The 12th International Symposium on Virtual Reality, Archaeology and Cultural Heritage VAST (2011)
M. Dellepiane, F. Niccolucci, S. Pena Serna, H. Rushmeier, and L. Van Gool (Editors)

Global Consistency in the Automatic Assembly of Fragmented Artefacts

A. García Castañeda[1], B. Brown[3], S. Rusinkiewicz[2], and T. Funkhouser[2], T. Weyrich[1]
[1]University College London [2]Princeton University [3]K. U. Leuven

Abstract

Automatic reconstruction of fragmented objects is of great interest in archaeology, where artefacts are often found in a fractured state. In this paper, we focus on the problem of automatically agglomerating clusters of fragments from previously determined pairwise matches. Common to any automated cluster agglomeration technique is the challenge of error accumulation, making it increasingly difficult to discern false from true matches as the assembly grows. Many assembly algorithms therefore introduce a global relaxation phase to distribute alignment errors evenly across the cluster, minimising major inconsistencies. Nevertheless, error accumulation limits the problem size automated assembly systems can handle in practice. In this paper we show how two careful modifications of the traditional relaxation scheme help lift this limit considerably. In contrast to previous work, we integrate global relaxation earlier, in the search phase of the assembly process. In addition, we do not fix connections between assembled fragments, but rather leave them flexible throughout the assembly. By modifying two representative assembly algorithms, we demonstrate the effectiveness of our approach. Using the example of a challenging fresco dataset, we show that these modifications achieve larger reconstruction sizes than traditional strategies.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modelling—Geometric algorithms, languages and systems

1. Introduction

Automatic reconstruction of fragmented objects is of great interest in archaeology, where artefacts are often found in a fractured state, and the effort of manually putting the broken pieces together represents a significant amount of time and resources. Previous work has contributed methods to acquire object fragments efficiently, identify pairwise matches between fragments, and automatically assemble them into larger clusters. In this paper, we focus on aspects of the final cluster agglomeration phase.

Common to any automated cluster agglomeration technique is the challenge of error accumulation, which makes reconstructing objects from a large number of fragments increasingly difficult. Minute alignment errors between adjacent fragments add up, eventually leading to gaps and interpenetration of fragments throughout the cluster, preventing pieces from fitting together properly [HFG*06]. Many assembly algorithms hence introduce a *global relaxation* phase, distributing alignment errors evenly across the cluster to minimise major inconsistencies.

Reducing such inconsistencies is more than a cosmetic task: in practice, matches between fragments cannot be identified with absolute certainty, and incorporating global consistency criteria improves the discrimination between true and false matches. By recurrently relaxing the active cluster before considering additional fragments [GMB04, ZZH08, HFG*06], existing assembly systems keep alignment errors of true matches low and maintain match discriminability also for larger clusters. Even with global relaxation, however, error accumulation increases as the cluster grows, eventually rendering true and false matches indistinguishable. This effect limits the problem size automated assembly systems can handle in practice [ZZH08, Str09].

In this paper we show that two careful alterations of the traditional relaxation scheme help lift this limit considerably. Instead of relaxing the active cluster first and then evaluating potential fragment additions, we incorporate global relaxation in the match evaluation itself, assessing each potential addition by considering alignment errors *after* global relaxation with the match candidate. Although this approach of course impacts the algorithm's speed, we mitigate this factor by using an efficient global relaxation scheme so that the trade-off with increased cluster sizes becomes worthwhile. In addition, we differ from many previous approaches by maintaining the fragments as separate entities for the entirety of the assembly process, rather than fixing well-established subclusters, which is a common choice to reduce the costs of relaxation. Together, these modifications build a search strategy that puts more emphasis on global consistency than existing approaches.

By modifying two representative assembly algorithms, we demonstrate the effectiveness of our approach. We show reassembly results for a shattered fresco that has been custom-made for a previous study. Note that wall paintings represent a class of physical objects that usually exhibit a two-dimensional fracture topology, as do tablets, parchments, papyri, and pottery. As such an important object class in archaeology, we argue that the insights gained in this work extend to three-dimensional fracture topologies as well.

2. Previous Work

Much work has been done on the problem of automatically reassembling fragmented objects, starting with Freeman and Garder's [FG64] work in encoding and matching arbitrary jigsaw puzzle piece contours.

The reconstruction of an artefact can be divided into two main phases: local assembly and global assembly [HFG*06]. Local assembly is concerned with finding matches between pairs of fragments, or with searching for matches in the immediate neighbourhood of a fragment. Global assembly is the task of taking pairwise match hypotheses and creating as complete a solution as possible to the reconstruction problem.

