Multi-Feature Matching of Fresco Fragments

Corey Toler-Franklin[*] Benedict Brown[†] Tim Weyrich[‡] Thomas Funkhouser[*] Szymon Rusinkiewicz[*]

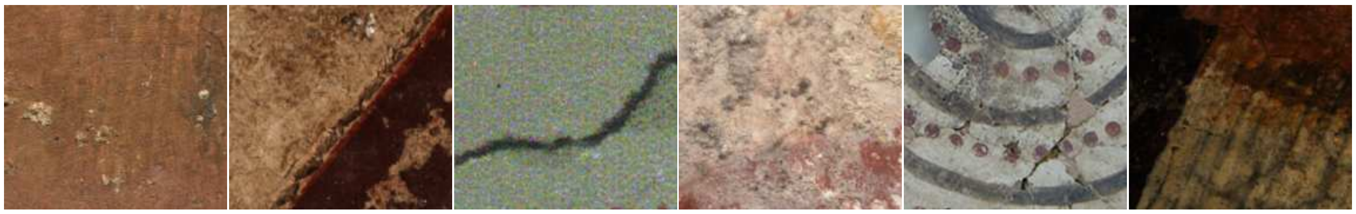

Figure 1: *Feature descriptors based on surface normal characteristics can capture a variety of physical characteristics such as brush strokes, string impressions, and erosion, that are used by archaeologists when assembling fresco fragments. Combining these with more traditional color-based features and 3D features using classification trees yields significantly improved matching performance.*

Abstract

We present a multiple-feature approach for determining matches between small fragments of archaeological artifacts such as Bronze-Age and Roman frescoes. In contrast with traditional 2D and 3D shape matching approaches, we introduce a set of feature descriptors that are based on not only color and shape, but also *normal maps*. These are easy to acquire and combine high data quality with discriminability and robustness to some types of deterioration. Our feature descriptors range from general-purpose to domain-specific, and are quick to compute and match. We have tested our system on three datasets of fresco fragments, demonstrating that multi-cue matching using different subsets of features leads to different tradeoffs between efficiency and effectiveness. In particular, we show that normal-based features are more effective than color-based ones at similar computational complexity, and that 3D features are more discriminative than ones based on 2D or normals, but at higher computational cost. Our results show good retrieval performance, significantly improving upon the match prediction rate of state-of-the-art 3D matching algorithms, and are expected to extend to general matching problems in applications such as texture synthesis and forensics.

1 Introduction

Advancements in low-cost, high-volume acquisition systems have made computer-assisted reconstruction of artifacts from small fragments practical. This problem is of particular interest to the field of archaeology, in which the reconstruction of artifacts such as shattered wall paintings reveals information about the history and culture of ancient civilizations. Historically, the process of reconstructing these wall paintings has been manual, occupying a major proportion of the human effort at excavation sites. As a result, wall painting reassembly is not even attempted at countless sites around the world, leading to a significant opportunity to advance our knowledge of ancient societies by improving the practicality of reconstruction.

[*]Princeton University. {ctoler, smr, funk}@cs.princeton.edu
[†]Katholieke Universiteit Leuven. bjbrown@esat.kuleuven.be
[‡]University College London. t.weyrich@cs.ucl.ac.uk

Several 2D and 3D computer-aided matching approaches have been explored and have proven successful in some domains. However, current matching algorithms have difficulty when matching artifacts that have deteriorated over many years. For example, they may consider features such as color, which frequently have changed over time even among neighboring fragments. Alternatively, they may operate exclusively on 3D geometry, which may not only have deteriorated, but is also challenging to acquire with the same fidelity and resolution as color images.

We address the problem of reconstruction by considering multiple cues based on color, shape, and — most interestingly — normal maps. The latter is a new source of information that has not been used for matching in previous work, and we argue that it combines high data quality and resolution with high discriminability and robustness with respect to certain types of deterioration. As has been recently demonstrated [Brown et al. 2008; Pintus et al. 2009], it is practical to use flatbed scanners to obtain normal maps of mostly-flat objects with 600 or 1200 dpi resolution. These normal maps reveal salient surface characteristics including string impressions, brush strokes, surface roughness, and fine cracks.

Our system begins with scanned images and normals of a collection of fragments, and computes a set of *feature descriptors*. Each descriptor may be computed over an entire fragment or over small patches sampled around the outer contour of the fragment: the tradeoff of sampling patches is greater discriminability for greater computation time. The feature descriptors range from general-purpose (such as variance in the normal map) to ones designed specifically for the domain of fresco fragment matching, and motivated by the visual cues used by archaeologists for reassembly (such as brush stroke direction). They are designed to capture characteristics including shape, surface decoration, surface texture, and deterioration (Figure 1).

We use similarity of these descriptors to suggest matches, and evaluate their performance on three different wall-paintings. The first is a geometric scene containing spirals and large areas of constant color, from a late-bronze-age Aegean civilization at Akrotiri, on the island of Thera (Santorini). The second is from a Roman villa at Kerkrade, The Netherlands, and is especially distinctive because of the strong brush marks visible both in the color and the surface relief. The third is a synthetic fresco, professionally created and shattered, for which a ground-truth reconstruction is available.

We perform a cross-validation analysis on databases of dozens to hundreds of fragments, drawn from the three different wall-paintings. Our results demonstrate the discriminative power of our collection of features, and suggest that matching performance is improved by the use of normal maps, in addition to features based on more conventional data sources such as color, thickness, and exterior contour shape. Moreover, we observe that the performance of individual features varies from dataset to dataset, suggesting a future extension to online learning.

Overall, the paper makes the following contributions:

- The introduction of a new type of input, *normal maps*, for matching small fragments of artifacts. We argue that normal maps are easy to acquire with higher resolution than 3D models, and are more robust to deterioration and discoloration than color.
- A set of easily computable descriptors, of both general purpose and domain specific types, that are effective for matching.
- Analysis and evaluation methods that demonstrate how well our features perform over state-of-the-art 3D match algorithms.
- A matching framework that is easily extendable to more generalized matching problems used in applications such as texture synthesis and forensics.

2 Previous Work

2D Matching: Traditional matching algorithms use 2D contours [Kong and Kimia 2001; Leitão and Stolfi 2002; Papaodysseus et al. 2002] as well as image-space features including color and texture [Fornasier and Toniolo 2005; Sağiroğlu and Erçil 2006]. However, these solutions are often sensitive to erosion and discoloration, a significant issue for fragments that have spent thousands of years exposed to natural elements. Moreover, they do not consider the wealth of 3D information available in geometric representations. Such cues are particularly important in our domain where impressions on the fragment surface provide strong matching cues.

3D Matching: Other approaches for assembling fractured objects incorporate full 3D descriptions. For example, Huang et al. [2006] reassemble solid objects by first identifying fractured regions, then generating clusters of feature patches for alignment-based matching. Although these feature clusters effectively describe the local geometry of the fracture surface, the algorithm does not consider other physical attributes of the dataset, and is burdened by the complexity of a full 3D matcher. Brown et al. [2008] exploits the orientation constraints of flat fragments to achieve a simple, fast matcher based on edge geometry. This matcher resamples the fragments edges in a regular grid structure, then exhaustively tests every possible alignment of a pair of fragments in a few seconds, in a correlation-like manner. This approach takes advantage of high resolution geometry to find precise alignments, and mirrors the common technique of finding matching fragments by testing for pairs that physically “lock” together. On the other hand, fragment edges are subject to erosion, and the brute-force nature of the algorithm means there is no early rejection for non-matching pairs.

Our approach retains the efficiency of a special-purpose matcher for flat objects, but focuses on fine surface details rather than edge information. It is complementary to existing geometry based matchers in two important ways. First, it matches features on the *external* surface of fragments rather than fitting fractured faces to each other. Second, it relies on high resolution normals captured with a flatbed scanner that could not be acquired reliably with current stereo-based scanners or fed into an alignment algorithm.

