Learning How to Match Fresco Fragments

T. Funkhouser,[1] H. Shin,[1] C. Toler-Franklin,[1] A. García Castañeda,[2] B. Brown,[3] D. Dobkin,[1] S. Rusinkiewicz,[1] and T. Weyrich[2]

[1]Princeton University, Princeton, NJ, USA
[2]University College London, UK
[3]Katholieke Universiteit Leuven, Belgium

Abstract

One of the main problems faced during reconstruction of fractured archaeological artifacts is sorting through a large number of candidate matches between fragments to find the relatively few that are correct. Previous computer methods for this task provided scoring functions based on a variety of properties of potential matches, including color and geometric compatibility across fracture surfaces. However, they usually consider only one or at most a few properties at once, and therefore provide match predictions with very low precision. In this paper, we investigate a machine learning approach that computes the probability that a match is correct based on the combination of many features. We explore this machine learning approach for ranking matches in three different sets of fresco fragments, finding that classifiers based on many match properties can be significantly more effective at ranking proposed matches than scores based on any single property alone. Our results suggest that it is possible to train a classifier on match properties in one data set and then use it to rank predicted matches in another data set effectively. We believe that this approach could be helpful in a variety of cultural heritage reconstruction systems.

1. Introduction

Reconstruction of fractured ancient artifacts, such as frescoes, is important because it helps archaeologists make inferences about past civilizations. Unfortunately, at archaeological sites with large numbers of fragments, reconstruction is difficult, since a large space of potential "matches" between fragments must be searched and evaluated.

To assist in this process, computer systems have been built that scan fragments with cameras and/or laser scanners and then use computer algorithms to assist reconstruction. For example, the Stitch Project [CWAB01] has provided semi-automatic algorithms for reconstruction of pottery, murals, and sculptures, and The Forma Urbis Romae Project has provided methods to search for matches in an ancient marble map of Rome [KL06]. These projects have provided automatic methods to search for potential matches between fragments sharing a fracture boundary, but usually predict matches with low precision (many false matches are proposed for every correct one found). Therefore, a great burden is placed on a user to sift through a large set of predictions to identify the correct ones.

The main objective of this paper is to provide a method to rank predicted matches between pairs of fragments with high precision and to provide a measure of confidence for each one. That is, given two scanned fragments and a proposed aligning transformation (a candidate "match"), we aim to provide a function that estimates the probability that the match is correct. This objective is challenging because excavated artifacts often have erosion, color fading, and other deterioration that make discrimination of correct matches difficult.

Our approach is to utilize a set of examples to train a classifier to predict the probability of a match based on a multitude of properties computable from scans of fragment surface colors and geometry. This approach follows the observations of two recent papers, one by Shin et al. that analyzed many properties of matches in assembled frescoes and suggested that they might be effective for classifying proposed matches [SDF*10], and one by Toler-Franklin et al. that tested the value of match properties based on surface color, normal maps, and edge geometry using a framework based on machine learning [TFBW*10]. Our contribution is to combine these two ideas into a system that predicts the probability of predicted matches with high precision.

We explore this machine learning approach for finding matches in three different data sets, one from a Late-bronze-age settlement in Greece (Akrotiri), a second from a Roman City in Belgium (Tongeren), and a third constructed

© The Eurographics Association 2011.

specifically for this project (Synthetic). Our findings indicate that classifiers based on many match properties are significantly more effective at ranking proposed matches than are scores based on any single property alone. They also suggest that it is possible to train a classifier on match properties in one data set and then use it to rank predicted matches in another data set effectively. For example, the classifier learned from examples in the Akrotiri data set can be used to rank matches in the Tongeren data set with 96% precision at 50% recall, whereas the previous state-of-the-art method provides only 2% precision at the same recall for the same test case [BTFN*08].

Overall, our main research contribution is investigating the idea of using machine learning to combine a large variety of match properties into a probability estimation. Secondary contributions include the descriptions of the match properties considered, many of which are novel, and the results of experiments suggesting that cross-training is an effective strategy for ranking proposed matches. We believe that that there are many applications of this approach in reconstruction of cultural heritage artifacts.

2. Related work

There has been a long history of work on computer-aided reconstruction of fractured objects in archeology [WC08]. Most previous work has focused on finding pairwise matches between adjacent fragments by aligning patterns in their surface colors [FT05], polygonal boundaries [PPE02], and/or fractured edges [HFG*06]. These methods have been successful in cases where the fragments have highly distinctive features [HFG*06], the reconstructed objects are surfaces of revolution [CM02], and/or when domain-specific features can be used to identify potential matches [KL06]. However, they have not been able to find matches with high precision amongst a large set of flat, mostly-monochromatic, partially-eroded fragments, such as those commonly found in frescoes.

