Repetition Maximization based Texture Rectification

Dror Aiger   Daniel Cohen-Or   Niloy J. Mitra

ESIEE / Google Inc.   Tel Aviv University   UCL

Abstract

Many photographs are taken in perspective. Techniques for rectifying resulting perspective distortions typically rely on the existence of parallel lines in the scene. In scenarios where such parallel lines are hard to automatically extract or manually annotate, the unwarping process remains a challenge. In this paper, we introduce an automatic algorithm to rectifying images containing textures of repeated elements lying on an unknown plane. We unwrap the input by maximizing for image self-similarity over the space of homography transformations. We map a set of detected regional descriptors to surfaces in a transformation space, compute the intersection points among triplets of such surfaces, and then use consensus among the projected intersection points to extract the correcting transform. Our algorithm is global, robust, and does not require explicit or accurate detection of similar elements. We evaluate our method on a variety of challenging textures and images. The rectified outputs are directly useful for various tasks including texture synthesis, image completion, etc.

1. Introduction

Textured surfaces are often rich in repetitions. Photographs of such surfaces commonly introduce various distortions due to camera projections. Such distortions disturb metric properties, i.e., relations involving angles and lengths, misrepresent repetitions present in the original scenes, and make image space analysis difficult. Hence, rectifying such distorted images is an essential first step for many computer graphics and computer vision tasks. Rectified textures can then be used for texture synthesis, image completion, etc.

Common rectification strategies require the user to manually mark an image space quadrilateral corresponding to a world space rectangle with known aspect ratio, or to identify sets of potential parallel lines. Providing such manual annotations can be tedious and even error prone especially in images without dominant linear elements (see Figure 1). Further, such a strategy only makes use of user-annotated local information. Although, for *low-rank* images, a sparse matrix based rectification [ZGLM10] can be very effective, most of our target images do not fall in this category.
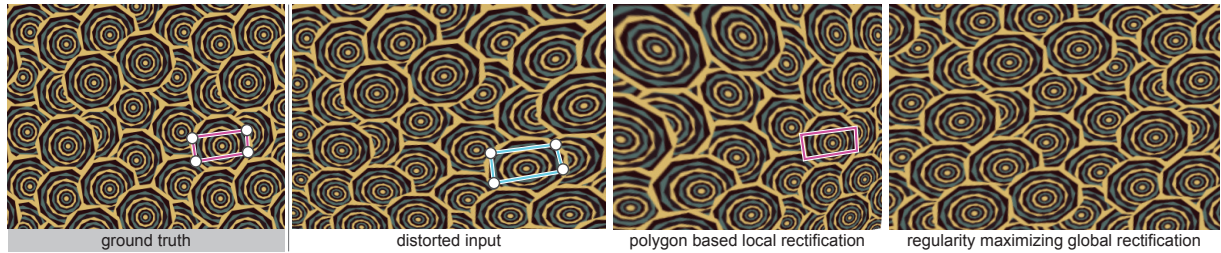
We correct images with (approximately) repeated elements, which are coplanar in the original scene, by searching for an allowable correcting transform that maximizes repetitions in the output, *without* explicitly solving for correspondence across the repeated image elements. Unlike existing methods, we exploit global clues across the input to produce robust results, even when the repetitions are only approximate.

Figure 1: *The distorted texture (top) is automatically unwarped (bottom) using a repetition maximizing rectifying transform. Our algorithm does not rely on the availability of vanishing lines or on any manual annotations.*

First, we use region descriptor features, e.g., image segments using statistical region merging [NN04], to extract candidate regions that are potentially similar in the original scene. Unwarping the image then amounts to searching for a transformation that adjusts the image to maximize the repetitions in the output, while being resilient to outliers. Note that we neither require the elements to be arranged in any regular grid, nor does the repetitions have to be exact (see Figure 1).

We map the above intuition into a computationally efficient procedure, alternately solving for pure projective and pure affine transforms (see also [LZ98]). Specifically, we map line segments extracted from the regional descriptors to *sur-*

| ground truth | distorted input | polygon based local rectification | regularity maximizing global rectification |

Figure 2: *Manual rectification is difficult in an image with no clear vanishing lines. Further, even when the user marks a repeated element (blue quad Q_d) along with a rectified base element (purple quad Q_o) on the ground truth texture, solving for rectifying transform T, such that $T(Q_d) = Q_o$, only uses local information, potentially distorting other regions. In contrast, our algorithm globally rectifies multiple repetitions, even when the repetitions are only approximate.*

faces in the respective transform domains. Solving for a rectifying transform then amounts to detecting specific patterns among the intersection points of triplets of such surfaces. We use consensus statistics to the set of (projected) intersection points to extract the unwarping transform. We evaluate the algorithm on a range of synthetic and real-world textures with noise, outlier, and missing data.

1.1. Related Works. Rectification of images distorted is important in many image analysis and texture synthesis applications (see [ZGLM10] and references therein). The most common image rectification method involves manually marking an image space quadrilateral corresponding to a world space rectangle with known aspect ratio [HZ00]. Other related attempts involve detecting finite vanishing points in the image, and searching for transforms to map the vanishing points to infinity, thus restoring image space parallelism among lines that are originally parallel in the world scene. Vanishing points of an image are detected either with the help of user annotations, or by searching for bundles of lines passing through a common point.

In a highly influential work, Liebowitz and Zisserman [LZ98] use a two-tiered approach to metric rectification that starts with affine rectification by recovering the vanishing line of the plane, and then achieve metric rectification by enforcing constraints based on *priors* involving known angles, equal angles, known length ratios of line segment groups, etc. Such methods, however, are inapplicable for rectification of textured images without dominant linear features or without explicit information about corresponding points, as is the focus of our work.

Criminisi and Zisserman [CZ00] propose a rectification algorithm for special regular textures, i.e., the world space planar texture elements being arranged on a grid. The algorithm first searches for vanishing points of the plane, and then uses two one-parameter searches to estimate the degrees of freedom of the vanishing line. Subsequently, Clark and Mirmehdi [CM01] detect the horizontal vanishing point of the text plane using an extension of 2D projection profiles, and use them for rectification of text images. Recently, Wu et al. [WFP10] introduce a rectification and symmetry

detection algorithm for urban facades observing that dominant repetitions are often along vanishing point directions. Chum and Matas [CM10] present an algorithm for affine rectification of a plane exploiting knowledge of relative scale changes with applications towards text rectification, detection of repetive patterns. These methods cannot be directly extended to rectify arbitrary textures or when vanishing directions are difficult to identify.