Reassembling Artifacts: Several computer-aided systems have been designed specifically for reassembling broken objects. One notable example is the Forma Urbis Romae project [Koller et al. 2006], where analysis of incision points and markings is used to match sparse data. While some aspects of these heuristics are of broader applicability (for example, in lining up fragments with string impressions), they are largely tuned to the specific needs of the Forma Urbis Romae. Another common application in the field of archaeology is the reassembly of broken pottery [Willis 2004; Karasik and Smilansky 2007]. Just as we take advantage of properties of fresco fragments to obtain an effective, efficient matcher, these algorithms rely on finding the axis of rotation and profile curve common to pottery.

Our approach improves upon these examples because we interpret observed qualities of our domain as a set of functions that are easy to compute, optimizing our system to use a combination of the most discriminative criteria for matching. Although we incorporate some 3D quantities, such as thickness, we maintain the ease and simplicity of a 2D system by only computing information in image space.

3 Overview

In this paper, we focus on obtaining feature descriptors from a database of scanned patches of objects, focusing on an archaeological fragment matching scenario. We describe our feature descriptors, match classification strategies, and the datasets on which we operate. We use three forms of data: *color maps* acquired using a high-resolution (600 dpi) 2D scanner, *normal maps* obtained from multiple scans using a variant of shape from shading [Brown et al. 2008], and *3D meshes* from a laser-triangulation range scanner. We work with these data types because they can be obtained in situ, at an archaeological excavation or in the context of some other digitization effort, with high fidelity, low cost, and considerable ease of acquisition.

3.1 Feature Descriptor Generation

Figure 2 presents a conceptual overview of our feature descriptor pipeline. Because practical datasets may contain thousands of fragments, we focus on designing a matching pipeline capable of scaling to these data sizes. Indeed, a brute-force solution that tested every possible alignment of every possible pair of fragments would quickly become infeasible, requiring perhaps 10^{10} to 10^{12} comparisons (a few thousand fragments times a few hundred orientations, squared). To overcome this growth we employ a sequence of matching stages, ranging from ones that can quickly reject a large number of implausible candidates to ones that precisely check individual matches. A key observation is that the early stages should require computation that grows *linearly* with the number of fragments, rather than quadratically.

We thus consider three possible classes of features. The first includes **per-fragment** features: those that are computed (once) for each fragment in our database. Fragments that differ greatly in the computed descriptors are assumed to have a low probability of matching, hence generation of plausible matching pairs of fragments could be accelerated with a fast clustering or indexing

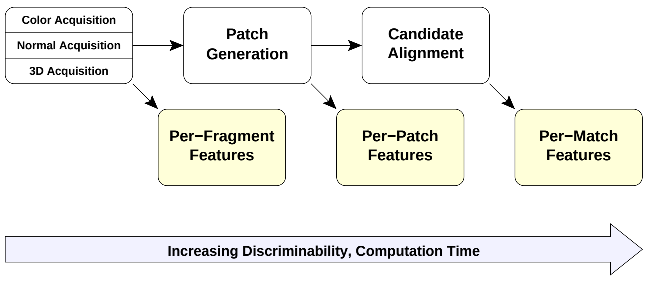

Figure 2: *An overview of feature generation. Beginning with high-resolution color scans, normal maps, and 3D models, we extract a variety of feature descriptors from each fragment. Descriptors may be extracted at the fragment level, for small sampled patches, or for a specific candidate match. Although we are able to match using all features simultaneously using machine learning techniques (Section 7), it is advantageous to first select possible matches with per-fragment and per-patch features, then compute the more expensive per-match features only for these possibilities.*

technique (although in this paper we focus on demonstrating the matching accuracy of the descriptors, rather than on evaluating their efficiency).

The second class of features we consider are **per-patch** features. These are computed not over entire fragments, but over small regions sampled around the boundary. Because these features consider more localized properties, we expect that they will be more discriminative of true matches. On the other hand, they also require more computation than the per-fragment features, since they must be evaluated at dozens to hundreds of locations around the perimeter of a fragment. Nevertheless, the descriptors are precomputed and cached, once per fragment, adding only a few seconds to the preprocessing time. The patches we use are circular and are sampled from the original 600 dpi images, every 5 mm along the perimeter of the fragment. For most features, we use patches 10 mm in diameter, and offset them 8 mm inward from the outside contour. This ensures that the resulting features are not corrupted by the very edges of the fragments, which are frequently broken off, eroded, or shadowed. For a few features we also use larger patches — 20 mm in diameter — to estimate properties more accurately and robustly, and for the area-based curvature descriptor the patches are centered on the contour instead of being offset inward. These variants are discussed below, in the descriptions of individual features.

Finally, we employ **per-match** features, which evaluate the plausibility of a candidate pair of fragments in a particular alignment. This stage is the most expensive, since it involves computing features per *pair* of fragments, and precludes the use of fast clustering or indexing methods. On the other hand, such descriptors (e.g., average distance between the fragments) can be more discriminative.

3.2 Match Classification

There are many types of machine learning tasks that can effectively use vectors of features: indexing, match scoring, classification, etc. As mentioned above, in the context of a large-scale fragment matching application we anticipate two main uses. First, in the early stages of matching the goal is to *quickly* determine large sets of potentially-matching pairs of fragments. In the ideal case, this stage would run in $O(n)$ time for n fragments, in contrast to the naive $O(n^2)$ strategy of checking every potential pair. Therefore, we anticipate that indexing and clustering methods are relevant, implying that we would like to determine which feature vectors are far apart, and which are nearby.

Later in the pipeline, the relevant task becomes separating matches from non-matches as effectively as possible. This may operate either via classification — predicting whether a proposed pair is likely to be a match or nonmatch — or via probabilistic regression — ordering proposed pairs from most to least likely. Either way, the most likely matches will, in the end, be presented to the user for ground-truth verification, meaning that all of the above strategies are amenable to incorporation in an “online learning” system that incrementally adjusts the importance of different features to adapt to the particular characteristics of each new database.

We therefore have four tasks — indexing, match classification, regression, and online adaptation to per-database feature importance — that all stem from the same set of features. In this paper we present results for match classification experiments, since it is likely that good performance on this task will lead directly to good performance on the others. We adopt an existing technique (decision trees) for producing trained classifiers, and explore classification performance using a cross-validation methodology. In most cases, the trees are simply trained on the absolute value of the difference between feature descriptor values for a pair, but a few cases require a more complex computation to convert the values of feature descriptors into a value likely to be predictive of a *match*. We also examine the typical variation in the different features.

3.3 Datasets

We evaluate our features using scanned frescoes from archaeological excavations at Akrotiri and Kerkrade, as well as a modern-day “synthetic” fresco data set.

Akrotiri: The Theran frescos were discovered on the island of Thera (modern-day Santorini), at the site of Akrotiri. Around 1650 B.C. the late-Bronze-Age Aegean civilization that occupied the island was destroyed by a volcanic eruption. The most important finds at Akrotiri are the extensive wall paintings, which, although broken into small fragments, have been well preserved by the volcano’s ash. In fact, the completeness of these wall paintings is unique in the ancient Mediterranean. However, the Theran wall paintings are known for their large fields of white or other solid colors, making manual reassembly especially difficult. Another distinguishing feature is the presence of surface impressions left by strings that were used as guides and placed in the wet plaster by artists. In this paper, we work with a dataset of 1200 fragments taken from a fresco with spiral motifs.

Kerkrade: The Kerkrade frescoes originate from a second-century Roman villa in Kerkrade, The Netherlands, near present-day Heerlen and Maastricht. They belong to the larger set of quality paintings from the Roman period found in the Netherlands, and are also a part of a select few that depict large-scale human figures. The Kerkrade fragments differ from those at Akrotri in two important ways: they are more eroded, and have visible brush strokes and texture resulting from the smoothing out of the plaster. We therefore, expect a different subset of the features to be important for matching. Our test set consists of 100 fragments.