Several researchers have observed that it is desirable to combine more than one computed property into a scoring function used to rank potential matches. However, most previous methods have utilized simple combination strategies based on hand-tuned weights and/or thresholds. For example, Brown et al. scored potential matches with a "ribbon-matcher error" that is the sum of two terms, one that measures the root mean squared distance (RMSD) between corresponding points in a fixed size "window" (50mm) of the scanned fracture surfaces, and another that is proportional to the difference between fragment thicknesses, after thresholds on these two terms and the amount of surface interpenetration are applied to rule out bad matches [BTFN*08]. McBride et al. scored potential matches based on a function with three terms: $\lambda_1 * C_{distance} + \lambda_2 * \sqrt{C_{length}} + \lambda_3 * \sqrt{C_{diagnostic}}$, where $C_{distance}$ measures the average length between corresponding points on contours of the two fragments, C_{length} measures the arc length of the contact region on the contours, $C_{diagnostic}$ measures how complex or jagged the contact region is, and λ_i are weights used to blend these three terms (square roots were added by the authors to the second and third terms after empirical results showed that they can "obscure the distance measure") [MK03]. While these methods can be hand-tuned for a small number of match properties on specific data sets, it would be impractical to use them to combine hundreds of properties. Moreover, it would be tedious to re-tune parameters by hand for different data sets, a task that probably could only be done by experts.

Recently, Toler-Franklin et al. explored the idea of using machine learning to incorporate multiple match properties into a scoring function [TFBW*10]. They computed a variety of properties of "patches" (small regions on fragment boundaries) and then trained a classifier to score patch pairs based on the differences between the computed properties. The focus of their study was on evaluating the discriminability of their new patch properties based on normal maps – they performed classification experiments on small sets of matches and non-matches and concluded that combining multiple properties with a classifier helps improve precision at high recall values. We aim to extend the ideas in that paper to consider many new, more discriminating match properties (and thus achieve higher precision at all recall levels) and to integrate machine learning into a full prediction system that finds novel matches in excavated frescoes.

3. Approach

In this paper, we investigate the idea of using a classifier trained on examples from the same or a different fresco to predict the probability that a proposed match between two fresco fragments is correct.

The main motivation for this approach is to provide automatic methods for combining large numbers of properties into a match scoring function. There are many properties of matches that can be computed easily, which may be useful for recognizing correct ones. However, it is difficult to know in advance which properties will be most discriminating for a given data set. So, instead of deciding in advance which properties we expect to be useful, we take the approach of computing as many properties as possible and then allowing a machine learning algorithm to determine which ones to use and how to combine them based on examples provided in a training set. With this strategy, there is little burden on the user to select/weight the most important properties, and the system can adapt to different data sets automatically based on properties of the examples.

In contrast to previous methods for ranking proposed matches, this approach utilizes information in previously discovered matches, combines multiple properties via optimization, requires no hand-tuning of parameters, and estimates the probability that a match is correct (rather than

© The Eurographics Association 2011.

providing only a score for ranking). Thus, we believe that it can be used by untrained operators to discover matches in new, large data sets.

There are several interesting questions to consider during an investigation of this approach – for example: 1) how are fragments scanned?, 2) how are candidate matches proposed?, 3) which match properties are measured?, 4) how is the training set composed?, and 5) how well do the properties of matches in one data set predict matches in another? These questions are addressed in the following five sections.

4. Data Sets

Our investigation is based on data collected with the scanning systems described in Brown et al. [BTFN*08] and [BLD*10]. They developed systems that quickly acquire color images for the front and back surfaces and laser scans of the fracture surface for a large number of fragments (Figure 1). They also provide processing tools to: 1) detect the front surface plane (grey textured surfaces), 2) to extract a regularly sampled "ribbon" representation of the 3D fracture surface (red and blue surfaces), and 3) to extract a 2D contour representing the shape of the fragment by intersecting the ribbon with a slicing plane 4mm from the front surface plane (orange curve).

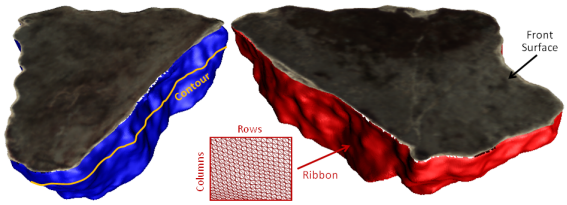


Figure 1: *Front surfaces and ribbons acquired for two matching fragments in the Akrotiri data set.*

We consider three data sets collected with these tools:

- **Akrotiri:** an excavation of a Late Bronze Age Aegean city in Greece destroyed by a volcanic eruption in approximately 1650 B.C. [Dou92]. We have scanned 1408 fresco fragments, amongst which there are 111 known matches.
- **Tongeren Vrijthof:** an excavation of a Roman building in Belgium destroyed by fire over the first three centuries A.D. [Lak10]. We have scanned 1306 fresco fragments, amongst which there are 203 known matches [BLD*10].
- **Synthetic:** a plaster fresco recently created and fractured specifically for research on fresco reconstruction [BTFN*08]. We have scanned 130 fragments, amongst which all 256 correct matches are known.

These data sets provide interesting test cases for our study because they come from disparate sources varying widely in space, time, culture, and scanning technologies. The Synthetic Fresco was made "in the style of" the Akotiri wall paintings, but more recently by more than 3,500 years. The Tongeren Vrijthof frescoes come from a Roman building in Belgium that was built and destroyed approximately 2,000 years ago, and show different materials and construction methods than the others. The Akrotiri wall paintings were destroyed by earthquakes preceding a volcanic eruption, Tongeren's were destroyed in phases over time by fire caused by warfare, and the Synthetic Fresco was broken intentionally. Fragments from Akrotiri have been stabilized by conservators, while fragments from the others hardly have. Finally, 3D geometry for the Akrotiri and Synthetic data sets were acquired with a laser scanner, while the Tongeren data set was scanned with structured light. Due to these differences, it is not obvious that properties of fragment matches found in one data set will provide good predictors for properties of matches in the others, and so we believe these data sets provide challenging cases for cross-training.