Motivated by early observations on texture gradient by Gibson [Gib69], researchers use distortion among texture elements to estimate local surface orientation. Usually foreshortening and scaling of texture elements are used to estimate affine transforms, and then estimate surface normals that are subsequently integrated to produce a smooth surface (see [MR97, LH05, LF06] and references therein). Such shape-from-texture approaches assume isotropy, homogeneity (i.e., distance between elements and their placement pattern is consistent), or reliable detection of (distorted) texture elements — assumptions that are often violated for textures with irregular repetitions, along with overlapping and outlier elements. Alternately, one can search for special arrangements of primitives in an image. For example, an arrangement of concentric circles, or simply a collection of circles on the world plane, maps to a set of image space ellipses, which can then be reliably detected even in the presence of overlaps. Based on this observation, Lourakis [Lou09] present a rectification algorithm by optimizing for a homography that maps ellipses to coplanar circles. Park et al. [PBCL09] propose an interesting MRF formulation to detect deformed lattices in real-world scenes. The method, however, deals in world-space distorting and not with camera distortions as is the focus of this work.

Recently, Zhang et al. [ZGLM10] propose TILT rectification as an elegant approach using a sparse matrix approximation for rectification. The algorithm assumes the undistorted input to have a low rank, and thus rectification amounts to searching for an allowable transformation that minimizes the rank of the resultant image. However, many textures do not belong to such a 'low-rank' class, especially images with outlier features, imperfect repetitions, etc. leading to undesired artifacts (see Figure 3 and supplementary material).

distorted input   our rectification result   'low-rank' rectification
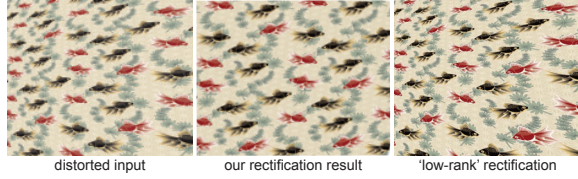
Figure 3: *Comparison of automatic rectification using our algorithm and TILT [ZGLM10]. Fishes on the left and on the right can be used for evaluating the extent of unwarping.*

2. Rectification Algorithm

We present an automatic rectification algorithm for images of scenes containing similar or repeated coplanar elements. Due to perspective distortion, in the projected image these similar elements appear different depending on their original orientations and distances from the camera. Technically, our goal is to extract a four parameter metric rectification up to four parameter similarity ambiguity (see Section 2.1), alternately optimizing for pure projective and pure affine components. True invariants under perspective and illumination variations, however, are difficult to extract. Hence, we do not directly rely on feature detectors like SIFT (see Figure 15) to extract potential repeated counterparts, but gather a set of candidate regions using a statistical segmentation procedure [NN04]. Further, we use consensus to extract the rectifying transformation based on a formulation that performs matching in the rectified domain.

Given an image I, we solve for a rectifying transform that maximizes repetitions, while factoring out image space translations and rotations across repetitions, i.e., we look for the transform T^* in the rectifying transformation family $\mathcal{G}$ such that repetition is maximized. Note that $\mathcal{G}$ depends on the prior for the source of distortion, e.g., foreshortening, wide angle distortion, etc.

One possibility is to manually identify (i) three repeated polygon instances, e.g., rectangles or trapezoids, on the input image, or (ii) one polygon on the input image and the corresponding ground truth polygon, and rectify the image to bring the polygons into agreement. When the differences between neighboring elements are small, however, such a method can be unstable and result in *only* locally consistent rectification. In Figure 2, we show that even when correspondence is perfect (manually provided in this example), the rectification is unsatisfactory, especially due to approximate repetitions among the undistorted elements.

Overview. Given a distorted image, we extract a set of candidate congruent line segments, which are coplanar in the world space (see Section 2.2). Our algorithm (see Figure 4) then rectifies the distorted input by restoring congruency among these candidate lines. Specifically, we map each line segment to a surface in a transform space and extract the intersection points between triplets of such surfaces. Reasoning on the patterns of such intersection points, we recover

transform space surfaces   projected intersection points   rectifying transform
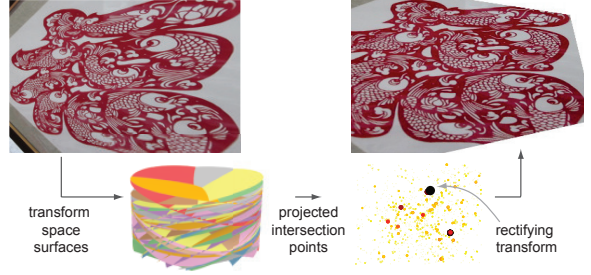
Figure 4: *Overview of our algorithm.*

the rectifying transform, alternately correcting for pure projective and pure affine components. We then generalize the procedure to handle images with repeated elements.

2.1. Rectifying Images with Congruent Line Segments

Assume that the projected set of line segments is represented by $S := \{s_1, s_2, \dots\}$. We use Euclidean length equality as the measure of repetition for line segments, which are congruent under rigid transforms in the world space. Thus, we look for the rectifying transform that brings the maximum number of line segments in S to equal but unknown length d up to an ϵ-approximation margin. Our goal is to find the transform that brings the largest numbers of such segments to an (unknown) equal length.

A naïve approach is to use a brute-force search over a 4-dimensional transformation space parameterizing a rectifying transform (up to a similarity ambiguity). Instead, we decompose the mapping into pure projective and pure affine parts, each having two degrees of freedom (see also [LZ98]), and alternately solve for them. One option is to use non-linear optimization to directly solve for the free variables. Unfortunately, we found such an approach (using Levmar [Lou04]) to be unreliable and requires frequent restarts (order of 100 restarts in our tests) in absence of a good initialization. Instead, we use the geometric structure of the problem to recover the respective rectification components.

