Segmenting Periodic Reliefs on Triangle Meshes

Shenglan Liu [1,2] Ralph R. Martin [1] Frank C. Langbein [1] Paul L. Rosin [1]

[1] School of Computer Science, Cardiff University, UK
[2] CMEE, Nanjing University of Aeronautics and Astronautics, China
{Shenglan.Liu, Ralph, F.C.Langbein, Paul.Rosin}@cs.cf.ac.uk

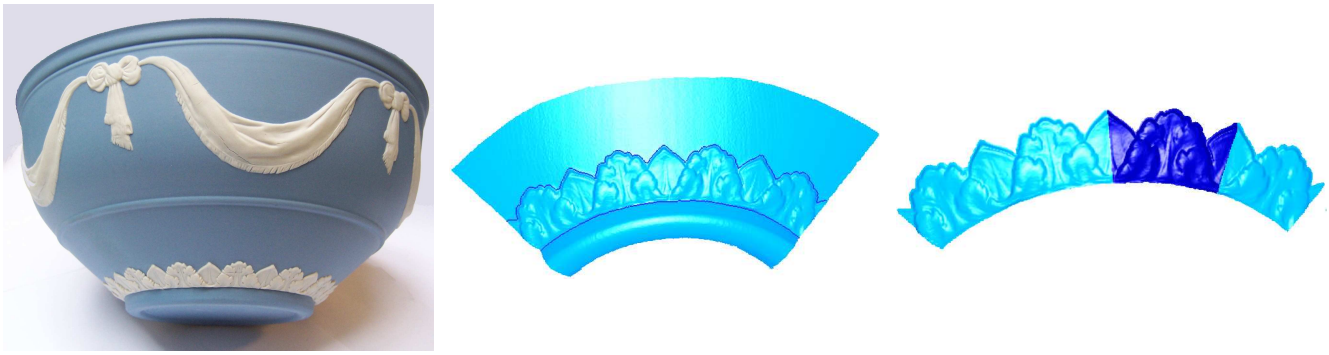

Figure 1: Periodic relief segmentation from a porcelain bowl: the bowl; segmented partial relief on mesh; an extracted repeat unit.

Abstract

Decorative *reliefs* are widely used for e.g. packaging and porcelain design. In *periodic* reliefs, the relief repeats a pattern, for example all the way around an underlying surface of revolution. Reverse-engineering of existing reliefs allows them to be re-applied to different base surfaces; we show here how to segment a single repeat unit of a periodic relief starting from a scanned triangle mesh.

We first briefly review how we segment the relief from the background surface using our previous work. The rest of the paper then concentrates on how we extract a single repeat unit from the relief. To do so, the user provides two points on one relief boundary which are in approximate correspondence on consecutive repeats of the relief. We first refine the relative locations of these points, and then determine a third corresponding point using relief boundary information. These are used to determine three initial cutting planes across the relief. Then surface registration strategies are utilised to refine the correspondence between adjacent repeat units. Finally, we refine the exact locations of the cutting planes by considering only surface information close to the cutting planes. This allows a repeat unit of the periodic relief to be extracted.

We demonstrate that our algorithm is successful and practical, using various real scanned models: user input can be quite imprecise, and we can cope with hand-made reliefs in which the pattern units are only approximately copies of each other.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid, and object representations

Keywords: Periodic relief, segmentation, mesh processing, surface registration.

1 Introduction

Sculptured *reliefs* are widely used in various industries such as sign-making, packaging and ceramics. In many cases, for example when extending or replicating an existing range of products, it is required to reuse the relief designs from existing objects and re-apply them to new CAD models. In many cases, CAD models of the original relief do not exist—for example, the original design for a range of porcelain may be many decades old. Thus, at present it is often necessary for a sculptor to hand-copy existing relief designs, a process that is time consuming, tedious and expensive. An attractive alternative is to reverse engineer the reliefs, allowing automatic extraction and reapplication of previously designed and manufactured reliefs to new CAD models. This approach which would provide benefits to industry in terms of cost savings, and also in time to market.

Generally, we may define a relief to be a part of a surface with sculpted features different from the underlying surface, and which is raised by a small height; this height is typically larger than the characteristic size of features on the background. There are various kinds of reliefs, and they can be imposed on diverse backgrounds. The simplest case is that of an isolated relief delimited by a single outer contour, lying on a smooth and slowly varying background. More complex, in terms of processing, are reliefs lying on a textured background. We have already addressed the segmentation problem—separating the relief from the underlying background—for such cases in [Liu et al. 2006] and [Liu et al. 2007].

Here, we consider the segmentation of another kind of frequently occurring relief, *periodic reliefs*, illustrated in Fig. 1(left). Here, a basic unit of relief pattern is repeatedly applied to the object. This may be cyclic, forming a closed pattern on objects such as vases, bowls, and bottles for which the underlying surface is a surface of revolution. However, more generally a periodic relief is any relief

in which relief units are repeated in a sequence along some arbitrary path (indeed, they could even be repeated two dimensionally to give an overall surface pattern, although we do not consider this case here). In this paper, we give a method designed to extract exactly replicated relief units distributed along, for example, *a circle around a surface of revolution* or *a linear path on a flat surface* to form a closed or open *relief frieze*. Nevertheless, reliefs along *arbitrary paths* may also be extracted using our methods, providing that over a sequence of three successive relief units two criteria are satisfied. Firstly, the shape of the underlying surface should not vary enough to introduce significant changes in the shape of successive relief units—the radii of curvature of the underlying surface should be large compared to the length of a relief unit, and not change much. Secondly, the relief path itself should not be too curved—a large amount of geodesic curvature in the relief path would alter the orientation of the repeat units with respect to the principal directions of the underlying surface, potentially significantly altering the shape of successive relief units. In cases meeting these criteria, successive relief units have sufficiently similar shape that their relationship is reasonably well approximated by a translation and rotation. In fact, many reliefs along more general paths meet these criteria sufficiently well in practice for our methods to work. Furthermore, we note that many existing reliefs of interest are handmade, and so do not exactly repeat anyway, even if e.g. encircling a surface of revolution. Our methods perform well in the face of such approximate regularity from the outset. Other more complicated cases, in which the relief units are scaled or significantly distorted along the path, are not considered in this paper, however.