Synthetic: This synthetic fresco, previously described by Brown et al. [2008], was created by conservators in a style similar to the one used at Akrotiri. The finished fresco was then broken into pieces to create fragments similar to the fragments found at that site. This fresco is characterized by large areas of white with smaller regions of color. Both string impressions and brush strokes are present on the fragment surfaces. Our ground truth set consists of 127 fragments.

4 Feature Descriptors

As mentioned above, our feature descriptors may be classified according to their type (per-fragment, per-patch, per-match) and the data from which they are computed (colors, normals, 3D). They range from “generic” descriptors that are frequent components of fragment-matching or puzzle-assembly systems, to descriptors that, while still general, were inspired by cues used today by conservators and archaeologists to perform manual matching.

In the below descriptions, we focus greatest attention on normal-based features: it is one of the claims of this paper that such features combine high classification performance with low acquisition cost and high matching efficiency (i.e., the features are largely per-fragment and per-patch, rather than per-match). These claims will be evaluated in the subsequent sections. Nevertheless, for completeness, we also describe the more “usual” features used by our system.

Average Color, Saturation, and Variance

Type: Per-Fragment and Per-Patch **Data:** Color

We begin with features traditionally used in image-based matching systems, such as the mean color (computed separately for each color channel) and color variance, both of which may be computed both per-fragment and per-patch. In addition, we use the color saturation as a feature. This was inspired by the observation that two adjacent fragments will often exhibit a similar amount of deterioration in their pigments: either they are both faded, or both retain their original colors. Though an imperfect descriptor, we include this in the hope that it may combine with other features to boost classification performance.

Contour Curvature

Type: Per-Patch **Data:** Color

The curvature of the fragment's outline, or contour, provides a *per-patch* descriptor that can group fragments of similar external shape, and has been frequently used for (2D) puzzle reconstruction. (Similar patches will, of course, have curvature of similar magnitude but *opposite* sign.) We have experimented with two alternative descriptors for 2D curvature along the fragment contour. An *area-based* descriptor finds the fraction of the fragment covered by a circular patch centered on the contour:

$$\mathit{CurvatureArea} = \frac{\mathit{Area(Fragment)}}{\mathit{Area(Patch)}}. \tag{1}$$

As shown by Manay et al. [2004], this quantity is just a function of curvature, in the limit of small patch size: values of 0, $\frac{1}{2}$, and 1 correspond to curvatures of $+\infty$, 0, and $-\infty$, respectively.

The second curvature descriptor only looks locally at three adjacent points A, B, and C on the contour:

$$\mathit{CurvatureContour} = 2\,\frac{\angle(C-B) - \angle(B-A)}{\|C-B\| + \|B-A\|}, \tag{2}$$

where the numerator is the angle between the segments $\overline{AB}$ and $\overline{BC}$, and the denominator is their total length. This discrete approximation to curvature is accurate for low-curvature regions, as is generally the case in practice. The points A, B, and C are picked at multiple scales: 2.5 mm, 5 mm, 8 mm, 10 mm, and 15 mm. Each one yields a separate descriptor, providing even more information about the contour shape to be used in matching.

Average Normal and Variance

Type: Per-Fragment and Per-Patch **Data:** Normal

Many of the datasets we have examined exhibit significant variation in surface roughness from location to location: some regions are smooth while others are rough because of visible brushstrokes, weathering, or the use of a different type of plaster. In order to characterize this, we look at the distribution of normals on the fragment or patch. However, we cannot simply consider the normal vectors themselves: one of their components is not known, in global coordinates, since the final orientation of the fragment is unknown. For this reason, we form a rotation-invariant quantity: the z component of the normals (i.e., the component perpendicular to the fragment's "flat" surface). We use the mean and variance of these z components as features.

Color/Normal Variation

Type: Per-Fragment and Per-Patch **Data:** Color and Normal

This descriptor captures the effect of *correlated* variation in color and normals, as frequently occurs when there are visible brush strokes or string impressions that were used as guides for painting. We begin by stacking the colors and normal z components for pixels in a fragment or patch into an $n \times 4$ matrix, then perform a Singular Value Decomposition:

$$\begin{pmatrix} c_{1,r} & c_{1,g} & c_{1,,b} & n_{1,z} \\ & \vdots & & \\ c_{n,r} & c_{n,g} & c_{n,b} & n_{n,z} \end{pmatrix} = U \begin{pmatrix} \sigma_1 & 0 & 0 & 0 \\ 0 & \sigma_2 & 0 & 0 \\ 0 & 0 & \sigma_3 & 0 \\ 0 & 0 & 0 & \sigma_4 \end{pmatrix} V^T. \tag{3}$$

We use the sum of the σ_i as a descriptor, yielding a compact, rotation-invariant, yet discriminative description of the nature of color and normal variation on the surface.

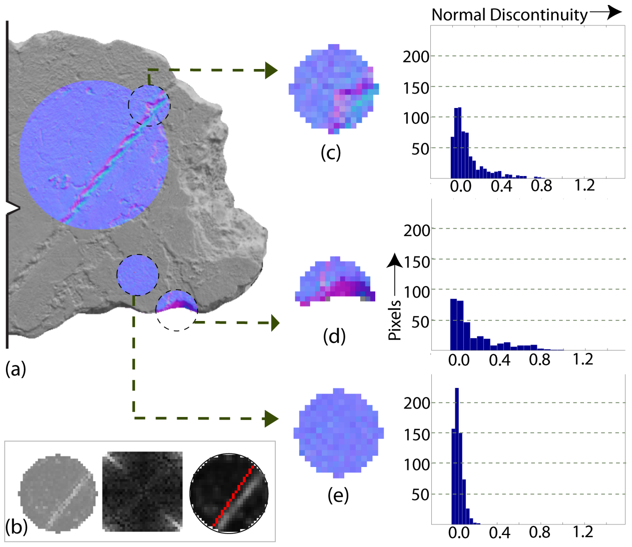

Figure 3: ***(a)*** *Fragment with a visible string impression, for which we visualize the normal-discontinuity and dominant-orientation descriptors.* ***(b)*** *The dominant-orientation descriptor detects the orientation of the string impression, allowing it to be matched to an impression on an adjoining fragment. Visualized are the mean component of the normal, the power spectrum, and the detected dominant orientation.* ***(c-e)*** *Normal-z histograms used by the normal-discontinuity descriptors. Note that the histogram in (c) has a long tail, while the smooth patch in (e) results in a histogram clustered around the origin. We are careful to inset patches away from the edge to avoid incorrect long-tailed distributions, as would result from sampling patches directly on the edge (d). We also have experimented with using larger patch sizes, as shown in (a).*

Normal Discontinuity

Type: Per-Fragment and Per-Patch **Data:** Normal

Many frescoes contain distinctive shapes on the front surface. For example, artists occasionally pressed a guide string against the wet plaster during fresco construction, in order to provide an outline for straight bands of color. We posit that a strong matching cue is the presence of such features, and we design descriptors that distinguish between relatively smooth patches (possibly with some noise in the normals due to deterioration) and those containing large discontinuities in the surface orientation.

Our analysis begins by computing a histogram of the differences $\{D_{ij}\} = \{\|n_{i,z} - n_{j,z}\|\}$ between neighboring normals in a patch. (We use only the z components of the normal vectors because they are invariant to rotation in the plane.) We then compute a number of statistical measures on this distribution, to determine the degree to which it either is strongly peaked around zero or contains long "tails" of high curvature. We have experimented with four measures to characterize the degree to which the normal difference distribution exhibits long tails:

1. The ratio between the normal differences' 80th and 50th percentiles:
$$\frac{D_{80\%}}{D_{50\%}}; \tag{4}$$
2. The fraction of discontinuity values greater than a threshold (0.46), determined experimentally by looking at fragments containing string impressions;
3. The third moment of the distribution; and
4. The fourth moment of the distribution.

Figure 3, (c) and (e), shows the normal discontinuity distributions for patches that both do and do not contain string impressions.