5. Generating Candidate Matches

Once fragments have been scanned, the first step in any reconstruction system is to search for a set of pairwise matches between fragments, which will be scored to produce a ranked list of candidate matches. Several methods are possible for this step, including ones that align extracted facets of fragment surfaces [PK03, HFG*06], ones that align points of high curvature [MP06, PSCC07], ones that align detected corner points [MK03], and ones that search exhaustively [BTFN*08, KK01].

In our study, we leverage the method of Brown et al., which utilizes the regularly sampled "ribbon" representation and an incremental alignment algorithm to quickly compute the RMSD for all possible pairs of boundary patches sampled at 0.25mm increments [BTFN*08]. The candidate matches produced by this algorithm are culled based on thresholds limiting the maximum RMSD, the maximum difference between fragment thicknesses, and the maximum volume intersection of aligned fragments. The pairwise matches passing these thresholds are then scored with the "ribbonmatcher error" described in [BTFN*08] and added to the candidate match set.

This match generation method represents the state-of-the-art for the data sets considered in this study. It exhaustively searches for alignments, culls out the obvious non-matches, and provides an initial estimate of the match quality. However, the precision of the ribbonmatcher error is not high enough to recover all correct matches within a candidate set that can be evaluated by a person in a practical amount of time. Our goal is to re-rank these candiate matches so that the correct matches can be found quickly.

6. Computing Match Properties

Our next step is to compute properties of matches that are likely to be useful for discriminating correct candidate

© The Eurographics Association 2011.

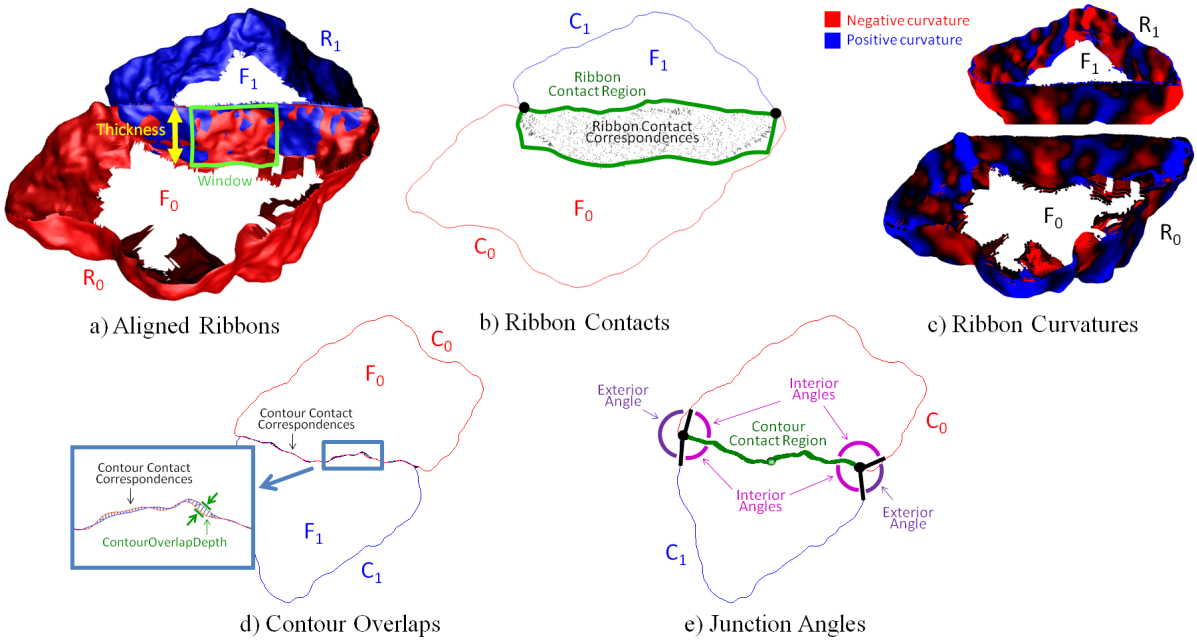


Figure 2: *Measurements utilized by match properties.*

matches. Of course, several properties have been investigated previously in the literature, including similarities of front surface textures, boundary contours, and fracture surface patterns, as described in Section 2.

In addition to these traditional properties, we consider novel ones motivated by the recent study of Shin et al. [SDF*10]. They analyzed the properties of matches between fragments in an already-reconstructed fresco and concluded that the fracture pattern resulted from a hierarchical breaking process. They observed that contact regions (edges) between adjacent fragments tend to be nearly straight and cover approximately 1/5th of the fragment perimeter, that fragments tend to match with other fragments of nearly equal size, and that "junctions" where multiple fragments join usually have three edges coming together in a T-junction (also noted in [KK01, MK03]). These observations imply very specific features of matches that we hope to capture in computable properties.

In all, we choose a set of 64 properties to compute for each match M. The following provides a brief list, grouped by the type of data from which they are derived. Details of how these properties are defined and computed appear in the Appendix.

