



# Pictorial and Apictorial Polygonal Jigsaw Puzzles from Arbitrary Number of Crossing Cuts

Peleg Harel<sup>1</sup> · Ofir Itzhak Shahar<sup>1</sup> · Ohad Ben-Shahar<sup>1</sup>

Received: 23 December 2021 / Accepted: 11 February 2024 / Published online: 22 March 2024  
© The Author(s) 2024

## Abstract

Jigsaw puzzle solving, the problem of constructing a coherent whole from a set of non-overlapping unordered visual fragments, is fundamental to numerous applications, and yet most of the literature of the last two decades has focused thus far on less realistic puzzles whose pieces are identical squares. Here we formalize a new type of jigsaw puzzle where the pieces are general convex polygons generated by cutting through a global polygonal shape/image with an arbitrary number of straight cuts, a generation model inspired by the celebrated Lazy caterer's sequence. We analyze the theoretical properties of such puzzles, including the inherent challenges in solving them once pieces are contaminated with geometrical noise. To cope with such difficulties and obtain tractable solutions, we abstract the problem as a multi-body spring-mass dynamical system endowed with hierarchical loop constraints and a layered reconstruction process. We define evaluation metrics and present experimental results on both apictorial and pictorial puzzles to show that they are solvable completely automatically.

**Keywords** Computational jigsaw puzzle solving · Lazy caterer · Crossing cuts · Pictorial and apictorial puzzles · Loopy constraints · Hierarchical loops

## 1 Introduction

It happens often in real life that an orderless set of given fragments should be matched correctly (typically with no overlaps) to reconstruct a desired (known or unknown) coherent shape. Indeed, this broader (yet informal) problem description of the common jigsaw puzzle game fits countless real-world applications, including in (but not limited to) archaeology (Willis & Cooper, 2008; Kleber, 2009; Sizikova & Funkhouser, 2016), biology (Gassner et al., 1996; Marande & Burger, 2007), earth sciences (Lindström, 2019), paleontology (Warren et al., 2014), security and forensics (Gao et al., 2010; Ali et al., 2014; Gioe, 2017), artificial intelligence (Zhao et al., 2020), speech processing (Zhao et al., 2007), document reconstruction (Kleber & Sablatnig, 2009), not to mention direct image editing (Cho et al., 2010) and artistic expressions in general (For example, this artist brilliantly uses different jigsaw puzzles to create sur-

real mashups. <https://mymodernmet.com/montage-puzzle-art-tim-klein/>). While one may conceive jigsaw puzzles of more abstract form, here we will refer to puzzles as *visuals* if both their fragments (a.k.a. pieces), and the reconstructed “wholes”, are “visual”, namely if they can be understood, analyzed, and reconstructed by a visual system (and in particular, the human visual system). In practical terms, this means that both the pieces and the reconstructed whole are geometrical entities, possibly endowed with some pictorial overlay. If only the global geometric shape of the fragments is used in the process, the puzzle is called ‘*apictorial*’. If, on the other hand, pictorial information on the fragments is also used for the reconstruction, the puzzle is called ‘*pictorial*’.

While solving real-life jigsaw puzzles has occupied humans for millennia, to our best knowledge it was first introduced as a *computational* task in 1964 by Freeman and Garder (Freeman & Garder, 1964), who discussed the attributes of *apictorial* jigsaw puzzles and proposed a solver for puzzles of *unrestricted shapes*. Limited by the computation power of the time, results were of course constrained to very simple and small puzzles. In the last two decades, however, the focus in the computational literature has shifted towards puzzles of *square* pieces that must be matched into a rectangular image. Since the pieces in such puzzles are all

---

Communicated by Jiri Matas.

---

✉ Ohad Ben-Shahar  
ben-shahar@cs.bgu.ac.il

<sup>1</sup> Department of Computer Science, Ben-Gurion University of the Negev, Beer Sheva, Israel

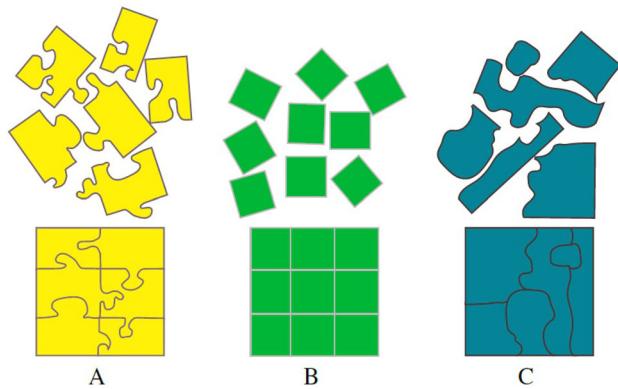
shaped identically as squares, their pictorial content becomes *the only* source of information available for the reconstruction. While starting modestly, the suggested solvers for such puzzles evolved rapidly in the past decade, and although they cannot guarantee optimal solutions, contemporary methods exist to solve “square jigsaw puzzles” of virtually any practical size.

As discussed below in the related work, the significant gap between unrestricted puzzles and square jigsaw puzzles was rarely addressed in the literature, although there are numerous methodological and applicative advantages for doing so. In this work, we attempt to do exactly that. We introduce a different puzzle formation process, and a new class of puzzles termed here *crossing cuts polygonal puzzles*, that are inspired by the celebrated Lazy Caterer’s sequence. Specifically, we consider global convex polygonal shapes (for apictorial puzzles) or convex polygonal images (for pictorial puzzles) that are sliced through by multiple straight cuts of arbitrary positions and directions, thus dividing the puzzle shape into many convex polygonal fragments. We discuss the synthesis of such puzzles, their properties with and without geometric noise, the qualitatively different reconstruction challenges they present for reconstruction, and a novel solver formulation that is based on abstracting the puzzle as a physical mechanical system. We discuss evaluation measures and present both qualitative and quantitative results on large and novel datasets that are made available to the community for future research.

## 2 Related Work

As mentioned above, the problem of puzzle solving is one where an orderless set of given fragments should be organized correctly with no overlaps to reconstruct a desired (known or unknown) global shape and possibly also with (typically unknown) pictorial content. In such cases, the pictorial data is yet another reconstruction cue whose degree of importance can vary relative to the apictorial (geometric) ones. Decades after Freeman and Garder’s seminal paper (Freeman & Gardner, 1964) the problem was proved NP-complete (Demaine & Demaine, 2007), leading the literature to focus on devising heuristics, ad-hoc methods, and computational schemes that indeed cannot guarantee optimal solutions in tractable time but nevertheless facilitate successful puzzle solving in many cases, including large scale puzzles of various types.

Broadly speaking, visual puzzles are generated by taking a coherent global (pictorial or apictorial) object and “breaking” it into a set of orderless or disorganized set of fragments. The details of this “breaking” are called in this paper the “forward” *puzzle generation process*, which sometimes incorporates additional actions before the puzzle is finalized,



**Fig. 1** The type of visual puzzle classes most studied in the literature. Shown are both the bag of pieces and the reconstructed puzzle. Sketched are only apictorial puzzles of the same global (in this case, rectangular) shape to emphasize the effect of the class on the possible geometry of the pieces. **A** Commercial toy puzzle. **B** Square Jigsaw puzzle. Note that while the sketch focuses on their geometry, these puzzles must be pictorial to avoid trivial settings. **C** Unrestricted puzzle

such as removing fragments, adding bogus ones, deforming fragments geometry, or distorting the pictorial content (what we will call “geometric noise” and “pictorial noise”, respectively). With this in mind, the types of puzzles addressed in the prior art can be categorized into three different classes of such generation processes. We call these classes “Commercial toy puzzles”, “Square Jigsaw puzzles”, and “Unrestricted puzzles” (see Fig. 1). Since puzzle reconstruction algorithms attempt to “reverse” the puzzle generation process, the classification also implies that the reconstruction process may be done differently in each class. One evident example is Square Jigsaw puzzles, which unlike their counterparts *must be* pictorial in order to escape trivial settings. In the following, we elaborate on each class, not necessarily according to their chronological order in the literature.

### 2.1 Commercial Toy Puzzles

Commercial toy puzzles, a set we denote  $\mathcal{P}_C$ , include puzzles one can buy at toy stores and designated as a leisure time activity. As this type of puzzle is specifically meant to be solvable by humans, it follows a common set of very specific constraints and rules (Goldberg et al., 2002). First, the outer border of the puzzle is rectangular. Second, the pieces in a reconstructed puzzle form a sort of rectangular grid so that all pieces except boundary ones have exactly four neighbors. Finally, pieces interlock with their neighbors by tabs (i.e., concave or convex protrusions), so that the shape of the matching tab allows a unique match.<sup>1</sup>

<sup>1</sup> It is becoming more common in recent times to find commercial jigsaw puzzles that deviate from these rules. However, we are unaware of corresponding computational literature that addresses such variations and thus ignore them here.

Although designed for humans, and very limited in their applications, Commercial Toy puzzles consume a fair share of the computational literature, and mostly after the mid-1980s. The uniqueness property suggests that a greedy approach can always solve such problems in low-order polynomial time simply by matching boundary curves. However, the need to scan the pieces and represent their boundaries numerically introduces geometric noise that leads to false positives during the search. With this in mind, the several toy puzzle solvers proposed in the literature share a common scheme. First, the pieces are classified as either border or inner pieces by analyzing their boundary and counting straight segments. Border pieces are then assembled first (just as humans would tend to do), for example by reducing the problem to the traveling salesman problem and solving it via heuristic methods (Wolfson et al., 1988). Once the border pieces are placed, the dimensions of the puzzle grid can be deduced and the inner pieces are placed in a grid using either a greedy or an exhaustive search method. Because noise could generate false positives, each piece placement involves a test for geometric violations (e.g., overlaps between pieces), a type of event that results in backtracking. Although the use of backtracking can entail exponential complexity, the shape of the tab is expected to be unique enough to make false positives rare (or even impossible), thus retaining a tractable solution process.

Given that the matching geometry is unique,  $\mathcal{P}_C$  puzzles need not contain pictorial content for a computer (or for that matter, even humans) to solve, as indeed was the case in several prior studies on the topic (Wolfson et al., 1988; Burdea & Wolfson, 1989; Webster et al., 1991; Bunke & Kaufmann, 1993; De Bock et al., 2004; Goldberg et al., 2002). That being said, pictorial extensions *do* exist, addressing the full real-life toy jigsaw puzzle challenge (except for the fact that toy puzzles usually include the reconstructed image printed on their box cover). Such pictorial toy puzzle solvers (Kosiba et al., 1994; Chung et al., 1998; Yao & Shao, 2003; Nielsen et al., 2008) can clearly utilize an additional constraint of visual coherence (e.g., continuity) across neighboring pieces to improve the accuracy of potential mates, lower the risk of false positives, and thus reduce the search space. Thus far, the biggest pictorial toy puzzle solved this way was sized at 320 pieces (Nielsen et al., 2008). Unfortunately, given the possibility of solving such puzzles perfectly in tractable time, no performance metrics (other than testing for perfect reconstruction) or statistical benchmark experimentation are typically performed.

## 2.2 Square Jigsaw Puzzles

Square Jigsaw puzzles, denoted  $\mathcal{P}_S$ , are the type of visual puzzles discussed most frequently in the last two decades. They are the simplest geometrically and based on a gener-

ation process that cuts a rectangular image into a grid of identically shaped square pieces. The problem is considerably different than the commercial toy puzzles since with identical boundaries to all pieces, the reconstruction must fully rely on the pictorial content.

Square Jigsaw puzzles also tend to share a similar algorithmic flow. First, a measure of dissimilarity between every two potential neighbors is pre-calculated. Then, the dissimilarity is used to derive neighbors' compatibility scores to represent the confidence that they should be paired. Then neighboring pieces are matched, placed, and aggregated to maximize global compatibility, either in a greedy fashion or by employing heuristics to globally search the solution space. Since state-of-the-art square-piece puzzle solving now tends to deal with rather large-scale problems, backtracking is typically avoided as the number of search paths is intractable. The many variants of this general scheme include solvers for Square Jigsaw puzzles with pieces of *known* size and piece orientation (Toyama et al., 2002; Fei et al., 2007; Zhao et al., 2007; Murakami et al., 2008; Alajlan, 2009; Cho et al., 2010; Pomeranz et al., 2011; Yang et al., 2011; Sholomon et al., 2013; Adluru et al., 2015; Andalo et al., 2016), puzzles where the orientation of the pieces is *unknown* (Gallagher, 2012; Mondal et al., 2013; Sholomon et al., 2014; Son et al., 2014; Yu et al., 2015; Son et al., 2016, 2018; Rika et al., 2019), challenges with mixed set of pieces from multiple puzzles (Gallagher, 2012; Mondal et al., 2013; Paikin & Tal, 2015; Son et al., 2016, 2018), missing pieces (Gallagher, 2012; Mondal et al., 2013; Paikin & Tal, 2015; Son et al., 2016, 2018) noisy pictorial content (Mondal et al., 2013; Brandão & Marques, 2016; Rika et al., 2019; Son et al., 2014; Yu et al., 2015; Son et al., 2018), gaps between pieces (Paumard et al., 2020; Derech et al., 2021), and even restricted deformations to the shapes of fragments (Gur & Ben-Shahar, 2017).

Indeed, earlier attempts to address the problem assumed known dimensions, known piece orientation, and even some prior knowledge regarding the solution. For example, Cho et al. (2010) used prior knowledge in the form of ground truth anchor pieces or low-resolution images of the solved puzzle. Color differences along abutting piece boundaries were used for the compatibility score and the reconstruction was based on achieving maximum likelihood for both piece compatibility and the prior knowledge. Shortly after, Pomeranz et al. (2011) were the first to solve the Square Jigsaw puzzle fully autonomously and *without* any prior knowledge (except for the puzzle dimensions) and increased the size of solvable puzzles an order of magnitude over the prior art to puzzles of thousands of pieces. Among the contributions were an iterative greedy approach, prediction of pictorial content across piece boundary for the dissimilarity metric, and the introduction of the *best-buddies* concept that influenced much of the later works and served as a precursor for the employment

of loopy constraints to reduce the search space (see below). Sholomon et al. (2013) introduced a different type of solver based on a genetic algorithm and the best-buddies idea, a combination that proved successful in solving even larger puzzles, exceeding the likely capacity of human solvers.

Pieces of unknown orientation add another layer of complexity to the Square Jigsaw puzzle problem, as the number of possible configurations increases by a factor of  $4^K$  (for puzzles of  $K$  pieces). Gallagher (2012) was the first to tackle such puzzles while introducing a gradient-based dissimilarity score and a greedy spanning tree-based solver. Yu et al. (2015) also dealt with unknown orientation by using a global optimization in the form of relaxed linear programming, and Sholomon et al. (2014) modified their original genetic algorithm to also handle unknown orientations in puzzles with a very large number of pieces.

Noise in the pictorial data increases the difficulty of the problem since the dissimilarity metric becomes less reliable and false matches are even more likely, prompting some studies to seek more robust compatibility metric (e.g., Mondal et al., 2013; Brandão and Marques, 2016; Rika et al., 2019). Toward that end, Mondal et al. (2013) combined two previously used dissimilarity metrics while Brandão and Marques (2016) measured the dissimilarity using a heat-based affinity measure that utilizes a pixel environment larger than the piece boundary. Rika et al. (2019) used deep learning as a mechanism to assess the compatibility between pairs of pieces, taking the whole piece as input. Taking a different approach, Yu et al. (2015) and Son et al. (2014, 2018) dealt with noise by applying a reconstruction algorithm that demands a consensus in an environment larger than the immediate neighbors of each piece. The former used a relaxed linear programming algorithm that rewards global piece consensus while the latter introduced the notion of loopy constraint - a requirement for compatibility consensus in loops of pieces.

Present-day state-of-the-art solvers for the Square Jigsaw puzzle can solve puzzles with over 20,000 pieces (Sholomon et al., 2014). For historical reasons, most of the prior art experimented with square pieces of  $28 \times 28$  pixels in size in order to allow enough pictorial data while measuring the compatibility of pieces. However, recent works now extend this convention to pieces as small as  $7 \times 7$  pixels (Son et al., 2014, 2018).

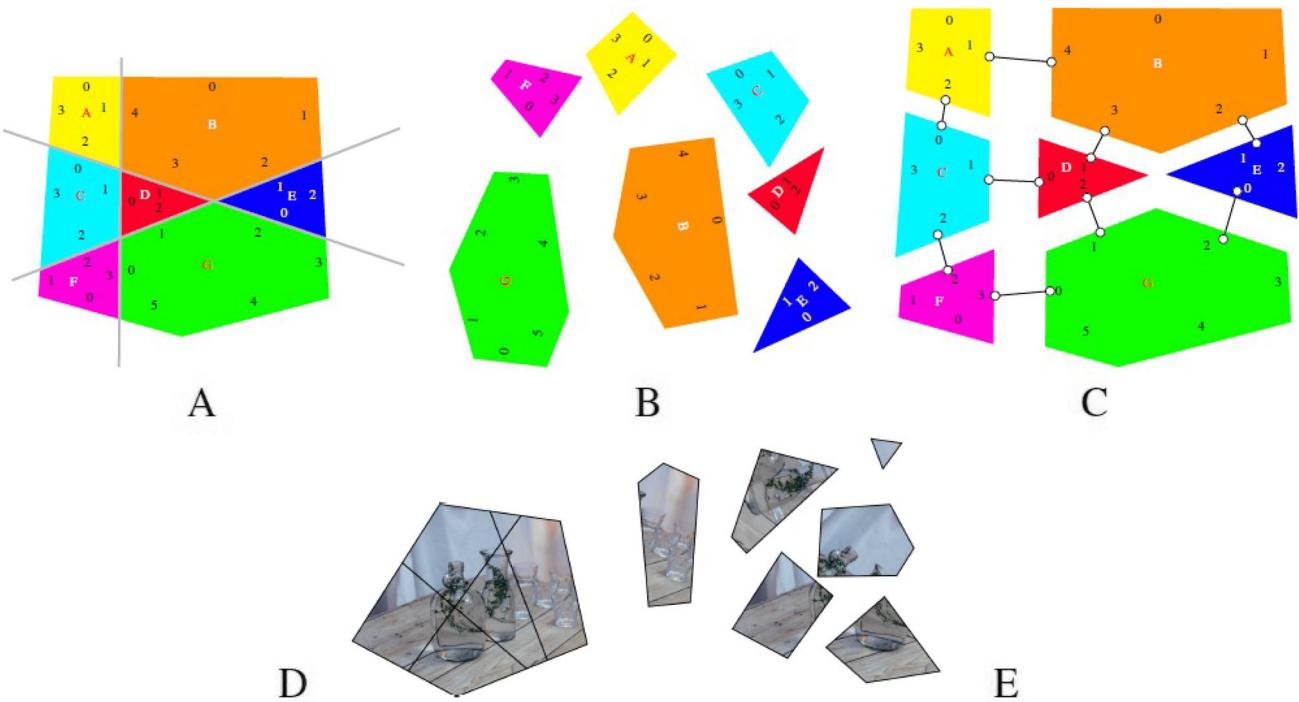
### 2.3 Unrestricted Puzzles

Unrestricted puzzles, the class we denote  $\mathcal{P}_U$ , contain puzzles that do not have a formal generation process or constraints on the shape of their pieces. In such puzzles, the representation of the pieces is far more complex (than  $\mathcal{P}_S$  or  $\mathcal{P}_C$ ), they can be matched to an arbitrary number of neighbors abutting arbitrary sections of their boundary, and the reconstruction of such 2D puzzles can be described as a general planar

adjacency graph of arbitrary maximal degree (unlike the degree 4 that characterizes the reconstructions of 2D puzzles in  $\mathcal{P}_C$  or  $\mathcal{P}_S$ ). Despite these complications, and somewhat unexpectedly, the very first work on computational puzzle solving (Freeman & Garder, 1964) belongs to this class.

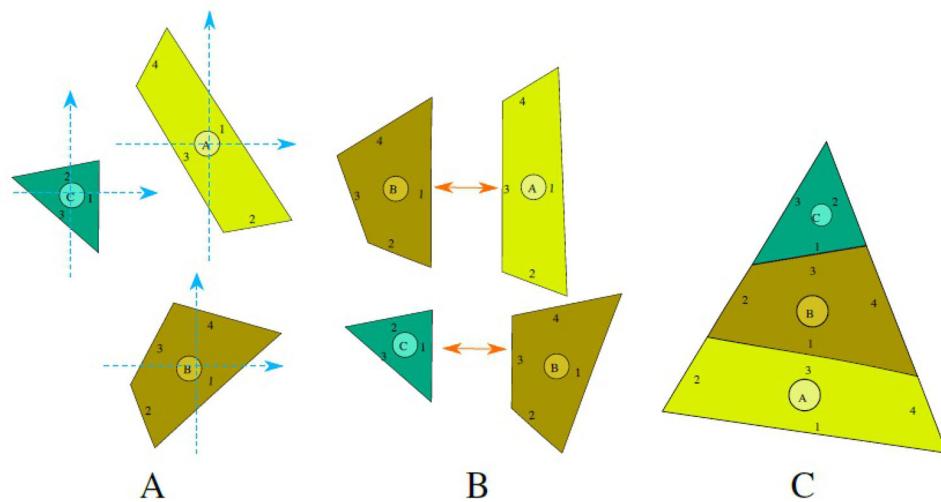
*Apictorial* unrestricted puzzle solvers typically use curve matching to find potential matching pieces (Freeman & Garder, 1964; Radack & Badler, 1982; Kong & Kimia, 2001). As mentioned above, the first to explore such an approach (or computational puzzle solving in general) were Freeman and Garder (1964), who also introduced a solver capable of dealing with a large variety of piece shapes and junction types. Their solver matches curves using a chain encoding scheme and then assembles the puzzle using a greedy algorithm that backtracks on errors, an exhaustive scheme possible only because of the very small scale problems considered. The solver tried to reconstruct coherently around junctions, thus seeking neighbors with loopy consensus, perhaps leading the way to the future use of loopy constraints in the field (Son et al., 2014). Owing to the small computational resources of the time, the single evaluation on a 9-piece puzzle of highly discriminated pieces did not permit later experimental comparison to contemporary contributions. Forty years later, Kong and Kimia (2001) used a coarse-to-fine approach to curve matching and a greedy merging of piece triplets and backtracking upon spatial overlap. While the geometrical treatment was significantly more rigorous, here too the experimentation was limited to few puzzles of up to 25 pieces, most of which had near-convex low-order polygonal shapes. An interesting question that emerges is whether or not such data represent realistic scenarios, at least on average. Of course, it is difficult to address such questions without some formal puzzle generation model, an observation that is key to the suggested research in this paper (see below).