2.1. Local Assembly

Most reassembly work to date has dealt primarily with methods of determining pairwise matches between fragments. There is a wide variety of techniques employed in this endeavour, including comparisons of shape, colour, texture, or more abstract features. Geometric methods commonly focus on the 2-D contour of the fragments, or on the surface area of the fragment edges. Pictorial match detection commonly looks at the colours around the edges of the fragments, or some analysis of the colour of the entire fragment surface. The material properties of the fragment sometimes play a role in matching algorithms [PKT02]. An in-depth discussion of pairwise matching techniques is outside the scope of this paper, but good explanations of the subject can be found in [WC08], [MK03], and [FSTF*11].

Independent from the particular matching technique, however, any form of local matching remains an error-prone process. Apart from natural sources of noise, such as erosion and discolouration, the process of digitisation introduces sampling errors into the fragment models. When comparing scanned fragments, these errors can cause both false and true matches to score equally well on a particular matching metric. As a result, virtually any successful automated assembler incorporates global criteria as well.

2.2. Global Assembly

One of the major difficulties of reassembly is that the problem scales poorly with the number of fragments being assembled, due to the combinatorial explosion of potential assemblies. Hence, global assembly algorithms tend to use greedy approaches that iteratively merge or grow clusters in the search for a solution that maximises a scoring function.

Typically, assembly systems model the process as a graph, with nodes representing fragments and edges representing candidate matches between pairs of fragments. Although the minimum reconstruction possible is a spanning tree of the connection graph, it is desirable to try to include as many matches as possible, since it is not guaranteed that all correct matches between fragments will be found. Assembly can take the form of constructing a connection graph from the candidate matches, and then solving for the most globally consistent set of edges [PK03]. When there are loops in the graph, it becomes easier to select which edge between a pair of fragments is the correct one, if any.

Hierarchical Clustering Hierarchical clustering generally takes the form of a greedy, best-first merging of pairwise matches. Similar to the traditional algorithmic technique of "divide and conquer", where a problem is subdivided recursively until it is simple enough to be solved directly, hierarchical clustering reduces the fragment count by replacing two matching fragments with one new fragment, effectively creating a "merge and conquer" strategy. The reconstruction begins with a pairwise match, typically the highest-scoring one. This match is then merged, and a new fragment is created to replace the two matching fragments. The matching features for the new, larger fragment are computed, and it is checked against all other fragments in the pool for matches. The procedure is repeated until no more matches can be found, or until only one fragment remains.

This technique is a popular choice [BBB05, ÜHT99, PPE02, KYKI01, BK93, SE06, SE08, MP06, dS09] and exists in a number of variants. Some implementations incorporate backtracking to help escape local minima. However, since the search space is so large, all implementations employ heuristics based on the quality of individual matches and/or small clusters and therefore are not guaranteed to arrive at the globally optimal solution.

Dense Cluster Growth To improve the chance of finding a globally optimal solution, some methods consider multiple matches at every iteration of a greedy algorithm. For example, [KK01, MK03, WF10] note that merging triplets of fragments produces fewer missteps in the search than merging pairs. Similarly, Goldberg et al. [GMB04] observe that considering two matches when adding each fragment to a cluster achieves better results than methods that consider just one match at a time, and greatly improves disambiguation of matches when reconstructing jigsaw puzzles, in particular in the case of fragments with large straight edges. Even so, they are not able to build large clusters with just this technique, due to accumulation of alignment errors in long sequences of matches.

Global Relaxation To assist building large clusters from pairwise matches, a few methods include a global relaxation step to optimise fragment alignments in each step of a greedy search [GMB04, HFG*06, Str09, ZZH08]. For example, after each fragment is added to a cluster, [GMB04] optimise the placement of all fragments in the cluster to minimise the sum of squared distances between corresponding feature points on fragment boundaries. [Str09] optimise the fragment transformations to minimise the sum of squared distances between all closest points on adjacent fragment contours. [ZZH08] optimise a match compatibility score based on fragment area and boundary overlaps. These methods do indeed improve the placement of fragments as clusters are built; however, the optimisation is performed in each step only after a fragment is added to a cluster, and thus does not directly influence the selection of the next fragment(s) to be added. As a result, the greedy algorithm can take missteps when it selects a fragment to add without considering the information in the resulting global alignment.

3. Overview

Drawing from the collective body of previous work, we have designed a fully automatic system for assembling clusters of fragments from a given set of proposed matches. We show how two critical design decisions, inclusion of global relaxation into the assessment of individual matches and including all individual fragments in relaxation at all times, improve the maximum achievable cluster size.