Pure projective transform. First, we address the case of a single set of originally congruent line segments, say of (unknown) length d under a pure projective transform,

$$P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l_1 & l_2 & 1 \end{bmatrix}$$

with l_1, l_2 denoting the homogeneous parameters of the line at infinity. Let any segment s be represented by its end points (x_1, y_1) and (x_2, y_2), with the P-transformed version being $s' = (x_1', y_1', x_2', y_2')$, i.e., $P(s) \rightarrow s'$. Mapping back to cartesian coordinates we have,

$$x_1' = \frac{x_1}{l_1 x_1 + l_2 y_1 + 1} \qquad y_1' = \frac{y_1}{l_1 x_1 + l_2 y_1 + 1}$$
$$x_2' = \frac{x_2}{l_1 x_2 + l_2 y_2 + 1} \qquad y_2' = \frac{y_2}{l_1 x_2 + l_2 y_2 + 1}.$$
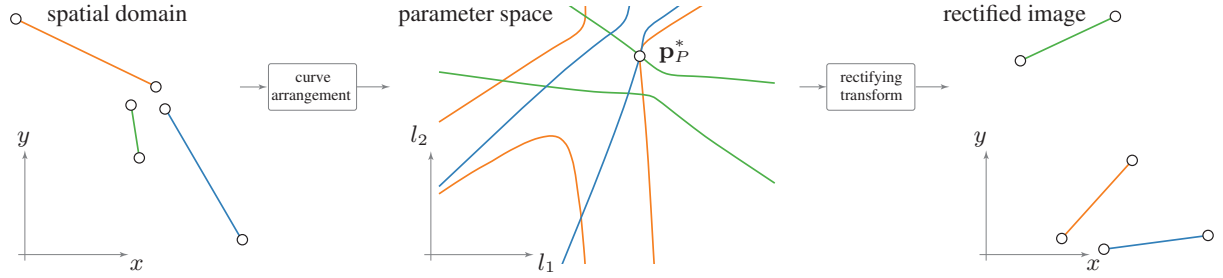
Figure 5: *Given a set of line segments (left), we search for a pure projective rectifying transform that makes the segments congruent, i.e., maps the segments to ones with equal length. (Middle) We map each segment s to a curve $c_P(s)$ in the (l_1, l_2) projective parameter space using its implicit form as in Equation 1. (Right) The rectifying transform, which corresponds to the point $\mathbf{p}_P^*$ through which maximum number of such curves passes, is then used to rectify the input image.*

Next, using the constraint that each line segment s in the rectified image has length d, we map s to a curve $c_P(s)$ in the (l_1, l_2)-plane (see Figure 5). Solving for the pure projective transform that takes the maximum number of segments to segments of length d amounts to finding the point in the (l_1, l_2)-plane that lies on the maximum number of curves $c_P(s_i)$ for all $s_i \in S$, up to an ϵ-margin.

Using shorthand $\phi(l_1, l_2) := \frac{1}{l_1 x_1 + l_2 y_1 + 1}$ and $\psi(l_1, l_2) := \frac{1}{l_1 x_2 + l_2 y_2 + 1}$, each segment $s \to s'$, i.e., $(x_1, y_1) \to (x_1', y_1')$ and $(x_2, y_2) \to (x_2', y_2')$ along with the condition that the mapped segment length equals d, we get an implicit representation of the curve $c_P(s)$ as:

$$
\begin{aligned}
d^2 &= (x_1' - x_2')^2 + (y_1' - y_2')^2 \\
&= (\phi x_1 - \psi x_2)^2 + (\phi y_1 - \psi y_2)^2
\end{aligned}
$$

$$
\Rightarrow c_P(s) : \phi^2(x_1^2 + y_1^2) \quad + \psi^2(x_2^2 + y_2^2) \quad (1)
$$
$$
-2\phi\psi(x_1 x_2 + y_1 y_2) = d^2,
$$

where ϕ and ψ represent $\phi(l_1, l_2)$ and $\psi(l_1, l_2)$, respectively.

Pure affine transform. In the case of pure affine transform A, the solution is similar, but simpler. Let,

$$
A = \begin{bmatrix} a & b & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}
$$

be a pure affine transformation applied in the homogenous $2D$ space and parameterized by the variables a and b. Using the notation that a transform A maps any segment $s = (x_1, y_1, x_2, y_2)$ to $s' = (x_1', y_1', x_2', y_2')$, we get

$$
\begin{aligned}
x_1' &= a x_1 + b y_1 & y_1' &= y_1 \\
x_2' &= a x_2 + b y_2 & y_2' &= y_2.
\end{aligned}
$$

Similar to the pure projective case, using the condition that the mapped segment length equals d, we get an implicit representation of the curve $c_A(s)$ as

$$
\begin{aligned}
d^2 &= (x_1' - x_2')^2 + (y_1' - y_2')^2 \\
&= (a(x_1 - x_2) + b(y_1 - y_2))^2 + (y_1 - y_2)^2
\end{aligned}
$$

$$
\Rightarrow c_A(s) : a(x_1 - x_2) \quad + b(y_1 - y_2) \quad (2)
$$
$$
\pm\sqrt{d^2 - (y_1 - y_2)^2} = 0.
$$

Thus, for each segment s in the plane, the corresponding curve $c_A(s)$ in the (a, b)-plane is a pair of lines (see Figure 6). To optimize the global measure, we search for matrix A^*, i.e., parameters a, b, that transforms the maximum number of segments in S to the same length d, up to an ϵ-margin.

Multi-d optimization. Given a set of line segments S, for each $s_i \in S$ under pure projective transform, we have a surface $c_P(s) : Q(l_1, l_2) = d^2$ in the (l_1, l_2, d)-space that characterizes all the (l_1, l_2)-s that bring the segment s_i to unknown length d using pure projection transforms (see Equation 1). Alternately, if there are line segments originally congruent, i.e., of the same length in the undistorted image, then their corresponding Q-surfaces must share common intersection points in the (l_1, l_2, d)-space. Further, if there are multiple congruent sets of segments, say set S_1 of original segment lengths d_1, and set S_2 of original segment length d_2, then all the surfaces corresponding to the segments in S_1 will pass through $(\hat{l}_1, \hat{l}_2, d_1)$, and all the surfaces corresponding to the segments in S_2 will pass through $(\hat{l}_1, \hat{l}_2, d_2)$, where $(\hat{l}_1, \hat{l}_2)$ is the pure projective rectifying transform. Thus, for a given image, although multiple clusters of congruent segments have different intersection points in the (l_1, l_2, d)-space, all the intersection points have the same projected foot-point $(\hat{l}_1, \hat{l}_2)$ on the (l_1, l_2)-plane (see
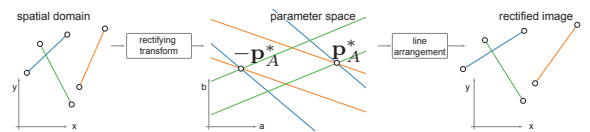
Figure 6: *(Left) Given a set of line segments, we search for a pure affine rectifying transform. (Middle) We map each segment s to a pair of lines $c_A(s)$ in the (a, b) affine parameter space using Equation 2. (Right) The rectifying transforms $\pm\mathbf{p}_A^*$, which correspond to the points through which maximum number of such lines passes, are then used to rectify the input image (the solutions are equivalent up to reflection).*

Figure 7). In case of pure affine transform, the surfaces are simply planes (see Equation 2).