Extending the basic computational flow of the above, solvers for unrestricted *pictorial* puzzles (Tsamoura & Pitas, 2009; Liu et al., 2011; Makridis & Papamarkos, 2006; Sağıroğlu & Erçil, 2010; Zhang & Li, 2014; Le & Li, 2019) use the pictorial content as well as geometrical boundaries to match pieces and reconstruct the puzzle. Sağıroğlu and Erçil (2010) used an extrapolation method to approximate the content of the pictorial data in a band around each piece. This allowed for a pictorial score by comparing the extrapolated bands to the content of prospective neighbors. Then, the Fourier transform translation property was used to find an alignment that maximizes the correlation between pieces while satisfying the geometrical constraints. The reconstruction itself was done in a greedy fashion, starting from a random configuration and improving the global score one piece at a time. To escape local minima, the reconstruction process was restarted multiple times with different random seed configurations. The experimental evaluation was limited to assemblies of 21 pieces, most of which had very distinctive



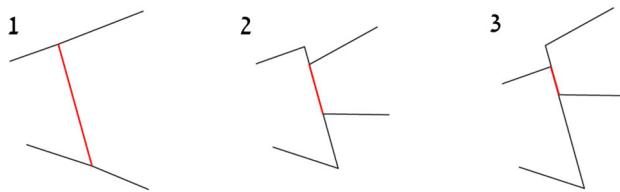
**Fig. 2** The elements of a crossing cuts puzzle. **A** The puzzle is created by cutting a convex polygon using multiple (here 3) straight lines. **B** The puzzle problem constitutes of an un-ordered and arbitrarily transformed set of pieces. Note that different pieces may vary vastly in size (e.g., compare pieces  $p_B$  and  $p_D$ ). **C** The mating graph matches pairs of edges of two different pieces and here

it includes  $\{e_A^1, e_B^4\}, \{e_A^2, e_C^0\}, \{e_B^2, e_E^1\}, \{e_B^3, e_D^1\}, \{e_C^1, e_D^0\}, \{e_C^2, e_F^2\}, \{e_D^2, e_G^1\}, \{e_E^0, e_G^2\}, \{e_F^3, e_G^0\}\}$ . Note that pieces end up having different numbers of edges and thus different numbers of mates. **D** A pictorial crossing cuts puzzle is one with an image (or any other visual content) covering the polygon that is being cut. **E** Once cut and shuffled, the pictorial pieces represent the puzzle to be solved



**Fig. 3** The representation of a crossing cuts puzzle and its solution, illustrated here for the simplest 2 cuts (and 3 pieces) puzzle. **A** Each piece  $\{p_1, p_2, p_3\}$  is represented by its vertices and edges in some arbitrary Euclidean coordinate system (which conveniently may be centered at the center of mass). **B** Each mating pairs two edges of two differ-

ent pieces. In our case it takes just the matings  $\{e_A^3, e_B^1\}, \{e_B^3, e_C^1\}\}$ . **C** After the application of the Euclidean transformations ( $t_i, R_i \forall i = 1..3$ , the puzzle is reconstructed (up to some global Euclidean transformation)

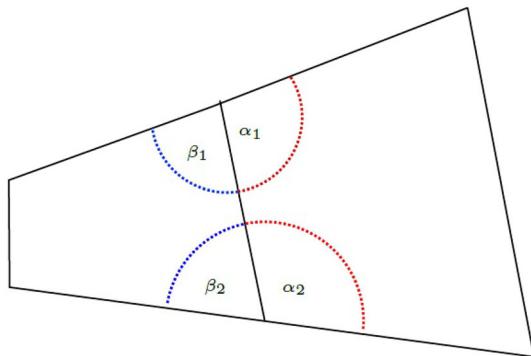


**Fig. 4** In (generic) crossing cuts puzzles only matings of type 1 are allowed, while configurations of type 2 or 3 are prohibited. In particular, mates must be of equal length and overlap

boundaries. A related approach with fragment extrapolation for registration of neighboring candidates was proposed by Derech et al. (2021).

Recently, Le and Li (2019) introduced a novel approach for fragment matching using a Convolutional Neural Network that utilizes both boundary shape and pictorial data with a hierarchical loops approach for the reconstruction. The solver was tested successfully on puzzles of up to 400 pieces, significantly bigger than prior work. Moreover, evaluation was performed quantitatively and on a relatively large number of puzzle problems, two advances over the prior art in the unrestricted puzzle literature. That being said, the test data published alongside the paper contains a relatively constrained shape for the pieces as all of them were roughly perturbed rectangles.

It should be mentioned that much work on (typically apictorial) unrestricted puzzles is performed in the archaeological domain, where computational tools have generated a revolution (Grosman, 2016) and visual puzzles are typically not 2D, but either 2.5D (“thick” 2D manifolds) or 3D. For their different focus we omit a detailed review of that literature, referring the reader to selected references from the last two decades (Papaioannou et al., 2001; Papaodysseus et al., 2002; Papaioannou & Karabassi, 2003; Koller & Levoy, 2006; Huang et al., 2006; Willis & Cooper, 2008; Kleber & Sablatnig, 2009; Mellado et al., 2010; Toler-Franklin et al., 2010; Castañeda et al., 2011; Funkhouser et al., 2011; Oxholm & Nishino, 2011; Brown et al., 2012; Shin et al., 2012; Palmas et al., 2013; Pintus et al., 2014; Mavridis et al., 2015; Andaló et al., 2016; Brandão & Marques, 2016; Li et al., 2020; Ylmaz & Nabiyev, 2023; Markaki & Paniagiotakis, 2023). That being said, the computational flow in most of these studies is similar and constitutes several steps, including scanning the artifacts to point clouds, processing these point clouds into meshes, segmenting the meshes to facets, and extracting geometrical features either on the facets or their boundary curves. Facets of different fragments are then registered through the raw geometrical point cloud data (e.g., using ICP) or the processed features. The final stage of combining the pairwise matches into a global assembly is done with a variety of methods, often including human intervention.



**Fig. 5** The mate angle constraint dictates  $\alpha_1 + \beta_1 = \alpha_2 + \beta_2 = \pi$

## 2.4 A Missing Link in the Puzzle Solving Chain?

The scientific background covered above suggests that even though the basic problem is one, research into visual puzzle solving has been conducted in “parallel tracks” that affected progress in ways that are not necessarily optimal. Related to this are observations like the following

- Solving commercial jigsaw puzzles computationally is very anecdotal in terms of its application value and serves mostly as an intellectual challenge.
- Markedly inconsistent with the popularity of  $\mathcal{P}_S$  in the literature, there are almost no real-life applications that can be abstracted as strict Square Jigsaw puzzles. For example, although most studies of Square Jigsaw puzzles cite archaeology as a possible application domain, archaeological puzzles are virtually never square, and to our best knowledge there is exactly one case in the computational archeology literature for which Square Jigsaw puzzle solvers may be relevant (Brandão & Marques, 2016).
- Unrestricted puzzles do have the potential to serve numerous applications, but their unrestricted generation model makes it difficult to study them rigorously, obtain useful insights, or allow any type of guarantees related to the solution process. Rigorous analysis means answering questions about puzzle properties (e.g., the expected number of pieces in a puzzle, the statistical properties of the area of puzzle pieces, etc.) and about the developed algorithms (e.g., how many false positive matches may occur to determine the worst time complexity of a particular algorithmic step). Such understanding of the problem or the solutions is typically missing for this class.
- The fact that  $\mathcal{P}_U$ ,  $\mathcal{P}_S$ , and  $\mathcal{P}_C$  are so different makes it practically impossible to apply tools (e.g., solvers or performance measures) from one class to another, even though both  $\mathcal{P}_S$  and  $\mathcal{P}_C$  are subsets of  $\mathcal{P}_U$  (and under certain relaxation one can even view  $\mathcal{P}_S$  as a subset of  $\mathcal{P}_C$ ). This gap manifests itself not only at the level of algorithms, but also in the representation of the prob-

lem (e.g., for I/O), data structures and formats used, and operational assumptions.

Generally speaking, a trade-off emerges between how constrained a puzzle class is, how relevant it is for real-life applications, how rigorous the analysis it permits, and how applicable are its solvers to other classes. Our present work tries to address this trade-off by suggesting a new puzzle generation model that is more restricted than  $\mathcal{P}_{\mathcal{U}}$  puzzles and thus allows rigorous analysis while being much more general than  $\mathcal{P}_{\mathcal{S}}$  and thus extends the applicability and usability to real-life challenges. In general, such an approach may refer to a new class of puzzles one may term *restricted modeled puzzles*, where some formal restrictions are defined for the puzzle generation process in a way that rigorous analysis is still possible while the range of applications remains viable. We hope that the present research will encourage the community to explore this direction further.

### 3 Model Formulation

Recall that the pieces (or fragments) of square jigsaw puzzles are all identical in shape, a setup that drives all reconstruction decisions to be solely pictorial. However, real-world puzzles usually have pieces of a more general form (e.g., Shin et al., 2012), leading to a different set of challenges. Here we try to formulate a new class of puzzles that is both general enough for more real-world cases and yet formal enough for rigorous analysis and exploration. We call this class the *crossing cuts puzzles*.

A crossing cuts puzzle is created by cutting through a convex polygon<sup>2</sup> with  $a \in \mathbb{N}$  arbitrary (random) straight cuts  $Cuts = \{c_1, \dots, c_a\}$ . The pieces of such puzzles are thus convex polygons where every piece (except border pieces) has a single neighbor along each of its edges. This puzzle generation model is inspired by the procedure that produces the Lazy Caterer's sequence<sup>3</sup> (Wetzel, 1978; Yaglom & Yaglom, 1987), but unlike the latter, in our case, the cuts are completely arbitrary and there is neither guarantee nor desire to maximize the number of pieces.<sup>4</sup> This proposed model can also be used to address and simulate the *realistic* and/or

<sup>2</sup> We note that one could apply crossing cuts to an arbitrary *non polygonal* convex shape, but then the curved edges would serve as a major clue for reconstruction, a relief we preferred to avoid in this work.

<sup>3</sup> Each number  $f(n)$  in the Lazy caterer's sequence, also known as the central polygonal numbers, is the *maximal* number of cake pieces a caterer can obtain by cutting the cake (or more abstractly, a disk) exactly  $n$  cuts. To do so a caterer must be “lazy” since the cuts cannot all intersect the center of the disk, as is usually done while slicing cakes.

<sup>4</sup> In some sense, the caterer in our case is even “lazier” than the “lazy caterer”, as she does not need to take measures to choose her cuts to maximize the number of pieces.

*physical* generation of puzzles already discussed in literature, such as square jigsaw puzzles. For example, while using a pair of scissors or a ruler and a blade, one can create a real-life square jigsaw puzzle by cutting a picture, where cuts are not strictly parallel or equidistant due to human error or lack of sensitivity. The result will be a *noisy* square jigsaw puzzle (see below), which is a crossing cut puzzle.

Geometrically, square piece puzzles are indeed a very special case of crossing cuts puzzles and thus the latter require a more general mechanism to represent them. Towards that end, and inspired by Freeman and Gardner (1964), we define the *mating graph* to be a planar graph whose nodes are the edges of the puzzle pieces and whose links,<sup>5</sup> dubbed *matings*, represent immediate neighborhood relationship. The connected pieces will be called *neighbors* or *neighboring pieces* while the edges matched by a mating will be called *mates*. Note that in the ideal case, when no geometric noise is present, a mating in the solved puzzle represents two overlapping mates with identical lengths.

Unlike in square piece (and also commercial toy) jigsaw puzzles, which have a constant number of neighbors for each piece (except boundary pieces), the mating graph of a crossing cuts puzzle is more general since the number of matings a piece can have varies (see Fig. 2A–C). Moreover, the number of possible Euclidean transformations of the pieces of crossing cuts puzzles adds additional complexity, since unlike for square or toy puzzles, it is infinite in cardinality and selected from a continuous range. Hence, on the one hand, the representation of the puzzle must account for these new degrees of freedom. On the other hand, the geometrical shape of the pieces provides more information that is not present in the square jigsaw problem and may facilitate reconstruction algorithms that rely only on the shape of the pieces. Just as in any other type of puzzles (see Sect. 2), an *apictorial* crossing cuts puzzle is one whose initial polygon contains no visual content (Fig. 2A, B) while a *pictorial* crossing cuts puzzle is generated from a polygon covered with an image. (Fig. 2D, E). In this paper, we start our analysis with apictorial puzzles and gradually incorporate pictorial aspects while arguing that under most realistic scenarios, pictorial content is critical for successful and efficient crossing cuts puzzle solvers.

To facilitate a constructive discussion towards computational solutions to our problem, one needs to differentiate the representation of the puzzle itself (in the sense of the riddle to solve) and its possible solutions. A crossing cuts puzzle is thus a representation of the unordered puzzle pieces after the complete polygon was cut (Fig. 2B, E). Formally, let  $P = \{p_1, \dots, p_n\}$  be a set of *pieces*, where each  $p_i$  is a convex polygon of  $N_i \geq 3$  vertices. By convention, we order

<sup>5</sup> To avoid terminological confusion, we use the term ‘links’ for the edges of the mating graph, while we reserve the term ‘edges’ for the boundary segments of puzzle pieces.

these vertices clockwise around the polygon's center of mass and denote them

$$V_i = \{\vec{v}_i^1, \vec{v}_i^2, \dots, \vec{v}_i^{N_i}\}.$$

Correspondingly we label the piece edges between these consecutive vertices by

$$E_i = \{e_i^1, e_i^2, \dots, e_i^{N_i}\} = \{(\vec{v}_i^1, \vec{v}_i^2), (\vec{v}_i^2, \vec{v}_i^3), \dots, (\vec{v}_i^{N_i}, \vec{v}_i^1)\}.$$

A *solution* to a crossing cuts puzzle requires positioning each piece in its "correct" position relative to all other pieces, and while this requires the determination of a Euclidean transformation (position and rotation) for each piece, in practice this will first require to resolve the neighborhood relationships between the pieces, i.e., the "correct" mating graph. An algorithm to obtain a solution thus needs to determine both

- i. the pairwise matings  $M = \{m_1, \dots, m_{|M|}\}$  of all pieces, i.e., all unordered pairs of edges  $m_q = \{e_i^j, e_k^l\}$  of two different pieces that should be matched (and in an ideal setting, truly overlap) in order to reconstruct the puzzle, and
- ii. the 2D Euclidean transformation of each piece  $p_i$ , from its given input representation  $V_i$  to the one in the reconstructed puzzle. The transformation of piece  $p_i$  involves a translation  $t_i \in \mathcal{R}^2$  and a rotation  $R_i \in \mathcal{S}^1$ . With the rotation typically represented by an orthonormal matrix  $R_i \in \mathcal{R}^{2 \times 2}$ , the pose of the piece in the reconstructed puzzle will be

$$p'_i = \{R_i \cdot \vec{v}_i^1 + \vec{t}_i, R_i \cdot \vec{v}_i^2 + \vec{t}_i, \dots, R_i \cdot \vec{v}_i^{N_i} + \vec{t}_i\}$$

Figure 3 illustrates both the puzzle and the aspects of its solution as just discussed. It should be noted that while the mating graph may have only one "correct" solution, the Euclidean transformations of the pieces can be correct up to some global Euclidean transformation that describes a rigid motion of the entire reconstructed puzzle.

## 4 Mating Constraints and a Greedy Solver

With the crossing cuts puzzles defined as above, and assuming no noise, idealized infinite precision in the representation of the geometrical objects, and random uniform distribution of the crossing cuts themselves, it is immediate to observe that the probability of (i) more than two crossing cuts to meet at a point, and (ii) having more than two edges with *identical* lengths, is nil in both cases. These properties of the *generic* (i.e., non-accidental) puzzle entail two key constraints for the formation of plausible matings:

**$C_1$ : The mate length constraint** Since plausible matings should match complete edges, it follows that they must match mates of the very same length (see Fig. 4).

**$C_2$ : The mate angle constraint** Since the mates of plausible matings have vertices emerging from just 2 crossing cuts, their adjacent edges must form a straight line (which simply overlaps with two different crossing cuts). It follows that the two pairs of adjacent angles of the neighboring pieces must complete to  $\pi$ , i.e., be supplementary (see Fig. 5).

In the following, we will refer to the mating constraints also as predicates, i.e.,

$$\forall i \in \{1, 2\} \quad C_i(e_i^j, e_k^l) = \text{true} \Leftrightarrow e_i^j \text{ and } e_k^l \text{ satisfy } C_i.$$

Clearly, the constraints just outlined entail the simple and *greedy* solver in Algorithm 1 that progressively moves pieces from the set  $U$  of unassigned pieces to the set  $R$  of the reconstructed assembly, while forming the mating graph  $G_M$ . This simple algorithm uses only constraint  $C_1$ , but versions using  $C_2$  are possible also. Either case, these greedy schemes are clearly sound, complete, and tractable, they do not need pictorial information and they can solve both apictorial and pictorial puzzles using the geometric information alone. As we discuss shortly, all this changes fundamentally once we introduce geometric noise to the puzzle.

---

**Algorithm 1:** A basic greedy algorithm for solving crossing cuts puzzles under ideal conditions

---

```

 $G_M \leftarrow \emptyset;$ 
 $R \leftarrow \text{an initial random piece from } P;$ 
 $U \leftarrow P \setminus R;$ 
while  $U \neq \emptyset$  do
    Pick an edge  $e_i^j$  of a piece  $p_i \in R$  that is not yet part of a mating;
    if (there is an edge  $j_k^l$  of some piece  $p_l \in U$  that has the same length as  $e_i^j$ ) then
         $R \leftarrow R \cup \{p_l\};$ 
         $U \leftarrow U \setminus \{p_l\};$ 
        Compute the Euclidean transformation that places  $p_l$  properly in the reconstructed assembly;
         $M_p \leftarrow \text{matings of } p_l \text{ edges that overlap edges in } R;$ 
         $G_M \leftarrow G_M \cup M_p;$ 
    else
        do nothing (because the edge  $j_k^l$  is a border edge);
    Return  $G_M$  and the corresponding Euclidean transformations of all pieces.

```

---

## 5 Noisy Crossing Cuts Puzzles

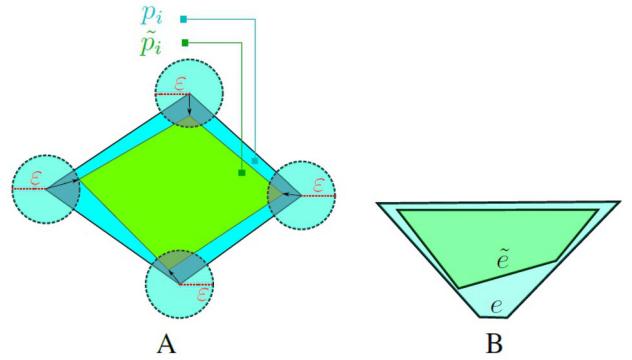
Real-world data, its measurement, or its representation, are never completely accurate. Even if the measurement or the digital representation of the pieces were devoid of errors, *real life* crossing cuts puzzles (or geometric puzzles in general) may incorporate deformations to the shapes of the pieces, as well as to their visual content (in pictorial puzzles). In fact, geometric noise also affects how pictorial information can be leveraged, even if no pictorial noise is present. For this reason we begin by formalizing the geometric noise, and extend the discussion to pictorial puzzles only later.

Clearly, geometric noise can be modeled in many different ways, though one particularly appealing is material degradation, and thus piece shrinkage, a process clearly relevant for applications involving physical pieces (e.g. in archaeology). To incorporate material degradation without escaping the crossing cuts framework, we model this deformation process by preserving the number of vertices of each piece, but shifting each of them *inward* by a random distance that is distributed (in our case, uniformly) in a given range. We note that the particular distribution of such noise may affect certain statistical properties (see Chapter 7 below), but otherwise it is less significant for the reconstruction algorithm discussed later.

### 5.1 Noise Formulation

Formally, a vertex  $\vec{v}_i^j$  of piece  $p_i$  is perturbed inwards by a distance  $\vec{\epsilon}_i^j$  that is bounded by some maximal noise level  $\varepsilon \geq 0$ . It is convenient to set that bound relative to a reference value that is based on the puzzles' geometrical properties. In our case, we use the puzzle diameter  $D$ , i.e., the distance between the furthest vertices. Formally, we define a relative bound  $\xi$  that sets the absolute noise level at  $\varepsilon = \xi \cdot D$ , and let  $\tilde{\xi}$  be the corresponding noise level relative to the average edge length (to be derived later). An original piece  $p_i = \{\vec{v}_i^1, \dots, \vec{v}_i^{N_i}\}$  ends up as the  $\varepsilon$ -noisy piece  $\tilde{p}_i = \{\vec{v}_i^1 + \vec{\epsilon}_i^1, \dots, \vec{v}_i^{N_i} + \vec{\epsilon}_i^{N_i}\}$  where the noise magnitude  $\|\vec{\epsilon}_i^j\| \sim U(0, \varepsilon)$  and the noise direction is selected from the sector originating from  $\vec{v}_i^j$  towards the two nearby vertices, namely  $\angle \vec{\epsilon}_i^j \sim U(\angle(\vec{v}_i^{j-1} - \vec{v}_i^j), \angle(\vec{v}_i^{j+1} - \vec{v}_i^j))$ . This limits the random perturbation angle of  $\vec{\epsilon}_i^j$  and constrains it to be *inward*, i.e., an erosion-like process “into” the material. Figure 6A illustrates how such noise could affect the shape of a quadrilateral (4-edges) piece.

Naturally, the incorporation of noise affects the validity of our constraints on mating. In particular, the number of potential mates now increases drastically and far from



**Fig. 6** The effect of noise on edge length. **A** Each of the vertices of a piece  $p_i$  is perturbed inwards along a uniformly distributed direction  $\angle \vec{\epsilon}_i^j$  and as far as a uniformly distributed distance  $\|\vec{\epsilon}_i^j\|$  to create the  $\varepsilon$ -noisy piece  $\tilde{p}_i$ . **B** A case where edge  $e$  increases in size after the application of noise, even though all the vertices collapsed inwards to end up as  $\tilde{e}$ . Clearly,  $\|\tilde{e}\|$  is bounded by  $\|e\| + 2\varepsilon$

uniqueness, and the implications on a reconstruction algorithm are paramount. In this sense,  $C_1$  and  $C_2$  must be revised, as discussed next.

### 5.2 $\tilde{C}_1$ : Mate Length Constraint Under Noise

Since now plausible matings should match edges that have been perturbed differently, the mate length constraint must be relaxed to accommodate these independent perturbations. Let  $e$  and  $e'$  be the matching edges before applying the noise while  $\tilde{e}$  and  $\tilde{e}'$  denote their corresponding  $\varepsilon$ -noisy edges. It follows that  $\tilde{e}$  and  $\tilde{e}'$  might have respective lengths  $\tilde{L}$  and  $\tilde{L}'$  that satisfy

$$|\tilde{L} - \tilde{L}'| \leq 4\varepsilon . \quad (1)$$

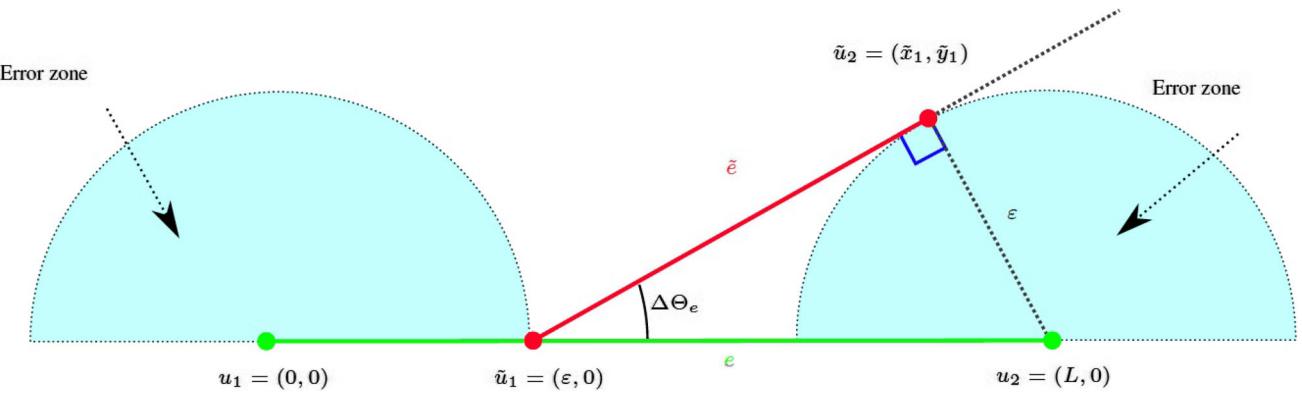
The maximum error ( $4\varepsilon$ ) can occur when one of the edges is shortened by  $2\varepsilon$  and the other is lengthened by  $2\varepsilon$ . Figure 6B exemplifies how edges may become longer even though the deformation represents the *erosion* of material.

### 5.3 $\tilde{C}_2$ : Mate Angle Constraint Under Noise

While it is clear that vertices of neighboring pieces may not meet if either sustains noise, and thus may no longer be expected to generate two supplementary angles in a strict way, one can still bound the deviation from that ideal behavior. To do so we first analyze the effect of noise on the degree of rotation of any single edge relative to its noiseless configuration and then leverage that result for the desired bound on the angles of mating edges under noise.