The system takes a set of digitised fragments and a set of candidate pairwise matches, and produces a set of globally-consistent clusters (Section 4). The candidate matches can be from any combination of pairwise match detection algorithms — the only requirement is that all of the matches be scored with the same ranking metric. Section 5 describes two cluster operators, global relaxation and a match completion, that distribute error throughout a cluster. Finally, our fitness function (Section 6) measures the degree of global consistency and is used in both example assembly strategies (Section 7).

4. Input Match Candidates

Because different pairwise matching algorithms typically find different subsets of the true matches, it is advantageous to use the output of several such algorithms simultaneously. We therefore opt for a very loose coupling of the pairwise matching and global assembly stages: any pairwise matching algorithm can be use to produce a set of candidate matches, and the output of multiple algorithms can be combined simply by re-ranking all proposals according to a single scoring metric. The global assembly relies exclusively on this rank and on the pairwise alignment transformations for each candidate match. As a result, it is only responsible for *recognising* matches, not *producing* them.

In contrast, “merge and conquer” systems fuse fragments together and generate new pairwise matches among the larger, merged fragments. This can be advantageous because multiple, short matching edges may be merged together to form a single, long matching edge that is easier to detect. However, global relaxation becomes more complex if fragments are merged, as does combining different pairwise matching algorithms when their use is interspersed with clustering. When clustering purely on the basis of precomputed candidates is insufficient, we suggest an intermediate approach where entire clusters are fused, followed by a new round of pairwise matching and global clustering.

5. Global Relaxation

We perform global relaxation using a physically-based nonlinear solver to simultaneously minimise all local alignment errors. To further improve its effectiveness at distributing local error, we start by applying a match completion operator to create a maximally dense set of connections within the cluster.

5.1. Physical Simulation

In the physical simulation, each fragment in a cluster is represented as a rigid body with unit mass and moment of inertia. Each match is decomposed into three springs: one linear spring to regulate the distance between the fragments, and two torsion springs for the angles of the fragments with respect to each other. The simulation accumulates the forces exerted on each fragment by the set of springs, then computes net linear and angular accelerations for each rigid body. A fourth-order Runge-Kutta ordinary differential equation solver is used to solve for the positions and orientations of the fragments at a point in time slightly after the input time. This procedure is iterated until the change in energy of the system converges within an epsilon of the previous iteration.

The simulation does not include collision detection between the fragments. In any such simulation, the gross majority of the effort is spent resolving collisions, so omitting their computation is a very significant reduction in cost. The candidate matches that are the input to the system already encode physical interpenetration constraints, so a spring-measured deviation is directly related to a corresponding deviation from the physically optimal alignment. As such, this form of global relaxation is a fast and efficient approximation of a physically-correct solution.

The low cost of the relaxation allows it to be used more often than would otherwise be possible. We perform global relaxation not only after the addition of each fragment to the cluster, but also during the search process. When examining a candidate fragment and the set of matches potentially linking it to the the cluster being grown, we perform global relaxation on this configuration before any scoring is performed.

In contrast to the "merge and conquer" strategies discussed in Section 2.2, our approach does not fix the relative poses of fragments as they are merged. As a result, any error in the match used to join the fragments is not carried forward into further assembly calculations because global relaxation can correct them as new fragments are added.

5.2. Match Completion

Global relaxation is most effective at distributing error when there are many matches linking the fragments, which leads to many loops in the connection graph. Error is distributed along these loops, so it is desirable to have a maximally dense set of connections in the cluster [HFG*06]: every fragment that could have a match linking it to its neighbour should have one. This operation finds all pairs of adjacent but unconnected fragments in the cluster, then gathers all possible matches from the input set that could be used to link each pair. The combinations of these candidate matches are enumerated, and the best-scoring valid match configuration is then added to the cluster, and global relaxation is performed.

6. Fitness Function

By determining the global consistency of a cluster, the global fitness function plays a key role in global assembly, incorporating the residual of the global relaxation (expressed as energy in the simulated springs) and a measure for fragment interpenetration within the clustering. More precisely, we consider the displacements of linear match springs, the displacements of angular match springs, and the largest interpenetration distance of fragments.

We define thresholds for each component of the fitness function. When evaluating potential additions to the cluster, these thresholds determine whether a configuration is valid. The threshold values are set generously, so as not to exclude any real matches, despite the error present in the cluster. Empirically determined values that fulfil this need are: a maximum interpenetration distance of 7.5 mm, a maximum linear displacement of 12.5 mm, and a maximum angular displacement of $\pi/3$ radians. Any match configurations that pass these threshold values are considered to be valid.