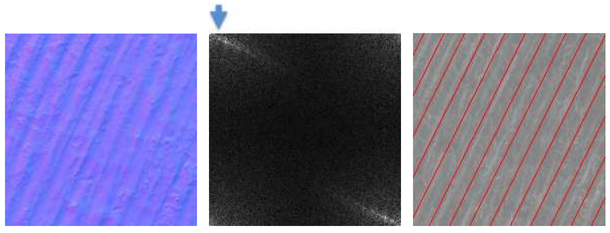

Figure 4: *Detecting brush-stroke amplitude and orientation. This illustration depicts the results of the dominant orientation detector on a sample taken from a fragment in the Kerkrade dataset. The first two images show the normal map and power spectrum image, respectively. We determine the strength, and direction of the brush-strokes by examining the frequency, amplitude and orientation, of the dominant frequency (not including the dc) in the power spectrum. The red lines in the image at right visualize the orientation of this dominant peak.*

Note that the presence of the string causes the distribution to have significantly more large values.

Dominant Orientation

Type: Per-Fragment and Per-Patch **Data:** Normal

This feature detects regular patterns in the surface, such as the brush strokes found on some fragments. We were inspired by examples such as the ones in Figure 9, which illustrates the variation in strokes left by both paintbrushes and tools for smoothing out the underlying plaster. We expect that the amplitude and frequency of these brushstrokes, as extracted from normal maps, will strongly group them according to local variation in their characteristics. In addition, the orientation of these brush strokes must be continuous across fractures, leading to another strong matching cue that essentially eliminates the search over orientation in possible matching fragments.

We begin by smoothing the z-normal image of the fragment or patch, applying a Hanning window, then finding its 2D Fourier transform $F_n(\omega)$. We then search for the frequency at which the energy is greatest. Because the patches are smooth, the highest-energy peak is usually the DC component, and low frequencies are generally stronger than high frequencies. For this reason, we apply a threshold to the frequency:

$$NormalDominantFrequency = \arg\max_{\omega > \omega_{min}} \|F_n(\omega)\|^2 \ . \tag{5}$$

In addition to using the frequency and amplitude of this peak as features, we also use its orientation. In order to make this invariant to rotation, we compute the difference between the angle of the dominant peak and the normal to the fragment contour. In other words, we store the angle between the dominant directional variation (e.g. brush strokes) and the fragment edge — this quantity is expected to be the same for a matching fragment. Figure 4 demonstrates the results of this process on a fragment from the Kerkrade dataset (see Section 3.3) containing strong brush strokes.

Cracking and Erosion

Type: Per-Fragment and Per-Patch **Data:** Color and Normal

Erosion of a plaster fresco frequently results in small pits in the surface, while the destruction of the original wall-painting produces an irregular pattern of cracks. The erosion descriptor uses morphological operators to quantify the degree of deterioration on the surface. Specifically, we extend versions of the black and white top-hat transforms [Serra 1983]:

$$\begin{aligned} T_{white}(f,\kappa) &= f - f \circ \kappa \\ T_{black}(f,\kappa) &= f \bullet \kappa - f, \end{aligned} \tag{6}$$

where $\circ$ and $\bullet$ are image-morphological open and close operators and κ is a structuring element.

Unlike previous applications, we achieve improved discriminability by combining color and normal values in the analysis. As shown in Figure 5, for patch-level erosion detection, we take the intersection of peaks in the threshholded black top-hat transform of the color map and the threshholded white top-hat transform of the normal map:

$$Erosion(I_c, I_n, \kappa) = \ T'_{black}(I_c,\kappa) \cap T'_{white}(I_n,\kappa) \tag{7}$$

where I_c is a grayscale version of the color buffer, I_n is the z component of the surface normal, and T' are the thresholded top-hat transforms. We found that an intensity threshold of 0.3 works well. For the structuring element κ, we use a circle (with a 3 pixel radius) to ensure our results are rotation invariant across fragments. We record two scores: the total number of pixels over all peaks (normalized by the number of visible pixels) and the average number of pixels per connected component. The first value records the density while the latter suggests the average size of each element.

At the fragment level, we use a multi-scale approach. In this case, we use a structuring element with ten different diameters and take the color black-top transform to be the sum of the transform taken across the ten scales. Similarly, the white top-hat transform is the sum of the white top-hat transform for the normal map over the ten scales. Figure 6 shows an example of fragment level erosion. We chose this approach over a single large scale because features are less likely to expand beyond their boundaries. Multi-scale morphological scale spaces work locally, are good at separating features from uneven backgrounds and do not exhibit the blurring across features characteristic of Gaussian kernels.

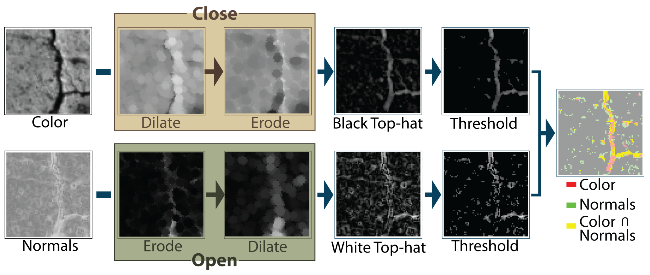

Figure 5: *Erosion detection using a circular structuring element to find cracks and pits in the surface. The black top-hat transform is generated by applying a closing operation to the color image and subtracting the result from the original color image. The white top-hat transform is generated by applying an opening operation to the normal map and then subtracting it from the original normal map. The final erosion map is the intersection of the black top-hat transform of the color map and the white top-hat transform of the normal map, and thresholding to reduce noise.*

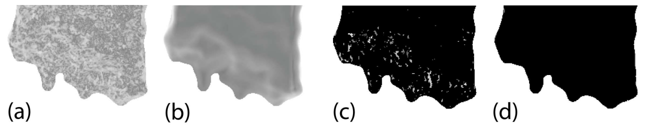

Figure 6: *Comparing erosion detection on 2D normal maps and 3D geometry. Note: We only show the z component of the normals. (a) The 2D normal map acquired with a flat-bed scanner has more detail than the smoother normals in (b) acquired with a 3D scanner. (c) Areas of erosion, highlighted in grey, are detected on the 2D normal map, but no erosion is detected on the 3D geometry (d).*

Thickness

Type: Per-Fragment and Per-Patch **Data:** 3D

In some datasets, the thickness of fragments varies considerably from location to location. Therefore, we use the 3D scan to determine the fragment's thickness at every point. We use the average thickness, per fragment or per patch, as a descriptor. Note that in this case it is especially critical that we offset each patch away from the edge of the fragment, since the estimated thickness is likely to be unreliable near the edges.

Ribbonmatcher Error and Volume Intersection

Type: Per-Match **Data:** 3D

To compare the performance of our per-fragment and per-patch descriptors to more descriptive per-match features, we look at two values computed via the brute force "ribbon-matching" approach of [Brown et al. 2008]. For both values, the optimal alignment of the two fragments at the patch locations is computed using the ribbon matcher with a 12.5 mm strip width. The first value ("ribbon error") we consider is the mean-squared distance between fragments along this strip, as computed by the ribbon matcher. The second value ("volume intersection") measures the amount of interpenetration between fragments. A vertical plane is oriented along the matching edge, and the interpenetration of the two fragments is sampled on this plane. We compute the average of all squared lengths that exceed 1 mm. The intuition behind this descriptor is that correctly matching fragments may have some slight interpenetration due to sampling error, erosion (which affects the alignment), and accretions on the fragment edge when it was scanned. However, correctly matching fragments should not have any *substantial* interpenetration. Considering only interpenetrations greater than 1 mm accounts for "explainable" interpenetration, and squaring the distances penalizes deep intersections more than shallow ones. Note also that the volume descriptor considers the entire fragments, not just selected patches or ribbonmatcher strips.

5 Case Studies of New Features

Several of the features we consider are motivated specifically by the fresco-matching application, rather than being "generic" features applicable to a variety of shape matching problems. Here we consider a few of these, and present anecdotal evidence for their performance.