#### i. Bound on the rotation of a single $\varepsilon$ – noisy edge

Let  $e = (\vec{u}_1, \vec{u}_2)$  be an edge (of some puzzle piece)



**Fig. 7** If the “clean” edge  $e$  (in green) stretches (w.l.o.g) from  $u_1 = (0, 0)$  to  $u_2 = (L, 0)$ , the vertices of the  $\varepsilon$ -noisy edge must lie inside the corresponding error zones (in cyan). When considering the angle of the  $\varepsilon$ -noisy edge  $\tilde{e}$  (in red), the worst case occurs when one of the vertices

(say,  $u_1$ ) is only perturbed horizontally by  $\varepsilon$ , while the other (say,  $u_2$ ) is perturbed to maximize the rotation, i.e., to a point  $\tilde{u}_2 = (\tilde{x}, \tilde{y})$  that makes  $\tilde{e}$  tangent to the error zone. This bound is expressed in Eq. 2 (Color figure online)

with coordinates  $\vec{u}_1 = (x_1, y_1), \vec{u}_2 = (x_2, y_2)$  and size  $\|\vec{u}_1 - \vec{u}_2\| = L$ , and assume (without loss of generality) that this edge is aligned with the origin and the  $X$  axis of some reference coordinate frame and thus stretches from  $\vec{u}_1 = (0, 0)$  to  $\vec{u}_2 = (L, 0)$ . The orientation of this edge is of course  $\angle e = 0^\circ$ , as illustrated by the green edge in Fig. 7.

Let us now denote by  $\tilde{e} = (\tilde{\vec{u}}_1, \tilde{\vec{u}}_2) = ((\tilde{x}_1, \tilde{y}_1), (\tilde{x}_2, \tilde{y}_2))$  the same edge after applying the noise. Except for accidental cases, the orientation  $\angle \tilde{e}$  will be different than  $\angle e$ , as was already exemplified in Fig. 6. Let  $\Delta\Theta_e(L, \varepsilon)$  be the bound on the difference between these two orientations over all possible  $\varepsilon$ -noisy edges  $\tilde{e}$ , i.e., over all combinations of the noisy vertices  $(\tilde{\vec{u}}_1, \tilde{\vec{u}}_2)$  that are possible under the noise model. In our case,

$$\Delta\Theta_e(L, \varepsilon) = \max_{\tilde{e}} |\angle \tilde{e} - \angle e| = \max_{\tilde{e}} |\angle \tilde{e}| .$$

To obtain the maximal (i.e. worst case) orientation change  $\Delta\Theta_e$  while the vertices of  $\tilde{e}$  remain in their respective error zones (cyan semi-disks in Fig. 7), it is needed to perturb one of the vertices only horizontally while the other is perturbed vertically as much as possible. This happens when  $\tilde{e}$  becomes tangent to the error zone as shown in Fig. 7 and thus the bound is:

$$\Delta\Theta_e(L, \varepsilon) = \begin{cases} \arcsin\left(\frac{\varepsilon}{L-\varepsilon}\right) & L > 2\varepsilon \\ \infty & L \leq 2\varepsilon \end{cases} . \quad (2)$$

Note that “short” edges ( $L \leq 2\varepsilon$ ) are special since the error zones intersect and thus the  $\varepsilon$ -noisy edge might take arbitrary orientation or simply vanish altogether. In

these cases, we set the bound to infinity ( $\infty$ ) to represent the fact that the angle constraint cannot contribute useful information and thus cannot be employed constructively. In practice, a finite value of  $\frac{\pi}{2}$  (the bound of arcsin) can serve our purpose equally well.

Equation 2 requires the length of the original (“clean”) edge  $L$ , but in practice only  $\tilde{L}$  can be measured. However, following the same arguments behind constraint  $\tilde{C}_1$  (Sect. 5.2), it holds that  $L \geq \tilde{L} - 2\varepsilon$  and this lower bound can be used as a worst case. We therefore conclude that an  $\varepsilon$ -noisy edge  $\tilde{e}$  with length  $\tilde{L}$  might be rotated relative to the original “clean” edge no more than

$$\Delta\Theta_e(L, \varepsilon) \leq \begin{cases} \arcsin\left(\frac{\varepsilon}{\tilde{L}-3\varepsilon}\right) & \tilde{L} > 4\varepsilon \\ \infty & \tilde{L} \leq 4\varepsilon \end{cases} . \quad (3)$$

## ii. Bound on the angle difference of two corresponding mates

Let  $e$  and  $e'$  be two “clean” mates and denote the corresponding lengths of the edges *before*, *at*, and *after* these mates as  $L_{-1}, L_0, L_1$  and  $L'_{-1}, L'_0, L'_1$ , respectively, as illustrated in Fig. 8A. Let  $\alpha_1, \beta_1$  and  $\alpha_2, \beta_2$  be the two pairs of supplementary angles these mates form with their adjacent edges at their vertices, also as illustrated in Fig. 8A. Recall that the mate angle constraint  $C_2$  dictates that

$$\alpha_1 + \beta_1 = \alpha_2 + \beta_2 = \pi .$$

Let  $\tilde{\alpha}_i, \tilde{\beta}_i, i \in \{1, 2\}$  be the angles corresponding to  $\alpha_i, \beta_i$  after applying the noise, as shown in Fig. 8B. It is clear that the bound on how different  $\tilde{\alpha}_i, \tilde{\beta}_i$  from their “clean”

versions  $\alpha_i, \beta_i$  is determined by the maximal change of orientation of each of their constituent rays (i.e., edges), as expressed in Eqs. 2,3. We thus get

$$\begin{aligned} |\alpha_1 - \tilde{\alpha}_1| &\leq \Delta\Theta_e(L_0, \varepsilon) + \Delta\Theta_e(L_{-1}, \varepsilon) \\ |\alpha_2 - \tilde{\alpha}_2| &\leq \Delta\Theta_e(L_0, \varepsilon) + \Delta\Theta_e(L_1, \varepsilon) \\ |\beta_1 - \tilde{\beta}_1| &\leq \Delta\Theta_e(L'_0, \varepsilon) + \Delta\Theta_e(L'_{-1}, \varepsilon) \\ |\beta_2 - \tilde{\beta}_2| &\leq \Delta\Theta_e(L'_0, \varepsilon) + \Delta\Theta_e(L'_1, \varepsilon) \end{aligned}$$

Combining with the mate angle constraint we obtain

$$\begin{aligned} |\pi - \tilde{\alpha}_1 - \tilde{\beta}_1| &\leq \Delta\Theta_e(L_0, \varepsilon) + \Delta\Theta_e(L_{-1}, \varepsilon) + \\ &\quad \Delta\Theta_e(L'_0, \varepsilon) + \Delta\Theta_e(L'_{-1}, \varepsilon) \\ |\pi - \tilde{\alpha}_2 - \tilde{\beta}_2| &\leq \Delta\Theta_e(L_0, \varepsilon) + \Delta\Theta_e(L_1, \varepsilon) + \\ &\quad \Delta\Theta_e(L'_0, \varepsilon) + \Delta\Theta_e(L'_1, \varepsilon), \end{aligned}$$

and finally we apply the bound in Eq. 3 to reflect the fact that the true edge lengths are unknown.  $\tilde{C}_2$ , the final mate angle constraint under noise thus incorporates the following two inequalities

$$\begin{aligned} |\pi - \tilde{\alpha}_1 - \tilde{\beta}_1| &\leq \Delta\Theta_e(\tilde{L}_0 - 2\varepsilon, \varepsilon) + \Delta\Theta_e(\tilde{L}_{-1} - 2\varepsilon, \varepsilon) \\ &\quad + \Delta\Theta_e(\tilde{L}'_0 - 2\varepsilon, \varepsilon) + \Delta\Theta_e(\tilde{L}'_{-1} - 2\varepsilon, \varepsilon) \\ |\pi - \tilde{\alpha}_2 - \tilde{\beta}_2| &\leq \Delta\Theta_e(\tilde{L}_0 - 2\varepsilon, \varepsilon) + \Delta\Theta_e(\tilde{L}_1 - 2\varepsilon, \varepsilon) \\ &\quad + \Delta\Theta_e(\tilde{L}'_0 - 2\varepsilon, \varepsilon) + \Delta\Theta_e(\tilde{L}'_1 - 2\varepsilon, \varepsilon). \end{aligned}$$

To conclude this analysis, and similar to the mating constraints in the “clean” case, we may refer to the noisy mating constraints as predicates:

$$\forall i \in \{1, 2\} \quad \tilde{C}_i(e_i^j, e_k^l) = \text{true} \Leftrightarrow e_i^j \text{ and } e_k^l \text{ satisfy } \tilde{C}_i.$$

#### 5.4 Noise-Induced Erased Pieces

An inevitable consequence of applying erosion to a puzzle with pieces of various sizes is the potential risk of piece disappearance. Not unlike in the physical world, relatively smaller pieces are at greater risk of being completely eroded and thus practically disappear. In practice, in our noise model, this happens if the random inward perturbation applied to a vertex pushes it beyond some other boundary of the piece, as illustrated in Fig. 9. The result of course is a noisy puzzle with *missing* pieces.

In our present work, missing pieces are not yet handled and solving puzzles with missing pieces is left for future work, as it is likely to require a completely different approach. At present, the possibility of missing pieces due to geometric noise can be reduced or even prevented by using a *softer* noise model based on a smaller adaptive bound (for example,

piece-adapted noise bound defined by the piece’s shortest edge).

#### 5.5 Pictorial Noisy Puzzles

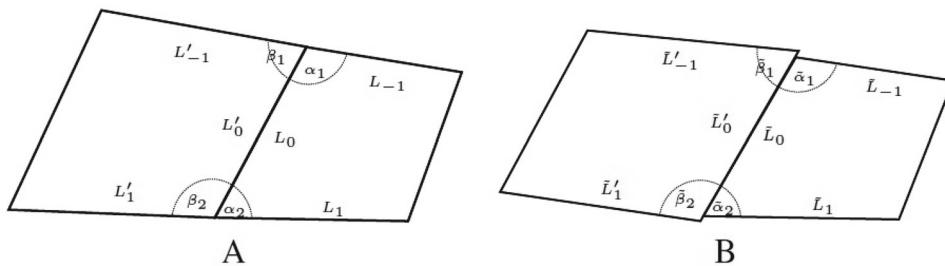
Just like apictorial crossing cuts puzzles, their pictorial counterpart can also be contaminated by geometric noise. A typical pictorial noisy crossing cut puzzle is depicted in Fig. 10A and since it is impossible to observe the noise when the pieces are shuffled, Fig. 10B puts several pieces in place to demonstrate the consequences. It is easy to observe that even if the pictorial content is immune to image noise, the geometric noise distances the pictorial content that *is* available in different puzzle pieces and thus complicates the way it can be used to determine plausible mates. We discuss a scheme that addresses the latter challenge in Sect. 8.

### 6 Data Synthesis

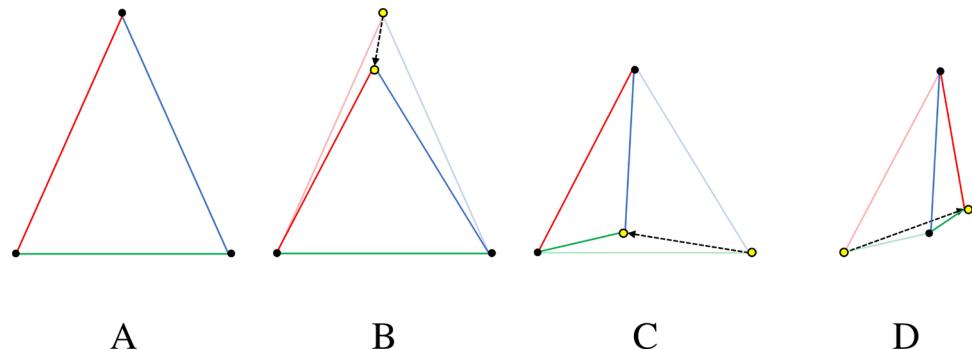
Since there is no previous work on crossing cuts puzzles, no data or benchmark results exist either. Part of our contribution here is a mechanism for data synthesis, as well as the first public dataset of crossing cuts puzzles. Such synthesis tools and datasets facilitate both the exploration of valuable properties of such puzzles and the experimental evaluation of reconstruction algorithms.

The synthesis process is based on a computational procedure that receives as input a description of the global polygonal shape  $S$  (which could be specified by the user or selected at random; see below) and the crossing cuts  $Cuts = \{c_1, \dots, c_a\}$  that dissect it. It returns both the *puzzle*, which can be given as input to reconstruction algorithms and the *ground truth solution* that can be used to evaluate the performance of puzzle solvers. As discussed in Sect. 3, the puzzle is a bag of polygonal pieces  $P = \{p_1, \dots, p_n\}$ , each represented properly by its vertices in some coordinate frame of reference. The ground truth solution constitutes a representation of the mating graph (and in particular, the set  $M$  of its matings), as well as the Euclidean transformations  $((R_1, t_1), \dots, (R_n, t_n))$  that place the pieces correctly in the reconstructed puzzle (or equivalently, the coordinates of the vertices of all pieces).

The process of synthesizing crossing cuts puzzles thus constitutes several aspects, all of which are described next for the sake of reproducibility. We note that pictorial puzzles are produced similarly to apictorial ones while the global polygonal shape is covered ahead of time by some pictorial content (e.g., from a user-provided image).



**Fig. 8** The effect of noise on the mate angle constraint. **A** Without noise, angles must comply to the original constraint  $\alpha_1 + \beta_1 = \alpha_2 + \beta_2 = \pi$ . **B** After applying the noise the  $\varepsilon$ -noisy angles are affected by the change in orientation in all edges that meet at both vertices of both mates, to result in the bound in the text



**Fig. 9** Example of a triangular piece being *disqualified* thus *erased* as a result of being exposed to relatively large noise, applied to each of the piece's vertices in clockwise order. **A** The original 'clean' piece. **B**

Noise is applied to the first vertex, pushing it down. **C** Noise is applied to the second vertex, pushing it left. **D** Noise is applied to the last vertex, pushing it right beyond the current piece border, thus eliminating it

## 6.1 A Graph Representation for Planar Divisions

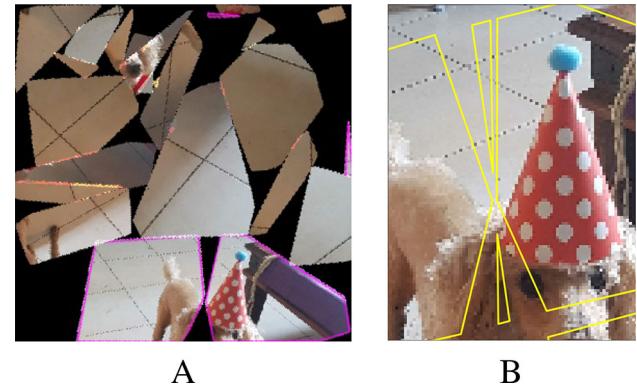
Let  $S \subseteq \mathbb{R}^2$  be a polygonal puzzle shape. The first stage of data synthesis is to construct a puzzle planar graph  $\mathcal{G}_{\text{puzzle}} = (\mathcal{V}, \mathcal{E})$  that represents both the boundary of  $S$  and the cuts that go through it. Note that  $\mathcal{G}_{\text{puzzle}}$  is different from the mating graph and is maintained for the synthesis process only. Toward that end, we first combine both the boundary lines of  $S$  (dashed blue lines in Fig. 11) and the crossing cuts themselves (dashed red lines in Fig. 11) into one set of lines:

$$\mathcal{C} = \text{Cuts} \cup \{\text{edge lines of } S\}.$$

The particular representation of lines is secondary, but in our case we represent each of them as a triplet  $(a_1, a_2, a_3)$ , where  $a_i$  are the coefficients of the line equation  $a_1x + a_2y + a_3 = 0$  in some global coordinate system.

The nodes of  $\mathcal{G}_{\text{puzzle}}$  are the intersection points of any two lines in  $\mathcal{C}$  that rest inside or on the border of  $S$  (see Fig. 11). Formally, this set of nodes is defined as follows:

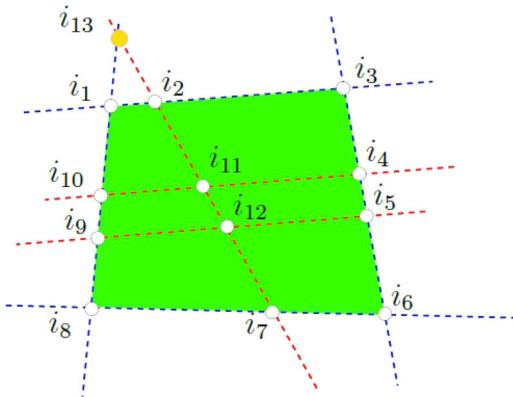
$$\mathcal{V} = \left\{ i \in S \mid \exists (c_1, c_2) \in \mathcal{C} \times \mathcal{C}, \quad (c_1 \neq c_2) \wedge (i = c_1 \cap c_2) \right\}.$$



**Fig. 10** Example of a noisy pictorial crossing cut puzzle. **A** A noisy pictorial crossing cuts puzzle is an unordered set of pictorial *noisy* pieces and recall that the noise is geometrical, not pictorial. The four pieces highlighted in purple are those shown in the next panel. **B** A closeup on four pictorial neighbors in their original position after they are cut from the original pictorial polygon and then contaminated with geometrical noise. Note how the noise sets the available pictorial content apart, thus complicating the decision about the affinity of the corresponding edges

The set  $\mathcal{E}$  of the edges of  $\mathcal{G}_{\text{puzzle}}$  link pairs of nodes that rest on the same line with no other nodes between them, or formally:

$$\mathcal{E} = \left\{ \{i_1, i_2\} \mid \exists c \in \mathcal{C}, \quad (i_1, i_2 \in c \cap \mathcal{V}) \wedge ([i_1, i_2] \cap \mathcal{V} = \emptyset) \right\}$$



**Fig. 11** The stages of representing a crossing cuts puzzle as a planar graph for the purpose of synthesis. In this selected example, the green quadrilateral is the global puzzle shape  $S$  whose boundary is defined by four lines (dashed blue). Three additional lines are defined as the crossing cuts (dashed red) and the intersection points of all these 7 lines that also lie inside or on the border of  $S$  are considered the nodes of the puzzle’s graph (and therefore  $\{i_1 \dots i_{12}\}$  are nodes in the graph, but  $i_{13}$  is not). The edges of the graph are defined by pairs of nodes that lie closest on the same line, i.e., by a pair of nodes that lie on the same line such that there is no other node in between them. Hence  $\{i_3, i_4\}, \{i_3, i_2\}, \{i_4, i_5\}$  are edges but  $\{i_3, i_5\}$  is not

where  $[i_1, i_2]$  is the line segment (as a set of points) between node points  $i_1$  and  $i_2$ .

## 6.2 Generation of Pieces and Ground Truth Matings

The extraction of the pieces from graphs that represent planar divisions has been addressed in the graph algorithms community and here we employ the optimal algorithm due to Jiang and Bunke (1993). This computational process receives the planar graph  $\mathcal{G}_{\text{puzzle}}$  from Sect. 6.1 and outputs all of the minimal polygonal regions, each represented as the ordered list of nodes that delineate it. One such region in Fig. 11 is  $(i_1, i_2, i_{11}, i_{10})$ .

The main construct in the algorithm is the notion of *wedge* (Jiang & Bunke, 1993), defined as a pair of different edges that meet at a node (e.g.,  $(\{i_1, i_2\}, \{i_2, i_3\})$ ) so that no other edge is encountered when rotating the first edge towards the second (e.g.  $(i_2, i_{11}, i_4)$  in Fig. 11 is a wedge, but  $(i_{10}, i_{11}, i_4)$  is not a wedge). A closed chain of overlapping wedges (e.g.  $((i_1, i_2, i_{11}), (i_2, i_{11}, i_{10}), (i_{11}, i_{10}, i_1))$ ) in Fig. 11 defines a minimal region, and thus a puzzle piece. The sorting scheme that locates the wedge chains was shown to have  $O(|\mathcal{E}| \log(|\mathcal{E}|))$  run-time complexity and  $O(|\mathcal{E}|)$  memory complexity. Please refer to the original paper for more details.

The application of Jiang and Bunke (1993) results is a set of puzzle pieces that are positioned in their original puzzle location, and thus, if the generated puzzle is pictorial, this is the point where the geometric representation of the pieces

serves to crop the original pictorial content that belongs to each piece. Either case, the segmentation of the original polygon into pieces in their “correct” position is now suitable for the computation and representation of the *desired* solution for the puzzle, at the very least for the evaluation of solvers output against the ground truth. Indeed, at this point of the synthesis process, any pair of neighboring pieces is positioned such that their mating edges strictly overlap. Hence the extraction of the ground truth mating graph can be done, for example, by finding all *overlapping* edges  $e_i^j$  and  $e_k^l$  that belong to *different* pieces. Formally, if  $E_m$  represents all edges of piece  $p_m$  (cf. Sect. 3) and thus  $E = (\bigcup_{m=1}^n E_m)$  is the set of all edges of all pieces, the ground truth matings are

$$M = \left\{ (e_i^j, e_k^l) \in E \times E \mid (p_i \neq p_k) \wedge e_i^k = e_k^l \right\}.$$

## 6.3 Piece Randomization and Geometric Noise

The final puzzle representation that is submitted to solvers should include no information about the ground truth position of the pieces. But with all pieces generated and the ground truth secured, puzzle pieces can now be shuffled and their Euclidean transformations randomized. To do so we first center each piece about its center of mass, i.e., each vertex is translated by the average of all vertices of the same piece. Then we apply a rotation transformation by some random angle selected uniformly in  $[0, 2\pi]$ . Needless to say, for pictorial puzzles the pictorial content is transformed correspondingly. If we denote the random rotation for piece  $p_i$  by  $RR_i$  and the translation to the center of mass by  $\vec{t}c_i$ , The ground truth positioning of the pieces is of course the inverse transformation  $\{[RR_i]^{-1}, -(\vec{t}c_i)\}$ .

If the desired puzzle should be “clean”, the process ends here and the list of randomly ordered and transformed pieces, each in its own coordinate system, serves as the final puzzle representation. However, if a noisy puzzle is required, each vertex of each piece is first translated by a random noise vector that obeys the constraints from Sect. 5.1. If the puzzle is pictorial, the application of the noise also crops the corresponding parts of the pictorial content. Only then the list of pieces is wrapped as the puzzle representation.

## 6.4 Datasets

We created several datasets using the procedure just described, where each serves a different purpose. The first dataset is tailored for the empirical exploration of statistical properties of crossing cuts puzzles while the others are designed to facilitate experimental evaluation (and if needed, of training) of crossing cuts solvers. The images for the pictorial content of all puzzles in all pictorial DBs were obtained from <https://>

[unsplash.com/](https://unsplash.com/) and <https://www.pexels.com/public-domain-images/>, or taken by the authors with a digital camera.

**DB1—circular puzzles for statistical properties** Sect. 7 presents a theoretical analysis of crossing cuts puzzles and their properties. To simplify and facilitate analytical analysis, it is performed on puzzles with circular global shape while the corresponding empirical properties were measured on synthesized puzzles whose shape was a unit triacontadigon (i.e., an approximation of a unit circle as a polygon of 32 sides). The random cuts in this case were selected by sampling two angles  $\phi_1, \phi_2$  and then passing a line through the corresponding points on the circumference of the circle, namely  $(\cos \phi_1, \sin \phi_1), (\cos \phi_2, \sin \phi_2)$ .

Following this procedure we generated a collection of 300 noiseless puzzles, 30 puzzles for 10 different numbers of crossing cuts  $a \in \{10, 20, \dots, 100\}$ . The number of puzzle pieces that resulted from this procedure varied from 36 to 2183. For each “clean” puzzle we then generated several noisy versions, with noise level varying in  $\xi \in [0\%, 0.1\%, 0.25\%, 0.5\%, 1\%, 2\%]$ . Recall that  $\xi$  is the noise bound relative to puzzle diameter, which in DB1’s case is always 2. In absolute terms, the noise in this dataset thus varied up to  $\epsilon \leq 0.04$ , but perhaps more informatively, when considered against the average edge length, the noise could approach 64% (i.e.,  $\bar{\xi} \leq 64\%$ , see Sect. 7.5).

With the noisy versions taken into account, the number of puzzles in DB1 thus totaled 1800. For their intended use (i.e., analysis of properties), all puzzles in this dataset need not be pictorial and thus this is an apictorial dataset. Selected examples are shown in Fig. 12.