Once the valid configurations for growing a cluster are retained, they are tested by adding them to the cluster to be grown, relaxing the test cluster, and then scoring the resulting cluster. The score of a cluster is defined as the sum of the uniform norm of these three values. This is the sum of the maxima of linear spring displacements, angular spring displacements, and largest fragment interpenetration distances.

It is worth noting that this scoring function produces a result that is independent of the size of the cluster. This way, there is no bias in constructing smaller or larger clusters, and two clusters of different sizes can be compared meaningfully, as it is the worst values of each that are being compared.

7. Assembly Strategies

The main novelty in our approach is that we perform global relaxation on a cluster with a new fragment *before* deciding if that fragment is correctly placed, rather than after. While requiring an increased amount of relaxation steps, this strategy is greatly aided by the high efficiency of the relaxation operations described in Section 5.

This idea is independent of the clustering approach taken, and we demonstrate its efficacy by incorporating it into two representative assembly algorithms, one being a variant of agglomerative, best-first hierarchical clustering, the other being inspired by triplet-based merge strategies, incrementally growing a cluster one fragment at a time. Despite having different merge strategies, both algorithms share the common aspect of incorporating global relaxation into the search, evaluating each potential configuration in a global context.

7.1. Hierarchical Clustering

Our basis implementation of hierarchical clustering follows the common structure of agglomerative hierarchical clustering. We perform greedy best-first hierarchical clustering, where every fragment is placed into a cluster, the best candidate match is taken from the set, and the clusters are joined by means of that match, and then global relaxation is performed on the resulting cluster. This process is repeated until no more matches remain.

We extend this algorithm by evaluating each potential configuration's global fitness (Section 6) *after* performing global relaxation on what the result would be if the configuration were constructed. Once a valid configuration is encountered, we merge the clusters and then apply the match completion operator (Section 5.2), followed by a final relaxation pass.

7.2. Dense Cluster Growth

Our approach is inspired by the observation by Goldberg et al. [GMB04] that placing a puzzle piece in the context of two neighbouring pieces, rather than considering a single match at a time, sufficiently constrains the problem to allow for a greedy approach.

Initially, seed clusters are formed from the best-ranking candidate matches: top-scored fragment pairs are merged to build seed clusters until all fragments are part of a seed. Not all of these matches will be correct, and will not grow to be large clusters, but enough of the seed clusters will be based on correct matches for the reconstruction to proceed.

Starting from each seed cluster, we run *dense cluster growth* by iteratively adding the best configuration of a fragment and two matches that link it to two fragments in the cluster. These configurations are computed by examining each fragment in the cluster for valid candidate matches to fragments outside the cluster, and then looking for valid candidate matches from the external fragment to a second fragment in the cluster. Each valid configuration is evaluated according to its global fitness by adding the external fragment to the cluster, linked by the two matches in the configuration, and the resulting cluster is subjected to thresholding and scored (Section 6). The valid configurations are stored in a list, ranked by their global fitness scores. After all fragments in the cluster have been examined, the best-scoring configuration, if any, is added to the cluster, and the cluster is relaxed. If there are no valid configurations for adding a fragment to the cluster, the cluster is no longer considered for growth.

Note that the original pairwise match rankings are only used for seed cluster creation; during the growth phase, ranking is solely based on the global fitness function, which includes both pairwise relations and global consistency.

Similarly to the case of hierarchical clustering, we optionally extend this algorithm by adding global relaxation to the scoring phase: whenever a valid configuration is scored, we first run global relaxation *before* determining its global fitness. Once a fragment has been added, we perform match completion and globally relax again.

8. Results

To evaluate the performance of the system, we attempt to assemble a known dataset: the 129-piece synthetic fresco used by Brown et al. [BTFN*08]. This fresco was created by conservators, then shattered, and many fragments have been removed to increase realism of the case. The remaining fragments were digitised, and a set of ground-truth matches was created, based on a manual reassembly. Due to the missing fragments, the fresco is not one connected cluster; in this evaluation, we focus on the largest cluster of 118 fragments (see Figure 1).