To be able to re-apply a periodic relief, our aim in segmentation is to obtain a single repeat unit from the scanned triangle mesh: when the relief is reapplied to a differently-sized or differently-shaped object, a different number of repeats may be required. Thus, there are two subproblems to be considered. Firstly we must segment the relief from the background, using its outer contours. Then we must extract a single repeat unit from the repetitive relief pattern.

The relief's outer boundaries can be detected using our previous segmentation methods: see [Liu et al. 2006] for the smooth background case, and [Liu et al. 2007] for textured backgrounds. These methods separate reliefs from their underlying surface using an active contour (or *snake*) driven by energy terms based on significant two relief characteristics, the raised step at the relief boundary, or the difference in surface properties between the relief and the background surface.

Thus, this paper concentrates on the second subproblem, the one of extracting a single relief repeat unit. As noted, if the underlying surface and relief path vary slowly, successive repeat units approximately match each other after applying some rigid transformation. Thus, we use a *surface registration* method to match adjacent repeating units. The repeating units can then be identified by cutting across the relief pattern at corresponding points between adjacent repeats; we do so simply by finding corresponding cutting planes across the relief.

In our final output, the relief generally does not have an exactly identical profile at these cutting planes, for a variety of reasons: because the relief may be hand-made and thus not exactly regular, because of measurement errors, because of errors in registration, and because of minor distortion in the relief units due to the underlying surface shape and relief path. Thus, to make successive repeats a good fit to each other, when placed along a path on a new object, the shape of a repeat unit may need adjustment, particularly in the areas adjacent to the cutting planes. One approach would be to blending the shapes of the left and right-hand ends of the repeat units. Detailed consideration of such adjustment is left as a separate problem for future work; we simply note here that in many ways it is analogous to the well-known problem of 'looping' audio samples.

In Section 2 we review periodicity extraction in related fields. In Sections 3 and 4, we present our algorithm, and discuss details such as the relief boundary period extraction, repeating unit refinement, and cut localization. Results are demonstrated in Section 5. Discussions and conclusions are given in Section 6 and 7.

2 Related Work

The problem of *periodicity* detection and extraction has been studied in the related fields of image processing and computer vision for several decades. *Co-occurrence matrices* have large diagonal entries when sampling corresponds to the period of a regular repetitive texture, as noted by early researchers [Zucker and Terzopoulos 1980], and are often used to extract the repetitive cells from images with a regular repetitive texture. For example, Handley [1998] used one particular co-occurrence matrix feature, the *dissimilarity*, to characterise the texture periodicity, and then to extract the repeating fundamental unit by finding local maxima.

Many other researchers have adopted another popular periodicity description, the *autocorrelation function*. One such example is provided by [Hsu et al. 2001], where the autocorrelation function is used on an overcomplete wavelet decomposition of an image, together with an efficient peak finding method, to determine the repetitive structure unit in an image texture. Another periodic pattern detection method can be found in [Liu et al. 2004], where a computational model based on the theory of crystallographic groups is presented. In computer vision, periodic motion detection on image sequences has been studied, such as in [Cutler and Davis 2000], where the authors propose a use of time-frequency analysis to a self-similarity measure.

However, the above methods often rely on the regular grid structure of a 2D image, and are thus difficult to directly extend to an irregular 3D triangle mesh. Nevertheless, we find use for the autocorrelation function, by applying it in our algorithm to seeking repetition along the relief boundary curves, after we have resampled points along them at regular intervals. However, it would be much trickier and, and time consuming, to resample the whole mesh into a regular grid.

In computational geometry, relatively little research has focused on periodicity detection in 3D shapes. However, symmetry detection shares certain similarities to our problem, in that both deal with repeating features, and both involve local shape matching under one or a set of transformations. Many different methods have been proposed to find whole-object symmetries with respect to planes through the center of mass. For example, Sun et al. [1997] converted the symmetry detection problem to finding correlations in Gaussian images, Martinet et al. [2005] introduced a method based on generalised moments, while recently Podolak et al. [2006] utilised a planar reflective symmetry transform to capture the reflective symmetries. A method for partial symmetry detection and extraction proposed by Mitra et al. [2006] matches simple local shape signatures pairwise to accumulate evidence for symmetries in an appropriate transformation space, then extracts potentially significant symmetries by a clustering method. We note that choosing repeating units in symmetry determination in either the global or partial case does not necessarily require full repeating units to be found. Furthermore, some algorithms work well for exact symmetries, but may not be readily adaptable to approximate symmetries. In our case, we require the repeating units to be determined as completely and accurately as possible within the bounds of their