- **Fragment Properties** measure the compatibility of the two fragments based on the differences in their thicknesses and front surface colors and the ratios of their front surface areas.
- **Contour and Ribbon Contact Properities** measure how well 2D contours (or 3D ribbons) representing the fragments align with one another based on "contact correspondences," points from the surfaces of two different fragments that are aligned by M. The compatibility of the surfaces in contact is measured with the RMSD between corresponding contact points, the differences between curvatures at corresponding contact points, the size of the region containing contact points, the ratios of that size with respect to the fragment sizes, the density of points in that region that are indeed in contact, and how closely the contact points fit a straight line/plane.
- **Contour and Ribbon Window Properties** measure the RMSD of corresponding points on the 2D contours (or 3D ribbons) representing the fragments within fixed width "windows" centered in the middle of the contact region (e.g., green square in Figure 2a).
- **Contour Overlap Properties** measure how much the contours inter-penetrate one another based on the average depth, maximum depth, and total area of the contour overlaps (see inset in Figure 2d).
- **Contour Convexity Properties** measure how convex the union of two fragements is in comparison to the convexity of the individual fragments.
- **Junction Angle Properties** measure interior and exterior angles at boundaries of contact regions (Figure 2e).
- **Ribbonmatcher Properties** are provided directly by the ribbonmatcher. They comprise the RMSD of corresponding points within a 50mm window on the ribbon, the estimated volumetric intersection of the two fragments, and a composite error used to rank matches in [BTFN*08], as described in Section 2.

© The Eurographics Association 2011.

7. Learning a Match Classifier

Our final steps are to learn a match classifier based on examples in a training set and then to use that classifier to estimate the probability of new candidate matches being correct. This is a classical machine learning problem. However, there are several interesting issues to consider when building a practical system for finding matches with this approach: 1) choosing a training set, 2) choosing a classification model, and 3) training the classifier.

In this study, we consider scenarios where the classifier is trained on matches known in one fresco (the training set) and then used to find matches in a different fresco (the test set). For both the training and test sets, we run the ribbonmatcher to generate candidate matches and compute the properties listed in the previous section for all of them, culling matches that have a property value outside a predetermined allowable range (where ranges are chosen *very* conservatively so as to not discard any correct matches). We train a classifier that associates the 64 properties defined for each match (a feature vector) with a label indicating whether the match is correct or not. Then, for each candidate match in the test set, we apply the classifier to estimate a probability that it is correct.

Our method for building the classifier from the training set is based on M5P regression trees as implemented in Weka [WF05]). This model builds a decision tree that splits feature space into distinct regions and then fits a linear regression model for each region independently. It was chosen for our study because it automatically performs feature selection for both the decision tree and the linear regression models, it fits non-linear relationships between input and output variables (piecewise linear), and it provides an explanation for how the model operates (the decision tree). Of course, nothing in our study is dependent on this particular choice, and we believe that several other alternatives could have been used just as effectively.

A practical concern in building the classifier is to provide an appropriate number of training examples. In our case, the training sets typically have very few examples of correct matches amongst vast numbers of incorrect candidates. Thus, providing all the examples as training data would guide the classifier towards predicting "not correct" (since it optimizes overall classification rate). Alternatively, random subsampling could miss the most useful data (since correct matches are rare). To alleviate these concerns, we utilize at most 50,000 candidate matches when training the classifier, selecting all of the correct matches and filling out the remainder with randomly selected incorrect matches.

Figure 3 shows a decision tree learned by our system when trained on examples from the Synthetic Fresco. Examining the tree, it is interesting to note that nodes of the tree take into account different types of properties, which suggests that several properties can be combined to make better predictions than any property alone.

```
RibbonContactRMSD <= 0.429 :
  RibbonContactRMSD <= 0.375 :
    RibbonContactPlanarity <= 0.517 :
      ContourContactRMSD <= 0.286 :
        ContourContact4mmHorizCurvL2 <= 0.009 : LM1 (29)
        ContourContact4mmHorizCurvL2 > 0.009 : LM2 (112)
      ContourContactRMSD > 0.286 : LM3 (560)
    RibbonContactPlanarity > 0.517 :
      RibbonContactArea <= 446.36 :
        RibbonContactRMSD <= 0.36 :
          RibbonJunctionMinInteriorAngle <= 2.232 :
            ContourContactRMSD <= 0.217 : LM4 (17)
            ContourContactRMSD > 0.217 :
              ContourContactMinLenAreaFract <= 0.309 : LM5 (20)
              ContourContactMinLenAreaFract > 0.309 :
                RibbonContactRMSD <= 0.331 : LM6 (12)
                RibbonContactRMSD > 0.331 : LM7 (20)
          RibbonJunctionMinInteriorAngle > 2.232 : LM8 (29)
        RibbonContactRMSD > 0.36 : LM9 (91)
      RibbonContactArea > 446.36 : LM10 (53)
  RibbonContactRMSD > 0.375 :
    RibbonContactArea <= 235.969 : LM11 (3015)
    RibbonContactArea > 235.969 :
      RibbonContact1mmMeanCurvL2 <= 0.121 : LM12 (603)
      RibbonContact1mmMeanCurvL2 > 0.121 : LM13 (151)
RibbonContactRMSD > 0.429 : LM14 (7416)
```