Assume that the number of the segments in each cluster S_i, which under some optimal transformation maps to segments of unknown length d_i, is a constant fraction of N, with N being the total number of line segments. Then, there are $O(N^3)$ surface triplets, each of which intersects at a point in the $(\hat{l}_1, \hat{l}_2, d)$-space with the same projected foot-print $(\hat{l}_1, \hat{l}_2)$ characterizing the rectifying transform. A random sampling analysis involving Chernoff bound (cf. [MR95]) shows that sampling only a *constant* number of surface triplets, computing their intersection points, if any, and projecting the intersections to the (l_1, l_2)-plane, give, with high probability, a good approximation of the optimal $(\hat{l}_1, \hat{l}_2)$ rectifying transform. We now describe the details.

First, we prune out surface triplet intersections that have no neighbors in the (l_1, l_2, d)-space within a threshold distance using a range query data structure [AMN*94]. In this stage, a conservative threshold is sufficient to prune out only the clear outliers (default value included in supplementary demo). Next, we perform another clustering in the projected (l_1, l_2)-plane. Let, $\mathbb{P}$ denote the set of all the remaining points in (l_1, l_2, d)-space, and let $\mathbb{P}_p$ be their projection on the (l_1, l_2)-plane. Accounting for an approximation margin δ, we have to find $\mathbf{p}^* \in \mathbb{P}_p$ such that the number of points in a rectangle of side 2δ centered around $\mathbf{p}^*$ is maximized. Such a point $\mathbf{p}^*$ corresponds to the pure projective transform that restores congruency among the maximum number of line segments.

The above optimization can be performed by explicitly intersecting the surface triplets. However, in the pure projective case it is cumbersome to analytically express the intersections of associated $Q(l_1, l_2)$ surfaces. Instead, we use an approximation to compute intersections of surface triplets in constant time. We partition the (l_1, l_2)-plane into quad-cells, and in each cell, approximate each $Q(l_1, l_2)$ function using a linear function in l_1, l_2. In such a cell c, the approximation $\tilde{Q}^c(l_1, l_2)$ is computed using a least squares plane fit to a constant number (4×4 in our experiments) of $(l_1, l_2, Q(l_1, l_2))$ samples in the cell. Although one can use an error-bounded adaptive partitioning strategy, in all our examples we found a fixed partitioning to be sufficient (see supplementary demo). Now, given any surface triplets $\{Q_1, Q_2, Q_3\}$, in *constant time* we approximate their intersection points, i.e., elements of $\mathbb{P}$, using the analytic intersection points between their corresponding fitted planes $\{\tilde{Q}_1^c, \tilde{Q}_2^c, \tilde{Q}_3^c\}$ over each cell c in (l_1, l_2), if the intersection lies in the cell c. For pure affine, the original surfaces being simply planes, we directly solve for their intersections without partitioning the (l_1, l_2)-plane. We project the recovered intersection points to the (l_1, l_2)-plane to get point set $\mathbb{P}_p$.

Next, we build an efficient range data structure [AMN*94] for points $\mathbb{P}_p$, and for each point in $\mathbb{P}_p$ we count the number of points inside a rectangle of size 2δ by querying the

data structure in logarithmic time. Our approximate optimum $(\hat{l}_1, \hat{l}_2)$ is then the point with maximum numbers of neighbors (see Figure 7). Thus, the total running time to approximate the rectifying transform $\mathbf{p}^*$ is $O(N' \log N')$, including the time to build the range data structure, where N' denotes the constant number (100 in our experiments) of random triplets drawn from $O(N^3)$ possibilities.

Seemingly, an alternate RANSAC strategy can take a random set of three segments, solve for rectifying transforms, retain the best transform, and then improve the solution using a local optimization. However, our experiments indicate the following shortcomings of the method: (i) for each triplet of segments to recover the rectifying transform parameters we still have to use the same transform domain approach as described above; (ii) each of the computed rectifications uses only three line segments and can be fragile since we allow approximately repeated elements in the input texture (see Figure 2); (iii) the local optimization is brittle. Instead, our proposed algorithm involving the two stage clustering once in the (l_1, l_2, d) and later in the projected (l_1, l_2)-spaces is resilient to outliers without significant computational overhead. Note that our algorithm is reminiscent of Hough transform due to our transform domain analysis.

General perspective transforms. A general plane perspective transform or homography (H) can be decomposed as, $H = SAP$, where S, A, P respectively denote similarity,
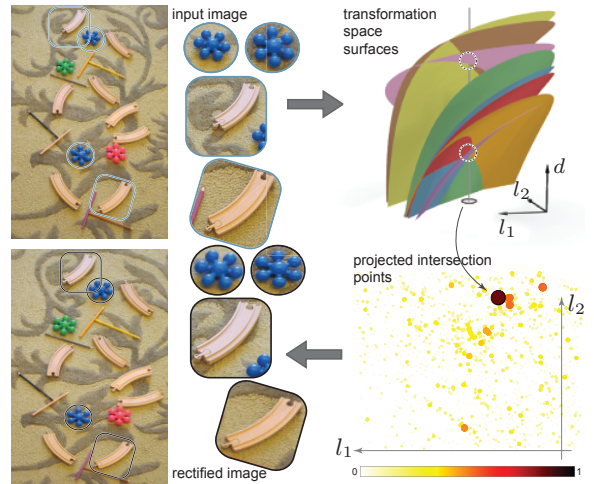
Figure 7: *We show two pairs of similar objects before (top-insets) and after (bottom-insets) rectification. The best rectification parameters (l_1, l_2) transform both instances to congruent ones. Line segments extracted from the input image are mapped to surfaces in the transform space, and intersection points for such surface triplets are projected to the transform parameter space. (Bottom-right) Projected points are color coded and scaled based on local point density. The densest region, in dark, represents parameters of the recovered pure projective transform for the real-world image.*