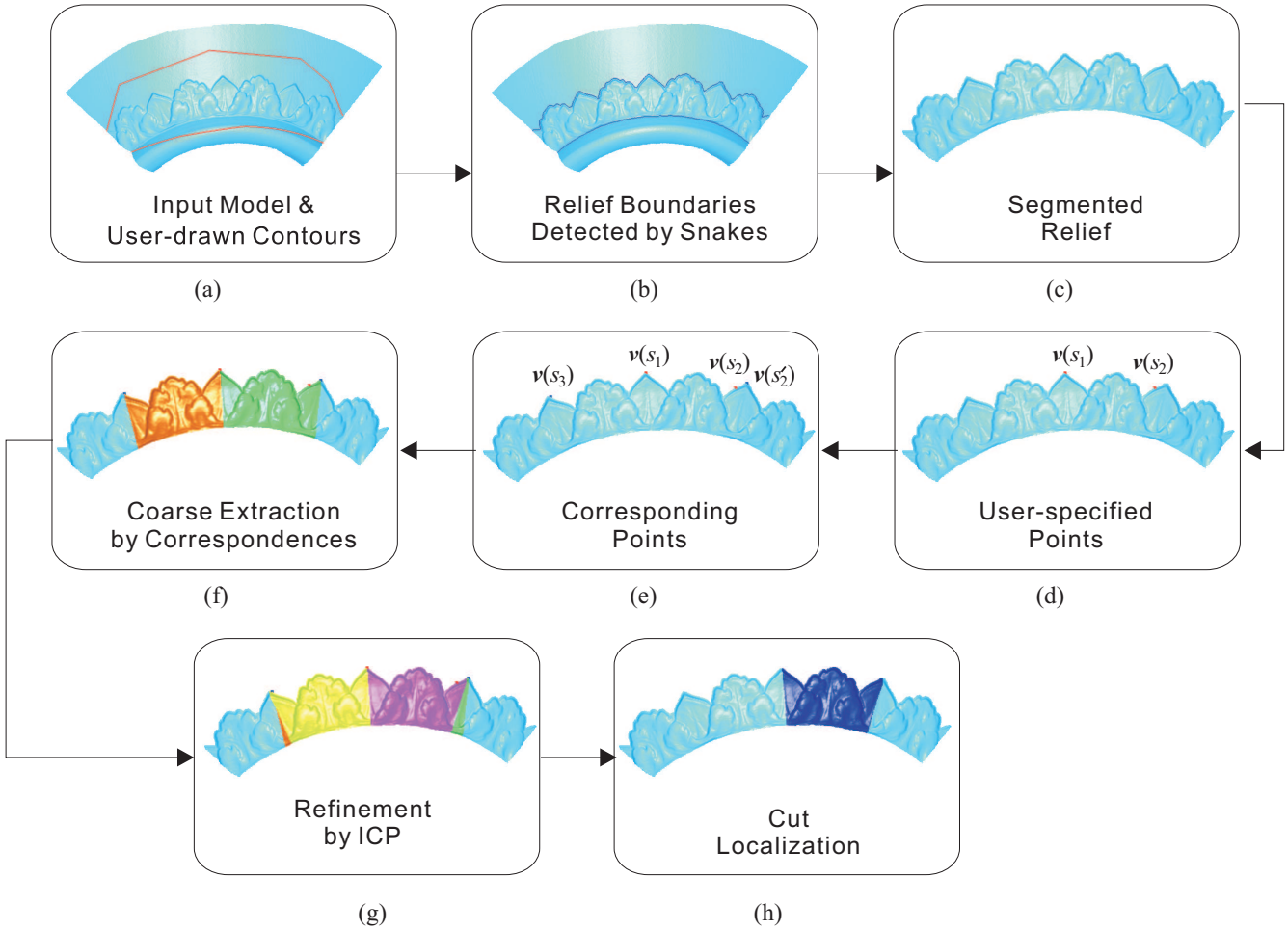

Figure 2: Periodic relief segmentation—sequence of operations

approximate nature, as their boundaries need to be well matched in future reapplication.

3 Algorithm Overview

As mentioned earlier, there are two main steps in our approach for periodic relief segmentation: segmenting the relief from the background surface it lies upon, and then extracting a single repeat unit from the relief. The sequence of operations used in our algorithm is illustrated in Fig. 2, where the first three phases belong to the first step, and the remaining phases, the second step.

When segmenting the relief from the background, we can deal with both smooth and textured backgrounds, although in both cases we assume the shape of the background surface varies slowly. In the smooth background case, we use snakes which start from a pair of user-drawn contours, which lie outside the upper and lower boundaries of the relief; their exact locations are unimportant. These evolve until they match the boundary of the relief, determined by using an energy defined to locate each snake at a step feature. If the background is textured, two approaches can be used to segment the relief. The first is to evaluate surface properties at mesh vertices, and use them to define an energy which distinguishes relief from textured background. This is used to drive each snake so that it coarsely matches the relief boundary; it is then finally adjusted to match the desired relief boundary by again seeking a step feature. The second approach is to initially smooth the surface to eliminate the background texture, and allowing the approach used for a smoothed mesh to be applied to find coarse boundary positions; these are again further optimised on the original textured mesh by seeking a step.

Our method for repeat unit extraction starts from two points specified by the user on one relief boundary, which are in approximate correspondence on consecutive repeats of the relief. Three phases are then involved: coarse relief unit extraction, refinement and cut localization. In the first phase, we first improve the relative locations of the two chosen points on the appropriate boundary curve. We then obtain a new corresponding point on the same boundary curve for a further adjacent relief unit, and from these determine three initial cutting planes across the relief. Next, in the refinement phase, a surface registration strategy based on the *iterative closest point* (ICP) algorithm [Besl and McKay 1992; Eggert et al. 1996] is utilised to find a more accurate match between the coarsely extracted repeating units, allowing us to obtain an optimised unit by refining the locations of the cutting planes. Finally, in the cut localization phase, we update the match between the left- and right-hand edges of the chosen repeat unit using only surface data close to these edges. This produces a better match near the *start* and *finish* of each repeat unit. This is important as in use, copies of the repeat unit are placed next to itself. Details are discussed in the next Section.

4 Algorithm Details

4.1 Whole Relief Segmentation

In this subsection, we remind the reader of certain details of snakes and geometric texture classification from our previous work on relief segmentation. Further information can be found in [Liu et al. 2006] and [Liu et al. 2007].

A *snake* is an energy-minimizing spline controlled both by internal forces, such as rigidity and elasticity of the curve which both make it smooth, and tend to shrink, and external forces such as constraint and feature forces which help to drive it towards some desired feature. To deal with the particular problem of relief segmentation, we tailor the snake by careful definition of energy terms, and also carefully control the evolution process. For extraction of reliefs on a smooth background, a feature energy term is specifically designed to detect the step at the edge of a relief. Secondly, a deflation force is used to make the snake move inwards, with strength determined dynamically to balance the internal energy in such a way that the snake is insensitive to choice of initial contour. Thirdly, a refinement phase is used, with different energy terms, to make the snake explore any relief contour concavities.

For reliefs on a textured background, the geometric texture classification method in our algorithm uses a *support vector machine* (SVM) classifier to identify parts of the mesh as belonging to relief or background. The input vectors used for classification are various geometric properties of three kinds: local surface properties, integral properties and statistical properties. The local surface properties are differential geometric quantities such as normals and curvatures, and are estimated using the immediate 1-ring neighbors. The integral properties are computed at multiple scales taken over local neighborhoods [Pottmann et al. 2005]. The statistical properties include both first order statistics (such as mean, variance, energy and entropy) [Materka and Strzelecki 1998] and co-occurrence matrix features (such as energy, entropy, contrast and homogeneity) [Haralick et al. 1973], and are extensions of ideas from image texture analysis.