Normal-based Features: While we have found that color cues are meaningful for some datasets and perform no better than chance on others, we have uniformly observed that normal-based features provide reasonable performance. For example, Figure 7 shows the best match found between a pair of fragments from the Kerkrade fresco using only color cues (center) and using normal-based features (right); only the latter is correct. We hypothesize that even in datasets that exhibit considerable color variation throughout the fresco, the variation within a single fragment is usually insufficient to yield the correct alignment.

Thus, we suggest that normal-based features combine the computational efficiency and ease of use of conventional 2D features while improving upon classification performance in many cases.

Erosion: We analyzed our erosion detection features on a number of fragments exhibiting strong cracking, strong color variation without erosion, and mild erosion (Figure 8). In each case we show the results of running our top-hat operators on the colors and normals, as well as their intersection. We found that the intersection of top-hat transforms applied to both the colors and normals gave good sensitivity to detection of cracking and erosion. At top, the normals detected all of the cracks, while the color served to limit sensitivity to additional normal variation. At center, the lack of variation in the normals successfully suppressed the detector in areas of color

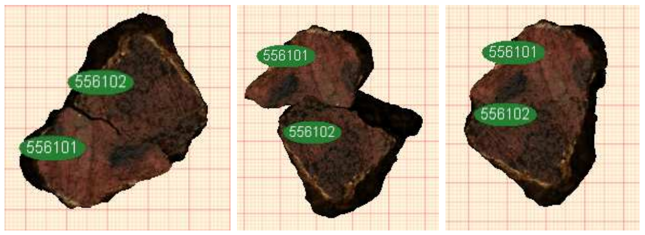

Figure 7: ***Left:*** *ground-truth match between two fragments of the Kerkrade fresco.* ***Center:*** *best match, only considering this pair of fragments (at all orientations) and color-based features.* ***Right:*** *best match, considering normal-based features — notice that the correct match was found.*

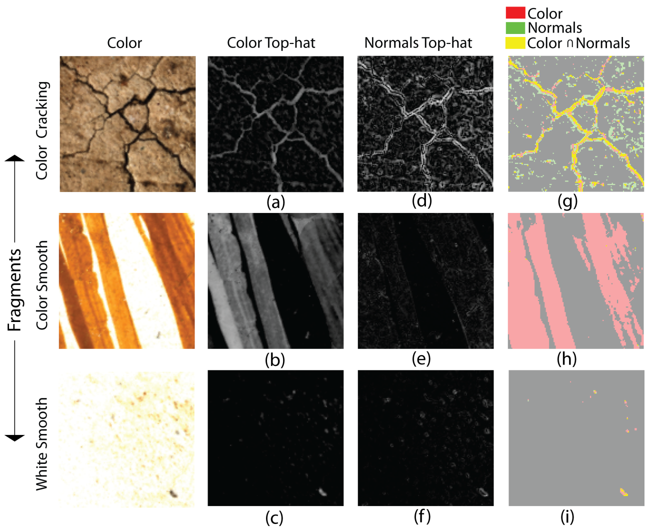

Figure 8: *When computing erosion, we take the intersection of the top-hat transforms of the color and normal maps to avoid capturing sharp variations in color due to dirt or stains, and high frequency noise in the normal maps. (a - c) Black top-hat transforms of color maps for three fragments (color and cracking, color and smooth, white and smooth). (d - f) The white top-hat transforms of the corresponding normal maps. (g - i) Intersection of black and white top-hat transforms.* ***Top:*** *Erosion is detected when there are cracks and pits in both the color and normal maps as shown by the yellow pixels in (g).* ***Middle:*** *No erosion is detected on the smooth fragment with color. There are no green or yellow pixels in (h) but several red pixels, representing the lines of color detected by the black top-hat operator.* ***Bottom:*** *Only a few pits are detected on the smooth white fragment.*

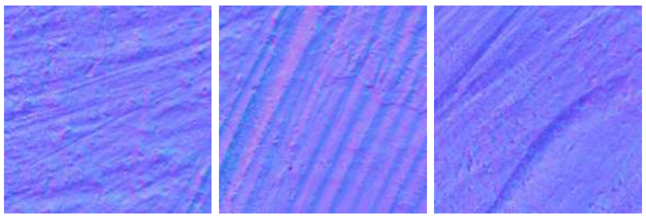

Figure 9: *Normal maps illustrating the wide variation in brush strokes on the Kerkrade fresco, including strokes left by a paintbrush (left) and strokes left when smoothing out the underlying plaster (center and right). Not only is the direction of these strokes (measured as the angle between their dominant orientation and the fragment contour) a strong cue for matching, but the amplitude and frequency help distinguish between these three types of strokes as well.*

Table 1: *Statistics of correspondence values (typically absolute values of differences of feature descriptor values) for random matches and non-matches in the "Synthetic," "Akrotiri," and "Kerkrade" datasets.*

	Feature	Synthetic (min / mean / max / stdev)	Akrotiri (min / mean / max / stdev)	Kerkrade (min / mean / max / stdev)
Color	ColorAvgR	0.000 / 0.054 / 0.978 / 0.157	0.000 / 0.170 / 0.613 / 0.131	0.000 / 0.135 / 0.505 / 0.097
	ColorAvgG	0.000 / 0.071 / 0.977 / 0.186	0.000 / 0.158 / 0.581 / 0.126	0.000 / 0.077 / 0.277 / 0.058
	ColorAvgB	0.000 / 0.092 / 0.968 / 0.199	0.000 / 0.122 / 0.589 / 0.103	0.000 / 0.040 / 0.174 / 0.034
	ColorVariance	0.000 / 0.040 / 0.628 / 0.083	0.000 / 0.026 / 0.167 / 0.022	0.000 / 0.008 / 0.085 / 0.009
	ColorSaturation	0.000 / 0.074 / 0.883 / 0.160	0.000 / 0.131 / 0.589 / 0.096	0.000 / 0.084 / 0.410 / 0.078
Curv	CurvatureContour	0.000 / 0.032 / 0.382 / 0.033	0.000 / 0.098 / 0.572 / 0.085	0 000 / 0.089 / 0.513 / 0.076
	CurvatureArea	0.000 / 0.069 / 0.414 / 0.059	0.000 / 0.069 / 0.410 / 0.058	0.000 / 0.074 / 0.445 / 0.061
Normal	ColorNormalVariance	0.000 / 0.051 / 0.896 / 0.098	0.000 / 0.036 / 0.227 / 0.029	0.000 / 0.036 / 0.242 / 0.035
	NormalMeanZ	0.000 / 0.038 / 1.155 / 0.067	0.000 / 0.143 / 0.634 / 0.118	0.000 / 0.074 / 0.383 / 0.064
	NormalVariance	0.000 / 0.056 / 0.483 / 0.073	0.000 / 0.064 / 0.712 / 0.064	0.000 / 0.059 / 0.301 / 0.049
	NormalDiscont8050Ratio	0.000 / 0.704 / 8.000 / 1.198	0.000 / 0.406 / 2.000/ 0.361	0.000 / 0.590 / 4.000 / 0.606
	NormalDiscontThresholded	0.000 / 0.053 / 0.833 / 0.109	0.000 / 0.201 / 0.893 / 0.149	0.000 / 1.086 / 0.584 / 0.098
	NormalDiscontThirdMoment	0.001 / 1.665 / 9.649 / 1.504	0.001 / 0.389 / 5.085 / 0.575	0.000 / 1.171 / 3.926 / 1.029
	NormalDiscontFourthMoment	0.004 / 15.647 / 147.952 / 18.466	0.000 / 1.746 / 50.572 / 4.222	0.001 / 3.546 / 61.146 / 4.760
	NormalDominantFrequency	0.000 / 0.111/ 1.394/ 0.227	0.000 / 0.711 / 8.314 / 0.914	0.000 / 0.301 / 2.798 / 0.358
	NormalDominantOrientation	0.000 / 0.727 / 2.931 / 0.549	0.002 / 0.826 / 3.028 / 0.649	0.002 / 0.767 / 3.033 / 0.593
	NormalDominantAmplitude	0.000 / 0.007 / 0.096 / 0.001	0.000 / 0.015 / 0.083 / 0.012	0.000 / 0.012 / 0.007 / 0.010
	ColorNormalErosionDensity	0.000 / 0.01 / 0.211 / 0.023	0.000 / 0.007 / 0.089 / 0.007	0.000 / 0.004 / 0.087 / 0.010
	ColorNormalErosionShape	0.000 / 0.006 / 0.211 / 0.016	0.000 / 0.036 / 0.24 / 0.037	0.000 / 0.002 / 0.065 / 0.005
3D	Thickness	0.001 / 1.392 / 8.432 / 1.399	0.002 / 2.953 / 12.821 / 2.221	0.000 / 4.700 / 22.199 / 4.022
	RibbonError	0.003 / 8.412 / 282.209 / 16.806	0.000 / 11.701 / 136.566 / 14.536	0.082 / 15.742 / 112.204 / 16.765
	RibbonVolIntersection	0.000 / 253.265 /5681.626 /648.324	0.000 / 293.093 / 14545.190 / 804.263	0.000 / 151.188 / 3183.708 / 308.539