**DB2—general apictorial puzzles for solvers evaluation** Unlike the specifically crafted puzzle shape used for the analysis of puzzle properties, the evaluation of puzzle solvers requires randomly shaped (yet convex) puzzles. To achieve this goal we first sampled a random number (between 4 and 50) of randomly positioned points in some predetermined workspace  $[0, W] \times [0, H]$  and then computed their convex hull to generate a random global convex polygonal shape (which in our case ended up having from 3 to 14 sides).  $W$  and  $H$  are given as parameters to the synthesizer but they bear very little significance. In our case, we fixed them both at  $W = H = 100$ .

The random cuts  $Cuts = \{c_1, \dots, c_a\}$  were also selected as uniformly distributed random lines in the same workspace, but to ensure they indeed penetrate the random polygon we first selected two random points *inside* the polygon and defined the cut as the line that goes between these points.

While this procedure can be activated on demand and with arbitrary parameter values, we used it to generate a collection of 100 random puzzles, whose number of cuts varies from 5 to 50 (10 instances from each case) and their number of pieces extends from 11 (in the easier puzzles) to 936 (in the more challenging ones). For each “clean” puzzle we also

generated several noisy versions, with noise levels varying in  $\xi \in [0\%, 0.1\%, 0.25\%, 0.5\%, 1\%, 2\%]$ . With the noisy versions taken into account, the number of puzzles in DB2 thus totals 600.

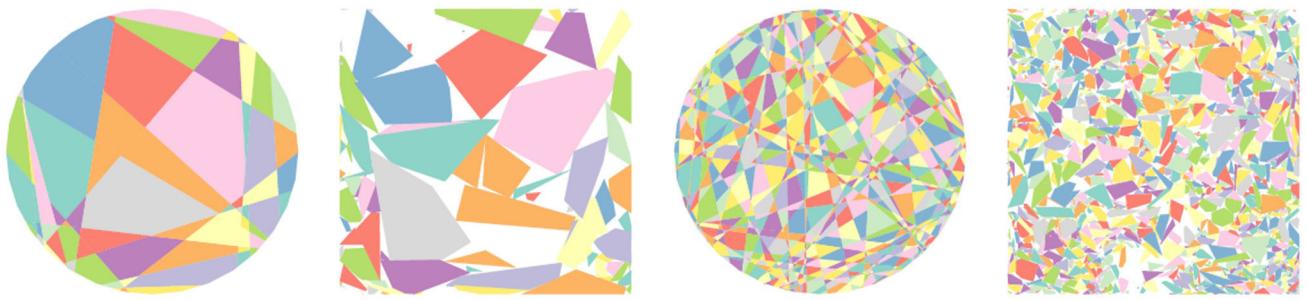
Figure 13 shows one example from DB2 and aspects of its generation process.

**DB3—Perturb grid pictorial puzzles for solver evaluation** The procedure just described was used also for the generation of a pictorial dataset for evaluation (and if needed, also for training) of pictorial crossing cuts puzzle solvers, where the pictorial content that covers the puzzle (and consequently, its pieces) was provided as an image and handled as described earlier in Sect. 6. However, to facilitate a better examination of the contribution of the pictorial content, in this first pictorial dataset, we reduced the role of the geometry by designating crossing cuts that generate edges of relatively similar lengths (both within and between pieces). This was done by defining the cuts to form a perturbed grid over the global polygonal shape, resulting in a narrower histogram of edge lengths and hence many more mating candidates when only geometry is considered. Without pictorial content, such puzzles will consider many more mating candidates, require a solver with significantly more computational resources, and (if the latter are bounded) may completely prohibit a solution unless pictorial considerations are incorporated too. At the same time, with crossing cuts that are roughly parallel, we are also guaranteed that the bounded geometrical noise does not erode pieces completely, a situation that generates puzzles with missing pieces that are outside the scope of our present solver. For all these reasons we tested our pictorial puzzle solver on DB3, but already generated DB4 below for future generalizations.

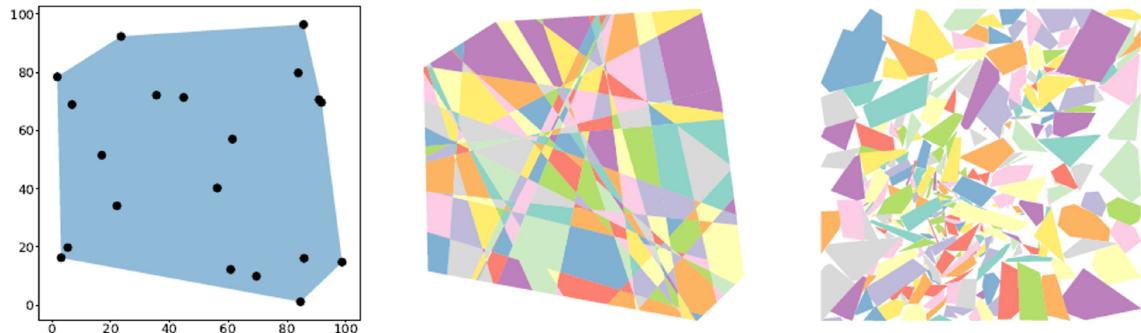
Following this scheme, we generate a collection of 600 random perturbed grid pictorial puzzles, whose number of cuts vary from 10 to 100 (with 10 instances from each case) and number of pieces that extends from 35 to 2601. As before, noise level varied in  $\xi \in [0\%, 0.1\%, 0.25\%, 0.5\%, 1\%, 2\%]$ . Figure 14 shows a selected example.

In addition to the 3 datasets above, we created two additional datasets for future work by the community. These DBs are described next.

**DB4—General pictorial dataset** DB4 is another pictorial dataset for future research where the cuts are completely arbitrary and no special care is taken to downplay the role of geometric constraints. The importance of this dataset for future work is twofold. First, since the geometrical information becomes more significant and informative again (compared to DB3, for example), it will take more “aggressive” methods to exploit the pictorial content *effectively*. Second, in arbitrary crossing cuts puzzles some pieces may turn small enough to completely disappear after the application of the geometrical noise (as indeed happens in 81.5% of puzzles in this dataset). Thus, this DB4 also facilitates



**Fig. 12** Two selected synthesized circular puzzles for the analysis of puzzle properties. Shown are the puzzle as a bag of pieces and the corresponding ground truth solution. The numbers of cuts used for these examples are 20 and 100 while the corresponding number of pieces are 63 and 1806, respectively



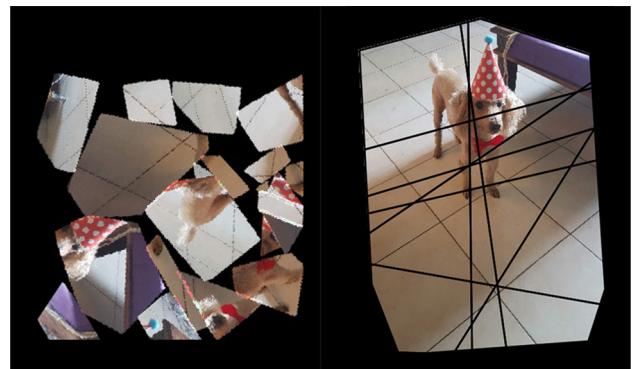
**Fig. 13** One selected synthesized polygonal puzzle (30 cuts, 200 pieces) from DB2. The left column shows the process of generating the random puzzle shape in the workspace using the convex hull of

random sample points. The middle column illustrates the ground truth solution and the right column shows the puzzles as a bag of pieces as provided to potential solvers



**Fig. 14** An example of a perturbed grid pictorial puzzle (with 8 cuts and 24 pieces), both as a bag of pieces and the corresponding ground truth solution

future research on crossing cuts puzzles with *missing pieces*. Toward that end we generated a collection of 600 random polygonal pictorial puzzles, whose number of cuts vary from 10 to 100 (10 instances from each case), number of pieces extends from 35 to 3907, and noise level in the range  $\xi \in [0\%, 0.1\%, 0.25\%, 0.5\%, 1\%, 2\%]$ . A selected example of such a puzzle is shown in Fig. 15.



**Fig. 15** One selected example of a general pictorial puzzle (with 7 cuts and 18 pieces). Note that some small pieces appear only in the ground truth row as the noise removed them completely from the bag of pieces in the puzzle to solve

**DB5—Square piece pictorial dataset** As mentioned in Sect. 3, from a geometrical point of view, strictly square piece puzzles are a very special case of crossing cuts puzzles where geometry plays no role and the pictorial content is the sole source of information for reconstruction. For “backward compatibility”, and for their more general scheme of representation, using crossing cuts puzzles to represent square

piece puzzles may be useful, especially if geometrical noise is to be allowed. We therefore generated such a collection of 3600 random square piece pictorial puzzles, whose number of cuts vary from 20 to 200 (10 instances from each case), number of pieces extends from 100 to 10,000, and noise level in the range  $\xi \in [0\%, 0.1\%, 0.25\%, 0.5\%, 1\%, 2\%]$ . Naturally, the different number of cuts generated pieces of different sizes, where in our cases extended from  $5 \times 5$  to  $30 \times 30$  pixels, yet another generalization of the prior art in square pieces puzzles that tended to focus on one size of  $28 \times 28$  pixel only (though as mentioned in Sect. 2, one exception does exist (Son et al., 2016)).

We note that all 5 datasets are open and available for the community at the public-domain portal [icvl.cs.bgu.ac.il/polygonal-puzzle-solving](http://icvl.cs.bgu.ac.il/polygonal-puzzle-solving). This portal also will host additional datasets of varying characteristics as they become available.

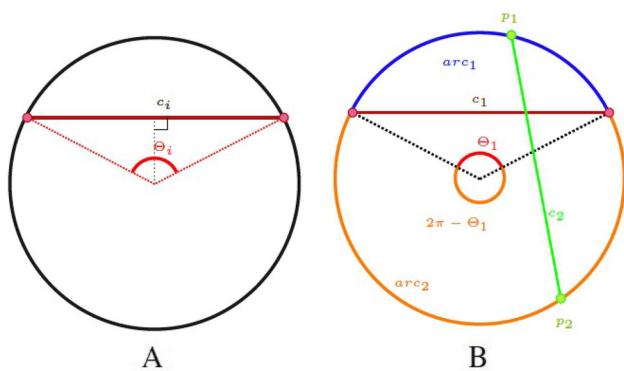
## 7 Puzzle Properties

One of the advantages of the generation model that defines crossing cuts puzzles is the better ability to analyze their properties. Since the model is stochastic, their properties are typically probabilistic, but nevertheless can provide insights on both the problem itself and about potential solutions (or limitations thereof). Here we explore such properties both analytically and, when needed, empirically. In this section, we assume that the global puzzle shape is a unit circle (or a polygonal approximation thereof), whose symmetry simplifies some of the analytical analyses. Most results, however, are indicative of all crossing cuts puzzles (up to a factor of half of their diameter). Empirical properties are evaluated on the DB1, the circular puzzles dataset that was described in Sect. 6.

### 7.1 Expected Cut Length

The first measure of interest is the length of a random cut  $c_i$  through the global puzzle shape. When the latter is a unit circle,  $c_i$  is determined by two points sampled uniformly on the circumference of the circle. In other words, the cut is determined by the chord between points  $\vec{p}_1 = (\cos \phi_1, \sin \phi_1)$  and  $\vec{p}_2 = (\cos \phi_2, \sin \phi_2)$ , where the two angles are uniformly distributed random variables  $\phi_1, \phi_2 \sim U(0, 2\pi)$ . The length of cut  $c_i$  is therefore another random variable defined by the function  $l_i = \|\vec{p}_2 - \vec{p}_1\|$ , and one may seek its expected value.

Since circles are symmetric, without loss of generality we can align the coordinate system parallel to the cut and consider only horizontal chords that lie in the circle's upper half, i.e., when both  $\vec{p}_1$  and  $\vec{p}_2$  have identical positive  $y$  coordinates, as in Fig. 16A. If we now assume (w.l.o.g.) that



**Fig. 16** Expected cut length and probability of cut intersection. **A** A unit circle cut  $c_i$  with a central angle  $\Theta_i$  can be considered w.l.o.g to be horizontal, leading to an expected length as in the text. **B** Two cuts  $c_1$  and  $c_2$  intersect if and only if the vertices of the second cut (in green) lie in different arcs (in blue and orange) generated by the first cut (in red) (Color figure online)

$\phi_2 > \phi_1$ , then  $\Theta_i = \phi_2 - \phi_1$  is the central angle of the cut and therefore  $l_i = 2 \sin(\Theta_i/2)$ . Since  $\Theta_i \sim U(0, \pi)$ , it follows that the expected length of a random cut through the unit circle is

$$E[l_i] = \int_0^\pi l_i(t) \cdot f_{\Theta_i}(t) dt = \int_0^\pi 2 \sin\left(\frac{t}{2}\right) \frac{1}{\pi} dt = \frac{4}{\pi} \approx 1.273 .$$

### 7.2 Probability of Cut Intersections

Given two uniformly distributed random cuts  $c_1$  and  $c_2$ , one may seek the probability of their intersection. This question is interesting for understanding how the number of pieces grows with the number of cuts, as intersecting cuts contribute more pieces than non-intersecting ones. Again, we can assume w.l.o.g that one of the cuts, say  $c_1$ , is horizontal and lying in the upper half of the circle (marked red in Fig. 16B). Let the central angle of  $c_1$  be  $\Theta_1 \sim U(0, \pi)$  and note how this cut divides the circle to two arcs -  $arc_1$  of angle  $\Theta_1$  (blue in Fig. 16B) and  $arc_2$  of angle  $2\pi - \Theta_1$  (orange in Fig. 16B).

Denoting the vertices of  $c_2$  as  $p_1$  and  $p_2$ , we first note that an intersection between  $c_1$  and  $c_2$  occurs if and only if  $p_1$  belongs to  $arc_1$  and  $p_2$  belongs to  $arc_2$  (or vice versa). Seeking the probability of such an event, let  $I_{c_1, c_2}$  be an indicator function for the intersection between  $c_1$  and  $c_2$ . Clearly, this function depends on the extent (or size) of the two arcs and indeed

$$\begin{aligned} P(I_{c_1, c_2} | \Theta_1) &= 2 \cdot P(p_1 \in arc_1 | \Theta_1) \cdot P(p_2 \in arc_2 | \Theta_1) \\ &= 2 \cdot \frac{\Theta_1}{2\pi} \cdot \frac{2\pi - \Theta_1}{2\pi} = \frac{\Theta_1(2\pi - \Theta_1)}{2\pi^2} \end{aligned}$$

It follows that the expected value for the intersection event is

$$\begin{aligned} E[I_{c_1, c_2}] &= P(I_{c_1, c_2}) = \int_0^\pi f_{\Theta_1}(t) \cdot P(I|\Theta_1 = t) dt \\ &= \int_0^\pi \frac{1}{\pi} \frac{t(2\pi - t)}{2\pi^2} dt = \frac{1}{3}. \end{aligned}$$

Hence, we conclude that only 1 out of 3 pairs of random unit circle cuts will intersect, an event perhaps less frequent than intuitively anticipated in such circumstances. An intuitive justification nevertheless arises once we consider 4 endpoints on the shape's circumference and all 3 combinations of crossing lines they facilitate. Indeed, only one of these combinations induces a crossing.

### 7.3 Expected Total Number of Cut Intersections

Following the probability of the cut intersection event (Sect. 7.2), we now can seek the total number of intersections expected in a puzzle of  $a$  cuts. Clearly, it is simply the sum of all pairs of intersecting cuts, that is

$$N_{\text{intersect}} = \frac{1}{2} \sum_{i=1}^a \sum_{j \neq i} I_{c_i, c_j}.$$

The expected value of this random variable, i.e., the expected number of intersections in puzzles with  $a$  crossing cuts, thus becomes:

$$E[N_{\text{intersect}}] = \binom{a}{2} \underbrace{P[I_{c_1, c_2}]}_{\frac{1}{3}} = \frac{a(a-1)}{6}.$$

Note that this number is far smaller than  $\binom{a}{2}$ , the maximum number of intersections possible between  $a$  cuts.

### 7.4 Expected Number of Edges

Given a crossing cuts puzzle generated by  $a$  crossing cuts, we next wish to express the number of piece edges in the entire puzzle. This measure is fundamental to the number of matings and therefore is a substrate of the computational complexity of reconstruction algorithms.

First, observe that each edge is a subset of some cut between two consecutive intersections along its length. In particular, if a cut  $c_i$  is intersected  $k$  times, the number of edges that emerge from this cut will be  $k + 1$ . To obtain the total number of edges  $N_{\text{edges}}$  in the puzzle one needs to sum up the edges on all cuts, i.e.,

$$N_{\text{edges}} = \sum_{i=1}^a \left( 1 + \sum_{c_j \neq c_i} I_{c_i, c_j} \right)$$

$$= a + \sum_{c_i} \sum_{c_j \neq c_i} I_{c_i, c_j} = a + 2N_{\text{intersect}}. \quad (4)$$

Since  $I_{c_i, c_j}$  is a random function, so is  $N_{\text{edges}}$ . We can therefore seek its expected value, i.e., the expected number of edges in the entire puzzle:

$$\begin{aligned} E[N_{\text{edges}}] &= E[2 \cdot N_{\text{intersect}}] + a \\ &= 2 \cdot \frac{a(a-1)}{6} + a = \frac{a^2 + 2a}{3}. \end{aligned} \quad (5)$$

### 7.5 Expected Edge Length

With the expected number of edges resolved, we can now seek the expected edge length as the expected ratio between the accumulated edge lengths to their number. Fortunately, the former is simply the summed length of all cuts and thus, if the puzzle constitutes  $a$  cuts, we obtain an average edge length of

$$l_{\text{avg}} = \frac{\sum_{i=1}^a l_i}{N_{\text{edges}}},$$

where  $l_i$  was obtained in Sect. 7.1 and  $N_{\text{edges}}$  was derived in Sect. 7.4. While the expected value of a ratio is *not* the ratio of expected values, it *is* its first-order Taylor approximation (Benaroya & Han, 2013). Thus:

$$E[l_{\text{avg}}] = E\left[\frac{\sum l_i}{N_{\text{edges}}}\right] \approx \frac{E[\sum l_i]}{E[N_{\text{edges}}]} = \frac{12}{\pi(a+2)} \quad (6)$$

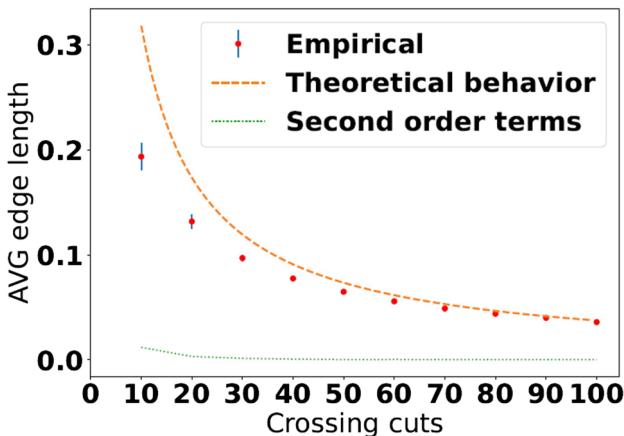
which conforms well with the empirical results of the same measure as shown in Fig. 17. The second-order Taylor approximation

$$E[l_{\text{avg}}] \approx \underbrace{\frac{E[\sum l_i]}{E[N_{\text{edges}}]}}_{\text{First order terms}} - \underbrace{\frac{\text{Cov}(\sum l_i, N_{\text{edges}})}{(E[N_{\text{edges}}])^2} + \frac{\text{Var}(N_{\text{edges}}) \cdot E[\sum l_i]}{(E[N_{\text{edges}}])^3}}_{\text{Second order term}}$$

constitutes two second-order terms that turn out to cancel each other to a diminishing sum as the number of cuts increases, thus facilitating the approximation in Eq. 6. This also is exemplified empirically in Fig. 17. As mentioned in the introduction to this section, empirical results are evaluated on DB1, the circular puzzles dataset, described in Sect. 6

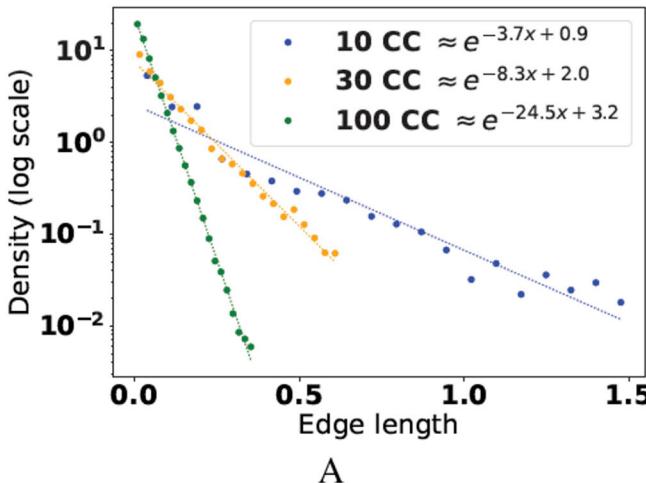
### 7.6 Edge Length Distribution

While the expected edge length can be computed analytically, it is far more complicated to do so for the entire distribution of edge lengths. The importance of this distribution lies in how it influences the number of possible mates under geometric noise, a quantity that is likely to increase the narrower the distribution becomes. We have therefore measured this property empirically using our synthesized datasets and Fig. 18A



**Fig. 17** Empirical average edge length with a growing number of cuts, as evaluated on DB1, compared to the first-order theoretical behavior from Eq. 6 that improves in prediction power as the number of cuts increases. Error bars are  $\pm 1$  SE. The green line shows the diminishing second-order terms of the Taylor approximation (Color figure online)

reports these findings. Note how in general the distribution is exponential, preferring shorter edges and (not surprisingly) becoming narrower with a larger number of cuts. Clearly, when cuts are no longer selected uniformly, the distribution can definitely change shape. For example, strictly square noiseless puzzles will of course have a delta distribution for their edge lengths. Perturbed square noisy puzzles, i.e., those in our DB3, exhibit the distribution shown in Fig. 18B.



A

**Fig. 18** Probability density of edge length in crossing cuts puzzles. **A** Empirical distribution for growing number of cuts, as evaluated on noisy puzzles from DB1. Note how the distribution gets narrower with the number of crossing cuts, as depicted by a steeper slope in the log

## 7.7 Min, Max, and Expected Number of Pieces

One of the significant properties of jigsaw puzzles that clearly affects the complexity of their representation (and thus of possible solutions) is their number of pieces. Clearly, even if the number of crossing cuts is set, different cut patterns can create puzzles with varying numbers of pieces. To estimate this number, and inspired by Moore (Moore, 1991), we use Euler's Formula for planar graphs:

**Theorem 1** (Euler's Formula) *If  $G = (V, E)$  is any planar graph, then  $G$  has  $|E| - |V| + 2$  regions where  $|E|$  is the number of links in the graph and  $|V|$  is the number of nodes.*

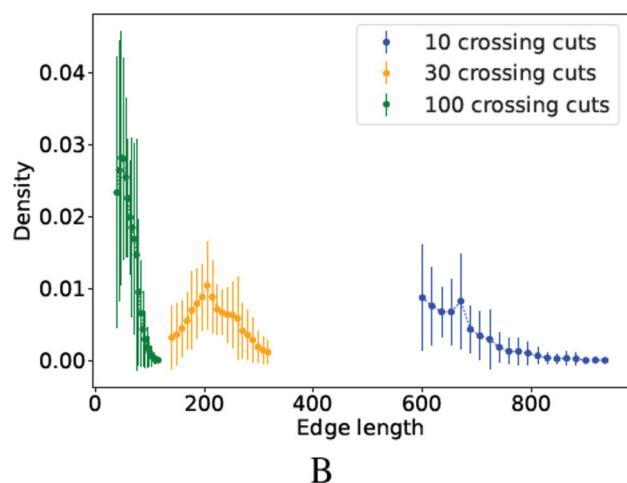
Note that in our crossing cuts puzzle case, the number of nodes for Euler's formula is the number of inner intersections ( $N_{\text{intersect}}$ ) plus the  $2a$  intersections of the cuts with the boundary of the puzzle. The number of links is the number of internal edges ( $N_{\text{edges}}$ ) plus the  $2a$  piece sides generated by the cuts along the puzzle boundary. Using Euler's formula, and applying Eq. 4, we thus get

$$N_{\text{pieces}} = \underbrace{(N_{\text{edges}} + 2a)}_{|E|} - \underbrace{(N_{\text{intersect}} + 2a)}_{|V|} + 2 - 1 \quad (7)$$

$$= N_{\text{intersect}} + a + 1 \quad (8)$$

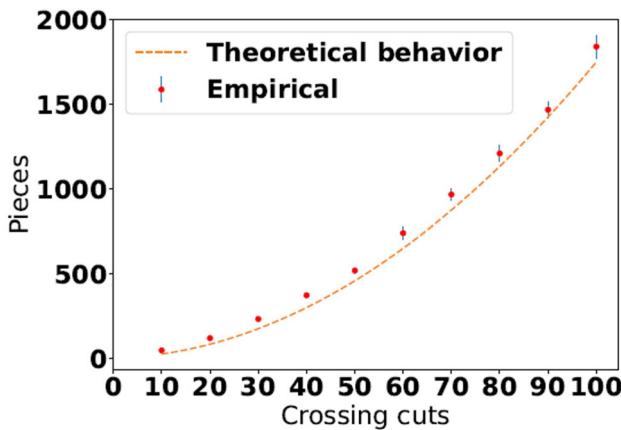
Note that the subtraction of the last 1 in Eq. 7 is required since Euler's formula also counts the region outside the puzzle/graph.