distorted input   two stage optimization   4-parameter optimization
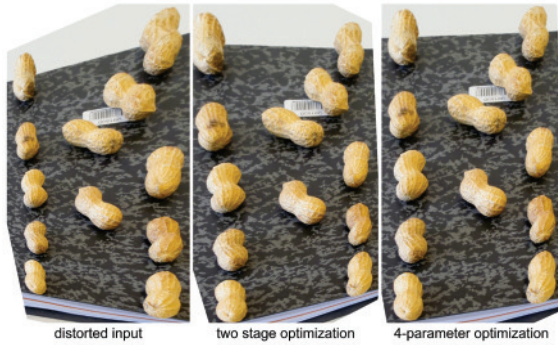
Figure 8: *Comparison between alternating pure projective and pure affine rectifications (2 iterations) vs. computationally inefficient full rectification using 4-parameters.*

pure affine, pure projective transforms [LZ98]. Thus, up to similarity ambiguity, we have a four parameter optimization, two each for the affine and the projective parts. One can express the congruency condition for rectifying a set of line segments in terms of the fours parameters (and d), similar to Equations 1 and 2, and look for a combination that maximizes the number of surfaces passing through the point. The approach, however, is computationally inefficient as the parameter space involves four variables, resulting in a collection of hyper-surfaces in five dimensions. Instead, we use an iterative solution, alternately optimizing for the pure projective and pure affine parts. The individual optimizations are robust against outliers as we rely on a voting based clustering approach. We found this alternating strategy to work well in practice, stopping after a fixed number of iteration steps, 2 in all our experiments. In Figure 8 we compare the rectification results obtained using our full optimization vs. alternating between pure projective and pure affine rectifications. In the full optimization, the algorithm is almost identical, except computing intersections among surface triplets are expensive. Empirically, the full optimization, even for 30-50 lines segments, is about 15-20x slower than the alternating pure projective/affine iterations. Unless stated, we use the alternating scheme for our results.

2.2. Rectifying an Image with Repeated Content

We now describe how to extract a set of line segments S from an image I with repeated content, such that our rectification algorithm for a set of originally congruent line segments (Section 2.1) can be used to rectify the input image. Since defining invariant features robust to large projective distortions, noise, and to outlier elements is difficult (cf. [HO07]), we instead focus on regional descriptors and account for outlier elements using a consensus mechanism.

We use Statistical Region Merging [NN04] (using publicly available source code) to obtain regional segments that maps the segmentation problem to a robust and efficient inference problem. The algorithm is known to be resilient to noise,


distorted input   regional descriptors   rectified output
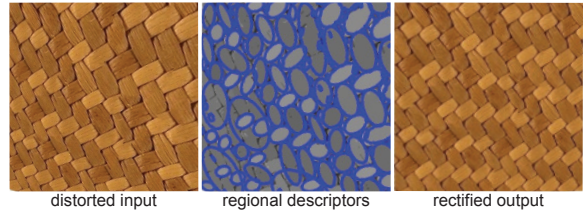
Figure 9: *We use statistical region merging to extract regional segments on an input image, and then fit an ellipse to each extracted segment. Note that the ellipses can have overlaps, vary in size, and also capture outlier elements.*

handles occlusions, and can simultaneously capture large to fine scale features. We least squares fit an ellipse to each such region (see Figure 9). Let, $E_s = \{s_1, s_2, \dots\}$ denote the major axes lengths of *all* such fitted ellipses. We now rectify the line segments E_s, alternately solving for pure projective and pure affine parts, as detailed in Section 2.1. It is conceivable to use other affine invariant detections (cf. [MTS*05]). For example, in our experiments component tree [NC06] and MSER [MCUP02] produced comparable results.

Since major axes of ellipses are *not* invariant under projective transforms, we use a weak perspective assumption and identify potential repeated elements by our two stage clustering approach. Although we are not guaranteed to find the exact optimal, recomputing the ellipses for the updated regional descriptors in each iteration works well in practice (see results and demo).

Discussion. *Homography* refers to a transformation, which under perspective projection, maps a plane to another plane.

affine rectification

metric rectification
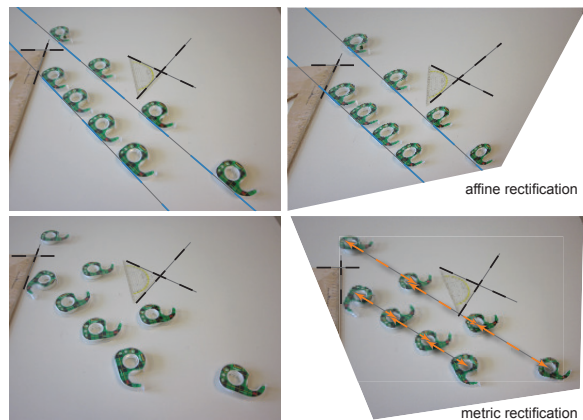
Figure 10: *(Top) Our algorithm achieves affine rectification when the input real-world scene contains coplanar repetitions only under translations. Parallelism is restored, but angle and metric relations remain distorted. (Bottom) For scenes with repetitions under translation and rotation (left), our algorithm produces metric rectification restoring both angles and lengths. For visualization, we overlay lines and arrows to highlight interesting relations.*
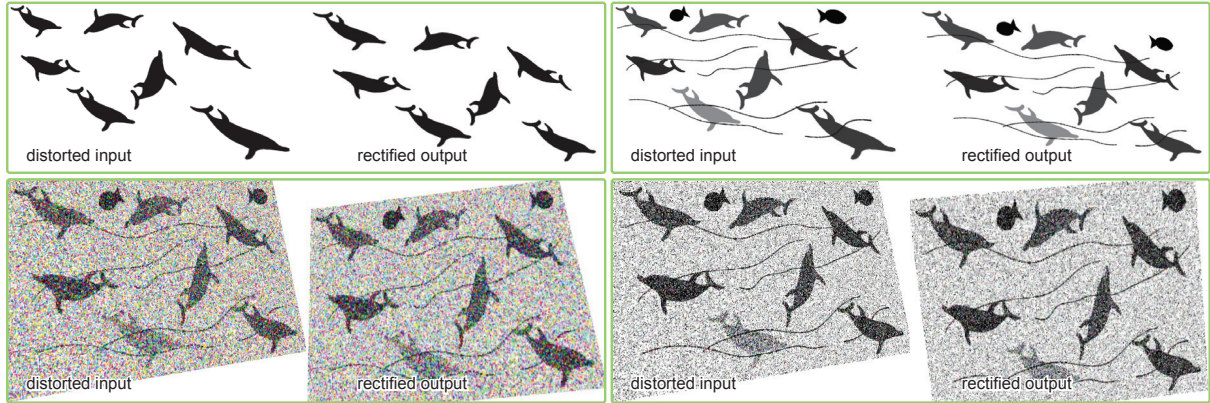
Figure 11: *Our rectification algorithm is robust to intensity change, noise and outliers. The base image (not shown) was synthesized using rotated, reflected, translated copies of a dolphin icon. In each case, noise and outlier elements were added, and the resultant image was distorted. In the corresponding rectified output, the dolphins become (near) congruent.*