Two sets of matches are used in reconstructing this fresco: a set of 256 ground-truth matches, and a set of 11,474 high-scoring matches. The set of ground-truth matches for this

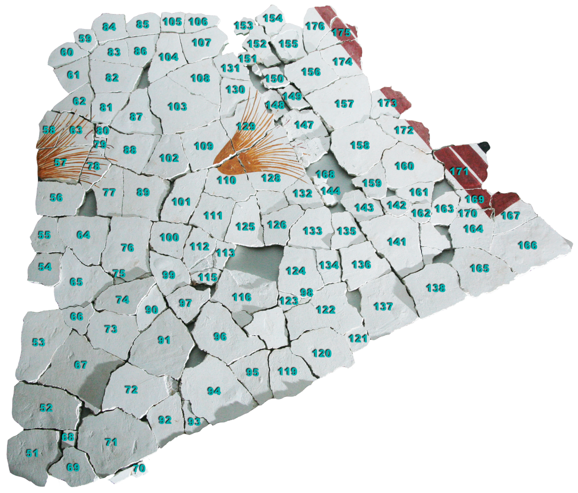


Figure 1: The ground-truth synthetic fresco used by Brown et al. [BTFN*08]. (The fresco is disconnected; shown here is only the largest cluster.)

fresco was obtained by manually positioning pairs of fragments in their known relative orientations, and then optimising their poses by means of the iterative closest points registration algorithm [BM92]. The set of high-scoring matches is computed with an algorithm similar to Brown et al.'s RibbonMatcher [BTFN*08]. It is the combined result of three runs of the algorithm, using window sizes of 12.5 mm, 25 mm, and 50 mm sampling windows. As with all known pairwise matching algorithms, the RibbonMatcher does not have perfect recall, so some genuine matches between fragments were not found. Where these genuine matches were absent, matches from the ground truth set were added as a supplement. This is to ensure that cluster growth is not inhibited by lack of viable matches. The combined matches were then scored by a decision tree based on numerous fragment features, similar to the one used by Funkhouser et al. [FSTF*11]. The first 152 ranked matches in this set are true ones, but the precision falls away quickly after that. There are 256 matches in the ground-truth set, so approximately 40% of the correct matches are not the best-scoring ones.

Given the design of the dense cluster growth algorithm, which requires that an added fragment be joined to the cluster in two places, it is not possible to reconstruct the entire fresco from any given starting point, as it becomes impossible to progress into areas where fragments are connected to only one other. There are several regions that can be reconstructed using the dense cluster growth method; here we are reporting on the largest clusters only, as the problem of error accumulation is less pronounced for small clusters.

We show the effectiveness of our modified match scoring

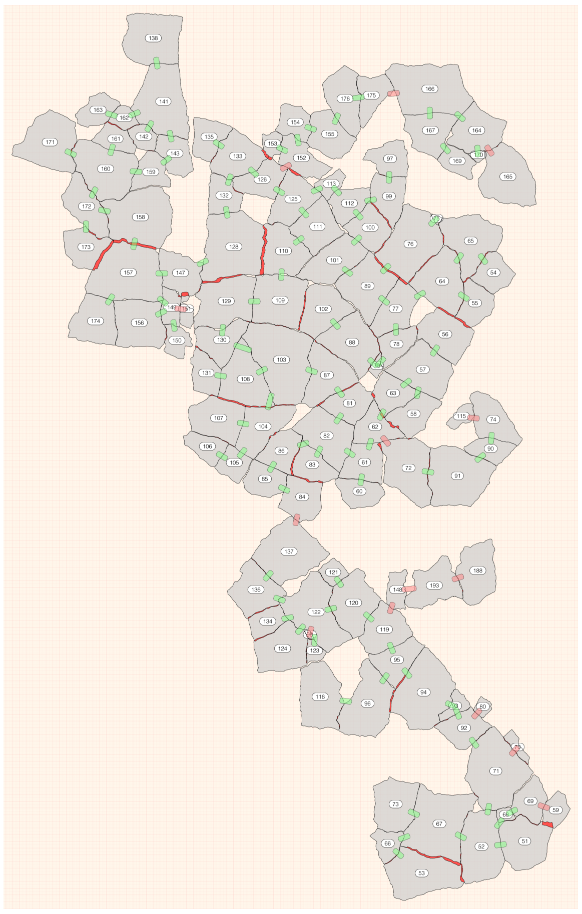


Figure 2: Hierarchical Cluster growth, when performing relaxation only after each merge operation (see Figure Legend below). The assembled cluster contains 118 fragments and 117 matches, 13 of which are incorrect.

approach for both assembly strategies by comparing results of the baseline versions with the modified algorithms.

With classic hierarchical clustering alone, all 118 fragments are placed with the minimum possible 117 ($n-1$) matches, 13 of which are incorrect (see Figure 2). When global relaxation is incorporated into the search phase, and the match completion operator is applied after each merge operation, performance improves: all 118 fragments are placed, but with 188 matches, 10 of which are incorrect (see Figure 3).