With this in mind, we next observe that one extreme case includes puzzles where no cut intersects others ( $N_{\text{intersect}} =$



B

scale. **B** Empirical edge length distribution of perturbed square puzzles with a growing number of cuts, as evaluated on DB3. Observe how the distribution of edge lengths in these puzzles is no longer exponential, with a modal behavior that is dominated by a particular edge length



**Fig. 19** The empirical expected number of puzzle pieces with a growing number of cuts, compared to the theoretically expected behavior (Eq. 8). Error bars are  $\pm 1$  SE

0), and thus the minimal number of pieces is  $N_{pieces} = a + 1$ . At the other extreme, every cut intersects all others, and the  $\binom{a}{2}$  intersections yield the following quadratic upper bound on the number of pieces (which is exactly the Lazy caterer's sequence)

$$\max_{c_i, \dots, c_a} N_{pieces} = \binom{a}{2} + a + 1 = \frac{a^2}{2} + \frac{a}{2} + 1 .$$

However, with  $N_{intersect}$  being a random variable (that depends on the random cuts), it is more interesting to examine the *expected* number of pieces:

$$\begin{aligned} E[N_{pieces}] &= E[N_{intersect}] + a + 1 \\ &= \frac{a(a - 1)}{6} + a + 1 = \frac{a^2}{6} + \frac{5a}{6} + 1 . \end{aligned} \quad (9)$$

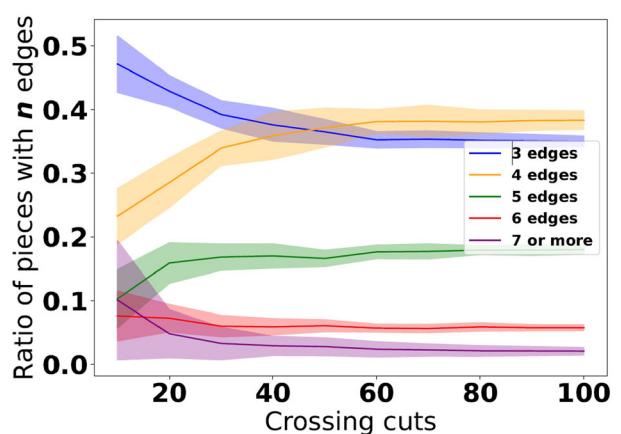
This behavior can also be verified empirically, as shown in Fig. 19. Finally, as the number of cuts increases, and when  $a \rightarrow \infty$ , the ratio between the expected and the maximum number of pieces becomes

$$\lim_{a \rightarrow \infty} \frac{E[N_{pieces}]}{\max N_{pieces}} = \frac{1}{3}$$

which is the same as the probability for cut intersection found in Sect. 7.2.

## 7.8 Expected Number of Edges Per Piece

As discussed in Sect. 3, the crossing cuts puzzle model cuts the puzzle shape into convex polygonal pieces. Clearly, these pieces can have a different number of edges and there is no a-priori inherent limit to this number (except the number of cuts, of course).



**Fig. 20** Expected ratios of pieces with a particular number of edges as a function of the number of crossing cuts. Note how quadrilaterals are always the majority, followed closely by triangular pieces and the less frequent pentagons. These three classes of polygons quickly converge to account for approximately 95% of all pieces. Note how the ratios converge quickly and remain essentially invariant to the number of cuts. Shaded bands represent  $\pm 1$  SE

To explore this property we conducted an empirical evaluation on DB1, i.e., by using the 30 synthesized circular crossing cuts puzzles synthesized for each of the different number of cuts. Empirically, the most frequent pieces are either quadrilateral or triangular, depending on the number of cuts, whereas asymptotically, quadrilaterals are the most abundant. The probability of encountering puzzle pieces with more than 6 edges is diminishing quickly from 10% in puzzles with few cuts, to approximately 2% as the number of cuts increases. The results up are shown in Fig. 20 and demonstrate that the distribution converges quickly and then remains stable from approximately 60 cuts.

## 7.9 Number of Potential Matings Per Edge

Since any puzzle reconstruction algorithm will seek (as part of its different computations) to match the edge of a given piece to edges of other pieces, the complexity of such an algorithm will relate intimately to the number of potential matings each edge may have. Clearly, the higher the number of admissible candidate mates, the more difficult the identification of the correct one is likely to be. Following the discussion in Sect. 5.1, the raw number of geometrically admissible mates each edge may have is determined by the two mating constraints  $\tilde{C}_1$  and  $\tilde{C}_2$  and it is naturally affected by the level of the noise. In fact, since the number of expected edges in the puzzle is quadratic in the number of cuts (cf. Sect. 7.4), a naive extension of Algorithm 1 from Sect. 4 that also incorporates backtracking when wrong matings are identified, will grow intractably in complexity by a factor of  $O(k^{a^2})$  if the number of potential matings per edge is  $k$ .

We empirically explored the expected average number of matings by counting the average number of possible matings for each puzzle in DB1 while employing  $\tilde{C}_1$  and  $\tilde{C}_2$ . Not unexpectedly, the results provided in Fig. 21A indicate that the noise level affects the number of potential mates very rapidly and very drastically (where the decline after the peak is because greater noise erodes more pieces completely, as shown in Fig. 21B).

Here it is also worth re-emphasizing that the noise levels in our model are measured relative to the puzzle size, or its *diameter*, and therefore might appear small. In reality, they are not small at all, because noise affects individual pieces, that typically are very much smaller than the entire puzzle. Thus, considering also the average edge length (cf. Sect. 7.5), noise level  $\xi$  relative to the puzzle diameter is comparable to the following bound

$$\bar{\xi} = \frac{4 \cdot \xi \cdot \pi(a+2)}{12} \quad (10)$$

relative to average edge length, which is perhaps a more tangible and informative measure. For example, in a puzzle of 20 crossing cuts (84 pieces on average) and a noise level of  $\xi = 1\%$ , the noise relative to average edge length is  $\bar{\xi} \approx 10\%$ , namely a rather significant noise.

Indeed, the high number of potential matings in the presence of noise suggests a similarly high branching factor in a naive “search and backtrack” algorithm, which will clearly become intractable for handling noisy (i.e., realistic) crossing cuts puzzles, even if the number of cuts is modest. Our goal is to seek heuristics that make the reconstruction more manageable after all, and as we will see later on, this can be achieved by utilizing multiple geometric constraints *simultaneously*, and by leveraging the *pictorial* content to generate and apply yet more constraints on the matings.

## 8 Puzzle Reconstruction Under Noisy Conditions

Recall that a “realistic” crossing cuts puzzle constitutes a representation of the input pieces (and some bound on the erosion noise), and it seeks as output both the correct matings and the geometric transformation of each piece. As mentioned above, at first sight one may wish to extend the initial greedy Algorithm 1 from Sect. 4 while using the relaxed “noisy” constraints ( $\tilde{C}_1$  and  $\tilde{C}_2$ ) to find candidate matings, and if needed employ backtracking upon failures (e.g., piece collisions). However, as analyzed above, the expected number of candidate matings per edge (cf. Sect. 7.9) clearly makes this naive extension intractable. Moreover, under noise, it is unclear what is the desired position (i.e., Euclidean transformation) of each piece, or how to compute it in the first

place, even if the mating relationships are resolved correctly. Figure 22 illustrates some of these challenges.

To address these difficulties we approach the problem in stages, and in particular, we begin with the simpler problem of solving the puzzle *when the correct matings are given also*. More concretely, we first suggest a solution to this sub-problem by representing it as a multi-body spring-mass system where energy minimization is sought while the spring attractive forces apply between corresponding vertices. The solutions obtained this way are then used as scores for searching and determining the correct matings while incorporating a hierarchical (and progressively growing) set of circular constraints among adjacent pieces. For pictorial puzzles, we also add another set of pictorial constraints on top of the geometrical ones.

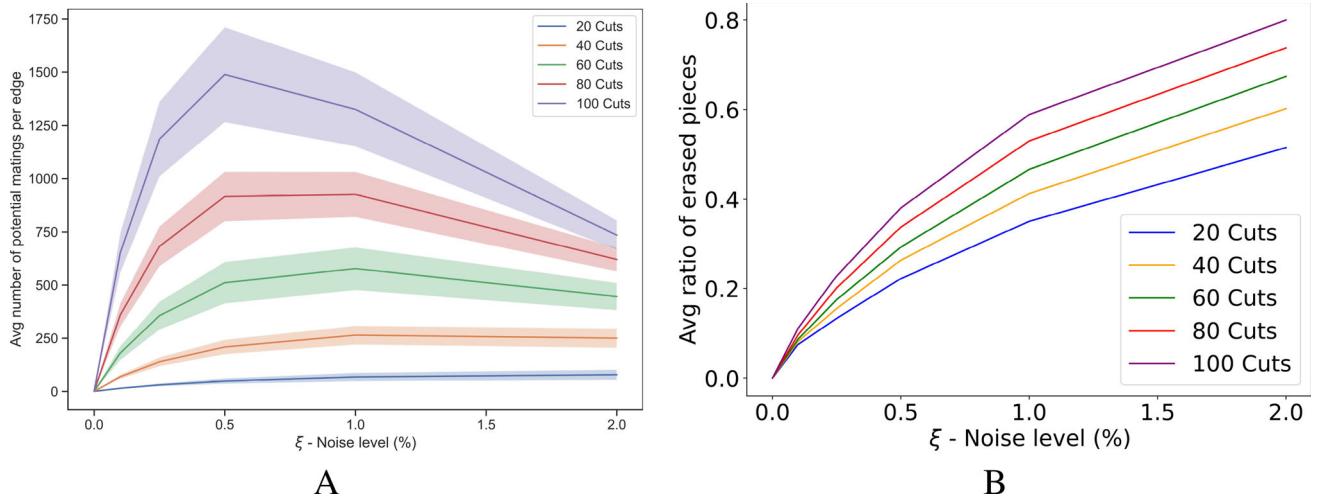
### 8.1 Noisy Puzzle Solving with Known Matings

Let  $P = \{p_1, p_2, \dots, p_n\}$  be the set of pieces and let  $M = \{m_1, \dots, m_{|M|}\}$  be a set of (known) pairwise matings  $m_q = \{e_i^j, e_k^l\}$  between their corresponding edges. We seek a computational scheme that obeys the given matings and places the pieces in some “optimal” or “good” way next to each other. Intuitively, we would like to do so in a way that minimizes the total distance, i.e., the  $L_2$  displacement error, between corresponding mating vertices, or more formally, to find the set of Euclidean transformations  $(R_i, \vec{t}_i)$  that satisfy

$$\operatorname{argmin}_{(R_i, \vec{t}_i)} \sum_{(\vec{v}_i^j, \vec{v}_k^l)} \| (R_i \vec{v}_i^j + \vec{t}_i) - (R_k \vec{v}_k^l + \vec{t}_k) \|^2 , \quad (11)$$

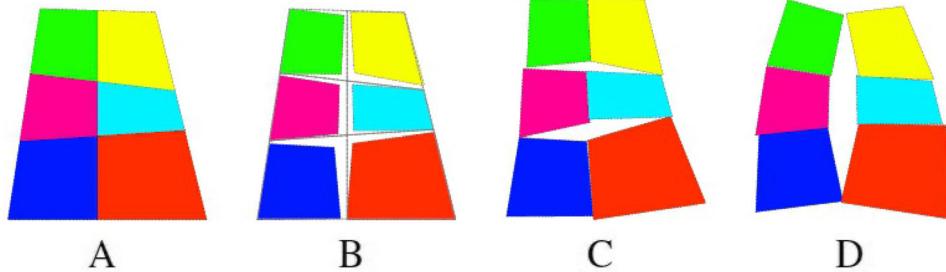
where  $\vec{v}_i^j$  and  $\vec{v}_k^l$  are the corresponding vertices of the matings defined by  $M$  while  $(R_i, t_i)$  and  $(R_k, t_k)$  are the euclidean transformations of pieces  $p_i$  and  $p_k$  that own these vertices. Unfortunately, this is no simple least squares minimization, as the unknowns include rotation matrices and the sought-after transformations must satisfy the constraint that they are *identical for all vertices of the same piece*.

As a result of its specifications, this optimization problem defies analytical solutions and we therefore resort to tools from other disciplines. In particular, we propose to abstract the rearrangement problem as a *multi-body spring-mass system*. To do so we first represent our puzzle pieces as 2D rigid bodies with uniform density, and therefore with mass that is proportional to their area. We then connect all pairs of corresponding vertices (i.e., those matched by the matings) with springs of zero length and identical elasticity (i.e., having the same *spring constants*). Since the elastic potential energy of such a spring-mass system is  $U(x) = \sum_l \frac{1}{2} kx_l^2$ , where  $x_l$  is the displacement from equilibrium length of spring  $l$ , it is identical (up to a constant) to our objective function in Eq. 11. We therefore apply numerical methods for solving multi-



**Fig. 21** **A** The average number of geometrically admissible matings for each edge as a function of noise level. Each graph shows the potential number of mates that satisfy both  $\tilde{C}_1$  and  $\tilde{C}_2$ , summed over all edges. Both the initial rapid growth and the subsequent gradual decline (which is particularly visible in puzzles with more cuts) indicate the harmful

effect of noise. Shaded bands are  $\pm 1$  SE. **B** The average ratio of completely eroded pieces due to applied noise, out of the puzzles' original (noiseless) number of pieces. As expected, the application of larger noise completely erodes more pieces, thus also decreasing the number of potential matings per edge



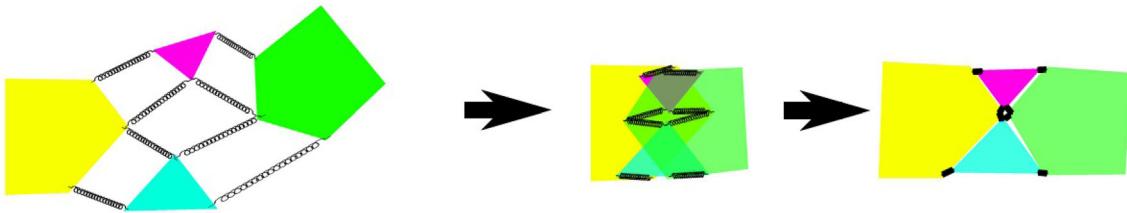
**Fig. 22** The placement of noisy pieces can be ambiguous. When noise is applied to this simple crossing cuts puzzle (**A, B**), there could be different placements of the noisy pieces that might count as “correct” (**C, D**)

body spring-mass problems, while the initial pose (position and rotation) of each piece is chosen randomly inside the arena. The physical system is then set loose and with some damping (i.e., loss of energy due to friction) it converges to its minimal energetic state, as illustrated in Fig. 23.

In practice there are off-the-shelf tools to solve the above system numerically, practically simulating the dynamical process that the system undergoes from initial condition until convergence, and here we use the Box2D physics engine [13]. For puzzles, it is undesired to obtain solutions with overlapping pieces, but adding this constraint to a random initial state is unstable numerically. We therefore run the process first while allowing the pieces to overlap. The convergence state of this run is energetically minimal but might include small overlaps. We then use it as the initial state for a second run, this time while forbidding overlaps. The end result is our solution and Fig. 24 shows several snapshots from this dynamical process.

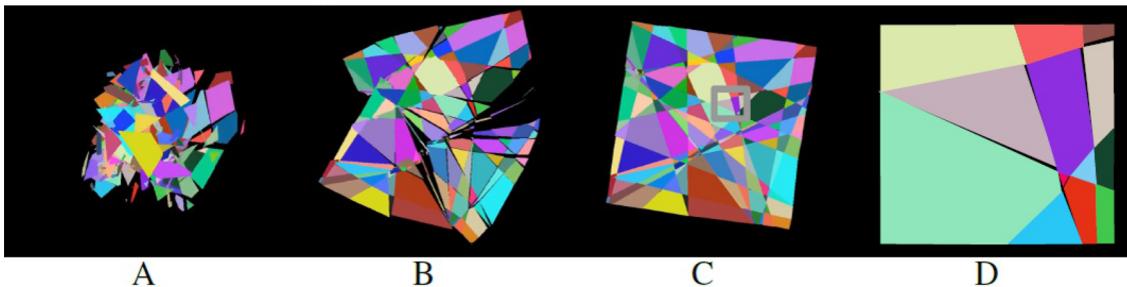
## 8.2 Noisy Puzzle Solving with Unknown Matings

Let  $P = \{p_1, \dots, p_n\}$  be the set of puzzle pieces and let  $\varepsilon$  denote the noise level. Unlike the conditions in the previous section, we now assume no knowledge of the matings and thus our goal is twofold: to find the correct matings  $M = \{m_1, \dots, m_{|M|}\}$  between the edges *and* the geometrical transformation of each piece. To do so, we endow the basic constraint matching procedure (based on  $\tilde{C}_1$  and  $\tilde{C}_2$ ) with a modified version of a hierarchical loops scheme (Son et al., 2018), where the mass-spring minimization approach from Sect. 8.1 is used to score the loops based on their success to position the pieces properly, as defined below. If the puzzle is pictorial, we also rank and filter those matches using the pictorial content next to the geometrical one.



**Fig. 23** The puzzle with given matings is abstracted as a spring-mass system that evolves over time towards a convergence state. If the pieces are far apart (left), the springs pull them closer. When the pieces overlap

(middle), the springs pull them apart again. With some damping (i.e., loss of energy due to friction), the system eventually converges to a state of minimal energy



**Fig. 24** Several snapshots of the numerical simulation for a noisy puzzle of 25 cuts and 940 pieces. **A** Initial state. **B** Intermediate state. **C** Final state. **D** A zoomed section of the region marked in the final state shows the approximated final placement due to the noise

### 8.2.1 Hierarchical Layered Loops

As is usually done in jigsaw puzzle solvers, we start by finding candidate mates for each edge by aggregating the set of all unordered pairs of edges that satisfy the constraints  $\tilde{C}_1, \tilde{C}_2$  (cf. Sect. 5.1). We denote this set by  $\tilde{M}$

$$\tilde{M} = \left\{ \{e_i^j, e_k^l\} \mid e_i^j, e_k^l \in E \wedge \tilde{C}_1(e_i^j, e_k^l) \wedge \tilde{C}_2(e_i^j, e_k^l) \right\},$$

and recall that the higher the noise level, the more numerous are the potential matings, as analyzed in Sect. 7.9.

As mentioned earlier, in crossing cuts puzzles with uniformly distributed random cuts, the probability of more than two cuts meeting at a point is nil (cf. Sect. 4). It directly follows that all *inner* puzzle junctions constitute exactly four pieces. We utilize this property to identify ordered lists of 4 mating candidates that form such junctions, or *loops*, as illustrated in Fig. 25. Formally, a mating loop in the *clockwise* direction is a 4-tuple

$$(m_1, m_2, m_3, m_4) = (\{e_A^{j_A}, e_B^{j_B}\}, \{e_B^{j_B}, e_C^{j_C}\}, \{e_C^{j_C}, e_D^{j_D}\}, \{e_D^{j_D}, e_A^{j_A}\}) \quad (12)$$

such that  $m_k \in \tilde{M} \forall k = 1..4$  and the following conditions hold:

Cond 1: No piece appears twice, i.e.  $p_A \neq p_B \neq p_C \neq p_D$  (i.e.,  $A \neq B \neq C \neq D$ ).

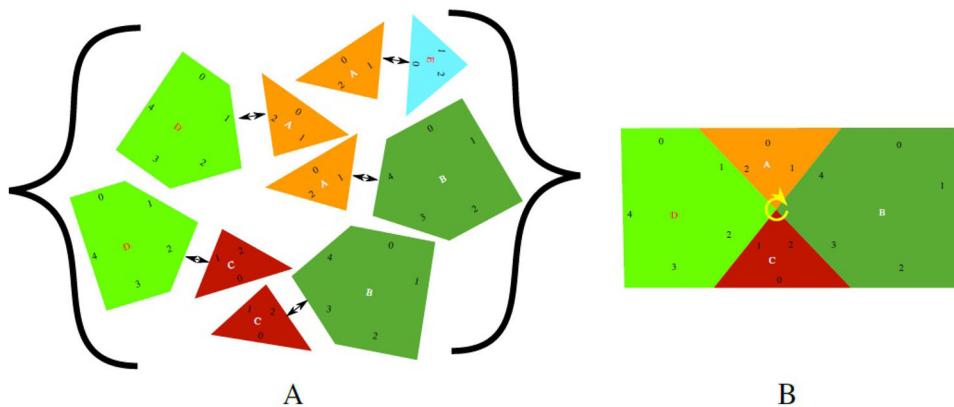
Cond 2: If a mating in the loop “enters” a piece  $p$  through its  $e_p^i$  edge, the consecutive mating “exists” the same piece through the adjacent edge  $e_p^j =$

$e_p^{(i-1) \bmod N_p}$ , where  $N_p$  is the number of  $p$ ’s edges (and also vertices; cf. Sect. 3). In other words, it “exits” through an edge immediately *counterclockwise* to  $e_p^i$  along the piece border. See edges  $e_B^4$  and  $e_B^3$  in Fig. 25B for an example.

Cond 3: The loop begins and ends with the same piece. This is in fact true by the definition in Eq. 12 as both the first and last matings contain the same edge of piece  $p_A$ .

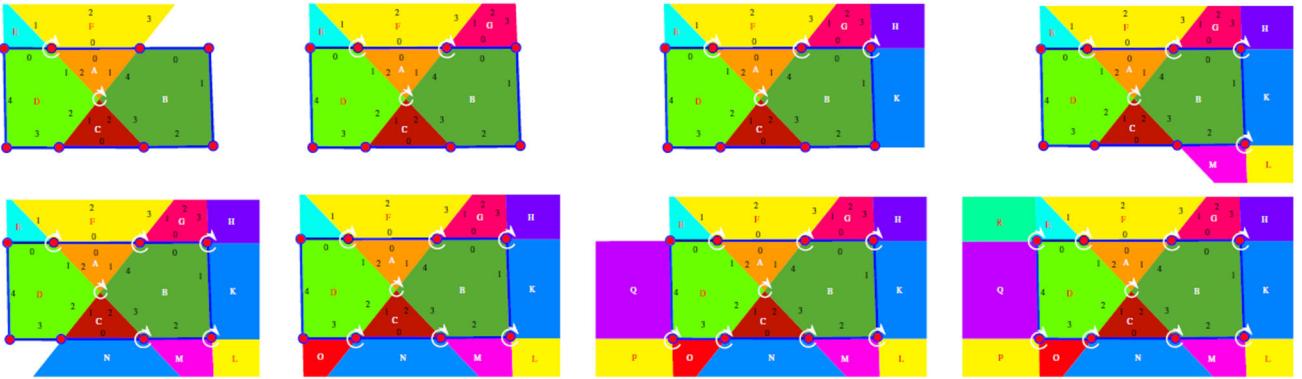
Since these basic loops are the building blocks for the puzzle reconstruction, and since their number is polynomial in the number of candidate matings, we search for them exhaustively among all  $O(|\tilde{M}|^4)$  possible mating 4-tuples, keeping only those that satisfy all of the above constraints. However, to nevertheless spare 75% of combinations and avoid searching and storing all 4 circular shift permutations of the same loop, we force loops to start with the edge having the lowest index.