The process of solving for a homography to map the image of a planar surface to a fronto-parallel view, thus removing all in-plane projective distortions, is known as *metric rectification*, except for a scale and rotational ambiguity. Such rectified images allow direct measurement of quantities including angles, lengths, and areas. *Affine rectification*, a more relaxed rectification procedure, only restores parallelism relations of the original scene by ensuring all vanishing points to be at infinity.

Observation #1. A set of purely *translated* copies of a line segment, transformed by a general perspective transformation (affine and pure projective), *cannot* be metric rectified by finding a transformation that brings the segments to the same length, as reasoned next: Let S be a set of translated copies of the same segment. Since any affine transformation A preserves parallelism, A also preserves the equality of length, i.e., $A(S)$ also contains only translated copies of line segments. Hence, we cannot uniquely recover the affine part, which is ambiguous (see Figure 10; cf. [LF06]).

Observation #2. A set of congruent copies of a line segment, with at least three of the segments in *general* arrangement, transformed by a general perspective transformation (pure affine and pure projective), *can* be metric rectified by finding a transformation that brings the segments to the same length, as reasoned next: Due to observation #1, we only have to show that there is no affine ambiguity in this case. Let S be a set of segments, originally of the same length, with at least three of them not pairwise parallel. If three (originally) equal length segments appear in general positions in the world coordinates, the affine parameters (a, b) for the rectifying transform (Equation 2) that bring them to the same length is the projection of the point of intersection of three planes in the (a, b, d)-space to the (a, b)-plane. Since the three plane equations are independent, we get a unique (a, b, d) solution, yielding the desired affine rectification parameterized by (a, b) (see Figure 10).

Finally, one can refine the solution using a non-linear optimization that explicitly searches over the 4-parameters to restore congruence among the rectified line segments (say, using [Lou04]). Since at this stage we already have a good solution, such a local refinement leads to visible improvements only in some cases.

3. Evaluation

We tested our method on a variety of input images and textures with varying amount of complexity and different types of repetitions, structured or otherwise. We first tested our method on a synthetic data set, where the ground truth data was available for comparison. We added noise and outlier elements to the input images, and verified the quality of the extracted rectifying transform (see Figure 11). Although the noise and outliers corrupt the extracted contours and resultant fitted ellipses, the unwarping results remain stable due to the consensus voting in the rectification extraction stage. Numerically, when $l_1 = l_2 = 1 \times 10^{-3}$ for pure projective transform the recovered values are $(1 \times 10^{-3}, 6 \times 10^{-4})$, $(8 \times 10^{-4}, 1 \times 10^{-3})$, respectively (top-row); and when $l_1 = -l_2 = -5 \times 10^{-4}$ the recovered values are $(-6 \times 10^{-4}, 6 \times 10^{-4})$, $(-5 \times 10^{-4}, 8 \times 10^{-4})$, respectively (bottom-row).
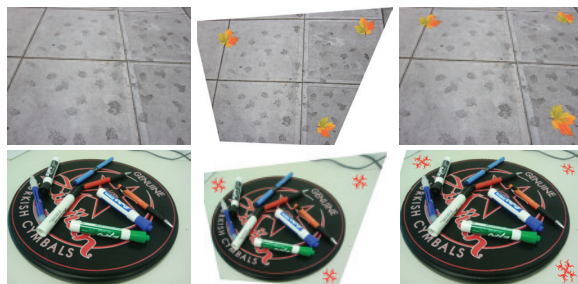
In order to test our algorithm on synthetic textures, we applied large distortions on well known texture patterns, and applied our algorithm (see Figure 12). In all cases, we observe that the rectification is satisfactory. As discussed, in case of translational patterns, we observe affine ambiguity. In most of these examples, vanishing lines are hard to detect or even to manually annotate. Alternately, a user may mark three of more repeated instances, but even then the rectification results are inferior due to use of only local information (see Figure 2), especially in presence of approximate repetitions. We further tested our algorithm on images with repeating elements (see Figure 12-bottom). Note that, in both

distorted input   rectified output   distorted input   rectified output

Figure 12: *(Top two rows) Automatic unwarping using the proposed algorithm on various well known synthesized textures (source: http://www.cc.gatech.edu/cpl/projects/graphcuttextures/). The textures were (synthetically) distorted by the same perspective transform. (Bottom) Rectification results on real-world images. See also supplementary demo and results.*

cases, the elements have non-negligible depth (compared to the size of the repeating elements) and the repetitions are quite approximate. The rectified outputs are satisfactory in both instances. In the chilli-pepper example, the quality of the rectification can be judged by how well the circles have been restored (we verified that the line segments were extracted based on the ellipses coming from the peppers). In Figure 13 we rectify the input (complex) images, add repeated patterns to the images, and then re-warp the images with the recovered projective transforms. Note that we do not require the input images to be segmented. Please refer to supplementary material for further examples and demo.
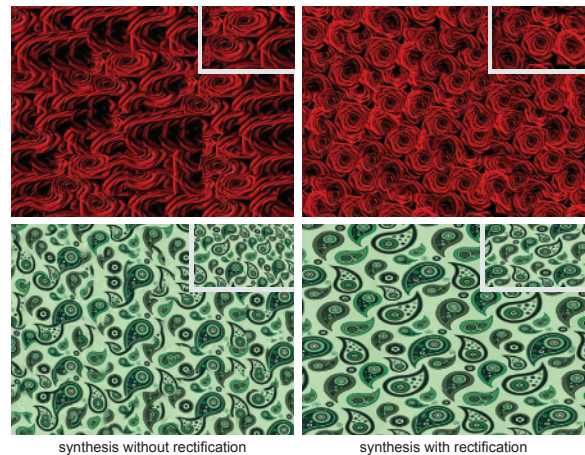
Performance. Our algorithm typically runs in less than half a minute on a standard laptop (3GHz Intel processor, 2GB RAM) for images with maximum dimension of 800 (downsampled if needed), including time to compute regional descriptors. Most of the computation time is spent in intersecting triplets of surfaces in pure projective case. The key parameter in our algorithm is the scale parameter for the extrac-

tion of regional descriptors. Although all our examples are with default setting (in author implementation of [NN04]), we can also perform a line search along the parameter, in each case compute the rectifying transform, and then retain the one with the best rectification based on density of points in the transform domain (see -q option in demo).