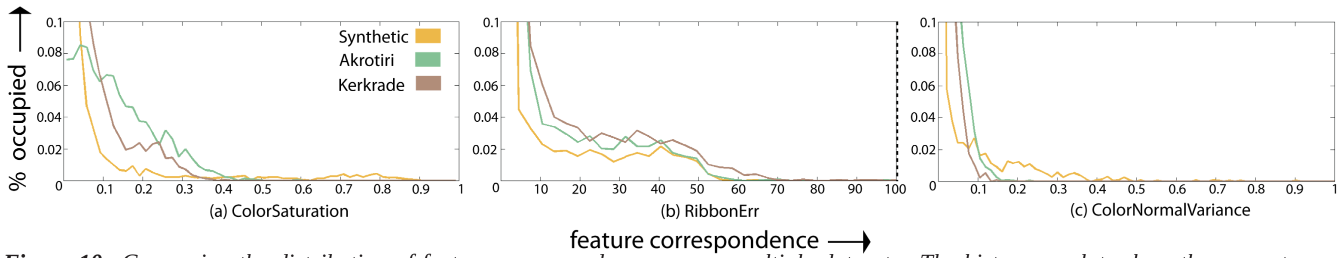

Figure 10: *Comparing the distribution of feature correspondences across multiple datasets. The histogram plots show the percentage of correspondences with the specified feature correspondence ranges for the Synthetic, Akrotiri and Kerkrade frescos. In this example 2,274 correspondences, consisting of both matches and non-matches were taken from each dataset. (a) There is more variation in correspondences for ColorSaturation for the Akrotiri and Kerkrade datasets than the Synthetic dataset which is primarily composed of white fragments. (b) The RibbonErr correspondences are similar across all three datasets. (c) The shape of the curves show that ColorNormalVariance correspondences are more similar for Akrotiri and Kerkrade than the Synthetic dataset. The variation of correspondence ranges across the different datasets suggests that re-weighting the contribution of individual features based on observed statistics of the dataset would adaptively improve match retrieval for a specific dataset.*

detail. At bottom, this white-colored fragment had only a few small pits, which were successfully detected in both colors and normals.

Brush Strokes: One of our frescoes — Kerkrade — exhibited strong variation in the types of brush strokes that were present. We observed a number of phenomena, including small strokes left by the artist's brush (Figure 9, left) and broader, deeper strokes left in the underlying plaster when smoothing it out (center and right). We also observed situations in which brushstrokes at different orientations were simultaneously visible. Though our current method does not detect these, returning only the strongest brushstroke direction present in a fragment or patch, we believe that it would be possible to extend the descriptor to handle these cases. In cases in which brushstrokes are present, we informally observe the orientation of these strokes to be one of the strongest matching cues available.

6 Summary of Features

In this section, we analyze the distributions of feature values across the three databases of frescoes introduced in Section 3.3. We observe that the importance of different features for discriminating matches from non-matches is different for each of the three databases, motivating the classifier-based evaluation methodology presented in Section 7.

Table 1 shows statistics for the feature correspondences computed on each fresco. In this analysis, we use 2,274 pairwise feature correspondences from each dataset, including both ground-truth matches and randomly sampled non-matches. All patches have a 10mm diameter and each patch center is offset 8mm from the boundary contour (except for curvature descriptors, which are sampled along the boundary contour). The values shown in the table are the minimum, mean, maximum, and standard deviation of "correspondence" values for each feature. For most features, this is just the absolute value of the difference between the feature values computed on both fragments: these are expected to be near zero for correct matches. For a few features, however, the correspondence value is the absolute value of the sum for the two fragments. This is necessary for features such as curvature, which are expected to have opposite signs on corresponding fragments.

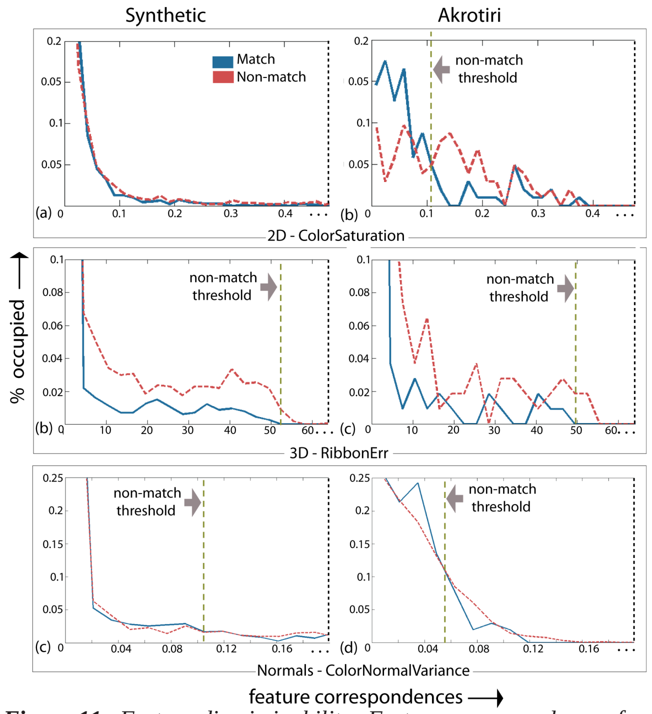

Figure 11: *Feature discriminability. Feature correspondences for equal numbers of matches and non-matches for the Synthetic (2,274 samples) and Akrotiri datasets (206 samples). (a) - (b) ColorSaturation is more descriminating on the Akrotiri dataset. The number of matches with correspondences near zero are greater than the number of non-matches. Conversely, there is little distinction between the two curves for the Synthetic dataset. (c) - (d) RibbonErr correspondences greater than 50 belong to non-matches for both datasets. (d) - (e) ColorNormalVariance values greater than 0.05 on the Akrotiri dataset are more likely to be non-matches. Values greater than 0.10 are more likely to be non-matches, however, the distinction is less obvious.*

As shown in Figure 10, the distribution feature correspondences is different for each fresco. For example, almost all ColorSaturation correspondences (a) for the Synthetic fresco are near zero while only 8% of the samples are clustered near zero for Akrotiri. This is not surprising, as most Synthetic fragments are white. The distribution of the RibbonError feature (b) is similar for each dataset while the range of the ColorNormalVariance feature (c) is most similar for Akrotiri and Kerkrade but different for the Synthetic fresco.

We also observe that different features are more *discriminating* on different datasets. Figure 11 compares histogram plots of selected features for ground-truth matches (blue) vs. non-matches (red) for the Synthetic and Akrotiri datasets. There are equal numbers of matches in each example, with 2,274 total samples for the Synthetic dataset and 206 total samples for the Akrotiri dataset. The match and non-match curves for ColorSaturation are almost identical on the Synthetic dataset, suggesting there is little information available to distinguish between a match from a non-match using this feature. Conversely, the number of matches whose correspondence lies near zero is significantly greater than non-matches for Akrotiri. We expect this behavior, since this dataset contains considerable pigmentation and hence color is a good matching cue. In addition, we observe that there is a clear threshold above which correspondence values are more likely to apply to non-matches than matches. In many cases, 100% of correspondences above the threshold are non-matches.