In our earlier work, such techniques were used to extract a relief with a single contour (which could be open or closed). Here, we extract two contours for each relief, which may either be closed (e.g. in the case of a relief encircling a vase), or open (e.g. if we have scanned just part of such a relief).

4.2 Coarse Relief Unit Extraction

It is simpler and faster to use boundary *curves* to extract the initial coarse repeating units than to process pieces of mesh *surface*. In the coarse relief unit extraction phase, the main task is to obtain three points in correspondence on one of the two relief boundaries, and from these, determine three initial cutting planes across the relief delimiting two adjacent, approximately corresponding, pieces of relief. We assume that the user starts by marking two points on the boundary curve which believed to be in approximate correspondence on subsequent relief units. Note that if the user selects arbitrary mesh points, these points generally will not exactly lie on the boundary curves determined previously, so they may have to be moved to the nearest point on the appropriate boundary first. (Alternatively, the user's selection can be constrained to lie on a boundary curve). The user should take into account that the final cutting plane at the left-hand end of the relief unit determined will go through the first of the user's chosen points.

Although periodic reliefs have repetitive boundary curves, in some cases the boundary may be featureless even though the relief itself is not: for example the boundary curve may be a circle, which only trivially shares the period of the relief. We assume here that at least one of the boundary curves has distinct features with the same period as the relief itself, and the user is instructed to select points from that particular boundary. We discuss how to proceed if this assumption is not satisfied later.

To accurately detect the period between corresponding boundary points on a boundary curve, we first resample the particular boundary curve to produce new points at equal arc-length intervals along it. The distance between points is set to the average distance between the original points on the boundary curve. Next, curvature of the boundary curve is estimated at each point, and an autocorrelation of the curvature on the resampled curve is computed. Note that the curvature is independent of a rigid body transformation, making it suitable for determining similarity of parts of the boundary an as-yet unknown transformation. Formally, the autocorrelation of the curvature is defined as follows:

$$A(x) = \frac{\sum_{s=0}^{N} k(s)k(s+x)}{\sum_{s=0}^{N} k^2(s)}, \tag{1}$$

where s is the index of a curve point, x is an integer denoting a candidate periodicity interval, and k denotes the curvature at that point.

The autocorrelation function $A(x)$ has a maximum at $x = 0$ and a number of peaks at other locations corresponding to the curve periodicity. Given the user-chosen point $\mathbf{v}(s_1)$ and its approximate correspondence $\mathbf{v}(s_2)$ on the consecutive repeat unit, suppose that we believe the approximate user chosen correspondence lies within a distance δ of the true correspondence; we set δ to $(|s_2 - s_1|/2)$ in our algorithm. Then $A(x)$ has a maximum in the range $[s_2 - \delta, s_2 + \delta]$ in Equ. 1. If we assume that the maximum is at $x = s_p$, then the refined position of the approximate correspondence is $s_2' = s_1 + s_p$. Later, we need three points in correspondence, so we also select another corresponding boundary point $\mathbf{v}(s_3)$ at the peak near $s_1 - s_p$. The large range we use for δ means that once $\mathbf{v}(s_1)$ has been user-selected, the point $\mathbf{v}(s_2)$ need not be specified very carefully at all. For example, in Fig. 2(d), the left red dot is $\mathbf{v}(s_1)$, and the right dot $\mathbf{v}(s_2)$ is quite far from the real correspondence. After improvement using the autocorrelation function, the correspondence is much more accurate as shown by the blue dots in Fig. 2(e).

If the relief boundaries are both featureless, the autocorrelation approach does not work, but we may still proceed provided that the user makes careful estimates of the locations of both initial points. In this case, to find the third point, we simply choose a point at an equal arc-length before $\mathbf{v}(s_1)$ as $\mathbf{v}(s_2)$ is after it. (We discuss the question of why we perform this improvement at all, later).

To estimate the curvature, we follow [Lewiner et al. 2005] and use least-squares fitting. In principle, we could use other rigid-body-invariant curve descriptors in place of the curvature, such as the semi-differential invariants proposed by Pajdla and Gool [1995b; 1995a]. For example, we have tried using the second semi-differential invariant calculated using sliding pairs based on fixed arc-length [1995a], but found that similar results were produced to those using curvature autocorrelation.

Having found three corresponding points $\mathbf{v}(s_3)$, $\mathbf{v}(s_1)$ and $\mathbf{v}(s_2')$ on one boundary, we now determine three cutting planes at these points. In each case we use the plane which passes through the particular point, and with plane normals, respectively, of $\mathbf{v}(s_3) - \mathbf{v}(s_1)$, $\mathbf{v}(s_3) - \mathbf{v}(s_2')$ and $\mathbf{v}(s_2') - \mathbf{v}(s_1)$. This simple approach gives cutting

planes approximately perpendicular to the relief strip, which is generally what the user expects.

As shown in Fig. 2(f), these approximately delimit two repeating units of the relief. Although these planes are not in especially good correspondence, i.e. are not exactly related by the same rotation and translation as the relief itself, they are adequate as a basis for delimiting approximate relief units for the next step.

4.3 Refinement

In this subsection, we refine the coarse relief unit match of the initial repeat units using an ICP algorithm.

The repeating units of a periodic relief are generally only *approximately* congruent, both potentially because of their hand-made nature, and also possibly because of variations in the underlying surface shape, and their orientation upon the underlying surface. We assume that the match is good enough that we can approximate it by a rigid body transformation described as follows:

$$Q = \mathbf{R}P + \mathbf{T}, \tag{2}$$

where P and Q are two point sets belonging to adjacent repeating units (obtained by cutting the relief with the cutting planes), $\mathbf{R}$ is a rotation matrix and $\mathbf{T}$ is a translation vector.

ICP is a well-known algorithm introduced by Besl and McKay [1992] for registration and shape matching. It is an iterative alignment algorithm based on minimizing the mean-square distance between two point sets. It works in three phases: 1) establishing correspondence by searching for pairs of nearest points, one from each set, 2) estimating the rigid transformation that overall best maps the first member of each pair onto the second and then 3) applies that transformation to update the position and orientation of the first point set. These three steps are then reapplied until convergence is obtained. The algorithm works quite effectively when given a good initial estimate. Here, we adopt an improved version developed by Eggert et al. [1996] which is more robust to errors in initial alignment, and produces a better transformation.