Let now  $\mathcal{L}$  be the bag of basic loops computed as above. We now exploit partial *overlaps* between loops to identify *correct* matings more robustly instead of relying on  $\tilde{M}$  matings alone. More specifically, the next stage of the puzzle reconstruction algorithm is searching for “higher-order” loops, i.e., loops of loops, or *hierarchical loops* (Son et al., 2018). Denoting the basic 4-tuple loops in  $\mathcal{L}$  as 0-loops, we now seek all possible  $x$ -loops by trying to *fully enclose*  $(x-1)$ -loops with partially overlapping 0-loops, as illustrated in Fig. 26. Toward that



**Fig. 25** 0-loops formation from pairwise matings. **A** A bag  $\tilde{M}$  of 5 potential matings, of which  $(e_A^1, e_E^0)$  is wrong but included because it satisfied both  $\tilde{C}_1$  and  $\tilde{C}_2$ . **B** The loop  $(e_A^1, e_B^4) \rightarrow (e_B^3, e_C^2) \rightarrow$

$(e_C^1, e_D^2) \rightarrow (e_D^1, e_A^2)$  is identified and supports the plausibility of its constituent matings. Note that the path ending with  $(e_A^1, e_E^0)$  does not close a loop because the mating  $(e_E^2, e_C^2)$  is not present in the bag



**Fig. 26** Hierarchical loops formation. Shown left to right, top to bottom, the formation of a higher-level loop (in this case, 1-loop) is done by scanning the boundary of the lower-level loop (in this case, 0-loop)

until looping all around it with consistent 0-loops that match existing matings and pieces. In particular, the last added 0-loop much also match the first added one

end, let  $(e_1, e_2 \dots e_k)$  be the list of edges along the boundary of some  $(x-1)$ -loop. For example, the boundary of the 0-loop in Fig. 25B is  $(e_A^0, e_B^0, e_B^1, e_B^2, e_C^0, e_D^3, e_D^4, e_D^5, e_D^0)$ . Starting with  $e_1$  and ending with  $e_k$ , we progressively construct a higher level  $x$ -loop by searching and merging a proper 0-loop from  $\mathcal{L}$  that matches a *sub-loop* of the current  $x$ -loop around  $e_i$ . For example, if we start from the boundary edge  $e_A^0$  in Fig. 25B, we look for 0-loops that not only include that edge but also include edges from piece  $p_B$ , i.e., the mating  $\{e_B^4, e_A^1\}$  and the edge  $e_B^0$ . As shown in Fig. 26, the loop that was found in this particular example constitutes  $(\{e_A^0, e_F^0\}, \{e_F^3, e_G^1\}, \{e_G^0, e_B^0\}, \{e_B^4, e_A^1\})$ . Typically, and unless it is near the corner of the  $(x-1)$ -loop, the 0-loops that are identified for merging will need to match an existing sub-loop of at least 3 edges and at least one mating. And yet, despite these multiple constraints, it is possible that more than one 0-loop in  $\mathcal{L}$  will match around some boundary edge of the current  $(x-1)$ -loop, and consequently, it is possible that more than one  $x$ -loop will fit around a given  $(x-1)$ -loop.

In such cases, we generate and store them all for subsequent processing.

The process just described constructs the hierarchical loops in “layers” to produce a bag of  $x$ -loops for each layer  $x$ . Each of the 0-loops in  $\mathcal{L}$  may produce several 1-loops, each of them may produce several 2-loops, and so forth, until a layered representation is established, as illustrated in Fig. 27A. This process terminates at level  $x_{\max}$  if not even a single  $(x_{\max}+1)$ -loop can be constructed, an event likely to happen if such loops overflow beyond the true (though unknown) puzzle boundary.

### 8.2.2 Ranking Hierarchical Loops

Although hierarchical loops require simultaneous consensus between growing numbers of participating matings, and thereby reduce significantly the possibility of wrong combinations, false positives are still possible due to the noise. To rank better and worse loops, we utilize the fact that each

of them is a small noisy puzzle of pieces  $P_{loop}$  and (known) matings  $M_{loop}$  (cf. Sect. 6.1), and that “correct” loops can be “solved” for their spatial transformations with little to no overlaps even when collisions are allowed when we follow the multi-body spring-mass mechanism from Sec 8.1. We therefore employ this scheme and rank the different  $x$ -loops by their convergence state. We first define the following “quality” measure

$$Q_{overlap}(P_{loop}, M_{loop}) = \sum_{p_i \in P_{loop}} \frac{|A(p_i) \cap (\bigcup_{p_j \neq p_i} A(p_j))|}{|A(p_i)|} \quad (13)$$

where  $A(p_i)$  represents the *region* (as a set of points) of piece  $p_i$  in its final pose ( $R_i, \vec{t}_i$ ) and the measure as whole is a modified Dice coefficient (Dice, 1945) between each piece and the rest of the pieces. Since the distance between all adjacent vertices in “correct” loops also must be small, we also consider the distances between corresponding vertices as defined by  $M_{loop}$  measured *after* collisions are prohibited:

$$Q_{dist}(P_{loop}, M_{loop}) = \sum_{\vec{v}_i, \vec{v}_{i'}} \|\vec{v}_i - \vec{v}_{i'}\|^2.$$

Combining both scores into one rank we get:

$$Q_{loop}(P_{loop}, M_{loop}) = w_1 \cdot Q_{overlap}(P_{loop}, M_{loop}) + w_2 \cdot Q_{dist}(P_{loop}, M_{loop})$$

and while the weights can prioritize one score over the other, in our evaluation we found that  $w_1 = w_2 = 1$  produces excellent results and that sensitivity to these values is very small.

### 8.2.3 Merging Hierarchical Loops

Even with the best hierarchical loop found at the maximum level, the process of puzzle reconstruction is not yet finished since the maximum level of hierarchical loops does not necessarily cover the entire puzzle (e.g. the 2-loop in Fig. 27A). To complete the process and obtain the matings for the complete puzzle we now attempt to merge hierarchical loops. The  $x$ -loops are first sorted at each level  $x$  according to their rank  $Q$  (Sect. 8.2.2), and this list is then scanned from the best and highest level loops (Fig. 27B).

More formally, let  $P_{agg}$ ,  $M_{agg}$  denote the pieces and matings of the merging (or aggregation) process, initialized to be the best  $x_{max}$ -loop. Scanning now the sorted list of all  $x$ -loops, each is merged into the aggregated structure if several conditions hold. Assuming the pieces of the current  $x$ -loop under consideration are  $P_{loop}$ , and they are connected with  $M_{loop}$  matings, this loop is merged into  $P_{agg}$ ,  $M_{agg}$  if

- at least one piece is shared with the aggregated structure, i.e.  $P_{agg} \cap P_{loop} \neq \emptyset$ ,
- at least one piece is novel, i.e.  $P_{agg} \cup P_{loop} \neq P_{agg}$ , and
- there is no contradiction between the matings in  $M_{agg}$  and  $M_{loop}$ , i.e. if  $\{e_A^i, e_B^j\} \in M_{loop}$  then either  $\{e_A^i, e_B^j\} \in M_{agg}$  or none of the matings in  $M_{agg}$  contains edges  $e_A^i$  or  $e_B^j$ .

The merging process continues through the lowest ranked 0-loop, and is then repeated from the start until  $M_{agg}$  no longer changes during a full scan. This process must converge since the aggregation can include each possible mating at most once.

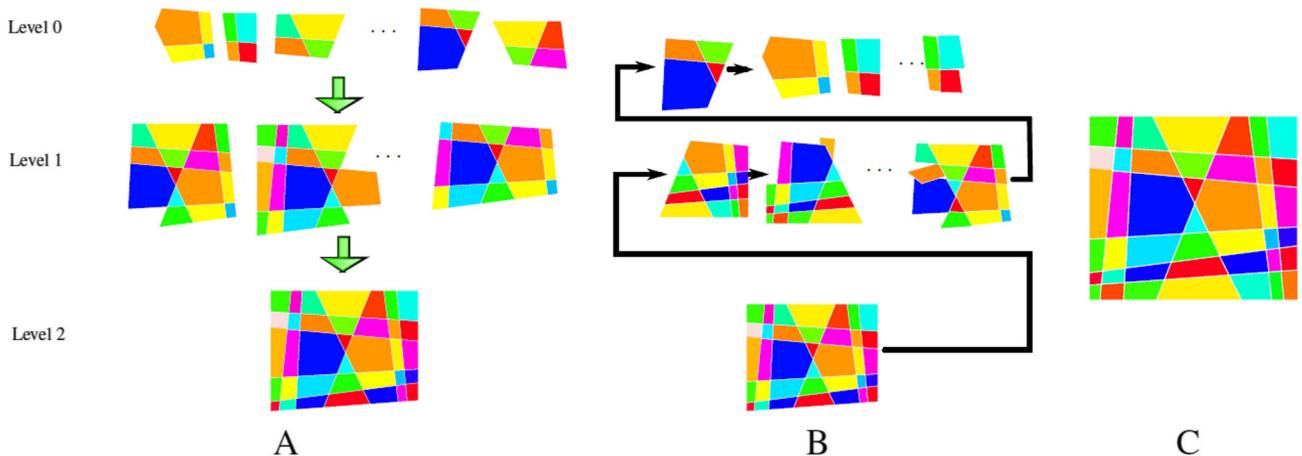
After the aggregated structure converges, the multi-body spring-mass process is performed one last time to position all the pieces  $P_{agg}$  properly based on the obtained mating  $M_{agg}$ . The result is the final reconstructed crossing cuts puzzle.

### 8.3 Incorporating Pictorial Constraints

As the analysis of puzzle properties showed, larger geometrical noise increases rapidly the number of potential mates that are found using the geometrical constraints ( $\tilde{C}_1$  and  $\tilde{C}_2$ ) (cf. Sec 7.9). Similar effect is induced by increasing the number of cuts. In these cases, using the pictorial content of the piece can provide a big advantage. In particular, while the initial set  $\tilde{M}$  of potential matings can be obtained using geometrical constraint, scoring and ranking these matings based on pictorial content may drastically reduce admissible matings and thus the computational effort of the reconstruction algorithm discussed in Sect. 8.2.

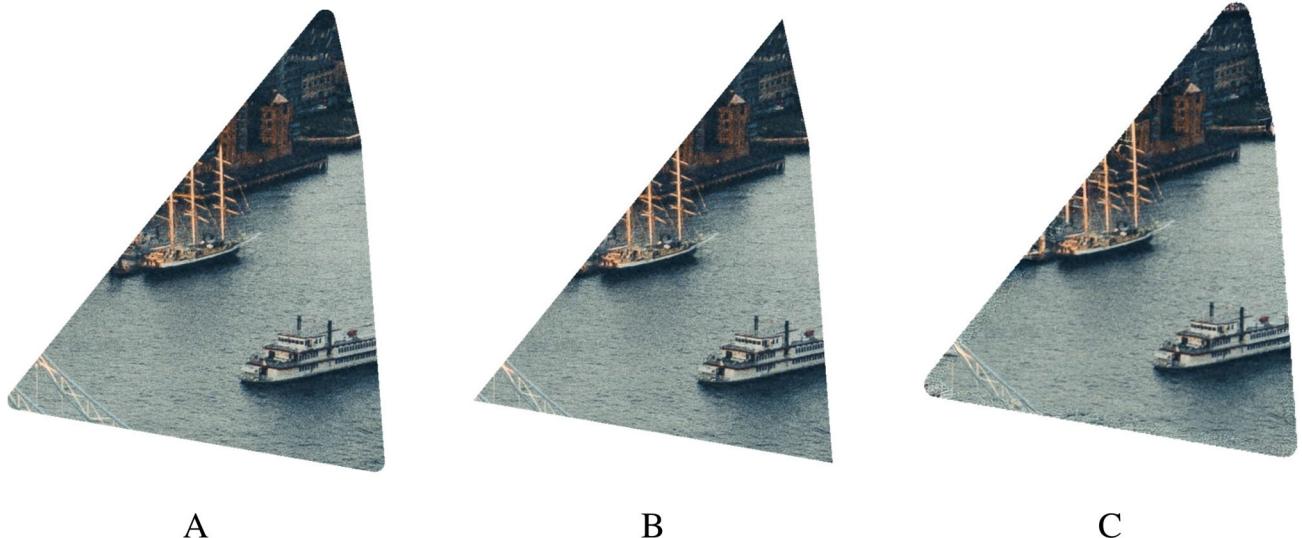
It should be emphasized from the outset that, unlike geometrical constraints, pictorial content alone cannot exclude matings with full certainty, as two genuinely neighboring pieces may legitimately have drastically different pictorial content even along their abutting boundaries. A solver can thus “take risks” and heuristically excludes pictorial matches below some predefined fidelity threshold, but strictly speaking, the pictorial content can help only in *prioritizing* certain matings over others, and therefore it can merge naturally into the ranking process described in Sect. 8.2.

Similar to methods proposed for solving other pictorial puzzles in the literature, mostly in the context of square jigsaw puzzles (see Sect. 2), a pictorial compatibility score can be based on some dissimilarity measure of the colors along the *edges* or *margins* of puzzle pieces, while paying less attention to the pictorial information deeper inside each piece. However, unlike in the common case studied in the square jigsaw puzzle literature, here our setting is far more challenging, for several reasons. First, pieces in crossing cuts puzzles are essentially never aligned with the pixel grid, making both the representation of the pictorial content and its



**Fig. 27** The outline of the reconstruction process. **A** Hierarchical loops of all levels are found, where each level is used as a starting point to search for the next level. Note that the max-level loop may not cover the entire puzzle. **B** The hierarchical loops are merged by iterating over

the loop levels in decreasing order. Each level is merged in increasing order based on the score. **C** The merged pieces are positioned using the spring-mass system



**Fig. 28** Extrapolation of visual information. **A** An original piece. **B** A geometrically noisy piece. **C** The extrapolated piece for extrapolation radius of  $\epsilon = 25$  pixels

use in a comparison measure, ill-defined and prone to aliasing (among other problems). Second, and even more critical, is the fact that the geometric noise renders the information that is vital for the comparison simply missing. In fact, it forces us to do what the square jigsaw puzzle literature has usually been avoiding deliberately, namely to use pictorial information *further away* from the piece boundaries. And third, the geometric noise also introduces uncertainty about the proper offset between neighboring pieces in the direction of the mates. In addition to Gur and Ben-Shahar (2017), who introduced the last consideration in their brick wall puzzle setup, only a handful of works address square piece puzzles with gaps between their pieces (i.e., eroded pieces that might have no direct contact), including Paumard et al. (2020) that

employs a deep network to predict the position of the pieces, and Song et al. (2023) that employs two types of deep networks and a genetic algorithm.

To deal with all these problems simultaneously, and inspired by similar ideas in the literature (Liu et al., 2011; Derech et al., 2021), we score a candidate mating  $m = \{e_i^j, e_k^l\}$  by extrapolating the information of the two corresponding puzzle pieces  $p_i$  and  $p_k$  to a spatial band beyond their boundaries and thus obtaining "dilated" pictorial pieces on which a compatibility measure can be applied more safely. There are many ways of doing such extrapolation, but most of those we experimented with perform too poorly to provide a reliable visual outcome that in turn can facilitate reliable pictorial compatibility score  $S(m) = S(\{e_i^j, e_k^l\})$  for mat-

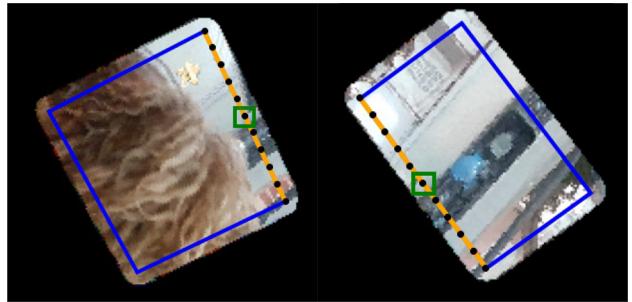
ing  $m$ . In our work, we first applied the basic inpainting method due to Telea (2004). We then fed the results, after resizing and padding, to a pre-trained deep Stable-Diffusion network (Rombach et al., 2022) to extrapolate the pieces beyond their original boundaries, as far as a band whose thickness is defined by the bound  $\varepsilon$  on the geometric noise. Figure 28 illustrates a selected result of the pictorial extrapolation and compares it to the original pictorial content. It goes without saying that in our case the available information is taken from the  $\varepsilon$ -noisy (i.e., “eroded”) piece while the pictorially extrapolated (i.e., “diluted”) piece usually extends even beyond the boundaries of the original noiseless piece.

With the extrapolated pieces computed, one can conceive many different ways to measure the compatibility of any two mates even without knowing the details of the geometric noise that affected them. For example, we note that by design of the extrapolation procedure, there must be some overlapping content between the two extrapolated pieces. Hence, one can attempt to register the two pieces and find the relative Euclidean transformation that places them next to each other with proper pictorial overlap along the extrapolated boundaries. This also may provide some information about the localization of pieces in the reconstructed puzzle, but at the same time, this approach is very sensitive and prone to errors (as the overlapping extrapolated pictorial information available for registration is both scarce and hypothetical) and therefore does not make the global optimization from Sect. 8.1 redundant. Because of such observations, it may be more effective to design a scoring mechanism that does not pretend to localize the pieces but is rather *invariant* to the relative transformation, for example by producing a *scalar* score for the dissimilarity embodied in any candidate mating. And while numerous such measures can be developed, we currently elected to implement the following simple scheme.

Given a candidate mating  $m = \{e_i^j, e_k^l\}$  from two different pieces  $p_i$  and  $p_k$ , we wish to compare the pictorial information around different corresponding points along  $e_i^j$  and  $e_k^l$ . For that, we sample both edges an equal number of times from one end to the other and compare visual windows around corresponding samples. More specifically, denote by  $W(\vec{v}) \in \mathcal{R}^{h \times h}$  the square pixel window around position  $\vec{v}$  and let  $F(\vec{v}) \in \mathcal{R}$  be its average color across the channels, i.e.,

$$F(\vec{v}) = \frac{1}{|W| \cdot 3} \sum_{\vec{w} \in W(\vec{v})} (R(\vec{w}) + G(\vec{w}) + B(\vec{w})) .$$

To evaluate the pictorial affinity of the two edges, we sample both  $e_i^j$  and  $e_k^l$  evenly  $G + 1$  times, including at their vertices  $(v_i^j, v_i^{j+1})$  and  $(v_k^l, v_k^{l+1})$ . We next consider all the mean color values of the windows  $W(\vec{v})$  along either  $e_i^j$  and  $e_k^l$  as two vectors in a  $G + 1$  dimensional space and compute the



**Fig. 29** The pictorial scoring process in action. Two running windows (in green) are simultaneously scanning the extrapolated pictorial content of the two candidate mates (in orange). The sample points are shown in black. The differences in window averages at all corresponding locations are aggregated into a  $G + 1$  dimensional vector, whose  $L_1$  norm serves as the dissimilarity measure  $S(m)$ . Note that the pictorial descriptor based on these window averages is invariant to the relative transformation, and although being simple it is descriptive enough to allow effective scoring and filtering or matings

$L_1$  norm of their difference. The result serves as our measure of dissimilarity  $S(m)$ . Formally,

$$S(m) = \sum_{k=0}^G \left| F \left( \frac{k}{G} \cdot v_i^{j+1} + \left( 1 - \frac{k}{G} \right) \cdot v_i^j \right) - F \left( \frac{k}{G} \cdot v_k^l + \left( 1 - \frac{k}{G} \right) \cdot v_k^{l+1} \right) \right| .$$

Note that the running windows may not be completely synchronized in relative position since the edges may have slightly different noisy lengths. In addition, the possibly different transformations of the two pieces suggest that the two pictorial windows will exhibit different aliasing and thus slightly different pixel values. However, the low pass filtering embedded implicitly in  $S(m)$  and the fact that a pictorial descriptor based on scalar window averages is invariant to the different relative transformations of each piece, provide robustness to both confounds. Clearly, one can conceive numerous other ways of implementing compatibilities for polygonal pieces and future work is likely to put additional attention on this challenge. However, despite being relatively simple, the  $S(m)$  measure is already descriptive enough to allow effective pictorial scoring of matings. A depiction of this process is provided in Fig. 29.

With a pictorial score for each candidate mating, and keeping in mind that  $\tilde{M}$  denotes all matings that satisfy the noisy *geometric* constraints, we now define the *pictorially constrained mating set*  $\tilde{M}_p$  by considering for each edge pair only the  $T$  matings that scored the best (i.e., lowest)  $S(m)$ , with  $T$  being some predefined number or an absolute percentile, that could depend on available computational or time resources. Formally, if  $H_T(X)$  denotes the set of  $T$  matings with the lowest  $S(m)$  score in a given set  $X \subseteq \tilde{M}$ , then  $\tilde{M}_p$  is defined as follows

$$\tilde{M}_p = \bigcup_{\{e \in E\}} H_T(\{m | m \in \tilde{M} \wedge e \in m\}) .$$

It goes without saying that the higher the geometrical noise (expressed by its bounds  $\varepsilon$  or alternatively,  $\xi$ ), the more significant the pictorial compatibility and pictorial mating filtering become. This quickly allows a drastic (typically an order of magnitude or larger) decrease in the number of potential matings, thus making much larger puzzles potentially solvable. Once the pictorially constrained set  $\tilde{M}_p$  is computed, the reconstruction process can proceed exactly as described for the apictorial case (cf. Sect. 8), including the global considerations encapsulated in the hierarchical layered loopy constraints (Sect. 8.2).

## 9 Evaluation Metrics and Experimental Results

This paper presented a new visual puzzle model, analyzed its properties, and suggested a solution scheme for both apictorial and pictorial variants. To test our approach, and having no prior work on crossing cuts or polygonal puzzles in the literature, our experimental evaluation focus on the formulation of performance metrics and reporting qualitative and quantitative results on the novel benchmark datasets presented in Sect. 6. Note that these datasets include both pictorial and apictorial puzzles with varying global shape, different numbers of crossing cuts, and a range of noise levels. Results of the naive algorithm for “clean” puzzles are not reported as it always reconstructs the puzzles perfectly.

### 9.1 Evaluation Metrics

As mentioned in Sect. 8, under geometric noise it is unclear what is the desired position (i.e., Euclidean transformation) of each piece in the reconstructed puzzle, and the multi-body spring-mass system aspires to obtain a solution that optimizes an intuitive objective. It still remains to score such solutions as to allow their quantitative evaluation and comparison, and for that purpose, one can assume the availability of a ground truth solution against which the evaluation is performed. As discussed in Sect. 3, any solution, be it the ground truth or one computed by a solver, constitutes both a mating graph and the Euclidean transformation of each piece, and thus the evaluation must take both into account. Unfortunately, this is not a straightforward task.

The evaluation of the mating graph is perhaps clearer, as we wish to compare two graph structures that could differ

only in their set of links.<sup>6</sup> Inspired by the Neighbor Comparison Metric from the square jigsaw puzzle literature (e.g., Cho et al., 2010; Pomeranz et al., 2011; Sholomon et al., 2013) we therefore define an evaluation metric for the computed matings as an area-weighted precision and recall measures of the computed matings:

$$Q_{precision} = \frac{\sum_{e_i^j, e_k^l \in M_{gt} \cap M_{sol}} (|A(p_i)| + |A(p_j)|)}{\sum_{e_i^j, e_k^l \in M_{gt}} (|A(p_i)| + |A(p_j)|)} \quad (14)$$

$$Q_{recall} = \frac{\sum_{e_i^j, e_k^l \in M_{gt} \cap M_{sol}} (|A(p_i)| + |A(p_j)|)}{\sum_{e_i^j, e_k^l \in M_{sol}} (|A(p_i)| + |A(p_j)|)} \quad (15)$$

where  $M_{gt}$  are the ground truth matings,  $M_{sol}$  are the matings of the solution found by the reconstruction algorithm, and  $A(p_i)$  represents the *region* (i.e., set of points) of piece  $p_i$  in its final pose, as in Eq. 13.