Figure 3: *Decision tree learned by training on matches in the Synthetic Fresco. Each line represents a branch conditioned on a property value. Lines with LMx (y) are leaf nodes, where LMx represents a linear regression model, and y indicates the number of matches classified with that regression model in the training set.*

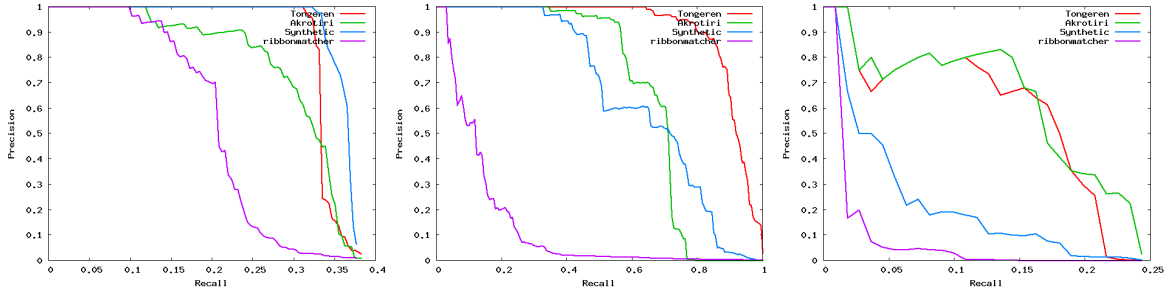
8. Experimental Results

We have executed a number of experiments with the proposed match classification approach in an effort to characterize how effective it is at ranking matches. In these experiments, we consider only the scenario in which the match classifier is trained on examples from one fresco (Akrotiri, Tongeren, or Synthetic) and then tested on candidate matches from another. We chose this scenario because it is the most challenging case for our approach – frankly, it is not obvious that our approach should work at all in this scenario, since matches in the training set may have different properties than the ones in the test set due to differences in colors, materials, erosion, scanning, etc.

For the sake of clarity, we limit these experiments to consider only matches proposed by the ribbonmatcher of Brown et al. [BTFN*08]. Those matches provide a good set of candidates associated with a state-of-the-art scoring function, the "ribbonmatcher error" (as described in Section 2). We evaluate the quality of our scoring function in comparison to this one using precision-recall analysis – i.e., we rank matches according to the scoring function and then plot the precision (true positives / (true positives + false positives) versus the recall (true positives / (true positives + false negatives) as candidates are considered in rank order.

The results are shown in Figure 4. Each plot represents a different test set, and each curve within a plot represents a different scoring function (in our case, a different training set). For example, the plot on the left compares the precision vs. recall for candidate matches in the Synthetic Fresco ranked by ribbonmatcher error (purple)

© The Eurographics Association 2011.



a) Synthetic b) Tongeren c) Akrotiri

Figure 4: *Comparison of scoring functions. The three plots show precision-recall comparisons for matches* predicted *in different frescoes. The curves in each plot compare results of ranking the predicted matches with four different scoring functions: the ribbonmatcher error (purple) and three classifiers* trained *on examples from different frescoes.*

vs. our classifier learned with three different training sets (red=Tongeren, green=Akrotiri, and blue=Synthetic). Likewise, the blue curve in each plot represents the results of ranking matches based on the decision tree learned from the Synthetic Fresco shown in Figure 3.

From these results, we observe that our classifiers significantly outperform the ribbonmatcher error. It is not surprising that they perform at least as well as the ribbonmatcher, since the ribbonmatcher error is one of the properties available to the classifiers. However, it is interesting that the classifier achieve significantly higher precision at almost every recall. This result indicates that multiple properties are providing valuable information for ranking matches.

It is also not surprising that the classifiers perform well when the training and test sets come from the same fresco (the blue curve in (a), the red curve in (b), and the green curve in (c)) – these results merely confirm that the classifier is able to characterize its training set. However, it is surprising that the precision achieved when training and testing on different frescoes is not dramatically different than when training and testing on the same fresco. For example, the blue curve in Figure 4a (Synthetic predictions trained on Synthetic examples) is not significantly higher than the red curve in the same plot (Synthetic predictions trained on Tongeren examples). This result is quite remarkable, since the frescoes were made by different people, in different styles, in different countries, in different millennia. It suggests that different frescoes have statistical properties similar enough to one another that cross-training is a valuable strategy for match discovery.

Table 1 shows compute times required for the processing by our algorithms (in seconds on 2.2 GHz AMD Opteron processors). Each row represents a different data set. The second column lists the number of fragments in the data set. The next three columns list the number of original matches proposed by the ribbonmatcher, the number of candidate matches satisfying our conservative thresholds, and the number of those that are correct, respectively. The rightmost three columns list processing times. As you can see, the vast amount of time in this experiment is spent processing candidate matches (computing properties) – it is approximately proportional to the number of candidates ($\sim$1 second per candidate match). Once properties are computed, training a classifier is fast (a few seconds), and applying a classifier to compute the score for a single candidate match is very fast ($\sim$1 millisecond). While the times to compute properties may seem large, they can be parallelized trivially – indeed they were computed on a cluster with 64 processors for this experiment, and thus took only around three hours of wall-clock time for the largest of these data sets (Akrotiri). Moreover, the time spent re-ranking matches for each test set was spent by the computer, not by a person – i.e., no time was spent by an operator tuning weights or tweaking parameters individually for each data set, which is often the most time-consuming part of using computer-assisted fragment matching. Rather, the "parameter tuning" was done automatically by the machine learning algorithms. In cases where human time is more valuable than computer time, this is a real advantage of the proposed approach.