The coarse relief unit extraction phase provides us with three corresponding points on one boundary curve. Thus, as the set of equally-spaced sample points along the curve segment $\mathbf{v}(s_3)\mathbf{v}(s_1)$ match corresponding points on the segment $\mathbf{v}(s_1)\mathbf{v}(s_2')$, these can be used to provide an initial estimate of the rigid body transformation for ICP. Following Besl and McKay [1992], we compute this initial transformation using the singular value decomposition (SVD) method.

A refined estimation of the transformation ($\mathbf{R}$ and $\mathbf{T}$) is then produced by the ICP algorithm.

We next apply this transformation and its inverse ($\mathbf{R}^{-1}$ and $-\mathbf{R}^{-1}\mathbf{T}$) to the point $\mathbf{v}(s_1)$, to obtain two new points in *accurate* correspondence with $\mathbf{v}(s_1)$; these replace $\mathbf{v}(s_3)$ and $\mathbf{v}(s_2')$. A refined cutting plane at $\mathbf{v}(s_1)$ is now constructed using the same method as described in Section 4.2, except using these new corresponding points. Finally, two refined cutting planes at the new corresponding points are then computed by applying the found transformation and its inverse to the newly computed plane at $\mathbf{v}(s_1)$. The two repeat units are now more accurately defined by these new cutting planes as shown in Fig. 2(g).

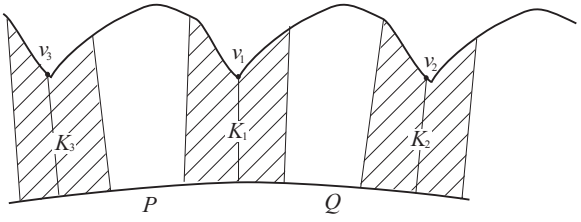

Figure 3: Key patches

4.4 Cut Localization

In the cut localization phase, we further improve the matching accuracy in regions adjacent to the cutting planes—this is where the relief units join on to each other when being reapplied, and it is more important to get a good match here rather than over the whole of each relief unit: again, note that the repeating units of a periodic relief are generally *only* approximately congruent, for reasons mentioned previously. We perform localization by again using the ICP algorithm, this time on selected areas adjacent to each cut.

The idea is illustrated in Fig. 3. Suppose the two repeating units P and Q are separated by cutting planes as shown at $\mathbf{v}_1$, $\mathbf{v}_2$ and $\mathbf{v}_3$. We now optimise the location of the cutting plane at the right-hand end of Q so that Q may be used as a repeat unit. We assume that the cutting plane at the left hand-end of Q remains fixed, to respect the user's selection of the location of v_1 at one end of the repeat unit; furthermore, the normal to this plane is the most reliable of all 3 cutting planes, having been determined by the vector between $\mathbf{v}_2$ and $\mathbf{v}_3$, which is a good approximation to the local direction of the relief strip at $\mathbf{v}_1$. To optimise the location of the right-hand cut on Q, for its use as a repeat unit, it is clear that carefully matching the regions K_1 and K_2 next to the cutting planes will produce a better result than minimizing the matching error between the whole of P and the whole of Q. We call patches K_1 and K_2 *key patches*.

Thus, using as an initialization the transformation found from the refinement phase, we now further optimise the cutting planes by using ICP on the key patches. The width of the key patches can be a fixed width chosen by the user, such as 10% of the repeat unit width. Alternatively, we may iteratively update the cutting planes, reducing the key patch widths at each step. While the latter takes a longer time than using a fixed key patch width, it generally produces better results.

To optimise the repeat unit Q in Fig. 3, we do the following:

1. Start with the whole of the key patch K_1 having half the width of Q (as shown): the cutting plane used to give the left edge of K_1 is found by interpolating by the cutting planes at $\mathbf{v}_3$ and $\mathbf{v}_1$. Similarly, obtain the the cutting plane for the right-hand edge of K_1. Find the left and right cutting planes for K_2 by applying the transformation computed in the refinement phase to the cutting planes for K_1.

2. Register the patches K_1 and K_2 by ICP, giving an updated estimate for the transformation relating the right-hand edge of Q to the left-hand edge.

3. Localize the cutting plane at $\mathbf{v}_2$ by applying the new transformation to the plane at $\mathbf{v}_1$.

4. Decrease the key patch width to half its previous value, centered on the current edges.

5. Repeat steps 1) to 4) until the key patch width is less than a given minimum width, for example, 10% of the repeat unit

width.

While the width subdivision method takes a longer time than choosing a fixed key patch width, it also provides us with some important information. If plane movement reduces as we subdivide the key patches, it indicates that a good match has been found between them, giving confidence that the extracted repeat unit is satisfactory. If plane movement increases, this indicates to the user that another part of the relief might be more suitable for providing a repeat unit. Indeed, the method can readily be automated to proceed to subsequent repeat units as candidates, until a suitable one is found, at the expense of taking longer.

5 Results

Our algorithm has been tested with various real scanned periodic geometric models using a PC with a 2.4GHz CPU and 1GB RAM.

Fig. 4 shows the process of periodic relief segmentation for a model with a quite regular repetitive relief. Fig. 4(a) and (b) show the extraction of the whole relief from the background. Fig. 4(a) shows the scanned model and the initial contours created joining several user-selected points to determine each contour. Fig. 4(b) shows the final positions of the snakes after evolution to the relief boundaries. Figs. 4(c)–(h) illustrate the steps of repeat unit extraction. First, two points on one relief boundary are specified by the user, as shown the red dots in Fig. 4(c). Note that we have deliberately chosen two points far from correct correspondence to show that the algorithm can correct such problems by boundary searching using autocorrelation. This gives a good estimation of the boundary periodicity and refines the correspondence to give the two blue points in Fig. 4(d), which are one period to the right and left of the left-hand red point. The algorithm then coarsely determines two repeat units as shown in Fig. 4(e). Matching the two coarse units using the ICP algorithm refines them as shown in Fig. 4(f). Further ICP matching in the localization phase, using key patches with 20% of the repeat unit width, gives the result shown in Fig. 4(g). Fig. 4(h) shows the final extracted repeat unit, which is the right-hand colored repeat unit in Fig. 4(g). In this model, the repeat units are fairly accurate copies of each other, so the repeat unit after the phases of coarse extraction, refinement and localization are all very similar. In Fig. 4, the original mesh had 102040 triangles and 51451 points, the segmented relief had 53936 triangles and 28036 points, and the repeat unit had approximately 5000 points. The time taken for the whole relief segmentation stage was about 2 minutes. In the relief unit extraction step, the most time consuming parts were uses of the ICP algorithm which took 17 seconds for matching the coarse units in the refinement phase, and another 10 seconds for key patch matching. The other computations took negligible time.

