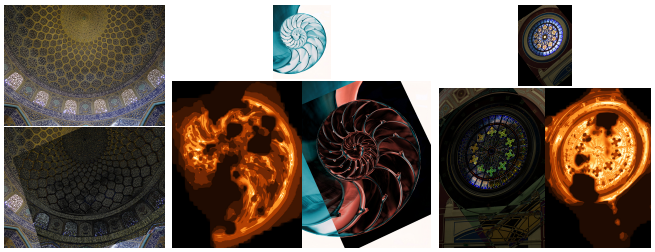
(a) Input (b) GrabCut (c) Our result

Fig. 12. Image segmentation with symmetry priors. Input image with inital segmentation rectangle for the GrabCut method [Rother et al. 2004] (a), the results from GrabCut suffer from obvious asymmetries (b), and the result of graph cut based segmentation [Boykov and Funka-Lea 2006] where our detected rotational symmetry (inset) is used as a prior for data term (c). The foreground label is assigned to extract the symmetric part (top) and the non-symmetric part (bottom). Source images: PSU symmetry dataset [Liu et al. 2013] (top), © Adobe Stock (bottom).

user's requirements. Figure 12 compares our results with the GrabCut algorithm [Rother et al. 2004] which requires a specification of an initial rectangle and then iteratively optimizes the appearance model. Our symmetry prior can significantly improve the result without additional manual intervention.

7 LIMITATIONS AND FUTURE WORK

While our method has been demonstrated to find high-quality results for many practical examples, there are scenarios which we found to be problematic. In some cases, the symmetry cannot be described by a planar homography, due to the original symmetric pattern being either non-planar (Fig. 13a), or irregular (Fig. 13b). Our method then finds a best planar fit, but this is only approximate fit. This can potentially be addressed by introducing local deviations to the planar transform representation, similar to work on near-regular translational symmetries [Hays et al. 2006; Liu et al. 2015; Park et al. 2009].



(a) (b) (c)

Fig. 13. Limitations. The dome of the mosque (a) is spherical and is not well captured by even the best planar fit. In the case of a Nautilus shell (b) our approach successfully detects the curve of the spiral, but is unable to exactly match the cell sections, as they are not perfectly regular. Illumination gradients in the window (c) generate spurious high-error areas, which result in "holes" in the spatial support map. Source images: (a) CC-BY-SA Adam Jones @ Flickr, (b) © Adobe Stock, (c) CC-BY Cat Burston @ Flickr.

Images with illumination gradients (Fig. 13c) may cause the Lucas-Kanade registration to converge to a perceptually non-optimal state, and cause "holes" in our confidence map. Gain and bias compensation could help remove these artifacts [Barnes et al. 2010].

Our registration step completely relies on the dense local symmetries computed with PatchMatch. However it is impossible to know if PatchMatch converged and found all symmetries in the image. Because of this, we run the search for a fixed number of iterations, but this still does not fully resolve the problem. Symmetries that were overlooked by the PatchMatch step will not be recovered by the later steps. Interleaving the local symmetry search with the region symmetry finding might resolve this and we leave this for future research.

Our technique is not designed to discover the primitive symmetries in an image. As a result, our method might discover composite transformations, for example, a rotation with twice the correct period, or a translation in both x and y-axes. We directly use these symmetries in our editing applications without attempting to decompose them into primitive transformations. Recovering the basic transformations is a challenging research problem that may help image understanding or reveal additional repetitions not detected by our algorithm. We leave such analysis for future work.

While the refinement of a single candidate transformation is relatively fast – about 60 seconds on average – the patch search produces an enormous number of candidates, numbering sometimes into the hundreds, that need to be examined. This may result in impractical runtimes. We hope that a better method for clustering patch search results – a difficult problem in its own right – could help reduce runtimes by reducing the number of initial candidates. Furthermore, a more robust method of classifying symmetries from non-symmetries would allow discarding candidates earlier in the process, providing additional speed-up.

8 CONCLUSION

We have presented an approach to detect generalized symmetry transformations in images captured under perspective. To our knowledge, our work is the first approach which enables completely automatic estimation of complex symmetry transformations – that

are combinations of rotations, translations and reflections – under perspective distortion. Our technique can also cope with partial occlusion and estimate regions of interest for detected symmetries. We have demonstrated the practical utility of our technique in various applications achieving results which were previously not feasible without using additional user intervention, and improving upon specific state-of-the-art methods. These promising results lead us to believe that our method has the potential to become a basic building block for various image manipulation algorithms and can help to automate tasks which were previously tractable only with extensive manual work.