Our analysis suggests that color and 3D features work best for the Akrotiri and Kerkrade databases and that, in general, normal-based features will work well on all databases. Some normal features are significantly stronger than others, however, depending on the surface features of the database. For example, we observed that dominant-orientation features are especially important for the Kerkrade dataset, with color-based features performing no better than chance. Curvature features are the least reliable.

We also compared classification results for each feature individually and in combination with other features. Individual features do not perform as well as combinations of features. We found that Patch level features are more robust than fragment level features. One exception to this rule is erosion, which was more effective when computed over the entire fragment. We anticipate that this feature is good at separating smooth fragments from eroded ones at the fragment level, but is too noisy at the patch level. We further discuss our classification approach, results from combining features, and comparisons of patch level vs. fragment level features in Section 7.

7 Classification Results

Classifiers and Evaluation Methodology: In order to evaluate the performance of our features for matching, we work with manually labeled sets of matching fragments, and randomly sampled non-matches. Except where stated otherwise, we use 10-fold cross-validation, with manually separated training and test sets. In each of these sets, we ensure that each *pair* of fragments, whether matching or not, is placed entirely within either the training or test set. This is done because a single matching pair of fragments may result in multiple matching *patches*, so we wish to ensure that the classification algorithms do not gain advantage from training and testing on patches from the same pair of fragments. Using various subsets of the color, normal, and geometric features described above, we train classifiers to distinguish between matches and non-matches.

We have explored four different classification algorithms, as implemented by the "Weka" open-source data mining package.[1] The algorithms are:

- **J48 decision trees**, which implement the C4.5 algorithm of Quinlan [1993]. This algorithm hierarchically subdivides the training set, at each node partitioning using the feature that results in the greatest difference in entropy among the subsets.
- **Random forests**, which train decision trees on multiple subsets of features, combining the results into a single probabilistic classifier.
- **Support vector machines**, which compute a high-dimensional separating plane between the two categories.
- **Logistic regression**, which fits the data with a generalized linear model consisting of the logistic function $p = 1/(1 + e^{-z})$ applied to a linear combination of the input feature values.

To determine which classifier would generalize well to all of our datasets, we evaluated robustness to overfitting, computational efficiency, and the availability of a real-valued probability instead of merely a binary yes/no classification. The latter is important for our application, since it allows us to create a rank-ordered list of hypothesized matches, which is then presented to a human for verification. Since it will typically be impractical for a person to check all predicted matches, the availability of a ranking is crucial.

Table 2 shows the performance of each classifier, using as input all of our per-fragment, per-patch and per-match features for a set of 2,274 ground truth samples (including equal numbers of matches and non-matches). The J48 decision trees had good matching performance, but we found them to be most prone to over-fitting the data. In addition, they provide only a binary decision, not a probability. Random forests were less prone to over-fitting and provided probabilities, but the probability values (resulting from combining

[1] `http://www.cs.waikato.ac.nz/ml/weka/`

Table 2: *Comparison of machine learning algorithmns: We evaluate the performance of each algorithm using manual cross validation on 2,274 groundtruth samples containing an equal number of matches and non-matches. In this example, we combine all per-patch, per-fragment and per-match features.*

Classifier	Ground-truth Matches		Ground-truth Nonmatches	
	Correct (TP)	Incorrect (FN)	Correct (TN)	Incorrect (FP)
J48	66%	34%	84%	16%
RandomForest	79%	21%	71%	29%
SVM	67%	33%	78%	22%
LogisticRegression	84%	16%	49%	51%

Table 3: *Classification performance on synthetic fresco, on a test set of 220 samples using logistic regression. There are 110 matches (one pairwise match per matching fragment pair) and 110 non-matches (also unique and randomly sampled. We apply the best model from our manual cross validation training session.*

Features	Ground-truth Matches		Ground-truth Nonmatches	
	Correct (TP)	Incorrect (FN)	Correct (TN)	Incorrect (FP)
AllColor	79%	21%	31%	69%
AllCurvature	54%	46%	62%	38%
AllNormal	80%	20%	48%	52%
Thickness	80%	20%	37%	63%
RibbonError	86%	14%	68%	32%
RibbonVolIntersect	94%	6%	35%	65%
AllCombined	90%	10%	78%	22%

multiple trees) were still strongly clustered. Support Vector Machines worked well and gave meaningful probabilities, but exhibited time, space and algorithmic complexities that make the method impractical for large datasets. In addition, they were sensitive to the choice of parameters, which were frequently difficult to set because of the different range of meaningful values for each feature. Logistic regression produces robust results and yields meaningful rankings, and this is the method used for the remaining results in this section.

Performance on Synthetic Fresco: We evaluate the performance of classifiers trained on different categories of features, on a set of ground-truth matches and non-matches from the Synthetic fresco. Because this is the dataset with the greatest number of known matches, we expect to learn the most meaningful results about feature performance by observing classification results on this fresco.

This test was conducted on 110 known matches and 110 known non-matches from this fresco. For maximum fairness, this set only includes one pair of matching or non-matching patches for each pair of fragments. Table 3 shows the number of correctly and incorrectly classified instances among the matches (true positives and false negatives, respectively) and among the non-matches (true negatives and false positives). The rows of the table represent classifiers trained on:

- All the “color” features listed in Table 1. These and the curvatures are the features considered by many traditional 2D-only matching algorithms.
- All of the “curvature” features listed in Table 1, evaluated at all different scales.
- All the “normal” features listed in Table 1. These are the new per-fragment and per-patch features we propose.
- The fragment thickness.
- The RibbonError and RibbonVolIntersection features, which represent two outputs computed by the ribbon-matching algorithm of [Brown et al. 2008] on the 3D models.

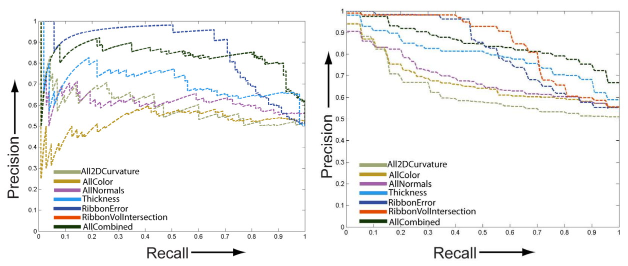

Figure 12: Left: *Precision-recall for a classification experiment on the Synthetic fresco including the same classes of features as in Table 3.* ***Right:*** *A re-weighting experiment, in which 203 locations of potential matches, predicted by the ribbon matcher, are ranked according to a trained classifier. Averages over 10-fold cross validation are presented.*

- A combination of all features listed in Table 1.

The table demonstrates that each type of feature has its strengths and weaknesses when it comes to both finding matches and rejecting non-matches. Curvature and color features, considering both true-positive and true-negative numbers, are barely performing above chance. The normal-based features perform better on non-matches, confirming our hypothesis that such features, while remaining easy to acquire, easy to compute, and easy to incorporate into a fast pruning stage based on per-fragment and per-patch information, incorporate substantially more information about matching fragments than does color.

Turning to 3D information, thickness performs moderately well, but not as well as normal-based features. The two ribbon-matcher features have substantially better performance, but note that these also have substantially higher computational cost: they are per-*match* features, not per-fragment or per-patch.

Finally, the combination of all features has the best overall performance, demonstrating that the classifier is successfully taking advantage of the best performance of each.

Precision-recall: Because the logistic regression classifier outputs not only a prediction but also a probability, we are able to evaluate our results on a “ranking” task that provides more insight than is available with simple confusion matrices. We present our results using precision-recall curves, in which points represent predicted matches in probability-ranked order, with the x coordinate (recall) representing the fraction of total matches found so far while the y axis (precision) indicates the fraction of all predictions so far that have corresponded to true matches. Higher curves therefore represent better results.