	#	# Matches			Compute Time (s)		
Data Set	Frag	Original	Candidate	Correct	Process	Train	Test
Synthetic	130	19K	12K	97	26K	6	1
Tongeren	1306	310K	188K	203	214K	27	394
Akrotiri	1408	2523K	1050K	27	772K	17	404

Table 1: *Complexity and compute time statistics.*

9. Conclusion and Future Work

In this paper, we have investigated the idea of using techniques from machine learning to construct a classifier that combines many properties of fragment matches to improve accuracy of predictions in a fresco reconstruction system. We have considered a large set of match properties, several of which are novel, and we have compared results for three different frescoes. Our experimental results indicate that multiple properties can be combined with a classifier

© The Eurographics Association 2011.

to produce better rankings than the previous state-of-the-art method used for the same data sets. They also indicate that matches found in one fresco can be used as training examples for finding matches in other frescoes, which opens up interesting possibilities for collaboration across different archaeological sites.

Our work investigates just one way of using machine learning for reconstruction of fragmented objects. In the short term, the next step would be to iterate between building a classifier based on known matches and applying the classifier to discover new matches. A system of this type could be linked to a visualization tool that helps archaeologists verify predicted matches for each iteration, both virtually and physically. In the longer term, it would be interesting to characterize fragment matches based on distributions of their properties. From comparison of distributions, it might be possible to learn relationships between how different frescoes were constructed and/or how they broke, and it might be possible to use transfer learning techniques to adapt classifiers learned from examples in one fresco as they are applied to a new one. Finally, it would be interesting to investigate whether the techniques proposed in this paper could be applied to other fracture reconstruction problems, such as failure analysis in forensics, assembly of broken bones in paleontology, or repair of shredded documents.

Acknowledgments

We wish to thank Professor Christos Doumas, Andreas Vlachopoulos, Colton Funkhouser, and the conservators and archeologists at the Akrotiri Excavation Laboratory of Wall Paintings for their input and collaboration. We also thank the persons and institutions that have collaborated with us or facilitated our research by giving access to the fresco material from the Tongeren Vrijthof excavation: Lara Laken (Radboud University Nijmegen), Guido Creemers and Igor Van den Vonder (Gallo-Romeins Museum Tongeren), Alain Vanderhoeven (Vlaams Instituut voor het Onroerend Erfgoed), and the city of Tongeren, Belgium. Finally, we thank Dimitris Gondicas, Peter Nomikos Jr., The Kress Foundation, Seeger Foundation, Thera Foundation, Cotsen Family Foundation, Samsung Scholarship Foundation, Google, National Science Foundation (CCF-0937139, CCF-0347427 and CCF-0702580), and the Research Foundation - Flanders for their support.

References

[BLD*10] BROWN B. J., LAKEN L., DUTRÉ P., VAN GOOL L., RUSINKIEWICZ S., WEYRICH T.: Tools for virtual reassembly of fresco fragments. In *Seventh International Conference on Science and Technology in Archaeology and Conservation* (December 2010).

[BM92] BESL P. J., MCKAY N. D.: A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Mach. Intell. 14*, 2 (1992), 239–256.

[BTFN*08] BROWN B., TOLER-FRANKLIN C., NEHAB D., BURNS M., DOBKIN D., VLACHOPOULOS A., DOUMAS C., RUSINKIEWICZ S., WEYRICH T.: A system for high-volume acquisition and matching of fresco fragments: reassembling Theran wall paintings. *SIGGRAPH '08: SIGGRAPH 2008 papers* (Aug 2008).

[CM02] CAO Y., MUMFORD D.: Geometric structure estimation of axially symmetric pots from small fragments. *Proc. IASTED SPPRA* (Jan 2002).

[CWAB01] COOPER D., WILLIS A., ANDREWS S., BAKER J.: Assembling virtual pots from 3D measurements of their fragments. *Conference on Virtual reality* (Jan 2001).

[Dou92] DOUMAS C.: *The Wall-Paintings of Thera*. Thera Foundation, 1992.

[FT05] FORNASIER M., TONIOLO D.: Fast, robust and efficient 2D pattern recognition for re-assembling fragmented images. *Pattern Recognition 38*, 11 (May 2005), 2074–2087.

[HFG*06] HUANG Q., FLÖRY S., GELFAND N., HOFER M., POTTMANN H.: Reassembling fractured objects by geometric matching. *SIGGRAPH '06: SIGGRAPH 2006 Papers* (Jul 2006).

[KK01] KONG W., KIMIA B.: On solving 2D and 3D puzzles using curve matching. *IEEE Computer Society Conference on Computer Vision ...* (Jan 2001), 583–590.

[KL06] KOLLER D., LEVOY M.: Computer-aided reconstruction and new matches in the Forma Urbis Romae. *Bullettino Della Commissione Archeologica Comunale di Roma* (2006), 103–125.

[Lak10] LAKEN L.: Wall-paintings in Atuatuca Tungrorum: preliminary report on the plaster fragments from the Vrijthof in Tongeren (Belgium). *Atti del X congresso internazionale dell'AIPMA (Association Internationale pour la Peinture Murale Antique) Napoli 17-12 sett. 2007, vol II* (2010), 865–869.