In the case of the dense cluster growth algorithm, without performing global relaxation as part of the search, most of the largest regions cannot be grown accurately (see Figure 4). If global relaxation is performed as a part of the search process and not only after the best fragment and its joining matches are found, there is a much higher rate of

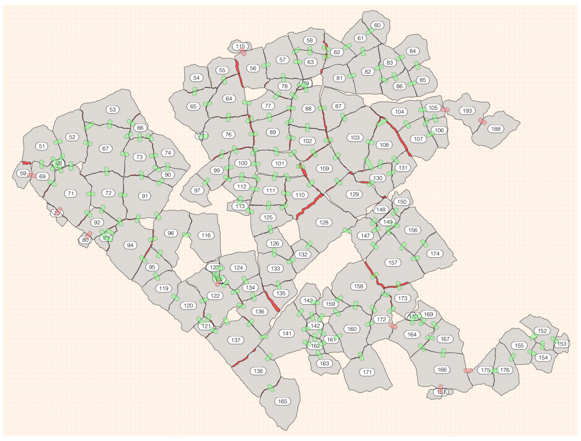


Figure 3: Hierarchical Cluster growth, when performing relaxation during the search process. The assembled cluster contains 118 fragments and 188 matches, 10 of which are incorrect.

success, as many more spurious matches are rejected (see Figure 5).

Performing global relaxation after adding each fragment is much less expensive than performing it for each candidate fragment configuration as part of the search process, but it does not yield equally good results. (Specifically, running times for hierarchical clustering were approximately 40 min without and 4½ hrs with our modification, while dense cluster growth requires 1 hr without and 3½ hrs with relaxation during the search phase when started from within the largest cluster.)

Overall, our modification significantly increases the reconstruction size of our test dataset, see Figure 6.

9. Discussion and Future Work

We have shown that using a more thorough approach to global consistency checks when evaluating potential matches increases the size of assembly problems that can be solved in the vicinity of local alignment errors.

The number of fragments successfully assembled automatically by the system may not be large as in the mostly-accurate reconstruction of 320-piece jigsaw puzzles by

Figure Legend

Matches indicated in light green are correct matches, and matches in light red are incorrect. Areas where fragments interpenetrate are shaded in red. Small numbers show fragment identifiers. Coarse grid lines denote centimetres.

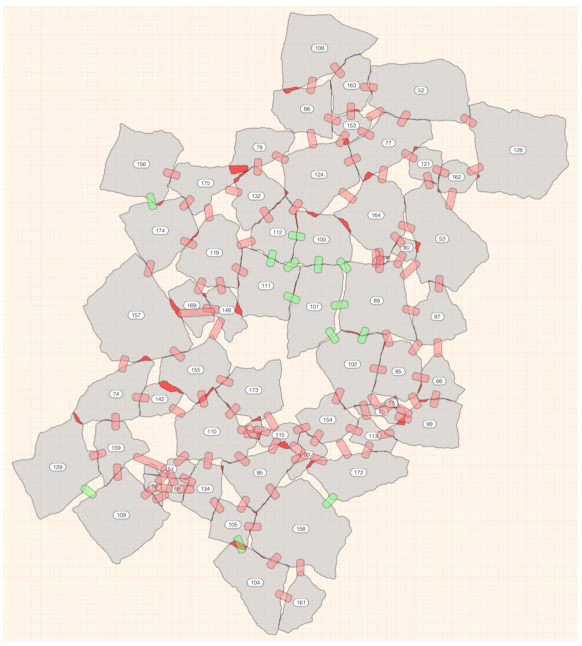


Figure 4: Cluster growth, (starting with the match connecting fragments 100 and 101) when performing relaxation only after selecting the best candidate fragment. The assembled cluster contains 58 fragments and 113 matches, 13 of which are correct, and 100 incorrect.

Nielsen et al. [NDH08], but it is worth noting that our dataset is particularly problematic. The fragments exhibit erosion, and many pieces have at least one straight edge, which greatly increases the complexity of the assembly search. Note that Zhu et al. [ZZH08] report that if the number of fragments being assembled is “not greater than about 50,” automatic global assembly is possible. They found matches along straight edges of fragments to be so problematic that they excluded them, and relied on being able to find a spanning tree of matches, rather than every possible match. Others resort to strongly penalise matches along straight edges [MK03]. Although it is easier to disambiguate candidate matches in a global context — the additional information helps ensure that false matches that score well locally are still penalised globally — error accumulation still quickly overwhelms the extra discriminatory power.