The scoring of the Euclidean transformation of pieces is more tricky. For example, we observe that even qualitatively perfect solutions by the spring-mass system may differ by a global Euclidean transformation due to arbitrary choice of a coordinate system in the representation of the pieces (cf. Sect. 3 and Fig. 3A). The situation becomes significantly more ambiguous once the solutions are not perfect (as is always the case under noise) and scoring needs to consider the placement of each and every piece of the puzzle separately.

With such challenges in mind, assume we have a solution we wish to score, i.e., the mating graph and the Euclidean transformation of all pieces in a reconstructed puzzle. Let  $\vec{u}_i^j$  be the vertices of piece  $p_i$  in the ground truth and  $\vec{v}_i^j$  the corresponding vertices of  $\tilde{p}_i$  in the obtained solution. We first globally align the obtained solution with the ground truth before comparing the placement of individual pieces. In other words, we wish to find a global Euclidean transformation  $(R^*, t^*)$  that aligns the reconstructed pieces “as close as possible” to the ground truth so they can be compared. To do so we employ SVD for Least-Squares Rigid Motion (Sorkine-Hornung & Rabinovich, 2017) to solve the following weighted minimization

$$(R^*, t^*) = \underset{(R, t)}{\operatorname{argmin}} \sum_{i=1}^n \sum_{j=1}^{N_i} w_i \| (R\vec{v}_i^j + t) - \vec{u}_i^j \|^2 \quad (16)$$

where the weights  $w_i$  are set to be proportional to the area of each piece to reflect the greater importance of larger pieces on the shape of the puzzle, i.e.,

$$w_i = \frac{|A(p_i)|}{\sum_{k=1}^n |A(p_k)|} . \quad (17)$$

<sup>6</sup> Recall that we reserved the term ‘edges’ for the boundary segments of the puzzle pieces while the edges of the mating graph are termed ‘links’ to avoid confusion.

Qualitatively, the more similar the mating graphs of the reconstructed puzzle and the ground truth, the better the global alignment will be and thus a better (i.e., smaller) score will be achieved by the optimal global transformation  $(R^*, t^*)$ . However, Eq. 16 in itself is not a convenient metric for the quality of the overall solution since it depends on the specific puzzle evaluated. In that sense, that score may allow the comparison of different solutions (say by different solvers) to the *same* puzzle, but it provides hardly any insights about the solution quality to an arbitrary puzzle, it does not allow ordering the solutions of different puzzles, and it prohibits aggregation of many solutions into statistical measures on whole datasets.

To overcome all these difficulties, we seek a more informative measure that is *normalized* to some canonical range (say  $[0, 1]$ ). We therefore consider the degree of area overlaps between the pieces in the solution vs. their ground truth counterpart, after the two solutions have been aligned with Eq. 16. Formally, if  $\tilde{p}'_i$  is the noisy piece  $\tilde{p}_i$  *after* being placed in the reconstructed puzzle, i.e.,

$$\tilde{p}'_i = \left\{ R_i \cdot \vec{v}_i^1 + \vec{t}_i, R_i \cdot \vec{v}_i^2 + \vec{t}_i, \dots, R_i \cdot \vec{v}_i^{N_i} + \vec{t}_i \right\}$$

then we define

$$Q_{pos}(R_1, t_1, \dots, R_n, t_n) = \sum_{i=1}^n w_i \cdot \frac{|A(p_i) \cap A(\tilde{p}'_i)|}{|A(\tilde{p}_i)|}. \quad (18)$$

where the weights are as defined in Eq. 17. This measure is conservative, in the sense that high scores always imply good solutions, but good solutions do not always receive high scores, as illustrated in Fig. 30. Future research may wish to explore improved metrics for such circumstances.

In summary, we use the mating measures  $Q_{precision}$  and  $Q_{recall}$  from Eq. 15 and the positions metric  $Q_{pos}$  from Eq. 18 as our quantitative measures for evaluating puzzle solutions. Next, we apply them to different test cases.

## 9.2 Experimental Evaluation of Piece Positioning

We first tested our crossing cuts solver for positioning puzzle pieces while assuming the matings are known, i.e., we only evaluated the degree to which the abstraction as a multi-body spring-mass system (Sect. 8.1) provides desired results, both qualitatively and quantitatively. Clearly, for this evaluation, it is irrelevant if the puzzle is pictorial or apictorial.

To implement this test we extracted from DB2 and DB3 puzzles their set  $\tilde{P}$  of noisy pieces *and* the ground truth matings  $M_{gt}$ , applied the positioning system (Sect. 8.1) and obtained the Euclidean transformation  $(R_i, t_i)$  of each piece  $\tilde{p}_i$  in the solution. Evaluation of the result was then based on Eq. 18.

The first of these evaluations examined the ability of the spring-mass system to converge to the desired spatial config-

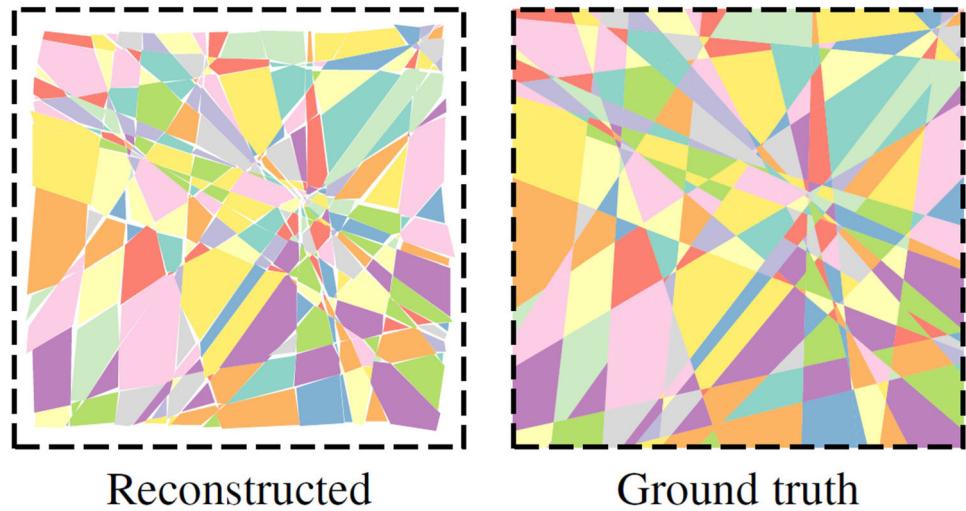
uration from the same initial state suggested by the algorithm, namely with the initial pose (position and rotation) of each piece chosen randomly inside the arena. Recall that the first run of the dynamical system allows pieces to overlap (a near-certain event under random piece positions). Upon convergence the same system is restarted but now while piece overlaps are prohibited. Figure 31 shows the initial and final configurations next to the ground truth of selected puzzles, and Fig. 32A presents the quantitative score  $Q_{pos}$  for puzzles with various noise levels. We were particularly interested in examining if the system might converge to improper local minima that depart qualitatively from the desired organization of pieces. This never happened and as shown in the examples, convergence is qualitatively correct even in the most complex cases.

The second test aims to empirically quantify a lower bound on the deviation from ground truth positions induced by the spring-mass system. To do so we applied the positioning computational to the same set of puzzles, though this time the initial state of the pieces was the ground truth position of the noisy pieces (where  $Q_{pos} = 1$ ), and the only computational step executed is the second phase where overlaps are prohibited. Intuitively, there could not be a better initial state for the pieces before the computation begins and thus Fig. 32B represents the best positioning scores possible. Figure 32C shows the performance difference between an optimal and a random initial states, thus representing how the deviation from the optimal positions is reflected in the positioning score. Note that in most cases the initial state of the positioning systems has negligible effect.

## 9.3 Experimental Evaluation of Apictorial Puzzle Solutions

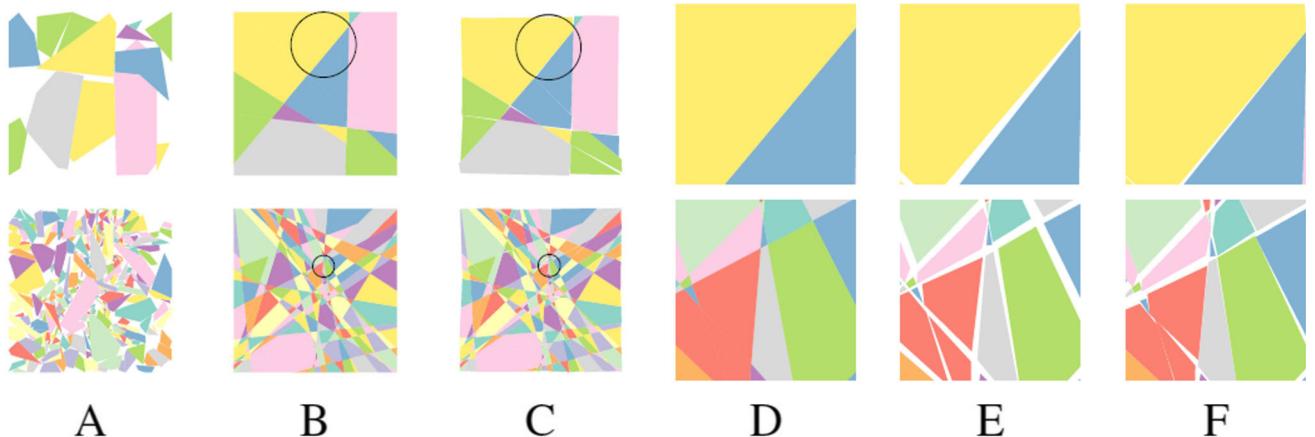
With a system to evaluate the solutions by the multi-body spring-mass system established, we turn to evaluate the full algorithmic solution under *unknown* matings (Sect. 8.2), where the input is just the noisy pieces  $\tilde{P}$  (and the bound on the noise level  $\xi$ ) while the output includes both the matings graph  $M$  and the Euclidean transformations  $(R_i, t_i)$  of each piece  $\tilde{p}_i$  in the solution. Here we first focus on apictorial puzzles and seek to evaluate both parts of the solution using the two evaluation measures, i.e., both the precision and recall from Eq. 15 and  $Q_{pos}$  from Eq. 18.

First qualitatively, Fig. 33 presents visual examples of successful reconstructions of selected puzzles of different global shapes, number of cuts, and noise levels. Note how the solution remains loyal to the (unknown) ground truth puzzle both in terms of its global shape and the organization of the pieces. The closeup insets show how the positioning system places the pieces at some distance, as would be desired due to the noise. Indeed, the configuration is not necessarily identical and in fact slightly perturbed relative to the ground truth,



**Fig. 30** Being conservative, the performance metric can score a solution low even if intuitively it is good. In this 30-cut and 326-piece noisy puzzle, the reconstructed solution is qualitatively similar to the noiseless ground truth and should be considered a success. And yet, this solution

scores just 0.56/1.00 by Eq. 18 because the noise is rather big, the gaps created between the eroded pieces is significant, and the springs tend to pull the pieces closer over the gaps and clump them in the center of the noiseless puzzle boundary (dashed line)



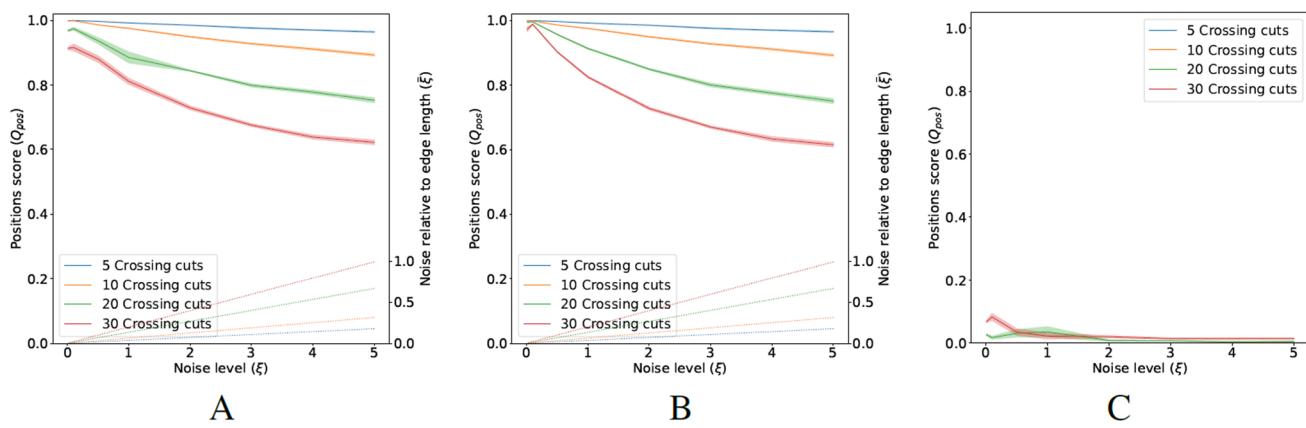
**Fig. 31** Qualitative performance of piece positioning for known matings. Since pictorial information plays no role in this test, we omit pictorial examples. **A** Initial random placement of pieces. Keep in mind that despite the messy organization the mates (i.e., the springs) are set correctly and we are interested to examine whether such a random initial state can lead to undesired local minima. **B** Ground truth assembly.

**C** Computed assembly after convergence of the second phase of the spring-mass system. The circle marks the area shown in the closeup section in the next column. **D** A closeup on the original ground truth. **E** A closeup on the corresponding section of the noised ground truth. **F** A closeup on the corresponding section of the reconstructed assembly

where the vertices' position (and thus the gaps) are determined automatically by the multi-body mechanical system while minimizing the energy of the springs.

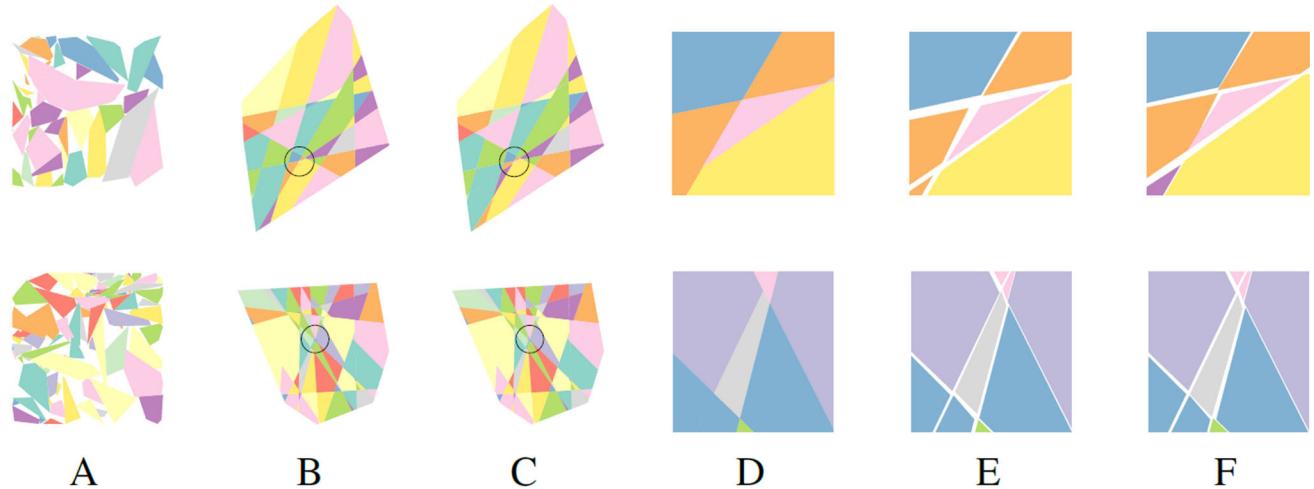
Figure 34A shows aggregated quantitative performance selected subsets of DB2. These results indicate that the mechanism based on the hierarchical loops, loop ranking, and loop merging obtains excellent results. Still, since the problem is intractable we cannot expect the heuristics to provide a perfect solution always, and Fig. 34B–D exemplifies one such unlikely failure.

It should also be mentioned that although the jigsaw problem is NP-complete, and thus complete or optimal solvers are expected to be exponential, relying on the crossing cuts geometrical constraints, and the looping and merging heuristics, can decrease the practical complexity significantly. Several of the steps become polynomial, and in particular, establishing 0-loops is bounded by a 4<sup>th</sup> order polynomial of the number of pieces. However, *in the noisy case* the number of possible mating combinations and the search in the merging step (cf. Sect. 8.2) remain exponential, where in practice they are influenced by the noise bound (cf. Sect. 7.9) and the number



**Fig. 32** Quantitative performance of piece positioning for known matings. The dotted lines show the value of  $\xi$  to appreciate how high it can get. **A** Average positioning score of the two-phase dynamical system as a function of noise level, computed from 10 random puzzles for selected numbers of crossing cuts/pieces. Shaded bands are  $\pm 1$  SE. **B**

Positioning score after the application of the second phase of the dynamical system from the ground truth initial positions. **C** The difference of the first two graphs indicates that in most cases the initial state of the positioning systems has no effect, except possibly for small noise cases



**Fig. 33** Examples of successful reconstruction results. **A** The unordered bag of puzzle pieces that the solver receives as input. **B** The ground truth assembly of (noiseless) pieces **C** The reconstruction result of the noisy puzzle. **D, E, F** A zoomed area of the noiseless ground truth, the noisy ground truth, and the solution. Unlike in the ground truth, the

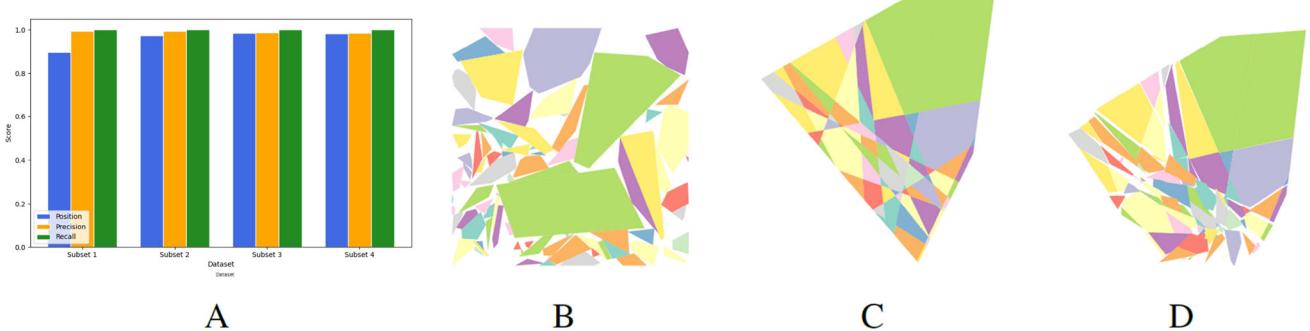
pieces in the noisy ground truth and in the solution do not abut each other, as would be expected because of the noise in their shape. In the solution, the gaps are determined automatically by the multi-body mechanical system while minimizing the global elastic energy of the springs

of cuts (cf. Sect. 7). This is why pictorial constraints are so valuable, and the better they can be utilized and reduce the number of candidate matings, the more efficient the solver can become for a given puzzle.

#### 9.4 Experimental Evaluation of Pictorial Puzzle Solutions

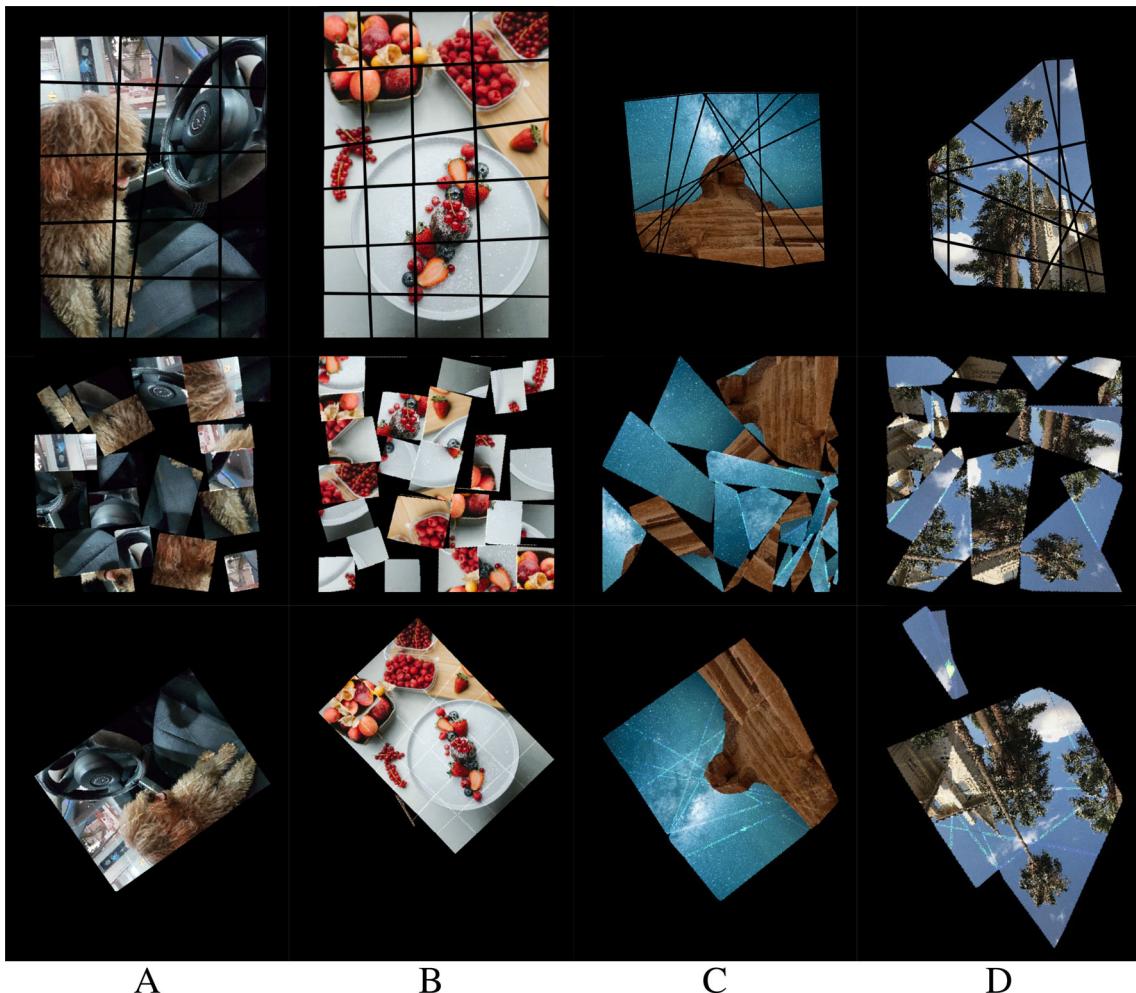
We finally turn to examine the reconstruction of pictorial puzzles and assess the role of pictorial constraints using puzzles from DB3 and DB4. Recall from Sect. 6 that DB4 is a general pictorial puzzle set, while DB3 is designed to play down the role of geometrical constraints by having pieces whose

edge length histogram is sharper (a condition that implies that each mate will have many more geometrically compatible matches). In such puzzles, the number of matings that satisfy constraints  $\tilde{C}_1$  and  $\tilde{C}_2$  is approaching the unfiltered set of matings, and thus the number of 0-loops, hierarchical loops, and possible geometrical solutions has easily overwhelmed the memory resources of the hardware we used for evaluation, which was a desktop computer with a 12th Gen Intel(R) Core(TM) i7-12700K Processor with a base clock speed of 3.60 GHz, and 32.0 GB RAM. Towards this end, we tested 10 puzzles from DB3 and DB4, all of which were solved only when the pictorial content was considered too. Selected qualitative solutions are shown in Fig. 35 while the



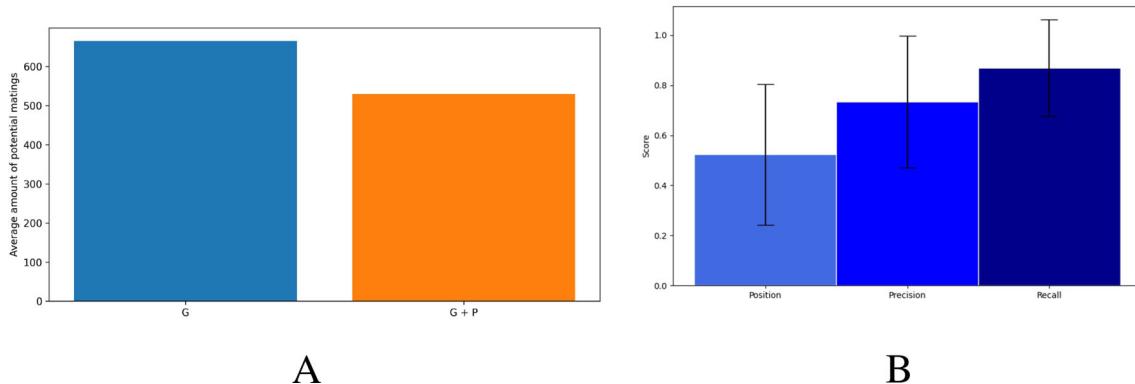
**Fig. 34** Reconstruction results on selected puzzles from DB2. **A** Results tested on 4 subsets of DB2, each containing 10 selected puzzles, sampled randomly from the original dataset with 5–50 cuts,  $\xi$  (noise level relative to the puzzle diameter) varying between 0 to 0.25% and  $\bar{\xi}$  (noise level relative to the average edge length) varying between 0 to 1.36%

The results show the positioning score (in blue), precision score (in orange), and recall score of the matings (in green). **B–D** An uncommon reconstruction failure. Shown are the unordered bag puzzle pieces that the solver receives as input (B), the ground truth solution (C), and the faulty reconstruction result (D) (Color figure online)



**Fig. 35** Selected examples of pictorial puzzle solutions (**A, B** - perturbed square jigsaw puzzles, **C, D** - general crossing cuts puzzles) having various numbers of cuts, piece size, noise levels, and visual contents. Top row shows the original image/puzzle. The second row represents the puzzle that was submitted to the solver, and the bottom

row shows the solution (slightly scaled down to fit the allotted space). Recall that solutions can be perfect up to a global Euclidean transformation. **D** shows one unusual failure caused by  $T$  being too restrictive thus excluding the correct mating



**Fig. 36** Pooled quantitative performance on pictorial puzzles. **A** Averaged on 10 DB3 puzzles, applying the pictorial constraints (P) on top of the geometrical ones (G) saves  $\sim 20\%$  of the original potential matings to test. This rate is of course based on the method used to filter the potential matings, based on their pictorial compatibility scores, and may vary accordingly. In this case, all of the potential matings scored

average saving in potential matings due to the pictorial constraints are depicted in Fig. 36A. These results are designated preliminary both because the pictorial filter is still simple, and because our present algorithm is not designed to deal with missing pieces, a condition that applies to many puzzles in DB4, once the level of the applied noise is increased.