Figure 12, left, shows results on an experiment similar to the one in Table 3, using the same sets of features. At right, we show a different way of using classifiers, namely a re-weighting experiment in which 203 locations of matches predicted by the existing “ribbon matcher” are ranked according to a classifier trained on different subsets of features. Average results for 10-fold cross validation are presented. In both cases, the results show that combining features leads to better precision than most individual features, especially at higher recall.

Per-Fragment vs. Per-Patch Features: To further examine the potential performance of the pipeline in Figure 2, we investigated the performance of per-fragment and per-patch features. Figure 13 shows precision-recall curves in which dashed lines include only per-fragment features, while solid lines include both per-fragment and per-patch features. We see that for color-based features, both sets perform relatively poorly, but for normal and thickness features, there is indeed more information available from per-patch features. In these cases, however, per-fragment features alone are still performing some degree of classification, suggesting that the

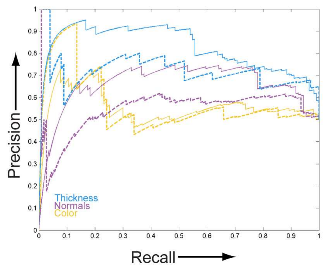

Figure 13: *Comparison of per-fragment features (dashed lines) with a combination of per-fragment and per-patch features (solid lines). In all cases, the addition of per-patch features improves classification performance, at the expense of additional computation time.*

pipeline of Figure 2 may, with appropriate thresholds, provide both efficiency and accuracy.

Generalization across Datasets: To examine the extent to which classifier performance generalizes across different datasets, we trained a classifier on the Synthetic fresco, using the three features of Figure 11. We then compared the performance of the classifier on the original Synthetic fresco, as well as the Akrotiri fresco (Figure 14). We observe that the performance is reasonable, with the classifier sometimes performing slightly better and sometimes slightly worse. In general, we expect better performance with custom-trained classifiers for each dataset, but these preliminary results suggest that adapting classifiers from one dataset to another may still lead to reasonable results. In particular, the results are usually sufficient to perform a "bootstrapping": finding enough ground-truth matches to enable a new, custom classifier to be trained. In the future, we expect to use the results of the analysis of variance of each feature across each dataset to be able to adapt classifiers even more directly, by re-weighting the contribution of each feature to the classification without a full re-training step.

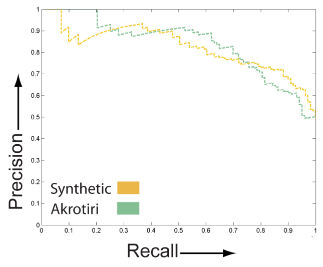

Figure 14: *Evaluating classifiers across multiple datasets. In this example a model was trained on the Synthetic dataset using a combination of features: ColorSaturation, RibbonError and ColorNormalVariance (the same features as in Figure 11). We show results of the trained classifier on the original (Synthetic) dataset, as well as the results of the same classifier on the Akrotiri dataset.*

8 Discussion and Conclusion

Manual fragment assembly rarely occurs based on a single cue. Even where there is an obvious and essential cue, such as the edge geometry of fresco fragments, a good assembler relies on judiciously combining every available cue. Working in the context of fresco fragments, we have introduced several new feature descriptors based on normals and color that encapsulate cues such as fragment erosion and surface impressions. We have also shown how to use machine-learning techniques to combine descriptors within a multi-cue framework, including using our new per-fragment and per-patch descriptors to complement existing per-match features.

There is some danger to relying on surface-based features. While matching fragments often erode in similar ways, that is not always the case. When only one fragment has eroded or discolored, we may not identify the match. This is of course inherent in relying on any kind of cue: edge- and contour-based matchers will fail if too much of the side (as opposed to the front) has eroded or broken off, whereas matching on surface properties might still succeed. We believe the best bet is to support many different cues so we can identify as many matches as possible.

Although we have presented our work in the context of fresco fragments, we believe the ideas translate to many other matching problems such as distinguishing the brush strokes of different artists on oil paintings, classifying chisel marks on sculptures, or matching textured objects to their impressions for forensic identification. Different types of objects will naturally require different features, but we expect the normal-based descriptors we have presented will be valuable for many types of material with an exterior surface containing relief or erosion.

For this reason, we anticipate that classifiers trained on one dataset will still perform well on another, but that improved performance could be achieved with an online learning approach: a pre-trained classifier is used to generate an initial classification, with re-training occurring as instances are confirmed to be either correctly or incorrectly classified. We leave this approach as future work, and present classification performance results for the synthetic dataset.

References

BROWN, B. J., TOLER-FRANKLIN, C., NEHAB, D., BURNS, M., DOBKIN, D., VLACHOPOULOS, A., DOUMAS, C., RUSINKIEWICZ, S., AND WEYRICH, T. 2008. A System for High-Volume Acquisition and Matching of Fresco Fragments: Reassembling Theran Wall Paintings. *ACM Transactions on Graphics (Proc. SIGGRAPH), Vol. 27, No. 3* (Aug.).

FORNASIER, M., AND TONIOLO, D. 2005. Fast, robust and efficient 2D pattern recognition for re-assembling fragmented images. *Pattern Recognition, Vol. 38, No. 11* (Nov).

HUANG, Q.-X., FLÖRY, S., GELFAND, N., HOFER, M., AND POTTMANN, H. 2006. Reassembling Fractured Objects by Geometric Matching. *ACM Transactions on Graphics (Proc. SIGGRAPH), Vol. 25, No. 3.*

KARASIK, A., AND SMILANSKY, U. 2007. 3D scanning technology as a standard archaeological tool for pottery analysis: practice and theory. *Journal of Archaeological Science*.

KOLLER, D., TRIMBLE, J., NAJBJERG, T., GELFAND, N., AND LEVOY, M. 2006. Fragments of the City: Stanford's Digital Forma Urbis Romae Project. In *Proceedings of the Third Williams Symposium on Classical Architecture, Journal of Roman Archaeology*, vol. Suppl. 61, 237–252.

KONG, W., AND KIMIA, B. 2001. On Solving 2D and 3D Puzzles under Curve Matching. *CVPR, Vol. 2, No. 10* (aug), 583.

LEITÃO, H. C. G., AND STOLFI, J. 2002. A Multiscale Method for the Reassembly of Two-Dimensional Fragmented Objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, No. 9* (Sept.), 1239–1251.

MANAY, S., YEZZI, A. J., HONG, B. W., AND SOATTO, S. 2004. Integral Invariant Signatures. In *Proc. ECCV*, 87–99.

PAPAODYSSEUS, C., PANAGOPOULOS, T., EXARHOS, M., TRIANTAFILLOU, C., FRAGOULIS, D., AND DOUMAS, C. 2002. Contour-Shape Based Reconstruction of Fragmented, 1600 BC Wallpaintings. *IEEE Trans. on Signal Processing, Vol. 50, No. 6*.

PINTUS, R., MALZBENDER, T., WANG, O., BERGMAN, R., NACHLIELI, H., AND RUCKENSTEIN, G. 2009. Photo Repair and 3D Structure from Flatbed Scanners. In *Proc. VISAPP*.

QUINLAN, J. R. 1993. *C4.5: Programs for Machine Learning.* Morgan Kaufmann.

SAĞIROĞLU, M. Ş., AND ERÇIL, A. 2006. A Texture Based Matching Approach for Automated Assembly of Puzzles. In *ICPR '06: Proceedings of the 18th International Conference on Pattern Recognition*, IEEE Computer Society, Washington, DC, USA, 1036–1041.

SERRA, J. 1983. *Image Analysis and Mathermatical Morphology*. Academic Press.

WILLIS, A. 2004. Stochastic 3D Geometric Models for Classification, Deformation, and Estimation. *Ph.D. Thesis* (may).