We presented two assembly strategies with distinct properties that make it conceivable to run both algorithms concurrently. While hierarchical clustering is able to interconnect clusters with single matches only (and hence may generally create larger clusters), our dense cluster growth algorithm applies stronger constraints to potential additions, ultimately leading to fewer false connections. Ultimately, however, any false matches added before termination are likely to be identified at the time of physical assembly.

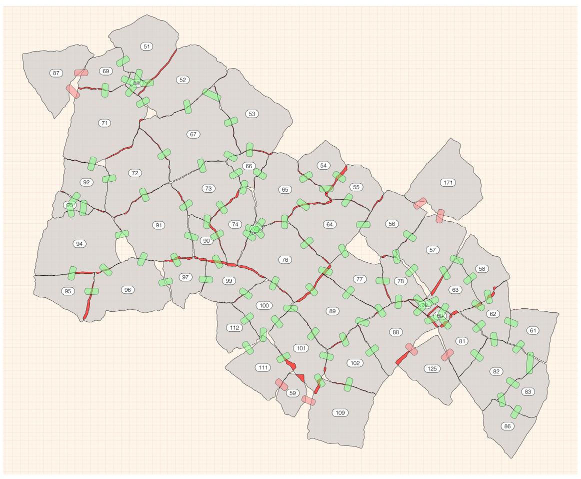
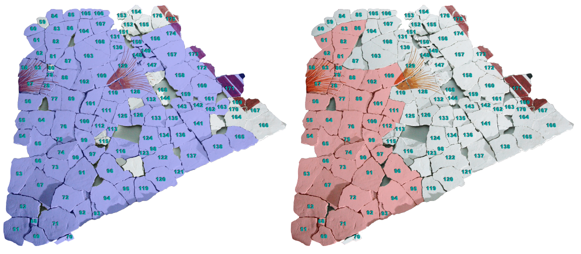


Figure 5: Cluster growth, (starting with the match connecting fragments 100 and 101) when performing relaxation during the search process. The assembled cluster contains 52 fragments and 102 matches, 94 of which are correct, and 8 incorrect.



(a) Hierarchical clustering (b) Dense cluster growth

Figure 6: Level of completion achieved with our modification. (a) Largest cluster found by hierarchical clustering. (b) Largest cluster assembled by dense cluster growth; note that the front where the assembly stopped runs through many four-way-junctions of cracks, where fragments cannot be added through two matches at the time.

As an algorithm based on pairwise matches, our approach differs from other approaches in that we postpone decisions on the suitability of a potential configuration until a match has been evaluated in its global context (including relaxation). As such, our system bears resemblance to previous approaches that do not rely on external match predictors, but find matches starting from the current state of the assembly. However, the constraints implied by the input pairwise matches rein in the search, so not all combinations need be evaluated.

A possible extension of the system is to perform a fully physically-correct global relaxation on the cluster after the addition of a fragment to the cluster, but to retain the fast approximation during the search.

10. Conclusion

We have presented a system for the fully-automatic assembly of fragmented 2-D objects, which operates on a set of fragments and a set of proposed pairwise matches, using two alternative assembly strategies. As a core contribution, we extended each of these algorithms by integrating global relaxation into the search phase of the assembly process.

We argue that, by putting more emphasis on global relaxation than previous approaches, we are making more effective use of global consistency information when disambiguating pairwise matches to guide the agglomerative assembly process. Furthermore, maintaining the fragments in the cluster as separate entities and not merging them once the cluster is grown allows them to be adjusted by subsequent global relaxation, as information introduced later in the assembly process might yield a more globally-consistent cluster.

Using a representative test dataset with considerable ambiguity in its pairwise matches, we have shown that this modification effectively increases the cluster size achievable in the presence of error accumulation. Since the system operates with a set of fragments and a set of proposed matches, these techniques are immediately applicable to all 2-D assembly problems, given digitised models of the fragments, and the output of a pairwise match proposal algorithm.