Fig. 5 shows a hand-made periodic leaf relief whose repeat units can, when examined in detail, be seen to be rather irregular in shape. Fig. 5(a) shows the model and the initial contour, and Fig. 5(b) shows the final snake on the relief boundary (there is only one boundary as this relief was at the edge of the mesh). The units separated during the coarse extraction phase, the refinement phase and the localization phase are shown in Figs. 5(c)–(e). It can be seen that the result after localization in Fig. 5(e) is improved compared to the result after refinement in Fig. 5(d). The final extracted unit, the left-hand unit in Fig. 5(e), is shown in Fig. 5(f). Here, we have selected the tip of the leaf to delimit the repeat units because it is readily recognizable to the boundary autocorrelation process.

Fig. 6 shows an example of periodic relief segmentation in which we do not use boundary autocorrelation. Fig. 6(a) shows the model and the initial contour. Fig. 6(b) shows the two points specified by user. However, the extracted relief boundaries are both rather featureless, and at the same time, very noisy. The autocorrelation function did not work in this case, so this step was omitted. The third point was simply chosen an equal arc-length to the left of the left-hand user chosen point, as the right-hand user chosen point was from it. This gave the coarse repeat units shown in Fig. 6(c). The refinement, localization and final results are shown in Figs. 6(d)–(f). Note that relatively large errors in the coarse patches, due to boundary autocorrelation not being used, are corrected at the refinement stage. In this example, we were unable to obtain a satisfactory result at the localization stage if key patch matching was performed just a single time—even when we tried several different key patch widths. Nevertheless, using the strategy of key patch width subdivision produced a good final result.

6 Discussions

In the coarse extraction phase, we use a boundary repetition detection method to refine the initial user-chosen corresponding points. However, in certain cases the boundary may not be periodic, or if so, relatively featureless, as shown in Fig. 6. Nevertheless, even without accurately corresponding points, good results can still be produced. Why then should we keep the boundary autocorrelation function for improving the initial correspondence? There are mainly two reasons. The first is to provide better coarse units for the downstream ICP algorithm—good initial estimations can lead to quicker ICP convergence and can save a great deal of ICP computing time. The second is that the ICP algorithm is less likely to converge to the wrong local minimum.

In our tests, the ICP algorithm seems to work robustly when the coarse units have at least 70–80% overlap. Thus, if we do not use boundary autocorrelation, the user must specify the correspondence points within a distance of 10–15% of a repeat unit from their accurate locations, which is not particularly difficult for the user.

In some cases, if the coarse stage produces good results, it may be possible to skip the refinement phase. For example, for the model in Fig. 4, if the key patches phase is used immediately after the coarse extraction phase, we obtain similar results.

7 Conclusions and Future Work

Our tests have shown that the proposed algorithm does a good job of extracting relief units. It requires very little user interaction, and is robust to poorly chosen user input. It works on hand-made reliefs in which the repeat units are not exact copies. We conclude by summarizing several important features:

1. A snake-based method is used to evolve a coarsely specified contour to each relief boundary, which gives good whole relief segmentation results.

2. A relief boundary repetition detection method, based on autocorrelation of curve curvature is used to extract coarse repeat units.

3. A surface registration strategy is utilised to refine the match between repeat units.

4. Further localization is done on key patches of the units, near their boundaries, to ensure a good match in the region where adjacent repeat units meet.