Texture Synthesis. Traditionally, texture synthesis deals with computationally expanding a small exemplar sample to create a large texture image, while preserving the structural pattern of the source image. Implicitly most existing texture synthesis algorithms assume that the source images are recti-

Figure 13: *(Left) Input photographs, (middle) rectified with pasted textures, (right) re-warped with distorted textures.*

synthesis without rectification   synthesis with rectification

Figure 14: *Texture synthesis with image quilting [EF01] with identical parameters using unrectified exemplars (see insets) as against the rectified exemplars (using our algorithm) on the left and the right columns, respectively.*
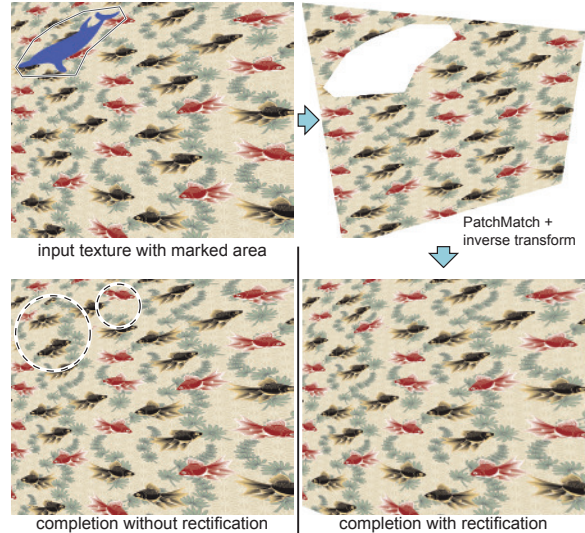
before rectification   after rectification

SIFT without rectification

SIFT with rectification
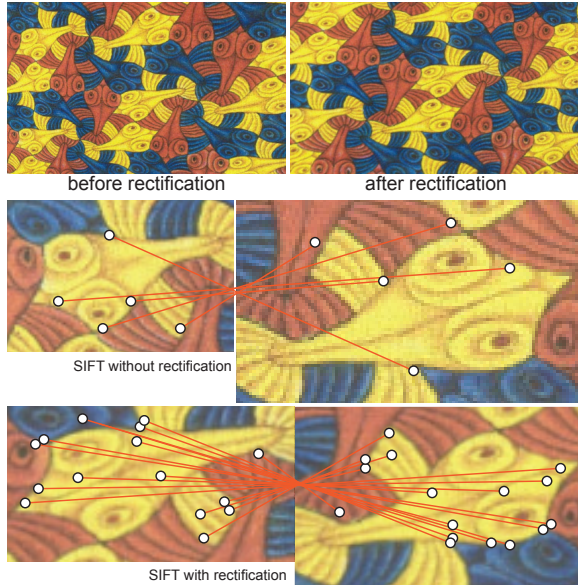
Figure 15: *Our rectification algorithm reduces perspective distortions, and can significantly improve performance of SIFT-based correspondence detection, a critical step for image processing algorithms, e.g., symmetry detection.*

fied, so that the structural and statistical properties of the images can be analyzed and used for synthesis. In Figure 14 we show image quilting [EF01] results without and with exemplar rectification, using same parameters for all the results. Not surprisingly, the synthesis results on the rectified inputs, produced by our algorithm, are superior in quality due to the presence of a larger variety of (undistorted) patches to choose from and piece together. The performance benefits are expected to be comparable with most other texture synthesis algorithms.

Texture Analysis. Our method rectifies inputs with repeated image content. In cases when the underlying textures have a regular pattern, the rectification helps expose the repetition pattern. State-of-the-art symmetry detection methods [LE06] rely on the availability of invariant features to seed the search for symmetry transforms. Commonly used critical point based features, like SIFT, are invariant to rigid and similarity transformations but can be unstable under perspective distortions (see Figure 15). In our algorithm, we handle such feature ambiguity by using consensus among the transform space surface intersection points, to get a global rectification.

Note that automatic rectification of images with repeated content naturally can help several other image processing tasks including image completion, image denoising, and superresolution that can make use of repeated data across an image. After analyzing and manipulating the undistorted input, one can re-warp the results to restore back the input perspective using the unwarp-mapping. In the future, we hope

input texture with marked area   PatchMatch + inverse transform

completion without rectification   completion with rectification

Figure 16: *Image completion without and with rectification. In both cases, the marked lasso polygon region (top-left) was filled using content aware completion via Patch-Match. Note that without rectification, the completed region contains artifacts, with unnatural double fishes (highlighted), due to lack of suitable proxies.*

to explore this avenue (see Figure 16 produced using Patch-Match [BSFG09] in Adobe$^{©}$ CS5).