Next to several successful solutions, Fig. 35 shows one rare failure result. Failures such as this could happen when the value of  $T$  is too small and correct matings are discarded by the pictorial filter. Indeed, although the filter eliminated many mating candidates, the aggregated quantitative performance on the tested puzzles from both DBs is good, as reported in Fig. 36B, C. Performance on DB3 puzzles is slightly lower since as implied above, they can hardly utilize the geometrical constraints.

## 10 Conclusions and Future Work

We introduced a new jigsaw puzzle model and analyzed its properties and the inherent challenges in solving them once pieces are perturbed with noise. To cope with such difficulties and keep the problem tractable, we abstracted it as a multi-body spring-mass dynamical system method endowed with hierarchical loop constraints and a merging process of layered puzzle loops. Results exhibit excellent solving power but also suggest that future work should utilize pictorial data on the pieces much more strongly to drastically reduce the number of potential mates per edge and turn the problem more tractable and thus truly suited for real-life applications. This work also introduces a new class of puzzle generation models that are partially constrained, well formulated, and have enough expressive power to allow more real-life applications while being subjected to more rigorous analysis. We

above a certain value (30%) were kept. **B** Precision and recall (Eq. 15) and position score (Eq. 18) for the pictorial puzzles from DB3. It should be noted that in cases where real puzzle matings had an unusually low pictorial compatibility score, they were filtered out of the potential matings, leading the solver to eventually fail to reconstruct the puzzle as a whole

hope this type of thinking about “restricted modeled puzzles” can expand puzzle-solving literature to new directions, where future work should also address even more general (but formal) generation processes and noise models, as well as how to handle missing pieces in all these cases.

**Acknowledgements** This work has been funded in part by the European Union’s Horizon 2020 research and innovation programme under grant agreement No 964854 (the RePAIR project). We also thank the Helmsley Charitable Trust through the ABC Robotics Initiative and the Frankel Fund of the Computer Science Department at Ben-Gurion University for their generous support.

**Funding** Open access funding provided by Ben-Gurion University.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## Appendix: Glossary

The following summarizes the different symbols and notations used in the paper, sorted alphabetically when applicable.

Symbol	Meaning
$a$	Number of crossing cuts that generate a puzzle
$A(p_i)$	The spatial region of piece $p_i$ in its pose ( $R_i, \vec{t}_i$ ) in the puzzle
$c_i$	cut $i$ that participated in generating the puzzle
$Cuts$	The set of cuts that generated a puzzle
$C_1, \tilde{C}_1$	The mate length constraint under ideal and noisy conditions
$C_2, \tilde{C}_2$	The mate angle constraint under ideal and noisy conditions
$D$	puzzle diameter (distance between furthest vertices)
$\Delta\Theta_e(L, \varepsilon)$	The bound of the orientation difference between $\angle e$ and $\angle e$ , i.e., between the orientation of the $\varepsilon$ -noisy edge and its original noiseless edge of length $L$
$E$	The set of all piece edges of the puzzle
$E_i$	The set of edges of piece $i$ of the puzzle
$e_i^j$	Edge $j$ from $\vec{v}_i^j$ to $\vec{v}_i^{j+1}$ of piece $i$ of the puzzle
$\mathcal{E}$	The edges (line segments between nodes) of $\mathcal{G}_{puzzle}$
$\varepsilon$	Bound on geometric noise in absolute units
$\tilde{\epsilon}_i^j$	Perturbation vector of vertex $j$ of piece $i$ due to the geometric noise
$G_M$	The mating graph of the puzzle
$\mathcal{G}_{puzzle}$	The planar graph that represents a synthesized crossing graph puzzle
$L, \tilde{L}$	The length of a clean and noisy edge, respectively
$\mathcal{L}$	The bag of puzzle loops obeying the geometrical constraints
$M$	The set of all matings that constitute a mating graph $G_M$
$M_{loop}$	The set of all matings in a hierarchical loop
$\tilde{M}$	The set of all matings candidates that satisfy $\tilde{C}_1, \tilde{C}_2$
$\tilde{M}_p$	The set of all matings candidates that satisfy $\tilde{C}_1, \tilde{C}_2$ and ranked best pictorially
$m_q$	Mating $q$ in a mating graph
$N_i$	Number of vertices (and edges) in piece $i$ of the puzzle
$P$	The set of puzzle pieces
$P_{loop}$	The set of pieces of an $x$ -loop
$p_i$	Piece $i$ of the puzzle
$\tilde{p}_i$	$\varepsilon$ -noisy piece $i$ of the puzzle
$Q_{loop}$	The quality score of a hierarchical loop
$Q_{pos}$	The normalized quality score (in [0, 1]) for piece positions in a puzzle solution
$R_i$	The rotation (matrix) applied to the vertices of piece $i$
$S(m)$	Pictorial compatibility score of mating $m$
$\vec{t}_i$	The translation (vector) applied to the vertices of piece $i$
$V_i$	The set of vertices of pieces $i$ of the puzzle
$\vec{v}_i^j$	Vertex $j$ of piece $i$ of the puzzle, ordered clockwise
$\angle \vec{v}$	Angle of the vector $\vec{v}$
$\mathcal{V}$	The nodes (intersection points) of $\mathcal{G}_{puzzle}$
$\xi$	Bound of geometric noise relative to puzzle diameter
$\tilde{\xi}$	Bound of geometric noise relative to average (expected) edge length

## References

- Adluru, N., Yang, X., & Latecki, L. J. (2015). Sequential monte carlo for maximum weight subgraphs with application to solving image jigsaw puzzles. *International Journal of Computer Vision*, 112(3), 319–341.
- Alajlan, N. (2009). Solving square jigsaw puzzles using dynamic programming and the Hungarian procedure. *American Journal of Applied Sciences*, 6(11), 1941.
- Ali, F. A. B. H., & Karim, F. B. (2014). Development of captcha system based on puzzle. In *2014 international conference on computer, communications, and control technology (I4CT)* (pp. 426–428). IEEE.
- Andalo, F., Taubin, G., & Goldenstein, S. (2016). PSQP: Puzzle solving by quadratic programming. *IEEE PAMI*, 39(2), 385–396.
- Andaló, F. A., Carneiro, G., Taubin, G., Goldenstein, S., & Velho, L. (2016). Automatic reconstruction of ancient Portuguese tile panels. *Graphics Appl: IEEE Comput*.
- Benaroya, H., & Han, S. Probability models in engineering and science.
- Brandão, S., & Marques, M. (2016). Hot tiles: A heat diffusion based descriptor for automatic tile panel assembly. In *European conference on computer vision* (pp. 768–782). Springer.
- Brown, B. J., Laken, L., Dutré, P., Gool, L., Rusinkiewicz, S., & Weyrich, T. (2012). Tools for virtual reassembly of fresco fragments. *International Journal of Heritage in the Digital Era*, 1, 313–329.
- Bunke, H., & Kaufmann, G. (1993). Jigsaw puzzle solving using approximate string matching and best-first search. In *International conference on computer analysis of images and patterns* (pp. 299–308). Springer.
- Burdea, B., & Wolfson, H. J. (1989). Solving jigsaw puzzles by a robot. *IEEE Transactions on Robotics and Automation*, 5(6), 752–764.
- Castañeda, A., Brown, B. J., Rusinkiewicz, S., Funkhouser, T., & Weyrich, T. (2011). Global consistency in the automatic assembly of fragmented artefacts. In *VAST*.
- Catto, E. Box2d. <https://github.com/erincatto/Box2D>.
- Cho, T. S., Avidan, S., & Freeman, W. T. (2010). A probabilistic image jigsaw puzzle solver. In *2010 IEEE computer society conference on computer vision and pattern recognition* (pp. 183–190). IEEE.
- Chung, M. G., Fleck, M. M., & Forsyth, D. A. (1998). Jigsaw puzzle solver using shape and color. In *ICSP'98. 1998 Fourth international conference on signal processing (Cat. No. 98TH8344)* (Vol. 2, pp. 877–880). IEEE.
- De Bock, J., De Smet, R., Philips, W., & D’Haeyer, J. (2004). Constructing the topological solution of jigsaw puzzles. In *2004 International conference on image processing, 2004. ICIP’04*. (Vol. 3, pp. 2127–2130). IEEE.
- Demaine, E. D., & Demaine, M. L. (2007). Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity. *Graphs and Combinatorics*, 23(1), 195–208.
- Derech, N., Tal, A., & Shimshoni, I. (2021). Solving archaeological puzzles. *Pattern Recognition*, 108065.
- Dice, L. R. (1945). Measures of the amount of ecologic association between species. *Ecology*, 26(3), 297–302.
- Fei, N., Zhuang, F., Renqiang, L., Qixin, C., & Yanzheng, Z. (2007). An image processing approach for jigsaw puzzle assembly. *Assembly Automation*, 27(1), 25–30.
- Freeman, H., & Garder, L. (1964). Apictorial jigsaw puzzles: The computer solution of a problem in pattern recognition. *IEEE Transactions on Electronic Computers*, 2, 118–127.
- Funkhouser, T., Shin, H., Toler-Franklin, C., Castañeda, A., Brown, B. J., Dobkin, D., Rusinkiewicz, S., & Weyrich, T. (2011). Learning how to match fresco fragments. *ACM Journal on Computing and Cultural Heritage*, 4, 7:1–7:13.
- Gallagher, A. C. (2012). Jigsaw puzzles with pieces of unknown orientation. In *2012 IEEE conference on computer vision and pattern recognition* (pp. 382–389). IEEE.
- Gao, H., Yao, D., Liu, H., Liu, X., & Wang, L. (2010). A novel image based captcha using jigsaw puzzle. In *2010 13th IEEE international conference on computational science and engineering* (pp. 351–356). IEEE.
- Gassner, N., Baase, W., & Matthews, B. (1996). A test of the “jigsaw puzzle” model for protein folding by multiple methionine substitutions within the core of t4 lysozyme. *Proceedings of the National Academy of Sciences*, 93(22), 12155–12158.

- Gioe, D. (2017). ‘The more things change’: HUMINT in the cyber age. In *The Palgrave handbook of security, risk and intelligence* (pp. 213–227). Springer.
- Goldberg, D., Malon, C., & Bern, M. (2002). A global approach to automatic solution of jigsaw puzzles. In *Proceedings of the eighteenth annual symposium on Computational geometry* (pp. 82–87). ACM.
- Grosman, L. (2016). Reaching the point of no return: The computational revolution in archaeology. *Annual Review of Anthropology*, 45, 129–145.
- Gur, S., & Ben-Shahar, O. (2017). From square pieces to brick walls: The next challenge in solving jigsaw puzzles. In *Proceedings of the IEEE international conference on computer vision* (pp. 4029–4037).
- Huang, Q., Flöry, S., Gelfand, N., Hofer, M., & Pottmann, H. (2006). Reassembling fractured objects by geometric matching. *ACM Transaction Graph.*, 25, 569–578.
- Jiang, X., & Bunke, H. (1993). An optimal algorithm for extracting the regions of a plane graph. *Pattern Recognition Letters*, 14(7), 553–558.
- Kleber, F., & Sablatnig, R. (2009). Scientific puzzle solving: Current techniques and applications. In *CAA*.
- Kleber, F., & Sablatnig, R. (2009). A survey of techniques for document and archaeology artefact reconstruction. In *ICDAR* (pp. 1061–1065).
- Koller, D., & Levoy, M. (2006). Computer-aided reconstruction and new matches in the forma urbis romae. *Bullettino Della Commissione Archeologica Comunale di Roma*, 2, 103–125.
- Kong, W., & Kimia, B. B. (2001). On solving 2d and 3d puzzles using curve matching. In *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001* (Vol. 2, pp. II–II). IEEE.
- Kosiba, D. A., Devaux, P. M., Balasubramanian, S., Gandhi, T. L., & Kasturi, K. (1994). An automatic jigsaw puzzle solver. In *Proceedings of 12th international conference on pattern recognition*, (Vol. 1, pp. 616–618). IEEE.
- Le, C., & Li, X. (2019). Jigsawnet: Shredded image reassembly using convolutional neural network and loop-based composition. *IEEE Transactions on Image Processing*.
- Li, Q., Geng, G., & Zhou, M. (2020). Pairwise matching for 3d fragment reassembly based on boundary curves and concave-convex patches. *IEEE Access*, 8, 6153–6161.
- Lindström, M. (2019). The geological development of the arctic. In *The Arctic* (pp. 3–25). Routledge.
- Liu, H., Cao, S., & Yan, S. (2011). Automated assembly of shredded pieces from multiple photos. *IEEE Transactions on Multimedia*, 13(5), 1154–1162.
- Makridis, M., & Papamarkos, N. (2006). A new technique for solving a jigsaw puzzle. In *2006 international conference on image processing* (pp. 2001–2004). IEEE.
- Marande, W., & Burger, G. (2007). Mitochondrial dna as a genomic jigsaw puzzle. *Science*, 318(5849), 415–415.
- Markaki, S., & Panagiotakis, C. (2023). Jigsaw puzzle solving techniques and applications: A survey. *The Visual Computer*, 39(10), 4405–4421.
- Mavridis, P., Andreadis, A., & Papaioannou, G. (2015). Fractured object reassembly via robust surface registration. In *Eurographics*.
- Mellado, N., Reuter, P., & Schlick, C. (2010). Semi-automatic geometry-driven reassembly of fractured archeological objects. In *VAST*.
- Mondal, D., Wang, Y., & Durocher, S. (2013). Robust solvers for square jigsaw puzzles. In *2013 international conference on computer and robot vision* (pp. 249–256). IEEE.
- Moore, T. L. (1991). Using euler’s formula to solve plane separation problems. *The College Mathematics Journal*, 22(2), 125–130.
- Murakami, T., Toyama, F., Shoji, K., & Miyamichi, J. (2008). Assembly of puzzles by connecting between blocks. In *2008 19th international conference on pattern recognition* (pp. 1–4). IEEE.
- Nielsen, T. R., Drewsen, P., & Hansen, K. (2008). Solving jigsaw puzzles using image features. *Pattern Recognition Letters*, 29(14), 1924–1933.
- Oxholm, G., & Nishino, K. (2011). Reassembling thin artifacts of unknown geometry. In *VAST*.
- Paikin, G., & Tal, A. (2015). Solving multiple square jigsaw puzzles with missing pieces. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4832–4839).
- Palmas, G., Pietroni, N., Cignoni, P., & Scopigno, R. (2013). A computer-assisted constraint-based system for assembling fragmented objects. *2013 Digital Heritage International Congress (DigitalHeritage)*, 1, 529–536.
- Papaioannou, G., & Karabassi, E.-A. (2003). On the automatic assemblage of arbitrary broken solid artefacts. *Image and Vision Computing*, 21, 401–412.
- Papaioannou, G., Karabassi, E.-A., & Theoharis, T. (2001). Virtual archaeologist: Assembling the past. *IEEE Computer Graphics and Applications*, 21, 53–59.
- Papaodysseus, C., Panagopoulos, T., Exarhos, M., Triantafillou, C., Fragoulis, D., & Doumas, C. (2002). Contour-shape based reconstruction of fragmented, 1600 bc wall paintings. *IEEE Transactions on Signal Processing*, 50, 1277–1288.
- Paumard, M.-M., Picard, D., & Tabia, H. (2020). Deepuzzle: Solving visual jigsaw puzzles with deep learning and shortest path optimization. *IEEE Transactions on Image Processing*, 29, 3569–3581.
- Pintus, R., Pal, K., Yang, Y., Weyrich, T., Gobbetti, E., & Rushmeier, H. E. (2014) Geometric analysis in cultural heritage. In *GCH*, pp. 117–133.
- Pomeranz, D., Shemesh, M., & Ben-Shahar, O. (2011). A fully automated greedy square jigsaw puzzle solver. In *CVPR 2011*, (pp. 9–16). IEEE.
- Radack, G. M., & Badler, N. I. (1982). Jigsaw puzzle matching using a boundary-centered polar encoding. *Computer Graphics and Image Processing*, 19(1), 1–17.
- Rika, D., Sholomon, D., David, E. O., & Netanyahu, N. S. (2019). A novel hybrid scheme using genetic algorithms and deep learning for the reconstruction of portuguese tile panels. In *Proceedings of the genetic and evolutionary computation conference*, (pp. 1319–1327). ACM.
- Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (June 2022). High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)* (pp. 10684–10695).
- Sağiroğlu, M. Ş., & Erçil, A. (2010). Optimization for automated assembly of puzzles. *Top*, 18(2), 321–338.
- Shin, H., Doumas, C., Funkhouser, T., Rusinkiewicz, S., Steiglitz, K., Vlachopoulos, A., & Weyrich, T. (2012). Analyzing and simulating fracture patterns of theran wall paintings. *Journal on Computing and Cultural Heritage (JOCCH)*, 5(3), 10.
- Sholomon, D., David, O., & Netanyahu, N. S. (2013). A genetic algorithm-based solver for very large jigsaw puzzles. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1767–1774).
- Sholomon, D., David, O. E., & Netanyahu, N. S. (2014). A generalized genetic algorithm-based solver for very large jigsaw puzzles of complex types. In *Twenty-eighth AAAI conference on artificial intelligence*.
- Sizikova, E., & Funkhouser, T. A. (2016). Wall painting reconstruction using a genetic algorithm. *Journal on Computing and Cultural Heritage (JOCCH)*, 11, 1–17.

- Son, K., Hays, J., & Cooper, D. B. (2014). Solving square jigsaw puzzles with loop constraints. In *European conference on computer vision*, (pp. 32–46). Springer.
- Son, K., Hays, J., & Cooper, D. B. (2018). Solving square jigsaw puzzle by hierarchical loop constraints. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- Son, K., Hays, J., Cooper, & D. B., et al. (2016). Solving small-piece jigsaw puzzles by growing consensus. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1193–1201).
- Song, X., Yang, X., Ren, J., Bai, R., & Jiang, X. (2023). Solving jigsaw puzzle of large eroded gaps using puzzlet discriminant network. In *ICASSP 2023 - 2023 IEEE international conference on acoustics, speech and signal processing (ICASSP)* (pp. 1–5).
- Sorkine-Hornung, O., & Rabinovich, M. (2017). Least-squares rigid motion using svd. *Computing*, 1, 1.
- Telea, A. (01 2004). An image inpainting technique based on the fast marching method. *Journal of Graphics Tools* 9.
- Toler-Franklin, C., Brown, B. J., Weyrich, T., Funkhouser, T., & Rusinkiewicz, S. (2010). Multi-feature matching of fresco fragments. In *SIGGRAPH 2010*.
- Toyama, F., Fujiki, Y., Shoji, K., & Miyamichi, J. (2002). Assembly of puzzles using a genetic algorithm. In *Object recognition supported by user interaction for service robots* (Vol. 4, IEEE, pp. 389–392).
- Tsamoura, E., & Pitas, I. (2009). Automatic color based reassembly of fragmented images and paintings. *IEEE Transactions on Image Processing*, 19(3), 680–690.
- Warren, L., Quaglio, F., Riccomini, C., Simões, M., Poiré, D., Strikis, N., Anelli, L., & Strikis, P. (2014). The puzzle assembled: Ediacaran guide fossil Cloudina reveals an old proto-Gondwana seaway. *Geology* 42, 5, 391–394.
- Webster, R. W., LaFollette, P. S., & Stafford, R. L. (1991). Isthmus critical points for solving jigsaw puzzles in computer vision. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(5), 1271–1278.
- Wetzel, J. E. (1978). On the division of the plane by lines. *The American Mathematical Monthly*, 85(8), 647–656.
- Willis, A., & Cooper, D. (2008). Computational reconstruction of ancient artifacts. *IEEE Signal Processing Magazine*. 25.
- Wolfson, H., Schonberg, E., Kalvin, A., & Lamdan, Y. (1988). Solving jigsaw puzzles by computer. *Annals of Operations Research*, 12(1), 51–64.
- Yaglom, A. M., & Yaglom, I. M. (1987). *Challenging Mathematical Problems with Elementary Solutions*, vol. 1. Dover Publications.
- Yang, X., Adluru, N., & Latecki, L. J. (2011). Particle filter with state permutations for solving image jigsaw puzzles. In *CVPR 2011*, (pp. 2873–2880). IEEE.
- Yao, F.-H., & Shao, G.-F. (2003). A shape and image merging technique to solve jigsaw puzzles. *Pattern Recognition Letters*, 24(12), 1819–1835.
- Yu, R., Russell, C., & Agapito, L. (2015). Solving jigsaw puzzles with linear programming. arXiv preprint [arXiv:1511.04472](https://arxiv.org/abs/1511.04472).
- Yilmaz, S., & Nabihev, V. V. (2023). Comprehensive survey of the solving puzzle problems. *Computer Science Review*, 50, 100586.
- Zhang, K., & Li, X. (2014). A graph-based optimization algorithm for fragmented image reassembly. *Graphical Models*, 76(5), 484–495.
- Zhao, F., He, X., Zhang, Y., Lei, W., Ma, W., Zhang, C., & Song, H. (2020). A jigsaw puzzle inspired algorithm for solving large-scale no-wait flow shop scheduling problems. *Applied Intelligence*, 50, 87–100.
- Zhao, Y.-X., Su, M.-C., Chou, Z.-L., & Lee, J. (2007). A puzzle solver and its application in speech descrambling. In *WSEAS international conference on computer engineering and applications* (pp. 171–176).

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.