ACKNOWLEDGEMENTS

We would like to thank Jakub Fišer for his help with application figures and James Pritts for providing results of his method. This research was funded by Adobe and has been supported by the Technology Agency of the Czech Republic under research program TE01020415 (V3C – Visual Computing Competence Center) and by the Grant Agency of the Czech Technical University in Prague, grant No. SGS16/237/OHK3/3T/13 (Research of Modern Computer Graphics Methods).

REFERENCES

Aseem Agarwala, Mira Dontcheva, Maneesh Agrawala, Steven M. Drucker, Alex Colburn, Brian Curless, David Salesin, and Michael F. Cohen. 2004. Interactive digital photomontage. *ACM Transactions on Graphics* 23, 3 (2004), 294–302.

Dror Aiger, Daniel Cohen-Or, and Niloy J. Mitra. 2012. Repetition Maximization based Texture Rectification. *Computer Graphics Forum* 31, 2 (2012), 439–448.

Simon Baker and Iain Matthews. 2004. Lucas-Kanade 20 Years On: A Unifying Framework. *International Journal of Computer Vision* 56, 3 (2004), 221–255.

Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. 2009. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics* 28, 3 (2009), 24.

Connelly Barnes, Eli Shechtman, Dan B Goldman, and Adam Finkelstein. 2010. The Generalized PatchMatch Correspondence Algorithm. In *Proceedings of European Conference on Computer Vision*. 29–43.

Yuri Boykov and Gareth Funka-Lea. 2006. Graph Cuts and Efficient N-D Image Segmentation. *International Journal of Computer Vision* 70, 2 (2006), 109–131.

Yuri Boykov, Olga Veksler, and Ramin Zabih. 2001. Fast Approximate Energy Minimization via Graph Cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 23, 11 (Nov. 2001), 1222–1239.

P.J. Burt and E.H. Adelson. 1983. The laplacian pyramid as a compact image code. *IEEE Transactions on Communication* 31 (1983), 532–540.

Hugo Cornelius, Michal Perďoch, Jiří Matas, and Gareth Loy. 2007. Efficient Symmetry Detection Using Local Affine Frames. In *Proceedings of Scandinavian Conference on Image Analysis*. 152–161.

Corinna Cortes and Vladimir Vapnik. 1995. Support-Vector Networks. *Machine Learning* 20, 3 (1995).

Soheil Darabi, Eli Shechtman, Connelly Barnes, Dan B. Goldman, and Pradeep Sen. 2012. Image Melding: Combining inconsistent images using patch-based synthesis. *ACM Transactions on Graphics* 31, 4 (2012), 82.

Tali Dekel, Tomer Michaeli, Michal Irani, and William T. Freeman. 2015. Revealing and Modifying Non-local Variations in a Single Image. *ACM Transactions on Graphics* 34, 6 (2015), 227.

James Hays, Marius Leordeanu, Alexei A. Efros, and Yanxi Liu. 2006. Discovering Texture Regularity as a Higher-Order Correspondence Problem. In *Proceedings of European Conference on Computer Vision*.

Kaiming He and Jian Sun. 2012. Statistics of Patch Offsets for Image Completion. In *Proceedings of European Conference on Computer Vision*. 16–29.

Jia-Bin Huang, Sing Bing Kang, Narendra Ahuja, and Johannes Kopf. 2014. Image Completion Using Planar Structure Guidance. *ACM Transactions on Graphics* 33, 4 (2014), 129.

J. B. Huang, J. Kopf, N. Ahuja, and S. B. Kang. 2013. Transformation Guided Image Completion. In *Proceedings of IEEE International Conference on Computational Photography*.

Vladimir G. Kim, Yaron Lipman, and Thomas Funkhouser. 2012. Symmetry-guided Texture Synthesis and Manipulation. *ACM Transactions on Graphics* 31, 3, Article 22 (June 2012), 14 pages.

Nahum Kiryati and Yossi Gofman. 1998. Detecting Symmetry in Grey Level Images: The Global Optimization Approach. *International Journal of Computer Vision* 29, 1 (1998), 29–45.

Seungkyu Lee and Yanxi Liu. 2010. Skewed Rotation Symmetry Group Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32, 9 (2010), 1659–1672.

Seungkyu Lee and Yanxi Liu. 2012. Curved Glide-Reflection Symmetry Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34, 2 (2012), 266–278.

David Liebowitz and Andrew Zisserman. 1998. Metric Rectification for Perspective Images of Planes. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. 482–488.