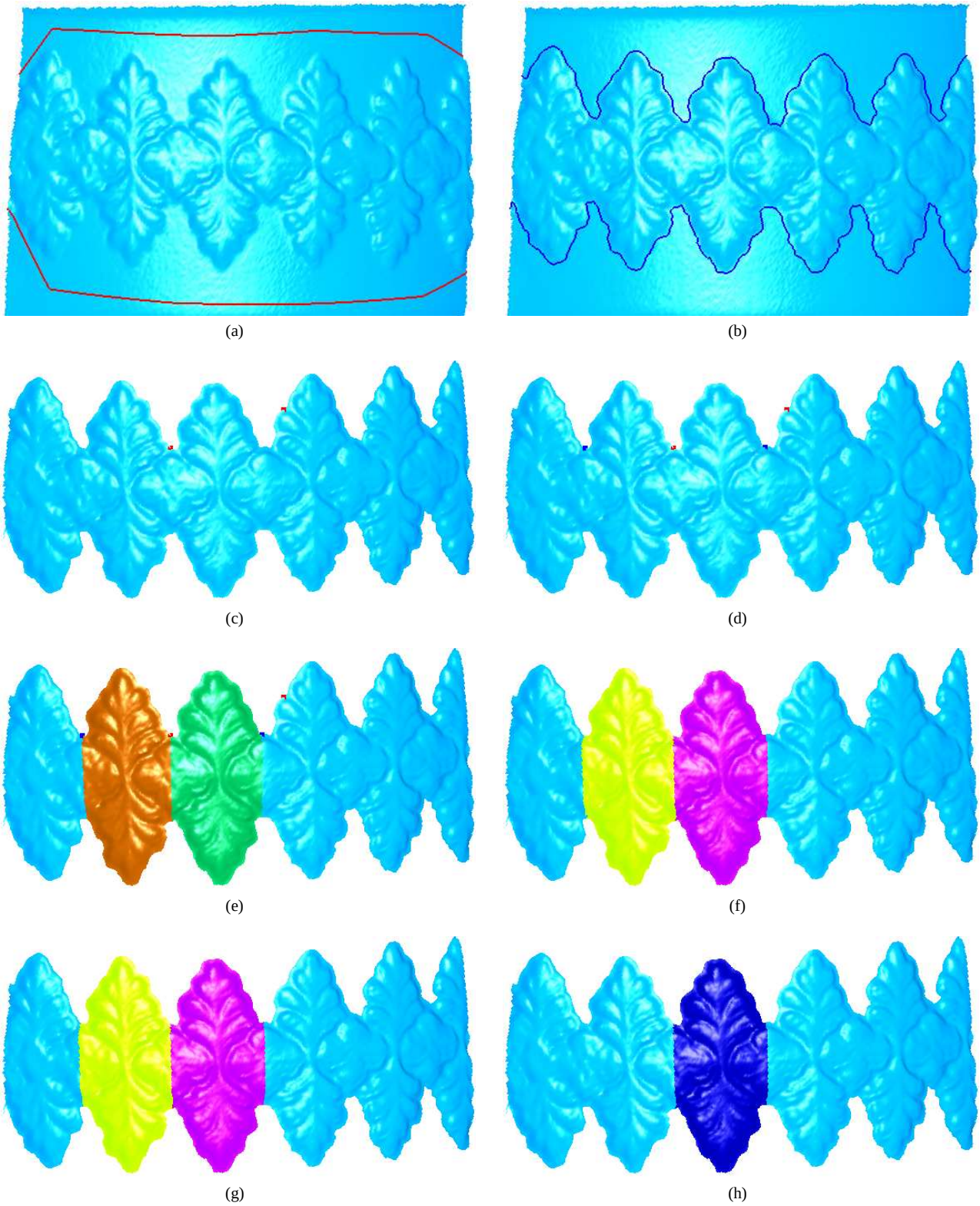

Figure 4: Periodic relief segmentation for a regular repetitive relief: (a) mesh and initial contour, (b) extracted relief boundaries, (c) two user-specified points in approximate correspondence on consecutive repeats, (d) accurate correspondences, (e) coarse repeat units, (f) refined repeat units, (g) verified refined units, (h) the final extracted unit.

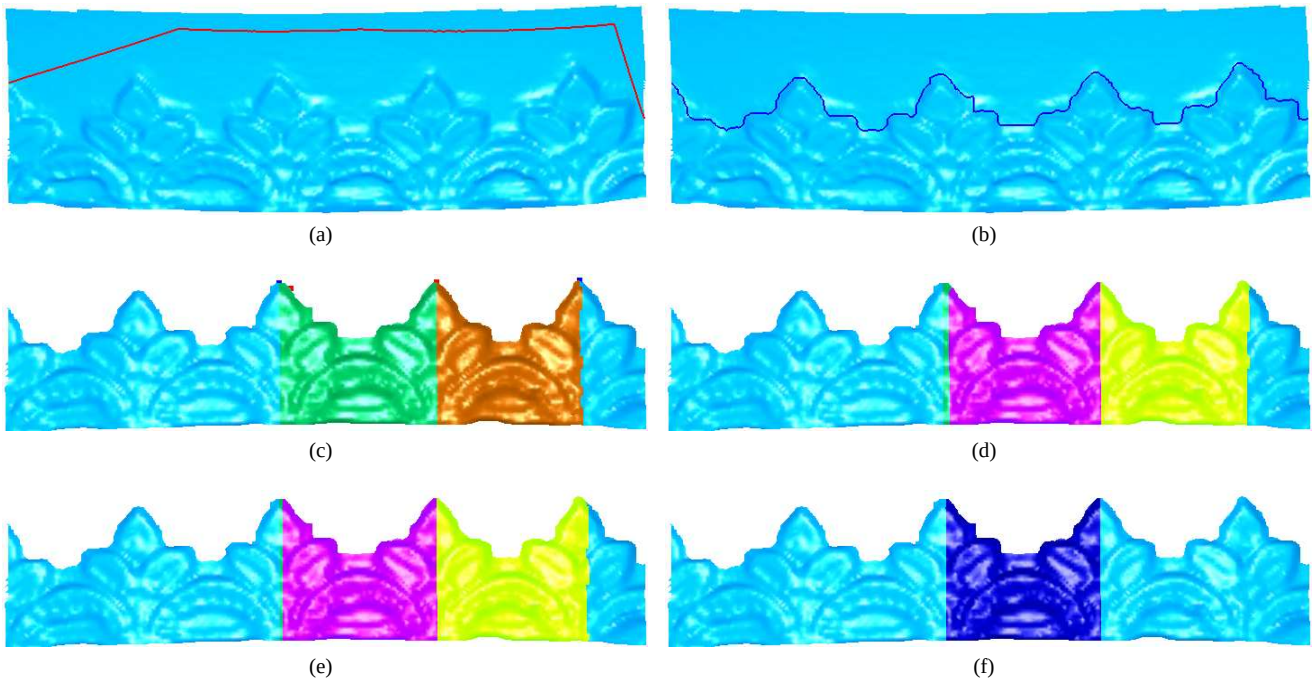

Figure 5: Periodic relief segmentation for an example with leaves: (a) mesh model and the initial contour, (b) extracted relief boundaries by snake evolution, (c) units separated from coarse extraction phase, (d) results from refinement phase, (e) results from localization phase, (f) final extracted unit