Limitations. Our methods have certain limitations: (i) For example in Figure 17-top, all the pinenuts differ from each other in the original scene, have non-negligible depth variations, and further the fitted ellipses are roughly circular, making the extracted major-axis line segments ambiguous. As a result, the obtained rectification is imprecise as can be judged by the reference circular ring. (ii) When the input has only translationally repeated elements, then the rectified
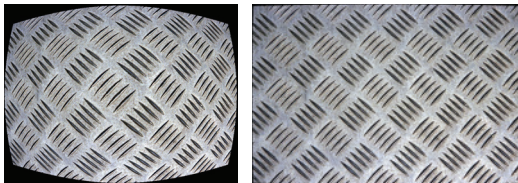

before rectification   after rectification
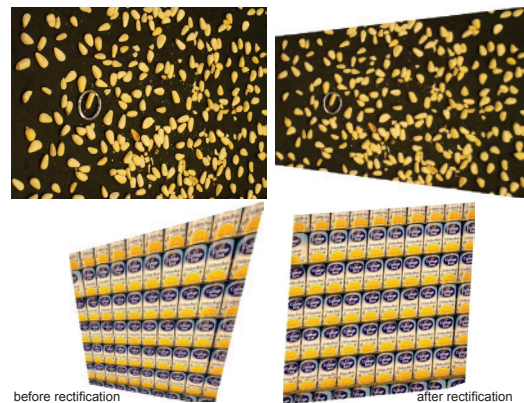
Figure 17: *(Top) Repeating elements with size comparable with depth variations are challenging. (Bottom) Elements arranged in translational grids result in affine ambiguity in the rectification space in absence of additional information.*

output has affine ambiguity (see Figure 17-bottom). Without additional information this ambiguity is impossible to avoid. (iii) As discussed, we make an approximation by working with major axes of fitted ellipses and alternately optimizing for pure affine/projective transformation. Hence, we cannot provide any guarantee of reaching the global optimal. In practice, however, we did not experience this problem.

4. Conclusion

We presented an algorithm for automatic rectification of textures with repeated coplanar elements. Our algorithm maps line segments arising out of regional descriptors to surfaces in a transformation domain, and solves for the repetition maximizing rectifying transformation using a consensus voting in transformation parameter space. We perform rectification by alternately optimizing over the pure perspective and the pure affine components. The proposed algorithm is particularly suitable for the analysis of textures, as demonstrated by the various examples.

Figure 18: *Our algorithm can be used for rectification of parameterized wide-angle distortion.*

In the future, we want to handle other types of distortions with low degrees of freedom — Figure 18 shows an initial rectification result for parameterized lens distortion. In general, for transformations with higher degrees of freedom, the resultant surfaces in the transform domain quickly become complex, making it challenging to efficiently compute their intersections. Additionally, we plan to further explore other texture and image analysis applications that can benefit from a similar dual domain reasoning.

References

[AMN*94] ARYA S., MOUNT D. M., NETANYAHU N. S., SILVERMAN R., WU A. Y.: An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. *Journal of the ACM 45*, 3 (1994), 891–923. 5

[BSFG09] BARNES C., SHECHTMAN E., FINKELSTEIN A., GOLDMAN D. B.: PatchMatch: A randomized correspondence algorithm for structural image editing. *TOG SIGGRAPH 28*, 3 (2009). 9

[CM01] CLARK P., MIRMEHDI M.: Estimating the orientation and recovery of text planes in a single image. In *BMVC* (Sep 2001), pp. 421–430. 2

[CM10] CHUM O., MATAS J.: Planar affine rectification from change of scale. In *Proc. ACCV* (2010), pp. 347–360. 2

[CZ00] CRIMINISI A., ZISSERMAN A.: Shape from texture: Homogeneity revisited. In *BMVC* (Sep 2000), pp. 82–91. 2

[EF01] EFROS A., FREEMAN W. T.: Image quilting for texture synthesis and transfer. In *SIGGRAPH* (2001), pp. 341–346. 8, 9

[Gib69] GIBSON E.: *Principles of perceptual learning and development*. New York: Appleton-Century-Crofts, 1969. 2

[HO07] HUBERT E., OLVER P.: Differential invariants of conformal and projective surfaces. *SIGMA 3* (2007). 6

[HZ00] HARTLEY R., ZISSERMAN A.: *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000. 2

[LE06] LOY G., EKLUNDH J.: Detecting symmetry and symmetric constellations of features. In *Proc. ECCV* (2006), pp. 508–521. 9

[LF06] LOBAY A., FORSYTH D. A.: Shape from texture without boundaries. *IJCV 67* (2006), 71–91. 2, 7

[LH05] LOH A. M., HARTLEY R.: Shape from non-homogeneous, nonstationary, anisotropic, perspective texture. In *Proc. BMVC* (2005), pp. 69–78. 2

[Lou04] LOURAKIS M. I. A.: levmar: Levenberg-marquardt nonlinear least squares algorithms in C/C++. http://www.ics.forth.gr/ lourakis/levmar/+, 2004. 3, 7

[Lou09] LOURAKIS M. I. A.: Plane metric rectification from a single view of multiple coplanar circles. In *IEEE ICIP* (2009), pp. 509–512. 2

[LZ98] LIEBOWITZ D., ZISSERMAN A.: Metric rectification for perspective images of planes. In *Proc. CVPR* (Santa Barbara, CA, Jun 1998), pp. 482–488. 1, 2, 3, 6

[MCUP02] MATAS J., CHUM O., URBA M., PAJDLA T.: Robust wide baseline stereo from maximally stable extremal regions. In *Proc. BMVC* (2002), pp. 384–396. 6

[MR95] MOTWANI R., RAGHAVAN P.: *Randomized Algorithms*. Cambridge University Press, 1995. 5

[MR97] MALIK J., ROSENHOLTZ R.: Computing local surface orientation and shape from texture for curved surfaces. *IJCV* (1997), 149–168. 2

[MTS*05] MIKOLAJCZYK K., TUYTELAARS T., SCHMID C., ZISSERMAN A., MATAS J., SCHAFFALITZKY F., KADIR T., GOOL L. V.: A comparison of affine region detectors. *IJCV 65*, 1/2 (2005), 43–72. 6

[NC06] NAJMAN L., COUPRIE M.: Building the component tree in quasi-linear time. *Image Processing 15*, 11 (2006), 3531 – 3539. 6

[NN04] NOCK R., NIELSEN F.: Statistical region merging. *IEEE PAMI 26*, 11 (2004), 1452 –1458. 1, 3, 6, 8

[PBCL09] PARK M., BROCKLEHURST K., COLLINS R., LIU Y.: Deformed lattice detection in real-world images using mean-shift belief propagation. *PAMI 31*, 1 (2009). 2

[WFP10] WU C., FRAHM J.-M., POLLEFEYS M.: Detecting large repetitive structures with salient boundaries. In *Proc. ECCV* (2010). 2

[ZGLM10] ZHANG Z., GANESH A., LIANG X., MA Y.: Tilt: Transform invariant low-rank textures. In *Proc. ACCV* (2010). 1, 2, 3