[MK03] MCBRIDE J., KIMIA B.: Archaeological fragment reconstruction using curve-matching. *Conference on Computer Vision and Pattern Recognition Workshop (CVPRW)* (Jan 2003).

[MP06] MAKRIDIS M., PAPAMARKOS N.: A new technique for solving a jigsaw puzzle. *ICIP* (Oct 2006), 2001–2004.

[PK03] PAPAIOANNOU G., KARABASSI E.: On the automatic assemblage of arbitrary broken solid artefacts. *Image and Vision Computing 21*, 5 (Jan 2003), 401–412.

[PPE02] PAPAODYSSEUS C., PANAGOPOULOS T., EXARHOS M.: Contour-shape based reconstruction of fragmented, 1600 b.c. wall paintings. *IEEE Transactions on Signal Processing 50*, 6 (Aug 2002), 1277–1288.

[PSCC07] PARIKH D., SUKTHANKAR R., CHEN T., CHEN M.: Feature-based part retrieval for interactive 3d reassembly. *IEEE Workshop on Applications of Computer Vision* (Jan 2007).

[SDF*10] SHIN H., DOUMAS C., FUNKHOUSER T., RUSINKIEWICZ S., STEIGLITZ K., VLACHOPOULOS A., WEYRICH T.: Analyzing fracture patterns in Theran wall paintings. In *International Symposium on Virtual Reality, Archaeology and Cultural Heritage (VAST)* (September 2010).

[TFBW*10] TOLER-FRANKLIN C., BROWN B. J., WEYRICH T., FUNKHOUSER T., RUSINKIEWICZ S.: Multi-feature matching of fresco fragments. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)* (December 2010).

[WC08] WILLIS A., COOPER D.: Computational reconstruction of ancient artifacts. *IEEE Signal Processing Magazine 25*, 4 (Jun 2008), 65–83.

[WF05] WITTEN I. H., FRANK E.: *Data Mining: Practical machine learning tools and techniques, 2nd Edition*. Morgan Kaufmann, 2005.

© The Eurographics Association 2011.

Appendix: Match Property Computation Details

This appendix provides detailed descriptions of how match properties are computed for a match, M, defined by two fragments, F_0 and F_1, and a rigid transformation, T.

For each fragment (F_i), our input data includes a color image (I_i) of its front surface, a "ribbon" (R_i) representing the 3D surface of fracture edge, and a "contour" (C_i) representing a 2D horizontal slice of the ribbon (Figure 1). As described in [BTFN*08], the ribbon is regularly sampled, with $m \cdot n$ vertices arranged in m rows and n columns to facilitate rapid property computation and correspondence finding. The contour is sampled similarly, with m evenly spaced vertices corresponding to rows of the ribbons.

Our first processing step is to find "contour contact correspondences," CC_i, for each contour C_i of M. For every vertex of C_i, we find the closest vertex in $T(C_{1-i})$ and add a contact correspondence to CC_i iff they are within 1mm of each other and have normals oriented in opposite directions $\pm 60^\circ$ (black lines in Figure 2d). We optimize these correspondences along with the transformation T with ten iterations of the ICP algorithm [BM92].

Second, we find "ribbon contact correspondences," RC_i for each ribbon R_i with a similar method. For every vertex in row r and column c of ribbon R_i, we find the closest vertex in column c of $T(R_{1-i})$ and add a contact correspondence iff they meet the same distance and normal compatibility criteria. We optimize these correspondences along with T with one iteration of ICP algorithm [BM92].

Third, we compute a "contact region," RR_i (and CR_i), for each ribbon R_i (and contour C_i) by searching for the longest sequence of consecutive rows in the contact correspondences allowing gaps less than $|C_i|/16$ (green regions in Figures 2b and 2e).

Fourth, we compute a "window" of correspondences $RW_i(s)$ (and $CW_i(s)$), for each ribbon R_i (and contour C_i) by establishing correspondences between vertices R_i[Center(CR_i)+r,c] and R_{1-i}[Center(CR_{1-i})-r,c] for all $r \in$ [0 - s] and all c within a 5mm swath centered on the middle column of the ribbon.

Fifth, we compute interior and exterior angles at the endpoints of the contact regions, CJ for contours and RJ for ribbons (we call them "junctions"). Specifically, we form line segments originating at the junctions (block dots in Figure 2e) and extending to points 5mm clockwise and counterclockwise on each contour (black line segments in Figure 2e) and then compute the interior and exterior angles between them (light purple and dark purple arcs in Figure 2e).

Finally, we compute match properties using the equations shown in Table 2.