D. C. Liu and J. Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Mathematical Programming* 45, 3 (1989), 503–528.

Jingchen Liu, George Slota, Gang Zheng, Zhaohui Wu, Minwoo Park, Seungkyu Lee, Ingmar Rauschert, and Yanxi Liu. 2013. Symmetry Detection from RealWorld Images Competition 2013: Summary and Results. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

S. Liu, T. T. Ng, K. Sunkavalli, M. N. Do, E. Shechtman, and N. Carr. 2015. PatchMatch-Based Automatic Lattice Detection for Near-Regular Textures. In *Proceedings of IEEE International Conference on Computer Vision*. 181–189.

Yanxi Liu, Tamara Belkina, James Hays, and Roberto Lublinerman. 2008. Image Defencing. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Yanxi Liu, Robert T. Collins, and Yanghai Tsin. 2004. A Computational Model for Periodic Pattern Perception Based on Frieze and Wallpaper Groups. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26, 3 (March 2004), 354–371.

Yanxi Liu, Hagit Hel-Or, Craig S. Kaplan, and Luc Van Gool. 2009. Computational Symmetry in Computer Vision and Computer Graphics. *Foundations and Trends in Computer Graphics and Vision* 5, 1–2 (2009), 1–195.

Yanxi Liu, Wen-Chieh Lin, and James Hays. 2004. Near-regular Texture Analysis and Manipulation. *ACM Transactions on Graphics* 23, 3 (Aug. 2004), 368–376.

Anthony Lobay and D. A. Forsyth. 2006. Shape from Texture Without Boundaries. *International Journal of Computer Vision* 67, 1 (April 2006), 71–91.

G. Loy and J.-O. Eklundh. 2006. Detecting Symmetry and Symmetric Constellations of Features. In *Proceedings of European Conference on Computer Vision*. 508–521.

Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. 1953. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics* 21, 6 (1953), 1087–1092.

Niloy J. Mitra, Leonidas J. Guibas, and Mark Pauly. 2006. Partial and Approximate Symmetry Detection for 3D Geometry. *ACM Transactions on Graphics* 25, 3 (July 2006), 560–568.

Niloy J. Mitra, Mark Pauly, Michael Wand, and Duygu Ceylan. 2012. Symmetry in 3D Geometry: Extraction and Applications. In *Eurographics 2012 - State of the Art Reports*, Marie-Paule Cani and Fabio Ganovelli (Eds.).

Jorge Nocedal. 1980. Updating quasi-Newton matrices with limited storage. *Math. Comp.* 35, 151 (1980), 773–782.

Nobuyuki Otsu. 1979. A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man and Cybernetics* 9, 1 (1979), 62–66.

Minwoo Park, Kyle Brocklehurst, Robert T. Collins, and Yanxi Liu. 2009. Deformed Lattice Detection in Real-World Images Using Mean-Shift Belief Propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31, 10 (2009), 1804–1816.

Mark Pauly, Niloy J. Mitra, Johannes Wallner, Helmut Pottmann, and Leonidas J. Guibas. 2008. Discovering Structural Regularity in 3D Geometry. *ACM Transactions on Graphics* 27, 3, Article 43 (Aug. 2008), 43:1–43:11 pages.

Patrick Pérez, Michel Gangnet, and Andrew Blake. 2003. Poisson image editing. In *ACM Transactions on Graphics*, Vol. 22. 313–318.

Dan Piponi. 2004. Automatic Differentiation, C++ Templates, and Photogrammetry. *Journal of graphics, GPU, and game tools* 9, 4 (2004), 41–55.

J. Pritts, O. Chum, and J. Matas. 2014. Detection, Rectification and Segmentation of Coplanar Repeated Patterns. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. 2973–2980.

C. Rother, V. Kolmogorov, and A. Blake. 2004. GrabCut: interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics* 23, 3 (2004), 309–314.

Yu Sun and Bir Bhanu. 2012. Reflection Symmetry-Integrated Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34, 9 (2012), 1827–1841.

Ching Lik Teo, Cornelia Fermüller, and Yiannis Aloimonos. 2015. Detection and Segmentation of 2D Curved Reflection Symmetric Structures.. In *Proceedings of IEEE International Conference on Computer Vision*. 1644–1652.

Tinne Tuytelaars, Andreas Turina, and Luc J. Van Gool. 2003. Noncombinatorial Detection of Regular Repetitions under Perspective Skew. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25, 4 (2003), 418–432.

Huisi Wu, Yu-Shuen Wang, Kun-Chuan Feng, Tien-Tsin Wong, Tong-Yee Lee, and Pheng-Ann Heng. 2010. Resizing by Symmetry-Summarization. *ACM Transactions on Graphics* 29, 6 (2010), 159:1–159:9.

Z. Zhang, A. Ganesh, X. Liang, and Y. Ma. 2012. TILT: Transform Invariant Low-Rank Textures. *International Journal of Computer Vision* 99, 1 (2012), 1–24.