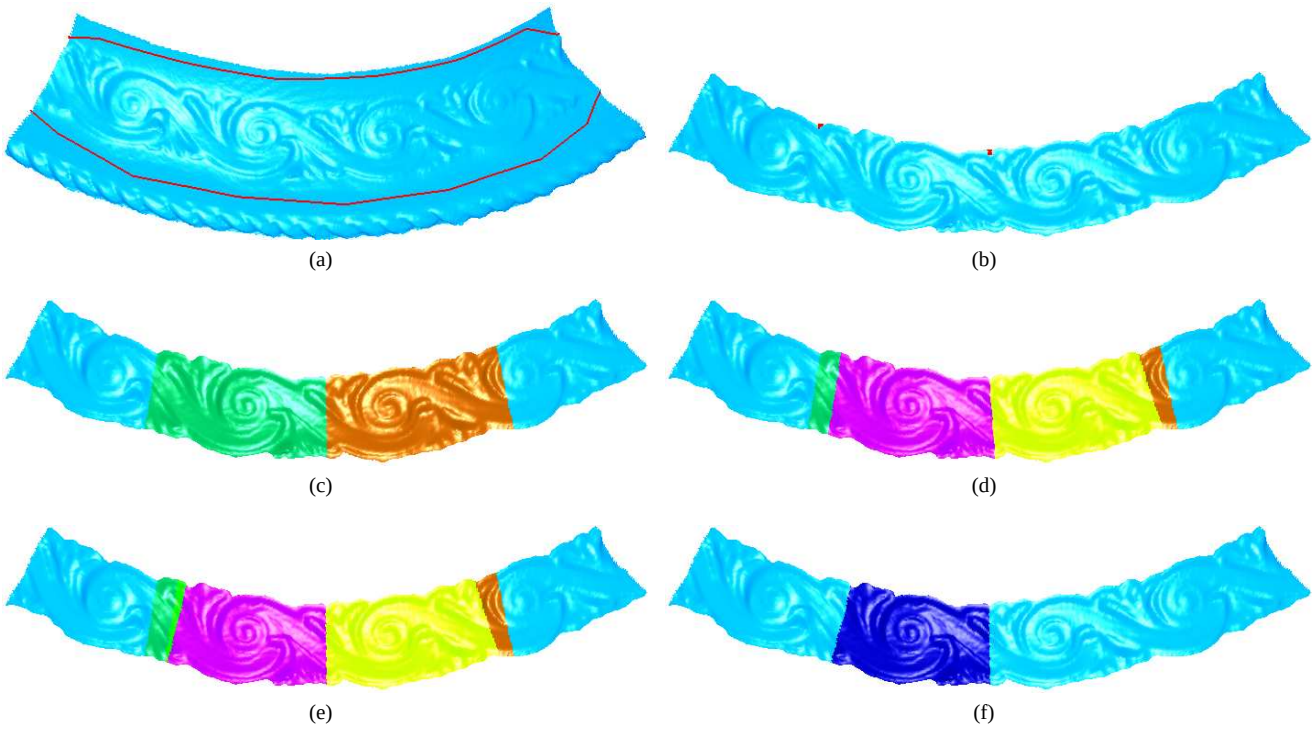

Figure 6: Periodic relief segmentation without using boundary repetitive extraction: (a) mesh model and the initial contour, (b) user-specified corresponding points, (c) units separated from coarse extraction phase, (d) results from refinement phase, (e) results from localization phase, (f) final extracted unit

Some patterns have local reflective symmetry as well as translational symmetry, e.g. the leaf pattern in Fig. 5. In future, we aim to investigate key matching of reflected units as well, in order to extract the primary semi-pattern.

Acknowledgements

The authors wish to acknowledge the support of Delcam plc, including many helpful discussions with Richard Barratt and Steve Hobbs, and the support of EPSRC grant GR/T24425, for this work.

References

BESL, P., AND MCKAY, N. 1992. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence 14*, 2, 239–256.

CUTLER, R., AND DAVIS, L. S. 2000. Robust real-time periodic motion detection, analysis, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence 22*, 8, 781–796.

EGGERT, D., FITZGIBBON, A. W., AND FISHER, R. B. 1996. Simultaneous registration of multiple range views for use in reverse engineering. In *Proceedings of the 13th International Conference on Pattern Recognition*, 243–247.

HANDLEY, C. 1998. The analysis and reconstruction of repetitive textures. In *Computer Graphics International*, 273–276.

HARALICK, R., SHANMUGAM, K., AND DINSTEIN, I. 1973. Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics, SMC-3 3*, 610–621.

HSU, J. T., LIU, L.-C., AND LI, C. 2001. Determination of structure component in image texture using wavelet analysis. In *Proceedings of International Conference on Image Processing*, 166–169.

LEWINER, T., JR., J. D. G., LOPES, H., AND CRAIZER, M. 2005. Least squares estimation of curvature and torsion. Technical Report, Department of Mathematics, Pontificia University, France.

LIU, Y., COLLINS, R. T., AND TSIN, Y. 2004. A computational model for periodic pattern perception based on frieze and wallpaper groups. *IEEE Transactions on Pattern Analysis and Machine Intelligence 26*, 3, 354–371.

LIU, S., MARTIN, R. R., LANGBEIN, F. C., AND ROSIN, P. L. 2006. Segmenting reliefs on triangle meshes. In *Proceedings ACM Symposiam Solid and Physical Modeling*, ACM, 7–16.

LIU, S., MARTIN, R. R., LANGBEIN, F. C., AND ROSIN, P. L. 2007. Segmenting geometric reliefs from textured background surfaces. To appear, Computer-Aided Design and Applications.

MARTINET, A., SOLER, C., HOLZSCHUCH, N., AND SILLION, F. 2005. Accurately detecting symmetries of 3d shapes. Tech. Rep. RR-5692, INRIA, September.

MATERKA, A., AND STRZELECKI, M. 1998. Texture analysis methods: A review. Technical Report.

MITRA, N. J., GUIBAS, L. J., AND PAULY, M. 2006. Partial and approximate symmetry detection for 3d geometry. *ACM Transactions on Graphics 25*, 3, 560–568.

PAJDLA, T., AND GOOL, L. V. 1995. Efficient matching of space curves. In *Proceedings of the 6th International Conference on Computer Analysis of Images and Patterns*, 25–32.

PAJDLA, T., AND GOOL, L. V. 1995. Matching of 3-d curves using semi-differential invariants. In *Proceedings, Fifth International Conference. Computer Vision*, 390–395.

PODOLAK, J., SHILANE, P., GOLOVINSKIY, A., RUSINKIEWICZ, S., AND FUNKHOUSER, T. 2006. A planar-reflective symmetry transform for 3d shapes. *ACM Transactions on Graphics 25*, 3, 549–559.

POTTMANN, H., HUANG, Q.-X., YANG, Y.-L., AND KOLPL, S. 2005. Integral invariants for robust geometry processing. Tech. Rep. 146, Geometry Preprint Series, Vienna Univ. of Techn.

SUN, C., AND SHERRAH, J. 1997. 3d symmetry detection using the extended gaussian image. *IEEE Transactions on Pattern Analysis and Machine Intelligence 19*, 2, 164–169.

ZUCKER, S., AND TERZOPOULOS, D. 1980. Finding structure in co-occurrence matrices for texture analysis. *Computer Graphics and Image Processing 2*, 12, 286–308.