References

[BBB05] Biswas A., Bhowmick P., Bhattacharya B.: Reconstruction of torn documents using contour maps. *IEEE International Conference on Image Processing 3* (Jan. 2005), 517–520. 2

[BK93] Bunke H., Kaufmann G.: Jigsaw puzzle solving using approximate string matching and best-first search. *Lecture Notes in Computer Science* (Jan. 1993). 2

[BM92] Besl P. J., McKay H. D.: A Method for Registration of 3-D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence 14*, 2 (1992), 239–256. 5

[BTFN*08] Brown B., Toler-Franklin C., Nehab D., Burns M., Dobkin D., Vlachopoulos A., Doumas C., Rusinkiewicz S. M., Weyrich T.: A System for High-Volume Acquisition and Matching of Fresco Fragments: Reassembling Theran Wall Paintings. *ACM Transactions on Graphics (Proc. SIGGRAPH 2008) 26* (Sept. 2008), 84:1–84:9. 5

[dS09] de Smet P.: Semi-automatic Forensic Reconstruction of Ripped-up Documents. *Document Analysis and Recognition, 2009. ICDAR '09. 10th International Conference on* (2009), 703–707. 2

[FG64] Freeman H., Garder L.: Apictorial Jigsaw Puzzles: The Computer Solution of a Problem in Pattern Recognition. *Electronic Computers, IEEE Transactions on EC-13*, 2 (Apr. 1964), 118–127. 2

[FSTF*11] Funkhouser T., Shin H., Toler-Franklin C., García Castañeda A., Brown B., Dobkin D., Rusinkiewicz S., Weyrich T.: Learning How to Match Fresco Fragments. In *Eurographics Area Track on Cultural Heritage* (2011). 2, 5

[GMB04] Goldberg D., Malon C., Bern M.: A global approach to automatic solution of jigsaw puzzles. *Computational Geometry: Theory and Applications 28*, 2-3 (Jan. 2004), 165–174. 1, 3, 4

[HFG*06] Huang Q.-X., Flöry S., Gelfand N., Hofer M., Pottmann H.: Reassembling Fractured Objects by Geometric Matching. *ACM Transactions on Graphics (Proceedings of SIGGRAPH) 25* (July 2006), 569–578. 1, 2, 3, 4

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

[KYKI01] Kanoh M., Yasuhara S., Kato S., Itoh H.: Similarity-based approach to earthenware reconstruction. *Image Analysis and Processing, 2001. Proceedings. 11th International Conference on* (2001), 478–483. 2

[MK03] McBride J., Kimia B.: Archaeological Fragment Reconstruction Using Curve-Matching. *Proceedings of the 2003 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'03)* (Jan. 2003). 2, 7

[MP06] Makridis M., Papamarkos N.: A new technique for solving a jigsaw puzzle. *International Conference on Image Processing* (Oct. 2006), 2001–2004. 2

[NDH08] Nielsen T., Drewsen P., Hansen K.: Solving jigsaw puzzles using image features. *Pattern Recognition Letters 29*, 14 (Jan. 2008), 1924–1933. 7

[PK03] Papaioannou G., Karabassi E.: On the automatic assemblage of arbitrary broken solid artefacts. *Image and Vision Computing 21*, 5 (Jan. 2003), 401–412. 2

[PKT02] Papaioannou G., Karabassi E., Theoharis T.: Reconstruction of three-dimensional objects through matching of their parts. *IEEE Transactions on Pattern Analysis and Machine Intelligence 24*, 1 (Jan. 2002), 114–124. 2

[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. 2

[SE06] Sağiroğlu M. Ş., Erçil A.: A Texture Based Matching Approach for Automated Assembly of Puzzles. *Pattern Recognition, International Conference on 3* (2006), 1036–1041. 2

[SE08] Sağiroğlu M. Ş., Erçil A.: Optimization for automated assembly of puzzles. *Continuous Optimization and Knowledge-Based Technologies* (Jan. 2008), 18–24. 2

[Str09] Strife D.: *Multi-piece Matching of Fresco Fragments: Reassembling the Theran Frescos*. Tech. rep., Computer Science, Princeton University, Computer Science, Princeton University, Jan. 2009. 1, 3

[ÜHT99] Üçoluk G., HakkÄś Toroslu I.: Automatic reconstruction of broken 3-D surface objects. *Computers & Graphics 23*, 4 (Jan. 1999), 573–582. 2

[WC08] Willis A., Cooper D.: Computational reconstruction of ancient artifacts. *Signal Processing Magazine, IEEE 25*, 4 (June 2008), 65–83. 2

[WF10] Weiss J., Funkhouser T.: Fresco Fragment Clustering: An Analysis of Triplets. *Undergraduate Report, Computer Science, Princeton University* (May 2010), 1–24. 2

[ZZH08] Zhu L., Zhou Z., Hu D.: Globally Consistent Reconstruction of Ripped-Up Documents. *IEEE Transactions on Pattern Analysis and Machine Intelligence 30*, 1 (2008), 1–13. 1, 3, 7