ContourContactLength	$0.5 \cdot (\|CR_0\| + \|CR_1\|)$
ContourContactDensity	$0.5 \cdot (\|CC_0\|/\|CR_0\| + \|CC_1\|/\|CR_1\|)$
ContourContactRMSD	$\sqrt{\sum_{i,j}(C_{i,i}[j] - C_{i,1-i}[j])^2}$, where $(C_{i,i}[j], C_{i,1-i}[j]) \in CC_i, i \in \{0,1\}, j \in \{0,..,\|CC_i\|\}$
ContourContactLinearity	$\sqrt{\sum_{i,j}(C_{i,i}[j] - L_i)^2}$, where $C_{i,i}[j] \in CC_i, i \in \{0,1\}, j \in \{0,..,\|CC_i\|\}$, and L_i is the minimizing line
ContourContactCurvL2 (4 properties)	$\sqrt{\sum_{i,j}(\text{Curv}(C_{i,i}[j],t,s) - \text{Curv}(C_{i,1-i}[j],t,s))^2}$, where $(C_{,ii}[j], C_{i,1-i}[j]) \in CC_i, i \in \{0,1\}, j \in \{0,..,\|CC_i\|\}$, $t \in$ { Horizontal }, and $s \in$ { 1mm, 2mm, 4mm, 8mm }
ContourContactLengthFraction (4 properties)	Stat($\|CR_i\|$)/Measurement(C_i)), where Stat $\in$ { Min, Max }, and Measurement $\in$ { Perimeter, $\sqrt{Area}$ }
ContourWindowRMSD (3 properties)	$\sqrt{\sum_{i,j}(C_{i,i}[j] - C_{i,1-i}[j])^2}$, where $(C_{i,i}[j], C_{i,1-i}[j]) \in CW(s), j \in \{0,..,\|CW(s)\|\}$, and $s \in$ { 4mm, 8mm, 16mm }
ContourMergeConvexity	Convexity($C_0 \cup C_1$)
ContourMergeConvexityFraction (2 properties)	Stat(Convexity(C_0) / Convexity($C_0 \cup C_1$), Convexity(C_1) / Convexity($C_0 \cup C_1$)), where Stat $\in$ { Min, Max }
ContourOverlapArea	$\|C_0 \cap C_1\|$
ContourOverlapDepth (2 properties)	Stat(Depth($C_{i,i}[j]$)), where $C_{i,i}[j] \in CC_i, i \in \{0,1\}, j \in \{0,..,\|CC_i\|\}$, and Stat $\in$ { Avg, Max }
ContourJunctionAngle (4 properties)	Stat(Angle(CJ_i, t)), where Stat $\in$ { Min, Max }, and $t \in$ { Exterior, Interior }
RibbonContactArea	$0.5 \cdot (\|RR_0\| + \|RR_1\|)$
RibbonContactDensity	$0.5 \cdot (\|RC_0\|/\|RR_0\| + \|RC_1\|/\|RR_1\|)$
RibbonContactLength	$0.5 \cdot ((\|RR_0 \rightarrow C_0\| + \|RR_1 \rightarrow C_1\|)$, where $RR_i \rightarrow C_i$ is the projection of RR_i onto C_i
RibbonContactRMSD	$\sqrt{\sum_{i,j}(R_{i,i}[j] - R_{i,1-i}[j])^2}$, where $(R_{i,i}[j], R_{i,1-i}[j]) \in RC_i, i \in \{0,1\}, j \in \{0,..,\|RC_i\|\}$
RibbonContactPlanarity	$\sqrt{\sum_{i,j}(R_{i,i}[j] - P_i)^2}$, where $R_{i,i}[j] \in RC_i, i \in \{0,1\}, j \in \{0,..,\|RC_i\|\}$, and P_i is the minimizing vertical plane
RibbonContactHCurvL2 (4 properties)	$\sqrt{\sum_{i,j}(\text{Curv}(R_{i,i}[j],t,s) - \text{Curv}(R_{i,1-i}[j],t,s))^2}$, where $(R_{i,i}[j], R_{i,1-i}[j]) \in RC_i, i \in \{0,1\}, j \in \{0,..,\|RC_i\|\}$, $t \in$ { Horizontal }, and $s \in$ { 1mm, 2mm, 4mm, 8mm }
RibbonContactCurvL2 (4 properties)	$\sqrt{\sum_{i,j}(\text{Curv}(R_{i,i}[j],t,s) - \text{Curv}(R_{i,1-i}[j],t,s))^2}$, where $(R_{i,i}[j], R_{i,1-i}[j]) \in RC_i, i \in \{0,1\}, j \in \{0,..,\|RC_i\|\}$, $t \in$ { Vertical, Mean }, and $s \in$ { 1mm, 2mm }
RibbonWindowRMSD (3 properties)	$\sqrt{\sum_{i,j}(R_{i,i}[j] - R_{i,1-i}[j])^2}$, where $(R_{i,i}[j], R_{i,1-i}[j]) \in RW(s), j \in \{0,..,\|RW(s)\|\}$, and $s \in$ { 4mm, 8mm, 16mm }
RibbonJunctionAngle (4 properties)	Stat(Angle(RJ_i, t)), where Stat $\in$ { Min, Max }, and $t \in$ { Exterior, Interior }
FragmentThicknessL2	(Thickness(F_0) - Thickness(F_1))^2, where Thickness(F_i) is the average number of columns with scanned vertex positions in each row of R_i
FragmentFrontColorL2 (12 properties)	(Stat(I_0, c) - Stat(I_1, c))^2, where Stat $\in$ { Mean, Median, Variance }, and $c \in$ { Red, Green, Blue, Luminance }
FragmentAreaFraction	min($\|C_0\|/\|C_1\|, \|C_1\|/\|C_0\|$)

Table 2: *Match property equations. Note that $|X|$ represents the size of a point set X (e.g., the length of a contour, or the area of a polygon).*

© The Eurographics Association 2011.